Concept Embedding Analysis Based Methods for the Safety Assurance of Deep Neural Networks

Gesina Schwalbe

2022

Cumulative Dissertation Thesis

Concept Embedding Analysis Based Methods for the Safety Assurance of Deep Neural Networks

Towards Safe Automotive Computer Vision Applications

Gesina Schwalbe

Submitted on April 13, 2022
Defended on October 24, 2022

Submitted in partial fulfillment of the requirements for the degree of
Doctor of Natural Sciences (Dr. rer. nat.)
—
Faculty for Information Systems and Applied Computer Sciences
University of Bamberg

Supervisor and Reviewer:
Prof. Dr. Ute Schmid

Second Reviewer:
Prof. Dr. Diedrich Wolter

Head of Examining Board:
Prof. Dr. Gerald Lüttgen

DOI: 10.20378/irb-57172
Handle: uniba/57172

URN: urn:nbn:de:bvb:473-irb-571725
DOI: https://doi.org/10.20378/irb-57172

Gesina Schwalbe: *Concept Embedding Analysis Based Methods for the Safety Assurance of Deep Neural Networks.*
Submitted in partial fulfillment of the requirements for the degree of Doctor of Natural Sciences (Dr. rer. nat.), Faculty for Information Systems and Applied Computer Sciences, University of Bamberg.

To Oli.

Abstract

Deep neural networks (DNNs) are handled as a key technology for computer vision (CV) in automated driving and similar safety critical domains. For admission to market, the safe operation of such systems has to be ensured. However, DNNs come along with properties like complexity and opaqueness. These make new approaches towards safety argumentation and evidence acquisition inevitable. This thesis both identifies and tackles key practical issues associated with safety assurance of convolutional deep neural networks (CNNs) in automotive computer vision applications.

The key practical issue handled here is that many safety requirements origin from symbolic domain knowledge which relate semantic concepts from natural language, such as "*eyes* usually belong to an *obstacle*". That is a problem since outputs of typical object detection CNNs are restricted to few semantic label classes, and both input and intermediate outputs are non-symbolic. This thesis bridges this gap using methods from concept embedding analysis (CA). CA research tries to associate semantic concepts with items in the DNN intermediate outputs, thus providing access to symbolic knowledge encoded in the model. A baseline CA method is chosen according to a broad literature analysis, and, in the course of several experimental studies, substantially improved regarding efficiency and performance. This allowed for the first time to successfully apply CA to state-of-the-art CNNs for object detection.

Based on the improved CA method, diverse approaches to provide evidence for different types of symbolic safety requirements are developed. Concretely, CA is used (1) to verify correct encoding of semantic relations, (2) to build global, interpretable, and inspectable proxy models, and it is used (3) as part of a framework for inspection and verification of compliance with symbolic fuzzy logic rules. Applicability of the approaches is shown and evaluated on several state-of-the-art object detector CNNs and backends of such.

Lastly, the contribution of CA-based evidence generation to safety assurance is highlighted. A template for a safety argumentation structure is developed, and a broad review of existing DNN specific evidence generation methods is conducted. This reveals the need for the CA-based methods developed in this thesis, and properly positions them in the overall safety argument.

Altogether, the works in this thesis constitute a solid step towards safer usage of DNNs in automotive CV applications.

Detailed Summary

English Summary

In the recent years, CNNs have shown outstanding performance in CV tasks. Additionally, they are in general applicable to hardly specifiable problems like object detection. This has made them a key candidate to enable perception in highly and fully automated driving (AD). For such safety critical applications it must be proven that the risk of the system causing harm is sufficiently low. However, the resounding performance of DNNs comes along with some inherent properties that impede or invalidate traditional evidence and evidence generation methods: Little experience, new types of errors specific to machine learning (ML) functions, and DNNs being black-boxes that are hard to inspect by human auditors. These issues render traditional safety engineering insufficient to assure safety of a CNN-based component. A new safety argument structure and evidence approaches are required for systems using CNNs in a safety relevant way.

A concrete example is the verification of symbolic domain knowledge like "*eyes* usually belong to an *obstacle*". This requires a mapping of symbolic concepts (like *eye*) to either inputs, outputs, or intermediate outputs of the function to assess. This makes methods from CA especially interesting: CA is a sub-field of explainable artificial intelligence (XAI) that associates visual semantic concepts such as "*eye*" with internal representations of a CNN.

This thesis pinpoints practical issues, and constitutes several steps towards application of CNNs in safety critical automotive perception: (a) a baseline CA method from literature is adapted, substantially improved, and applied to an object detection use-case for the first time; (b) more than three applications are developed and realized with CA that allow to verify usage of symbolic prior knowledge within a CV CNN; and (c) a safety argument template is established, exposing the contribution of symbolic prior knowledge for requirements formulation and verification.

(a) Concept Analysis for Object Detection: This thesis establishes a general definition of CA. A review of explainability methods, and CA methods in specific, reveals the baseline CA method Net2Vec [24] as a suitable candidate for producing safety evidences. This is a supervised, post-training concept segmentation approach for CNNs. It attaches a simple linear model, the concept model, to the output of a DNN layer. This simple model is trained on few labeled samples, and the resulting weight vector is considered a representation of the concept in the DNN latent space, the *concept embedding*. In order to apply this baseline to an object detection CNN, this thesis suggests several substantial improvements, and experimentally validates them in comparative studies. This includes adaptations to the preprocessing, modeling, and training, which allow working with large networks, high-resolution datasets, and object part concepts like "*eye*". Besides the baseline BRODEN dataset from [7], the chosen CA approach is evaluated on three additional datasets: an artificial one, and two automotive related realistic datasets. Labels for the latter two are created in the course of this thesis.

(b) Applications of CA to Safety Evidence Generation: Related work on CA only considers analyzing sensitivity and proximity between concept embeddings. This thesis substantially increases the portfolio of types of symbolic requirements that can be assessed. Several verification applications utilizing CA are proposed, developed, and demonstrated on networks of realistic size for object detection:

Verification of (global) internal semantics: Whether task-relevant concepts are respected in the CNN internal representations, the representations are invariant to task-irrelevant concepts, and whether the similarity between concept embeddings corresponds to that of their semantic concepts;

Inspection and verification of encoded logic: CA is used to establish a symbolic interpretable proxy model approximating the CNN internal logic. This is shown to achieve high fidelity, and can be used, e.g., for manual inspection.

Verification and runtime monitoring with respect to fuzzy logic rules: A framework is developed based on CA that allows to monitor in a self-supervised manner compliance with symbolic fuzzy logic rules without changing the original CNN. The semantic concepts used in the rules in the form of unary logical predicates need not be part of inputs or outputs of the CNN. Instead, post-hoc attached concept models serve as the missing unary predicates. The truth values of the fuzzy logic rules grounded on given samples serve as logical consistency scores. Experimental results showed that this could uncover a substantial amount of errors during runtime, and reveal interesting error cases.

(c) Safety Argument Structure: First, the specialties and specific error types of DNNs in the CV domain are pinpointed. Three main types of errors are formalized: issues in simple generalization, such as caused by overfitting; robustness issues; and issues internal symbolic logical reasoning. Based on the error types, a template for a top-down safety argument is proposed. The safety argument is rounded out by a bottom-up catalogue of evidence generation methods: Methods and considerations providing evidence for the safety case are reviewed, and classified according to practical safety assessment aspects. This includes the applicability to different lifecycle stages, and the capability of the method to either improve, prove, or reduce risk due to a property. Given the safety argument template, open challenges and missing evidence generation methods are systematically determined. This highlights the need for quantitative methods to verify requirements derived from symbolic prior knowledge.

To conclude, a couple of further proposals are made to bridge the gap between finding safety issues during verification and solving them. Altogether, the methods presented and proposed in this thesis should provide a solid step towards safer usage of CNNs in automotive CV applications. It is also shown that still further gaps in the body of evidence are waiting to be filled by effective methods in order to obtain a convincing safety argument. I conclude with a call for further research in the direction of XAI for the sake of safe, and in general responsible, artificial intelligence (AI).

German Summary (Zusammenfassung)

In den letzten Jahren haben sich tiefe neuronale Netze (DNNs) mit herausragender Performanz in Computer Vision Problemstellungen hervorgetan. Daher, und dank ihrer generellen Flexibilität in schwer spezifierbaren Aufgaben wie Objekterkennung in Bildern, sind sie inzwischen ein aussichtsreicher Kandidat für die Implementierung von Perzeption im hoch- und vollautomatisierten Fahren. Allerdings muss im Falle solch sicherheitskritischer Anwendungen nachgewiesen werden, dass das Restrisiko für einen Schaden durch die verwendete Technologie genügend gering ist. Traditionellerweise wird die Zuversicht in die sichere Funktionalität eines Produktes über verschiedene Arten von Sicherheitsnachweisen hergestellt. Diese können die Anwendung bewährter Design- und Entwicklungsansätze sein, das Befolgen standardisierter Prozesse oder Ergebnisse aus Verifikations und Validierungsschritten, z.B. Tests, manuelle Inspektion, oder formale und semi-formale Verifikation. Wenn es um DNNs geht, geraten traditionelle Nachweismöglichkeiten allerdings an ihre Grenzen. Fehlende Praxiserfahrung im Feld mit der neuen Technologie, neue technologiespezifische Fehlertypen, sowie fehlender Einblick in die komplexen Vorgänge innerhalb eines DNN erschweren die Prüfung durch menschliche Auditoren und invalidieren oder behindern viele der bisher üblichen Nachweisformen. Entsprechend sind Anpassungen an den Ansätzen zur Absicherung von sicherheitsrelevanten DNN-basierten Systemen nötig. Genauer bedarf es einer Überarbeitung zum einen der Struktur der technologiespezifischen Sicherheitsargumentation und zum anderen der Ansätze zur Evidenzgewinnung, um den Einsatz von DNNs im autonomen Fahren zu ermöglichen.

Ein konkretes Beispiel für Entwicklungsbedarf in der Absicherung von DNNs ist die Verifikation von symbolischem Domänenwissen. Ein Beispiel für solches Domänenwissen bei visuellen Eingaben ist "*Augen* gehören normalerweise zu einem *Hindernis*". Um die Einhaltung von derartigem Vorwissen zu überprüfen, müssen symbolische Konzepte (z.B. *Auge*) verknüpft werden mit Merkmalen in der Eingabe, der Ausgabe oder Zwischenausgaben der zu prüfenden Funktion. Im Computer Vision Bereich sind Bildeingaben und Zwischenausgaben von DNNs allerdings nicht-symbolischer Natur und die Anzahl an Konzepten in der symbolischen Ausgabe eines DNNs ist begrenzt. Abhilfe kann hier das Forschungsfeld der Konzeptanalyse verschaffen: Konzeptanalyse oder Konzepteinbettungsanalyse ist ein Teilgebiet der erklärbaren künstlichen Intelligenz und befasst sich damit, visuelle semantische Konzepte (z.B. *Auge*) mit Repräsentationen in den Zwischenausgaben eines DNNs, den Konzepteinbettungen, zu assoziieren.

Diese Doktorarbeit behandelt mehrere Schritte auf dem Weg zur Verwendung von DNNs, im Speziellen Faltungsnetzen (CNNs), in sicherheitskritischen Perzeptionsanwendungen:

(a) Eine Referenzmethode für Konzeptanalyse aus der Literatur wird angepasst, grundlegend verbessert und erstmals auf eine Applikation zur Objekterkennung angewendet.

(b) Mehr als drei Anwendungsfälle von Konzeptanalyse für die Absicherung von CNNs werden entwickelt und implementiert, mithilfe derer die Einhaltung symbolischen

Vorwissens überprüft werden kann.

(c) Es wird eine Vorlage für den DNN-spezifischen Teil eines Sicherheitsarguments entwickelt, wobei der Beitrag symbolischen Vorwissens für die Formulierung von Anforderungen und die Verifikation herausgearbeitet wird.

Zu (a) Konzeptanalyse für Objektdetektion: Im Rahmen dieser Arbeit wird eine allgemeine Definition von Konzeptanalyse entwickelt. Für die Wahl einer Referenzmethode wird ein weitreichender Überblick über Methoden der erklärbaren künstlichen Intelligenz (XAI) im Allgemeinen und Konzeptanalysemethoden im Speziellen gegeben. Anhand dessen sowie von anwendungsfallspezifischen Kriterien wird das Verfahren Net2Vec [24] als geeigneter Ausgangspunkt ausgesucht. Net2Vec ist ein überwachter Lernansatz für das Training einfacher linearer Modelle (Konzeptmodelle), die Zwischenausgaben eines CNN-Layers in semantische Segmentierungsmasken für ein Konzept umwandeln. Das Konzept muss hierbei lediglich über wenige Beispieldaten vorgegeben sein. Der resultierende Gewichtsvektor des Konzeptmodells wird als *Einbettung* des Konzepts aufgefasst, d.h. Repräsentation des Konzepts in der DNN Zwischenausgabe. Um diese Referenzmethode auf Objektdetektoren anwenden zu können, werden einige tiefgreifende Verbesserungen entwickelt und deren Wirksamkeit in Vergleichsstudien experimentell untersucht. Adaptionen betreffen u.a. die Vorverarbeitung, die Architektur der Konzeptmodelle und deren Training. Die Verbesserungen erlauben den Einsatz der Konzeptanalysemethode auf großen CNN-Architekturen, hochauflösenden Bildeingaben und Konzepten, die Objektteile beschreiben (z.B. *Auge*) anstelle einfacherer Textur- oder Farbkonzepte. Neben dem Standarddatensatz BRODEN [7] für Konzeptanalyse wird das neue Verfahren auf drei weiteren Datensätzen evaluiert: einem generierten Datensatz zu Gesichtsmerkmalen und zwei Datensätzen mit realen Bildern und Bezug zum autonomen Fahren. Für letztere werden im Rahmen dieser Doktorarbeit Labeling-Schemata entwickelt.

Zu (b) Anwendungen von Konzeptanalyse für Evidenzgenerierung: Vergleichbare Arbeiten beschränken sich auf die Analyse von Sensitivität ("Wie stark hängt ein Konzept vom anderen ab?") und Ähnlichkeit zwischen Konzepteinbettungen, was einige wichtige Formen symbolischer Sicherheitsanforderungen nicht abdeckt. Diese Lücken werden hier angegangen, indem ein breites Repertoir an Prüfanwendungen auf Basis von Konzeptanalyse entwickelt wird. Deren praktischer Nutzen wird auf typischen CNNs und CNN-Backends für Objekterkennung demonstriert. Folgende Anwendungen werden betrachtet:

Verifikation (globaler) interner Semantik: Hierbei wird nachgewiesen, dass (1) Konzepte, die für die Anwendung relevant sind, in der internen Repräsentation des CNNs widergespiegelt sind, z.B. *Augen* oder *Gesichtsmerkmale* bei Fußgängererkennung; und dass (2) das CNN invariant ist gegenüber Konzepten, die für die Anwendung irrelevant sind bzw. sein sollten, z.B. *Körpergröße*. Außerdem wird überprüft, ob (3) die relative semantische Ähnlichkeit von Konzepten auch in der internen Repräsentation wiederzufinden ist.

Inspektion und Verifikation kodierter Logik: Bei diesem Verfahren wird Konzeptanalyse genutzt, um globale, interpretierbare Ersatzmodelle zu erstellen, die die symbolischen Ausgaben der Konzeptmodelle entgegennehmen und das Ausgabeverhalten des Originalnetzes immitieren. Solch eine verständliche Approximation der internen symbolischen Logik des CNNs kann für manuelle Inspektion genutzt werden.

Verifikation und Laufzeitüberwachung hinsichtlich Fuzzy-Logik-Regeln: Es wird ein Framework basierend auf Konzeptanalyse entwickelt, das pro Ausgabe oder auf einem Testdatensatz eine Kennzahl liefert, wie gut sich ein gegebenes CNN an vorgegebene symbolische Fuzzy-Logik-Regeln hält. Das Verfahren ist selbstüberwacht, ändert das trainierte CNN nicht und die Konzepte, welche in Form einwertiger logischer Prädikate in den Regeln verwendet werden, müssen weder Teil der Ein- noch der Ausgabe des CNNs sein. Fehlende Prädikate werden durch nachträglich erlernte Konzeptmodelle repräsentiert. Wie experimentell gezeigt, kann die Kennzahl genutzt werden, um einen erheblichen Anteil an Fehlern aufzudecken, zur Laufzeit oder in einem Verifikationsschritt. Außerdem kann sie in der manuelle Inspektion dabei unterstützen, eine Vorauswahl interessanter Beispiele zu treffen und so Fehlerfälle zu erkennen.

Zu (c) Struktur der Sicherheitsargumentation: In einem ersten Schritt zu einer systematischen Sicherheitsargumentation für DNNs in Computer Vision Anwendungen werden Besonderheiten und spezifische Fehlertypen herausgearbeitet. Dabei ergeben sich als Haupttypen: Probleme mit einfacher Generalisierung, wie etwa verursacht durch Overfitting; Robustheitsprobleme; und fehlerhafte interne Repräsentation bzw. fehlerhafte interne symbolische Logik. Anhand dieser Fehlertypen wird eine Vorlage für ein Top-Down-Sicherheitsargument erarbeitet. Diese Argumentationslinie wird ergänzt durch einen Katalog von Methoden zur Evidenzgenerierung, klassifiziert nach praxisorientierten Kriterien: die Einordnung im Lebenszyklus des DNNs (z.B. Training oder Verifikation), sowie das eigentliche Ziel der Methode (Verbesserung des DNNs, Prüfung oder Minderung eines Risikos durch Maßnahmen auf Systemebene). Anhand der Argumentationsvorlage und des Methodenkatalogs werden offene Fragestellungen bzw. fehlende Evidenzgenerierungsmethoden identifiziert. Dies hebt vor allem einen grundlegenden Bedarf an Methoden zur quantitativen Verifikation von Anforderungen aus symbolischem Domänenwissen hervor. Die hier entwickelten Methoden füllen genau diese Lücke.

Zuletzt umfasst diese Doktorarbeit auch einige weitere Vorschläge, wie aufgedeckte Sicherheitsprobleme in DNNs auch in der Praxis behoben werden können. Zusammengefasst geht diese Arbeit einige wichtige Schritte auf dem Weg zu sicheren DNN-basierten Computer Vision Anwendungen in der Automobilindustrie. Nachdem außerdem weitere ungelöste Herausforderungen aufgedeckt werden konnten, wird zu weiterer Forschung in diese Richtung aufgerufen, um dem Ziel sicherer, verantwortungsvoller künstlicher Intelligenz näher zu kommen.

Publications

Peer-reviewed Publications

The following publications contain scientific contributions and results produced as part of the doctoral research of this thesis:

[Th01] **Gesina Schwalbe**. “Verification of Size Invariance in DNN Activations Using Concept Embeddings”. In: *AIAI 2021: Artificial Intelligence Applications and Innovations*. IFIP Advances in Information and Communication Technology. Springer, 2021. doi: 10.1007/978-3-030-79150-6_30. Reproduced in this work with permission from Springer Nature

Find details regarding contribution on p. 60, Subsec. 4.2.4, full text on pp. 233.

[Th02] Johannes Rabold, **Gesina Schwalbe**, and Ute Schmid. “Expressive Explanations of DNNs by Combining Concept Analysis with ILP”. in: *KI 2020: Advances in Artificial Intelligence*. Lecture Notes in Computer Science. Springer International Publishing, 2020, pp. 148–162. doi: 10.1007/978-3-030-58285-2_11. Reproduced in this work with permission from Springer Nature

Find details regarding contribution on p. 63, Subsec. 4.3.1, full text on pp. 246.

Note on contribution: First and second author contributed equally. My contributions concentrated on the CA approach and its experimental evaluation.

[Th03] **Gesina Schwalbe** and Martin Schels. “Concept Enforcement and Modularization as Methods for the ISO 26262 Safety Argumentation of Neural Networks”. In: *Proc. 10th European Congress Embedded Real Time Software and Systems*. Jan. 2020. url: https://hal.archives-ouvertes.fr/hal-02442796

Find details regarding contribution on p. 91, Subsec. 5.7.2, full text on pp. 315.

Note on contribution: The content was fully provided by me. The coauthor assisted in writing style and focus.

[Th04] **Gesina Schwalbe** and Martin Schels. “A Survey on Methods for the Safety Assurance of Machine Learning Based Systems”. In: *Proc. 10th European Congress Embedded Real Time Software and Systems*. Jan. 2020. url: https://hal.archives-ouvertes.fr/hal-02442819

Find details regarding contribution on p. 82, Sec. 5.4, full text on pp. 293.

Note on contribution: The content was fully provided by me. The coauthor assisted in writing style and focus.

[Th05] **Gesina Schwalbe**, Bernhard Knie, Timo Sämann, Timo Dobberphul, Lydia Gauerhof, Shervin Raafatnia, and Vittorio Rocco. "Structuring the Safety Argumentation for Deep Neural Network Based Perception in Automotive Applications". In: *Computer Safety, Reliability, and Security. SAFECOMP 2020 Workshops.* Springer International Publishing, 2020, pp. 383–394. DOI: 10.1007/978-3-030-55583-2_29. Reproduced in this work with permission from Springer Nature

Find details regarding contribution on p. 88, Sec. 5.6, full text on pp. 303.
Note on contribution: I contributed more than 50 % of the content, with a focus on DNN insufficiencies and a derived argument structure.

Preprints

The following preprints are planned for publication and contain scientific contributions and results produced as part of the doctoral research of this thesis:

[Th06] **Gesina Schwalbe**, Christian Wirth, and Ute Schmid. "Enabling Verification of Deep Neural Networks in Perception Tasks Using Fuzzy Logic and Concept Embeddings". In: *CoRR* abs/2201.00572 (2022). URL: http://arxiv.org/abs/2201.00572. *Submitted to 2022 European Conference on Computer Vision (ECCV).*

Find details regarding contribution on p. 67, Subsec. 4.4.2, full text on pp. 261.
Note on contribution: I contributed more than 90 % of content and experimental evalations. Coauthors contributed parts concerned with uncertainty calibration.

[Th07] **Gesina Schwalbe** and Bettina Finzel. "A Comprehensive Taxonomy for Explainable Artificial Intelligence: A Systematic Survey of Surveys on Methods and Concepts". In: *CoRR* abs/2105.07190 (2021). URL: http://arxiv.org/abs/2105.07190. *Accepted with minor revisions at Special Issue on Explainable and Interpretable Machine Learning and Data Mining of Springer Nature Journal* Data Mining and Knowledge Discovery*, ISSN 1573-756X.*

Find details regarding contribution on p. 16, Sec. 2.2, full text on pp. 128.
Note on contribution: Authors contributed equally.

[Th08] **Gesina Schwalbe**. "Concept Embedding Analysis: A Review". In: *CoRR* abs/2203.13909 (2022). URL: https://arxiv.org/abs/2203.13909. *Submitted to Springer Nature Journal Artificial Ingelligence Review Journal, ISSN 1573-7462.*

Find details regarding contribution on p. 17, Sec. 2.3, full text on pp. 186.

Related Work not Part of This Thesis

My following preprints contain work that is prior or related to the work in this thesis, but not considered part of this thesis:

[87] **Gesina Schwalbe** and Ute Schmid. “Concept Enforcement and Modularization for the ISO26262 Safety Case of Neural Networks”. In: *Proc. ECML PhD Forum 2019*. Sept. 2019. URL: https://ecmlpkdd2019.org/submissions/phdforum/

[86] **Gesina Schwalbe** and Martin Schels. “Strategies for Safety Goal Decomposition for Neural Networks”. In: *Extended Abstracts of 3rd ACM Comput. Science in Cars Symp.* Oct. 2019. DOI: 10.13140/RG.2.2.29392.74244

[43] Sebastian Houben, Stephanie Abrecht, Maram Akila, Andreas Bär, Felix Brockherde, Patrick Feifel, Tim Fingscheidt, Sujan Sai Gannamaneni, Seyed Eghbal Ghobadi, Ahmed Hammam, Anselm Haselhoff, Felix Hauser, Christian Heinzemann, Marco Hoffmann, Nikhil Kapoor, Falk Kappel, Marvin Klingner, Jan Kronenberger, Fabian Küppers, Jonas Löhdefink, Michael Mlynarski, Michael Mock, Firas Mualla, Svetlana Pavlitskaya, Maximilian Poretschkin, Alexander Pohl, Varun Ravi-Kumar, Julia Rosenzweig, Matthias Rottmann, Stefan Rüping, Timo Sämann, Jan David Schneider, Elena Schulz, **Gesina Schwalbe**, Joachim Sicking, Toshika Srivastava, Serin Varghese, Michael Weber, Sebastian Wirkert, Tim Wirtz, and Matthias Woehrle. “Inspect, Understand, Overcome: A Survey of Practical Methods for AI Safety”. In: *CoRR* abs/2104.14235 (Apr. 2021). URL: http://arxiv.org/abs/2104.14235

[88] **Gesina Schwalbe** and Christian Wirth. *Hybrid Learning for DNNs.* https://github.com/continental/hybrid_learning. 2021. *Source code of developed framework for concept embedding analysis.*

Acknowledgments

Before diving deeper into the technical results of this thesis, I would like to take one step back, recalling my long and winding journey that led to this work. This would never have been possible without the many hands helping me up and guiding my way. Hence, I here want to express my gratitude towards those many people—my supervisors, the examining board, collaborators, reviewers, friends and family—who supported me on my course.

First, I would like to express my heartfelt gratitude to my supervisor Prof. Dr. Ute Schmid for her guidance, helpful feedback, and constant encouragement, at any time and in any situation. She greatly helped me find the right focus and carve out a research profile in my interdisciplinary topic. Thereby, she never failed to enthuse me with new topics, and more than a few times straightened my confidence in my own work. Thanks, Ute, for getting me that far! My sincere thanks also go to my second supervisor Prof. Dr. Diedrich Wolter for his interest in my research, the interesting and inspiring discussions, and his fruitful encouragement to aim for ambitious research and publication targets. I also owe Prof. Dr. Gerald Lüttgen my gratitude for his interest in my work and for taking time to share helpful feedback and for taking over the lead of my doctoral examining board.

Besides the supervision on the part of University, I enjoyed constant, great and indispensable mentoring and support from within my company. Many thanks to my supervisors and mentors: Dr. Stefan Voget for driving me with his trust and high expectations; Dr. Martin Schels for many a good advice and his straight and helpful feedback; and Dr. Jing Xiao for her outstanding support and guidance for the topic of safety assurance, and for her advice and many wise words that helped me live through my phase as doctoral researcher. Appreciations also go to my colleagues for the discussions and help in technical issues, especially Christian Wirth, Umut Ikibas, Christian Hellert, and Antje Loyal. While I am glad for the many helpful reviews of my works provided by so many people, a special thanks goes to those who reviewed parts of the final synopsis of this thesis: Jing Xiao, Daniel Petrisor, Christian Wirth, Christian Hellert, Antje Loyal, Oliver Rümpelein, Stefan Voget, Martin Schels, and all other coauthors of my papers. Other than supervision, I was glad to have the opportunity for many interesting and fruitful collaborations that sparked my contributions in this thesis. Specifically, I would like to thank all my coauthors for joining me in nurturing our ideas up to publication readiness, even in the face of an upcoming deadline.

I also owe thanks to my employer, Continental AG, for making this dissertation possible. Specifically, I would like to thank Stefan Voget in the role of my line manager for constantly taking care of this. Part of the funding for my work was received by the German Federal Ministry for Economic Affairs and Energy within the project "KI Absicherung – Safe AI

for automated driving”. I am very grateful to the consortium for the open and broad collaboration, and to all my collaborators in the project for the many interesting and helpful discussions.

This thesis would not have been possible without the mentioned support from supervisors, the examining board, collaborators, and funding bodies. Finally, I would also like to highlight the more latent but none the less crucial role that was played by my family and friends. They formed me, and kept me up during this long journey with its ups and downs, earning my deepest heartfelt gratitude. Thus, thanks to all my friends and family for never giving up on me, even in times when I disappeared behind piles and nights of work. I would like to thank my mother, Regina, with all my heart for her encouragement and the inexhaustible believe and trust in my capability to achieve my goals. Thanks to my grandmother, Kornelia, for her open ears, and wise words. And finally, huge thanks also go to my love and partner, Oli, whose love, care, and jokes were some of the core ingredients to bring me through my time as student and doctoral candidate, including nearly two years of pandemic. Thanks, Oli, for conjuring a smile into my face even during my darkest (and then not so dark anymore) times, for many deep and funny discussions, and for making sure I do not loose myself in the course of this thesis!

Gesina Schwalbe, 2022

Contents

List of Figures

List of Tables

List of Abbreviations

A2D2	Audi Autonomous Driving Dataset [28]
AD	automated driving
Adam	Adaptive Moment Estimation [57]
AI	artificial intelligence
ASIL	Automotive Safety Integrity Level [49, Sec. 6.3.4]
bBCE	balanced binary cross-entropy (see Def. 3.3 (3))
BCE	binary cross-entropy
BRODEN dataset	BROad and DENsely labeled dataset [7]
bs	batch size
bw	batch-wise
CA	concept embedding analysis
CAV	concept (embedding) vector or concept activation vector
CNN	convolutional deep neural network
CV	computer vision
DNN	deep neural network
fBCE	focal loss [64] (see Def. 3.3 (5))
fpn	feature pyramid network
GSN	goal structuring notation [89]
GTSRB dataset	German Traffic Sign Recognition Benchmark dataset [93]
HAD	highly automated driving
IEC 61508	IEC 61508:2010 standard [45] on functional safety of electrical, electronic, and programmable electronic systems
ILP	inductive logic programming
Intersect Loss	intersection loss (see Def. 3.3 (4))
IoU	intersection over union or Jaccard index
ISO/PAS 21448	ISO/PAS 21448:2019 draft automotive standard [52] for safety of the intended functionality of a system
ISO 26262	ISO 26262:2018 international standard [48] for automotive functional safety
KI-A	KI-Absicherung publicly funded project [30]
KL	Kullback-Leibler divergence (see Def. 3.2 (3) and Def. 3.3 (7))
lr	learning rate

ML	machine learning
MS COCO dataset	Microsoft Common Objects in COntext dataset [65]
MSE	mean squared error (see Def. 3.3 (6))
SEooC	Safety Element out of Context [48, Sec. 3.138]
SGD	stochastic gradient descent
sIoU	set intersection over union
STPA	System Theoretic Process Analysis
SVM	support vector machine
XAI	explainable artificial intelligence

Part I.

Synopsis of the Thesis

1. Introduction

The overarching issue tackled in this cumulative dissertation is the question of how to argue and evidence safety for CNN-based perception systems, and, in specific, how to verify symbolic requirements on CNNs with non-symbolic image inputs. In this introductory chapter first Sec. 1.1 details the practical motivation and research questions at the heart of this work. Then, Sec. 1.2 shortly clarifies the scope of this thesis and the guiding example use-case. Sec. 1.3 gives an overview on the followed scientific approach to attack the research questions, and the resulting scientific contributions towards them made by this thesis. Lastly, this chapter concludes with the outline of the remaining thesis in Sec. 1.4.

1.1. Motivation and Research Questions

Since nearly a decade DNNs, and in specific CNNs, have shown outstanding achievements in the domain of CV. They are capable of solving hardly specifiable tasks only on the basis of examples, can deal with non-symbolic inputs like images, and can be scaled to tasks of high complexity like object detection in traffic scenarios [67]. These advantages allow them to outperform traditional manually crafted algorithms and most other machine learning methods [81]. Hence, DNNs are handled as key technology in several safety relevant domains like medical diagnostics [2], and environment perception in highly automated driving (HAD) and fully automated driving (AD) [56]. For safety critical applications in general, the manufacturer is required to provide proof that the risk of causing harm to persons, objects or the environment with the system is sufficiently low [48, 52]. Such proof can be, e.g., a safety case, i.e., a conclusive argumentation with a documented body of evidence [8]. This is typically used in automotive industries [48, def. 3.136]. According to Bishop and Bloomfield [8], evidence can be: compliance to a process, in particular best-practice measures during development; system design measures; and verification and validation results, especially from testing, inspection, and formal or semi-formal verification [51, Tab. 7], [82].

The overarching goal of this thesis is to make a step towards practical safety argumentation of CNNs, in specific for perception in AD that is based on CV. This is done by tackling the following successive needs that are further detailed in the upcoming subsections: safety requirements respectively a safety argument structure for DNN-based systems (Chap. 5); methods for the verification of symbolic safety requirements (Chap. 4); and practical applicable of these methods to object detection (Chap. 3).

1.1.1. Safety Argument Structure and Evidence Generation for DNNs

The resounding performance of DNNs comes at several costs, delicately impeding safety assurance and, thus, implementation in safety critical applications: (a) There is little experience with DNN-based perception in HAD and AD, in particular no reference system admitted to the market; (b) machine learned algorithms may exhibit new types of errors and invalidate assumptions of traditional verification [103]; and (c) DNNs in particular are black-boxes which are hard to inspect by human auditors [1]. Regarding (a), traditionally developed software has a long history in the automotive industry and is an integral part of existing standards and certification. However, there is little field experience available for DNNs, and consolidation of best practice is still on its way, such as in [104]. Point (b) is caused by the data dependence. A ML model is learned from correlations in the given training data. A bias in this training data may easily result in counterintuitive behavior (e.g., shortcuts) or even systematic misbehavior (e.g., exploitation of spurious correlations) [47, 3]. For one, this leads to new types of errors not yet considered in existing safety standards, like severe robustness issues. And secondly, this violates the basic assumption of extrapolation inherent to many traditional safety verification methods, especially testing. Lastly, the learning also has its part in the black-box property. Complex tasks like object detection on real world images require large networks with high-dimensional intermediate representations. The automatically learned information encoding is usually distributed, i.e., one unit participates in encoding several pieces of information [24]. High dimensionality and automated complex encoding leads to non-human-interpretable intermediate outputs, and altogether to (c): a black-box that is hardly accessible for safety inspection by humans [5]. Due to these issues, traditional safety engineering is insufficient to assure safety of a CNN-based component, and when starting this thesis it has been unclear how to amend and structure a safety argument for systems the safety of which relies on CNNs. This specifically holds for domains like open real world images that practically do not allow for completeness of testing [90]. Therefore, new and DNN specific safety requirements, new patterns for the safety argument, and new types of evidences and evidence generation methods are needed to safely leverage the power of DNNs in safety critical tasks. This gives rise to the following interrelated research questions tackled in Chap. 5:

> Research Question. *Given a DNN used for safety relevant CV, what are necessary general safety requirements for it, how to structure a safety argument based upon them, and what methods are available (or needed) to provide safety evidence?*

The striking quick rise and density of the activities towards safe AI in the recent years demonstrate the urgent practical need for solutions in the direction of DNN safety assurance and related fields like fairness, in specific for HAD. Examples of domains in which research towards safe AI was sparked are manifold. Development towards new standards for the use of ML have started both on national and international level, and for different aspects of systems and the system lifecycle (e.g., [59, 52, 104, 18, 17]); new lifecycle models were developed (e.g., [106]); different safety argument patterns were proposed (e.g., [106, 9, 37]); safety concerns associated with a DNN were collected (e.g., [103, 83]); and the development of safety methods for DNNs saw a rise. For the latter, the growing research fields of DNN adversarial robustness [95] and XAI [Th07, 66, 98] take over an important role in providing best-practice and evidence generation methods. By now, safety aware development of ML functions is considered part of the new notion of responsible AI [5]. This encompasses besides safety also ethical and privacy aspects, sharing their need for proper assessability of the created functions.

1.1.2. Verification with Respect to Symbolic Domain Knowledge

Even though perception tasks may be hard to specify, there usually is a lot of domain knowledge available [99]. This can be general knowledge about relevance of features (e.g., "*body part*s are indicative for *pedestrian*s, and *street light*s are not"); relations of features (e.g., "*arm*s and *leg*s are both *limb*s", "*head*s are usually attached to a *person*"); or even physical rules (e.g., "*person*s tend not to fly, at least not very *high*"). From a safety and accountability perspective, logical consistency with respect to such semantic domain knowledge must be ensured [Th03]. Hence, explainability of perception functions, i.e., "opening the black-box", must be considered inevitable for sufficient confidence in safety.

However, knowledge based on symbolic concepts from natural language semantics [105, 24] (e.g., *arm*, *leg*, *person*, etc.) may be hard to associate with inputs, intermediate or final outputs of a perception CNN: The network output is restricted to few semantic classes to reduce labeling costs; the inputs are non-symbolic (e.g., pixels); and the opaqueness does not allow to directly associate semantic knowledge with the complex network internals. XAI aims to mitigate the black-box property and lift knowledge and behavior encoded in the complex, non-symbolic inputs and network internals to a human interpretable symbolic formulation, as will be detailed in [Th07], Sec. 2.2. For one, this allows to manually inspect the internal reasoning, imitating traditional inspection measures and also useful for debugging. And secondly, the knowledge retrieved from the CNN can be compared with prior symbolic knowledge, allowing to verify safety requirements derived from symbolic knowledge. Concretely, this gives a starting point for the question guiding Chap. 4:

> Research Question. *For a CNN-based CV component, how to evidence compliance with requirements that are based on symbolic background knowledge in the image domain?*

Chap. 4 elaborates different requirement types together with applications to verify them that are developed in this thesis. That includes a modeling approach to formulate fuzzy domain knowledge on object detection as verifiable requirements. Lastly, this work also proposes and discusses solutions for fixing compliance issues of a trained CNN in a fine-tuning step.

1.1.3. Concept Analysis for Object Detection

One promising XAI idea that is closely considered in this thesis is that of concept embedding analysis (CA). CA in general seeks to directly associate semantic concepts (e.g., *eye*) with vectors or sub-spaces of the intermediate output of a DNN [Th08]. Notable baselines here are the supervised CA methods Net2Vec [24] and TCAV [55]. Both want to find the vector in the latent space that points towards a given visual semantic concept of interest. This vector is then called concept (embedding) vector or concept activation vector (CAV). Both do this by training a linear model, the concept model, to predict presence of the concept from activation map features. This linear model then describes a hyperplane separating between concept and non-concept examples, and the normal vector of this hyperplane is the searched-for CAV. The general approach of post-hoc supervised concept analysis with linear models has several benefits for application in safety assessment: The example-based approach allows to consider also hard-to-specify visual semantic concepts; being post-hoc, the methods are applicable to trained models without further training or architectural constraints; linear models are simple, introducing little further complexity, as needed for safety assessment; and the concept models may serve as additional (post-hoc attached) outputs of the model and allow for *quantitative assessment* of their outputs and CAV relations. The applicability of CAVs for (ethical) assessments was already proposed and demonstrated in [55]. Here, they suggest to measure the sensitivity of a DNN output with respect to an intermediate concept using the partial derivative along a CAV. While these CA methods are promising for safety verification, they have not yet been applied in a practical safety critical setting. In particular, they were only demonstrated on small classification networks and small resolution datasets, with efficiency problems like CAV size [31] or expensive preprocessing [Th01]. This prevents their application to object detectors and high-resolution image data, which is needed for automotive perception tasks. Also, the baseline for assessing properties regarding object part concepts suffers from poor performance results [24]. Given these shortcomings of the else promising CA methods, the question at the heart of Chap. 3 arises:

> Research Question. *How to associate symbolic knowledge with CNN internal representations in a way that is suitable for safety assessment purposes in practical CV perception tasks like object detection?*

1.2. Scope and Guiding Use-case

This thesis concentrates on post-training verification measures (either offline or online) for state-of-the-art object detection or segmentation CNNs on single frame camera inputs (which include objectness confidence outputs). Evaluations, hence, are done on typical CNN architectures utilizing a VGG16 [91], ResNet [39], or ResNeXt [108] backbone architecture (cf. Table 4.1). Requirements for evidence generation methods are derived from the guiding use-case of pedestrian detection detailed below (cf. Subsec. 3.1.1). For example, the semantic concepts considered for symbolic requirements on the object detector are human body part features like *eye*, *nose*, *arm*, *leg*.

This thesis concentrates on safety assurance aspects that are independent of the final hardware integration. Note that, thus, safety activities regarding lifecycle phases like integrated verification and validation are out-of-scope. An overview of considered lifecycle phases is later given in Subsec. 5.1.2.

Guiding Use-case Parts of this doctoral research were conducted in the context of the project KI-A [30][1] which aims to establish a provable safety argumentation that allows to assure the safety of CNN-based perception modules in HAD. The guiding use-case in the project is a CNN-based pedestrian detector component within an HAD system. This function acts as an emergency brake assistant component (the *pedestrian detector component*) for protection of vulnerable road users. It can automatically initiate braking of the vehicle. Hence, failures of the component may directly impose a risk to pedestrians (in case of false negatives) or hind road users (in case of false positives), making it safety relevant. As illustrated in Figure 1.1, the pedestrian detector component is sub-structured into: a CNN-based detection function; post-processing of the detection function output, e.g. distance estimation and tracking; and an online monitoring module for the detection function that feeds a final voting component that makes the final decision about braking. The CNN itself is also assumed to be safety relevant, receives single, high resolution images of diverse urban traffic scenes, and outputs bounding box predictions for vulnerable road users including pedestrians and cyclists. This thesis concentrates on safety measures regarding the trained CNN itself (offline verification) and the online monitoring (online verification), as highlighted in Figure 1.1.

1.3. Scientific Approach and Contributions

The core of this work is to investigate the following hypothesis:

> **Main Hypothesis.** Methods from concept embedding analysis (CA) can substantially contribute to the safety argument of CNN-based perception components via verification of requirements that are based on symbolic knowledge in the visual domain.

[1] Project "Methods and measures for the safety assurance of AI-based perception functions for automated driving (AI Validation)" funded by the German Federal Ministry for Economic Affairs and Energy, https://ki-absicherung-projekt.de

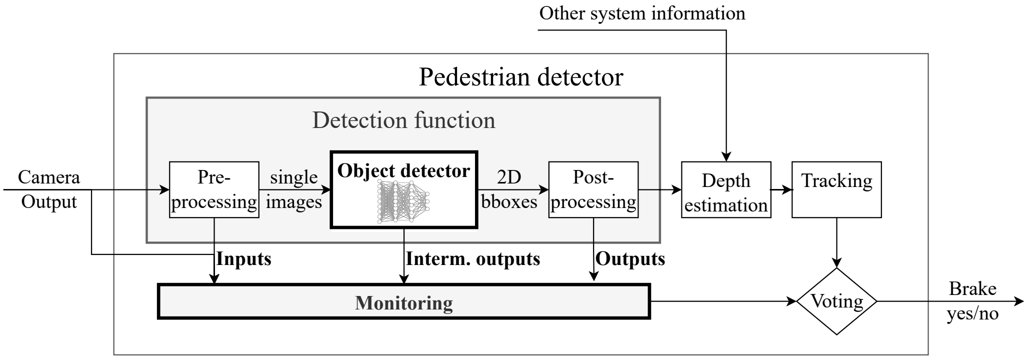


Figure 1.1. Data flow diagram depicting the considered pedestrian detector component structure, with relevant parts for this thesis marked in bold face.

1.3.1. Approach

The following gives a summary of the scientific approach and chronological line followed by the work in this thesis. It can be separated into four activities, reflected by the chapters in this synopsis: positioning of and within the research field CA (Chap. 2); optimization of a CA baseline method (Chap. 3); development of CA applications (Chap. 4); and, on the safety side, development of a safety argument structure and positioning of the contribution to safety assurance (Chap. 5).

The initial and continually guiding motivation was to enable safety assurance for DNN-based perception. Hence, the starting point was to systematically obtain an overview on existing methods for safety assurance of DNNs. This was compiled into [Th04], Sec. 5.4. The next step concentrated on finding concrete gaps, in the form of requirement types for which no assurance approaches existed. For that, I started a systematic derivation of safety requirements to match them with existing evidence providers (later iteratively improved). This is published in my work [Th03], Sec. 5.4. Here, a general need for methods to verify or enforce symbolic requirements is worked out.

Based on these findings, [Th03] also proposes several applications of CA for model verification and improvement that may fill this gap. The few baseline CA methods existing back then had not yet been tested on other domains. Thus, the next steps were to adopt the chosen baseline to different DNN architectures and datasets. In an initial evaluation setup, first necessary adaptations and options for improvements were systematically studied, as documented also in [Th03].

In parallel to the method development, I iteratively progressed on the safety argument structure. This allowed to fully capture the potential applications and contributions of CA for model verification and improvement. The results were discussed and published within the rising community for safety of machine learning in [Th05], Sec. 5.4.

On the method side, my further work now targeted the development and evaluation of concrete practical applications. That drove identification and evaluation of further substantial enhancements and variations of the baseline from [Th03]. Such included concept detection (in [Th02] and Sec. 3.6), stabilization (in [Th02]), and adaptation and optimization for transfer to larger models and object detectors (in [Th01]).

Concrete applications for post-hoc verification of CV DNNs were developed and evaluated in this order: (1) inspection via expressive interpretable proxy models in [Th02], Subsec. 4.3.1, (2) verification and inspection of encoded simple semantic relations and invariances in [Th01], Subsec. 4.2.4, and (3) monitoring and inspection of compliance with more complex fuzzy logic rules modeled on DNN and CA model outputs in [Th06], Subsec. 4.4.2.

Lastly, I captured the collected experience and related work on XAI and CA in review papers. This allows to position CA as a research field within the broader scope of XAI ([Th07], Sec. 2.2), and to clearly position the developed methods with respect to related work on CA ([Th08], Sec. 2.3; cf. Subsec. 3.1.1).

1.3.2. Contributions

Contributions in this thesis are successive building blocks towards the main hypothesis. The overarching goal is to enable CNN safety assurance, in specific with methods for the verification of symbolic requirements.

Establishing the research field of CA (Chap. 2).
A general theoretical perspective on CA is established, positioning it as a novel sub-field of XAI.

Improved CA for object detection (Chap. 3).
Net2Vec from [24] is a baseline method for post-hoc supervised segmentation of visual semantic concepts from trained CNN activations. This method is substantially improved with respect to efficiency on large concept datasets and object detection models, and with respect to performance on object part concepts and imbalanced concept datasets. Improvements are achieved by a combination of better optimization schemes, hyperparameter selection, and ensembling. This allowed for the first time to apply CA to state-of-the-art object detection tasks with large CNNs and datasets, and for the first time to achieve interesting results for object part concepts. The benefits are shown in comparative studies on different and diverse models (*see overview in Table 4.1*), including baseline classifiers trained on ImageNet [16] and several state-of-the-art object detector CNNs. Evaluations included two automotive real-world image datasets, the simple German Traffic Sign Recognition Benchmark dataset [93] (GTSRB dataset) (*[Th03]*) and the popular Microsoft Common Objects in COntext dataset [65] (MS COCO dataset) (*[Th01]*). For both of these datasets novel labeling schemes are developed, in order to tackle the scarce availability of suitable densely labeled datasets.

CA-based applications for verification of symbolic requirements (Chap. 4).
The following CA-based methods are developed to verify symbolic requirements of several types. Applicability is shown and evaluated each on at least one setting related to safety relevant automotive CV, and each including state-of-the-art object detection CNNs or CNN backends, like Mask R-CNN [38] and EfficientDet D1 [96].

[Th06]: A method is developed to verify and monitor compliance with fuzzy logic rules formulated on CNN outputs and concepts. It is shown that (1) a substantial amount of detection errors can be uncovered during runtime by the monitor, and this can already be achieved using just one rule; and (2) the compliance score (the fuzzy truth value of the formula) serves for efficient corner case selection: Selecting low-scored images revealed interesting and generalizable error cases. The method evaluation was conducted for a safety relevant logical occlusion robustness rule.

[Th02]: In order to allow manual inspection of the globally applied logic of a CNN, a method is developed that allows to build global, rule-based, interpretable surrogate models for large parts of a CNN. Concretely, post-hoc attached concept models are used to extract symbolic information from CNN intermediate outputs. This can then serve as inputs to symbolic surrogate models. To ease the association of concepts with positions in an image, a concept localization approach is developed (*[Th02]*). The here considered inductive logic programming (ILP)-based proxy models were shown to achieve high fidelity.

[Th01]: Prior work on concept embedding quality and CAV similarity measurement is used to verify requirements regarding internal representation of semantic concepts and semantic concept relations.

A safety argument structure for CNN-based perception (Chap. 5).
Based on systematically derived DNN-specific error types in CV, patterns for a safety argument structure are developed. This is done by jointly considering a top-down approach of breaking down top-level safety requirements, and a bottom-up approach identifying categories of evidence and evidence generation methods. The resulting argumentation template and method categorization scheme provide a basis for defining work products similar to traditional automotive standards [48, p. 3.185].

Lastly, a gap analysis of evidence generation methods on the base of the safety argument template is conducted. This shows the substantial contribution of CA-based methods towards completing the toolbox for assuring safety in automotive CNN-based CV applications.

1.4. Outline

The synopsis of this thesis will be structured as follows: First, Chap. 2 introduces the background on XAI and CA, including relevant definitions used throughout the other chapters.

Then, Chap. 3 gathers the improvements towards practical application of CA to object detection which are distributed over several publications that are part of this thesis. Here also the choice of baseline method is reasoned, and the methodological approach positioned within related work on CA. Chap. 4 introduces several CA-based methods to verify diverse symbolic requirements. Lastly, Chap. 5 positions the efforts regarding CA-based evidence generation methods within the larger scope of safety argumentation. It presents patterns for structuring and evidencing the safety argument of automotive CNN-based perception functions. Based on this, it is shown how the methods for CA-based verification can contribute to the overall safety evidence.

Paper Full Texts

The original full texts of papers that are part of this thesis (see publications overview on page xv) can be found in Appendix B. The reading order and positioning of the content within the overall work is pointed out by placeholder sections for each paper.

2. Background on Concept Analysis and Related Work

The research field of XAI by now constitutes a promising direction towards enabling human assessment of DNNs. This is of special interest for DNN-based applications in domains which are critical with respect to safety, security, or fairness. Efforts to open black-box functions for the sake of auditability have recently been summarized under the term responsible AI [5]. In the past years, starting with the work of Bau et al. in 2017 [7], an interesting sub-field of responsible AI has emerged [85]: Concept embedding analysis (CA) aims to associate the intermediate output representations of a DNN with human interpretable concepts. This idea enables an abundance of further assessment options, that pure behavior approximations like attribution heatmapping methods do not provide, as will be discussed in detail in Chap. 4.

This chapter gives the basic background and related work on concept embedding analysis, and how this field positions within XAI. That allows to clearly position the CA methods and applications developed in this thesis into a broader scope of related work. The method baseline used in Chap. 3 and Chap. 4 is chosen accordingly. In the following, first an overview on basic CA related definitions is provided in Sec. 2.1. Then, a broad introduction to the research field of XAI is presented in the paper [Th07] (Sec. 2.2). And lastly, my work [Th08] introduces CA in specific, providing a detailed review of related methods, applications, and datasets (Sec. 2.3).

2.1. Overview on Basic Definitions for Concept Analysis

A basic notion for concept analysis is that of a semantic concept, and for CV tasks that of *visual* semantic concepts.

Definition 2.1 (Concept). A *(semantic) concept* is a concept describable in natural language, i.e., a lexical synonym set as defined in the WordNet lexical database [105]. Some typical concept types are scenes, objects (e.g., *person*), object parts (e.g., *arm*), and object attributes like texture, material, or color [7]. A *visual semantic concept* is a concept that can be allocated to an image or image region. The detail of the allocation can be one of *classification* of a complete image (e.g., for scene concepts); or localization of a concept in an image via *detection* at the level of a region like a rectangular window, or *segmentation* at the level of a single pixel.

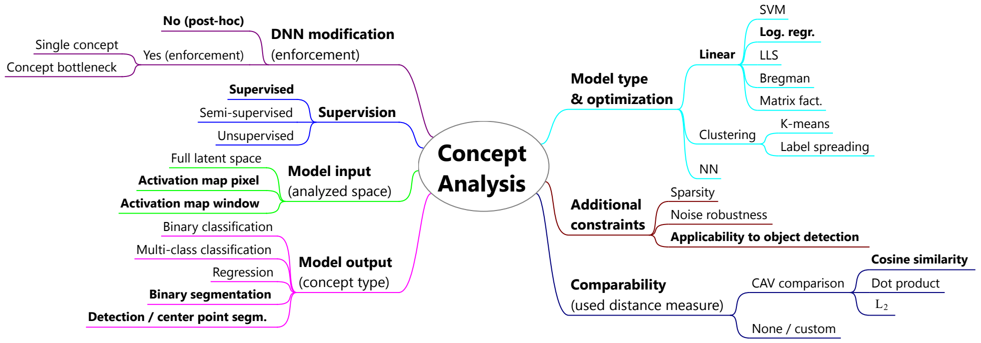


Figure 2.1. Overview on CA taxonomy aspects adapted from [Th08, Fig. 1] with aspects pursued in this thesis highlighted **bold**

Given a DNN, the underlying goal of concept analysis now is to answer the questions *whether*, *how*, and *with which properties* a semantic concept is represented in the intermediate outputs of the DNN. The core tool is the concept model associated to a concept. It accepts intermediate outputs of the DNN, and returns information about its concept.

Definition 2.2 (Concept Analysis)**.** *Concept (embedding) analysis* denotes an activity that tries to associate semantic concepts with vectors or sub-spaces in one preselected latent space of a DNN (i.e., layer intermediate output), with the constraints that this is done via a global helper model, the *concept model*, and the association is “simple” in the sense of not introducing further complexity.

Definition 2.3. The pair of a concept and its latent space representation obtained via concept analysis is called a *concept embedding*. In case concept analysis associates a concept globally or locally with a vector in a latent space of a DNN, this vector is called the *concept (embedding) vector or concept activation vector (CAV)* [24, 55].

Approaches to achieve the desired association of concepts and latent space representations are manifold. In the following, an overview is given on useful traits of concept analysis approaches that serve for comparing and categorizing them. Figure 2.1 summarizes the mentioned classification aspects and highlights the choices made for CA approaches used in this thesis (cf. Subsec. 3.1.1). An important notion needed to categorize concept models is that of the output format or concept label type.

Definition 2.4 (Concept Label Type)**.** The *concept label type* is the collection of attribute values a concept instance can take. The main classes of concept label types are single *binary* truth values [55, 24]; discrete *multi-class* values [53]; and continuous *regression* values [32].

As detailed in my work [Th08], important aspects that allow to categorize concept models include the following. Aspects agnostic to the concept model:

(1) The *supervision* of the concept model training;

(2) The *input format* of the concept model, i.e., which parts of the DNN latent spaces the concept model accepts, e.g. a window or a pixel in the activation map output of a convolutional layer;

(3) The *output format* of the concept model, i.e., the format of information about the concept, respectively the concept label type in supervised settings (cf. Def. 2.4);

(4) Whether the concept model is trained together with the DNN, modifying its architecture and weights (*inherent interpretability*), or is attached *post-hoc*.

Aspects specific to the chosen concept model type:

(5) The *type* of the machine learning model architecture chosen for the concept model;

(6) The *optimization* algorithm used to train the concept model (if any);

(7) Any *additional constraints* that need to be respected in the choice of concept model pre- or post-processing, architecture, or training algorithm, e.g., linearity or sparsity;

(8) The choice of a *comparison metric* applicable and used for the respective concept model type, e.g., L_2 distance in case of prototype CAVs for clustering models.

2.2. Publication: A Comprehensive Taxonomy for Explainable Artificial Intelligence: A Systematic Survey of Surveys on Methods and Concepts

The next content is contained in the following paper:

[Th07] **Gesina Schwalbe** and Bettina Finzel. “A Comprehensive Taxonomy for Explainable Artificial Intelligence: A Systematic Survey of Surveys on Methods and Concepts”. In: *CoRR* abs/2105.07190 (2021). URL: http://arxiv.org/abs/2105.07190. *Accepted with minor revisions at Special Issue on Explainable and Interpretable Machine Learning and Data Mining of Springer Nature Journal* Data Mining and Knowledge Discovery*, ISSN 1573-756X.*
Find the original full text attached on page 128.

Positioning in the Thesis

The survey reviews characteristics to classify XAI methods. A taxonomy is proposed and underpinned by diverse examples, including methods for CA. This outlines the properties making CA an especially suitable explainability method for symbolic requirements verification, and positions CA in the research field of XAI.

My Scientific Contributions

- A broad literature review on XAI method surveys and taxonomies, comparing surveys regarding their specialty and extensiveness;
- Definition of a detailed taxonomy summarizing all classification aspects for XAI methods mentioned in literature;
- Exemplary classification and short description of more than fifty diverse exemplary XAI methods according to the taxonomy aspects.

My Text and Content Contributions

The review methodology, review, taxonomy definition, and method selection and classification were developed in close collaboration with the coauthor, with equal distribution of contributions. I authored the sections *2.1 Related Work*, *3 Approach to Literature Search*, *4 A Survey of Surveys on XAI Methods and Aspects*, ca. 90 % of the texts in *5 Taxonomy*, and *6 Discussion and Conclusion*. In the taxonomy section I created the taxonomy illustrations and the overview table.

2.3. Publication: Concept Embedding Analysis: A Review

The next content is contained in the following paper:

[Th08] **Gesina Schwalbe**. "Concept Embedding Analysis: A Review". In: *CoRR* abs/2203.13909 (2022). URL: https://arxiv.org/abs/2203.13909. *Submitted to Springer Nature Journal Artificial Ingelligence Review Journal, ISSN 1573-7462.* Find the original full text attached on page 186.

Positioning in the Thesis

This review in detail highlights the XAI sub-field of concept embedding analysis, to which the core methods in this paper belong. The given classification scheme and broad survey of example CA methods clearly positions the developed approaches and datasets within the related work. Also, the paper carves out the challenge tackled in this thesis: utilizing CA for verification in complex tasks like object detection.

My Scientific Contributions

- General definition of the research field CA; demarcation against other sub-fields of XAI;
- Development of a detailed technology-oriented taxonomy for CA methods;
- A broad literature review with classification and comparison of CA methods, applications, related metrics, and image datasets;
- Derivation of open challenges and research direction proposals for CA, including
  - CA for further domains and model architectures (like larger object detectors),
  - CA for quantitative assessment applications.

My Text and Content Contributions

I was the sole contributor of texts and content to this paper.

2.4. Chapter Summary

The basic notions of (visual) semantic concept and concept embedding analysis (CA) that are used throughout this thesis have been introduced. A comprehensive taxonomy for methods from the research area of XAI is provided that allows to position CA-related methods within this larger context of related work. Within the work [Th08], CA is carved out as a broad and interesting sub-field of XAI with the common goal to associate parts of DNN intermediate outputs with semantic concepts. The identified classification criteria and related work allow to tailor method choices specifically to a given use-case. This is used later in Chap. 3 to demarcate the contribution of the methods developed in this thesis.

3. Concept Embedding Analysis for Object Detection

As discussed in Chap. 1 and Chap. 2, CA seems a good and natural candidate to enable verification of symbolic requirements. However, prior to this work methods for CA had not been tested—and were not immediately suitable, as will be shown in the following—for application to larger CNNs such as state-of-the-art object detectors. Hence, one of the main contributions of this thesis is to substantially improve CA for such applications. This chapter details these contributions which are distributed over my application papers [Th03, Th02, Th01, Th06].

The chapter starts with a motivation for the chosen CA baseline for supervised post-hoc concept segmentation, and positions it with respect to related methods in Sec. 3.1. Common definitions and notation regarding performance measures and loss terms are recapitulated in Sec. 3.2. As a basis for evaluation, Sec. 3.3 presents two labeling approaches to enrich datasets with concept labels. These were developed in this thesis to enable realistic evaluation of CA on automotive related datasets. The remainder of this chapter details the concrete contributions of this thesis towards improved CA in the selected application scope from Sec. 1.2. These are split into (1) Sec. 3.4 with basic adaptations to the baseline approach Net2Vec from [24] in order to allow for efficient application to object detection, (2) substantial performance improvements in Sec. 3.5, and, finally, (3) extension of concept segmentation to concept detection in Sec. 3.6.

3.1. Related Work and Preliminaries for Method Choice

This chapter clarifies the rationale behind the choice of baseline method later detailed in Sec. 3.5. For this, the considered constraints are collected in Subsec. 3.1.1. Desired method traits are derived, and a suitable base work identified which is then positioned in the related work in Subsec. 3.1.2.

3.1.1. Use-case Specific Requirements for Method Choice

The goal of this thesis is to establish methods for verification of symbolic requirements formulated for real-world object detection. In the following, requirements regarding concept model traits are derived which are imposed by the considered verification use-cases (cf.

Sec. 1.2). From this, choices of traits for methods in this thesis are derived (cf. Figure 2.1 for an overview).

(1) *Concept type: Object parts and attributes*
Since object detection is instance-based, the prevalent concepts types are object or object part instances, as well as attributes thereof. For verification, both predictive concepts are of interest, as well as concepts causally unrelated to the detection ground truth. Especially interesting predictive concepts are such that give rise to interesting spatial and hierarchical relations of objects, like respective object parts. In real-world settings, examples of causally unrelated concepts are surface properties like textures and material, and shape properties like object size.
Choice here: This thesis thus considers the object part concepts of *human body parts*, and the object attribute concept of *object distance* from the camera (for persons approximated via an estimated body size).

(2) *Post-hoc or inherent: Post-hoc*
As the goal is to verify already trained DNNs, the CA method should be *post-hoc*.

(3) *Output format: Binary concepts*
To easily use concept models as unary predicates in logical formulas, one should be able to interpret the outputs as truth values. Therefore, *binary concept labels* are chosen.

(4) *Supervision: Supervised*
For formal or semi-formal verification purposes, the logical relations to assess must be pre-defined. This already fixes the concepts needed, namely those used to formulate the logical relations. Even though this requirement is relaxed for verification via inspection, this thesis considers *supervised* concept analysis in order to ensure a joint method for both applications.

(5) *Type: Linear*
For safety verification, it is mandatory that the assessment methods do not introduce further complexity potentially leading to faulty audit results. Furthermore, the method should provide means to globally assess properties of the concept embedding, in order to reduce dependency on a sample selection scheme. This means, the concept model type must be global, well preserve the structure of the representation space, and be well interpretable. This makes *linear models* a canonical choice for binary concepts:

- They are considered to be *simple and naturally interpretable* [55, 5].
- Linear models for binary classification define (affine) hyperplanes that separate between positive and negative samples. In case of binary concepts, the normal vectors of linear concept models can serve as *global* CAVs that "points into the latent space direction of the positive concept value".
- In [24] it was found that the association of concepts to such linear CAVs *preserves semantic similarity*: Semantically similar concepts show a high cosine

similarity (cf. Def. 3.1). The linear CAV vector space structure within a CNN latent space thus seems an appropriate approximation of its semantic structure, with concept combinations (e.g., a *green arm*) modeled by vector addition, and basic semantic similarity by cosine similarity.

(6) *Input type: Activation map pixel or region*
The following are standard approaches to localize objects within an input image using latent space representations:

- Learn to globally score the presence of the object in the image, then find input parts with most attribution to a positive score [70]. The quality of the localization, however, is limited by the assumption that there is only one instance of the object in the image.
- In case of activation maps (i.e., an association of latent-space sub-dimensions to spatial positions), one can use the spatial grounding of activation map pixels and learn to score the presence of the object per activation map pixel or region. This results in a—possibly low-resolution—segmentation mask. If necessary, this can then be upscaled [24, 7]. This has the disadvantage of fairly rough localization maps due to the upscaling.
- Again, for activation maps, learn per-pixel or per-region scoring, but instead of static upscaling, additionally learn to predict exact offsets for each candidate position [116]. This, however, introduces quite some further complexity into the predictor model, which is unwanted for assessment methods.

For the approaches in this thesis, the second option is chosen, namely concept models that accept and make a *prediction for each pixel or region of the activation map*. This is simple, applicable to object detection, and efficient regarding parameters (just one parameter per filter).

(7) *Optimization: Allowing back-propagation*
A non-strict desirable for concept models in assessment settings is that they can directly be used for fixing inappropriate representations of a model. One way to ensure this, is that the attached concept models allow for back-propagation. In this case, the concept model performance can serve as regularization in a fine-tuning step for the DNN.
Choice here: In the case of this work, the simple approach from [24] is adopted, which attaches the concept model outputs via a convolution with weights optimized via standard *logistic regression*. Other than the alternative image-level support vector machine (SVM) implementation from [55], this easily yields backpropagatable pixel-wise concept models.

(8) *Additional constraints: Applicable to object detection*
Two important efficiency constraints that are discussed in detail in Sec. 3.4 are:
(i) Applicability to object detector CNNs with large dimensional intermediate out-

put spaces while maintaining acceptably low memory and computational resources consumption;

(ii) Applicability to high-resolution images of at least 400 px × 400 px size (the default size for many pre-trained object detectors[1]);

(iii) Good performance for the selected concept types, especially object parts.

Other than in [24, 7], sparsity of CAVs is investigated, but not enforced, e.g., by pruning. This would directly decrease fidelity of the concept model.

(9) *Comparison metric: Cosine similarity*
Cosine similarity between normal vector CAVs is chosen because it is the canonical candidate for linear concept models.

At the time of writing this thesis, the CA method best fulfilling the mentioned requirements is the post-hoc method Net2Vec [24] for pixel-wise binary concept segmentation using linear concept models. Interesting related approaches are outlined in Subsec. 3.1.2. The details of the approach and the improvements with respect to the baseline are discussed in Sec. 3.4.

3.1.2. Positioning with Respect to Related Work

This section summarizes some insights from my detailed review of CA in [Th08]. To give an impression of the breadth of the field, examples of main categories of related CA approaches are outlined.

An important class of inherently interpretable models integrating concept analysis ideas are concept bottleneck models [58]. The idea here is to intercept a feed-forward DNN by a layer in which each unit is trained to correspond to a semantic concept. In contrast to latent space disentanglement methods like variational auto-encoders, the concept units need not be independent [54]. There are both supervised approaches to achieve this [58, 69] as well as unsupervised ones utilizing clustering [10]. Also, both classification [58, 10] and segmentation [69, 23] methods exist. However, concept bottleneck models inherently change the architecture and training scheme of the target DNN, which contradicts requirement (2) from Subsec. 3.1.1 to do a post-hoc analysis.

One class of post-hoc CA methods is that of unsupervised mining of concepts from latent space representations. A standard approach is to first select concept candidates (e.g., superpixel image patches), then collect latent space outputs for many such candidate samples, and finally cluster these latent space vectors. Each resulting cluster is represents a semantic concept respected within the DNN. The cluster centers can be taken as CAVs. There are different approaches to obtain the concept candidates, like single object images [101], purely superpixel-based [29, 109], and attribution-based [27]. Also, different clustering methods are used, like k-means clustering [29], without or with custom regularization [109], or hierarchical agglomerative clustering [101]. The k-means clustering was generalized in [114]

[1] Compare the implementations of object detectors in the PyTorch modelzoo (https://pytorch.org/vision/stable/models.html).

to general matrix factorization. The benefit of concept mining is that those concepts are found that are considered relevant *by the DNN*. While this might be of interest for manual inspection, this hardly allows to check the quality of the representation with respect to pre-defined concepts, which is requirement (4) from Subsec. 3.1.1.

Supervised methods post-hoc attach a simple concept predictor to the DNN latent (sub-) space of interest. This is then trained on a labeled concept dataset. Here, also, clustering-based approaches may be used to find CAVs for the different concept values, like k-means clustering [34] or spectral clustering [53]. CAVs may also be obtained using locally linear approaches, like local derivatives [113] or attributions [107]. However, requirement (5) demands a linear model providing *globally valid* semantic concept representations.

One such method, and a baseline for many further approaches, is TCAV [55]. The concept model here is a binary classifier, and takes a complete latent space output. The classifier is chosen to be linear to maintain simplicity, and is trained using a SVM. In [55] the applicability of TCAV to fairness analysis was demonstrated, making it a good candidate for safety analysis methods. A disadvantage of the vanilla method is that the large input to the concept model: This leads to a huge amount of parameters and large CAVs. To solve this, successor works train the concept model not directly on the activation maps. Instead, a global average pooling is applied to the activation maps before feeding them to the linear models [31, 115]. Still, the implementation via SVMs is unhandy for pixel-wise or region-wise concept models that shall be translation invariant (cf. requirement (6)). A more practical implementation of the linear model is used in Net2Vec [24] by Fong and Vedaldi. They use an efficient and small 1×1 convolution, thus applying the same linear concept model to each activation map pixel. It is trained using logistic regression[2]. This simple implementation not only smoothly integrates into most deep learning frameworks[3], but also eases later fine-tuning based on the concept models (cf. requirement (7)). Thus, Net2Vec is taken as a basis for the work in this thesis. For this, some shortcomings of the vanilla method had to be fixed, like a sub-optimal choice of loss, and an expensive activation map denoising step. This is discussed in Sec. 3.4.

3.2. Definitions and Notation

This section gives an overview on the definitions and notation regarding performance measures and loss terms that are used in upcoming discussions. Throughout this work, mask vectors are used to represent (Boolean or fuzzy) sets of pixel positions. Before going into detail first of similarity coefficients, and then loss terms, some notation is introduced for expressing operations on mask vectors.

[2] The original implementation does not apply logistic regression but a simplified intersection loss without logarithm. See [Th03] for details.

[3] For example, my work in [Th03] was implemented using the TensorFlow-based Keras library (`https://keras.io`), and the work in [Th01] used PyTorch (`https://pytorch.org`).

Notation 3.1. Let $k, j \leq k, n_i, i_j$ be positive natural numbers, and $[0,1]^{n_1\times\dots\times n_k}$ the space of *k-dimensional masks of size* $\prod_{1\leq j\leq k} n_j$, with entry values in the unit interval $[0,1]$ It is assumed that for 2D masks the index position $(0,0)$ represents the top left corner and $(1,1)$ the bottom right one. Let $M, M' \in [0,1]^{n_1\times\dots\times n_k}$ be two masks, $i = (i_1, \dots, i_k)$ an index vector (a pixel position), $\pi_i\colon \mathbb{R}^{n_1\times\dots\times n_k} \to \mathbb{R}$ the projection onto the ith entry, and δ_C the indicator function for the Boolean condition C.

- *Indexing and predicate notation:* The value of M at the pixel position i is denoted $M_i := \pi_i(M)$. That way, a mask M may be interpreted as lookup table for the output of the pixel-wise (fuzzy) predicate $\mathtt{Is}_M(i) := M_i$.

- *Mask operations:* Operations op for multiplication and comparison operations op $\in \{\leq, \geq, <, >\}$ of masks with masks or with real-valued thresholds t are interpreted pixel-wise and as functions to real valued vector spaces, i.e., $(M \cdot M')_i := M_i \cdot M'_i$, $M \text{ op } M' := (\delta_{M_i \text{ op } M'_i})_i \in \{0,1\}^n$, and $M \text{ op } t := (\delta_{M_i \text{ op } t})_i \in \{0,1\}^n$ The negation, intersection, and union of masks are defined as[4]

  $$\begin{aligned}\neg M &:= \mathbb{1} - M \in [0,1]^n\\ M \cap M' = M \wedge M' &:= M \cdot M' \in [0,1]^n\\ M \cup M' = M \vee M' &:= M + M' - M \cap M' \in [0,1]^n\end{aligned}$$

  For operations of M with a mask $M' \in [0,1]^{n'_1\times\dots\times n'_k}$ of the same dimensionality but different size, the masks are zero-padded on the right up to the least common size. A padding in a higher-dimension formally is the embedding $e(M)_i := M_i$ if $i_1 \leq n_i, \dots, i_k \leq n_k$ and 0 else. Operators are then defined by $M \text{ op } M' := e(M) \text{ op } e(M')$.

- *Mask shifting:* Sometimes, mask content needs to be translated, e.g., within the 2D plane. The shifting operation $- \odot p$ by an index position $p \in [0, n_1) \times \dots \times [0, n_k)$ is defined as

  $$(M \odot p)_{(i_1,\dots,i_k)} := \begin{cases} M_{i-p} & i_1 \geq p_1, \dots, i_k \geq p_k \\ 0 & \text{else} \end{cases} \qquad \text{for} \quad i_j \leq n_j + p_j, j \leq k\ .$$

  Intuitively, this is an embedding of M into the mask space $[0,1]^{(n_1+p_1)\times\dots\times(n_k+p_k)}$ in a right- and bottom-aligned manner, filling remaining positions with zeros.

- *Mask area:* The mask area of M, i.e., the (fuzzy) amount of positive pixels in the mask, is denoted as $|M| := \sum_{i\leq n} M_i$.

[4] These are the fuzzy logical operations of the product or Goguen fuzzy logic [77, Ex. 2.15], applied pixel-wise. For Boolean masks these are identical to standard set operations.

- *True and false positives and negatives:* For the pair of masks M, M' with M interpreted as a prediction and M' as a ground truth mask, the true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) are defined as the real values

$$\begin{aligned} \mathrm{TP}_{M,M'} &:= \left|M \cap M'\right| \ , & \mathrm{TN}_{M,M'} &:= \left|(\neg M) \cap (\neg M')\right| \ , \\ \mathrm{FP}_{M,M'} &:= \left|M \cap (\neg M')\right| \ , & \mathrm{FN}_{M,M'} &:= \left|(\neg M) \cap M'\right| \ . \end{aligned}$$

  If it is clear from the context which masks are meant and which is prediction and which ground truth, the subscript is omitted, e.g., $\mathrm{TP} = \mathrm{TP}_{M,M'}$.

3.2.1. Similarity Coefficients

The cosine similarity is a measure for the similarity of vector directions, and is used to compare CAVs throughout this work. The idea is to measure the angle between two vectors. It is commonly used, e.g., to compare word vectors in word vector spaces [74], and was first used for CAV comparison in [24, 55].

Definition 3.1 (Cosine Similarity)**.** The cosine similarity between two vectors $v, w \in \mathbb{R}^n$, $n \in \mathbb{N}$, calculates as the normalized dot product, taking a value of 1 for parallel vectors, 0 for orthogonal ones, and -1 for v and w being anti-parallel:

$$\mathrm{CosSim}(v, w) := \frac{v \cdot w}{\|v\| \cdot \|w\|} \ .$$

Following the baseline from [24], the performance of semantic segmentation is measured using set intersection over union (sIoU).

Definition 3.2 (Mask Similarity Coefficients)**.** Let $M, M' \in \{0,1\}^n$ be two binary masks of the same size and dimension $n \in \mathbb{N}^k$. The following performance coefficients measure the similarity of the masks:

(1) **Set Intersection Over Union (IoU).** The *set intersection over union (sIoU)*, for 2D masks $M, M' \in \{0,1\}^{n_1 \times n_2}$ called intersection over union or Jaccard index (IoU), is defined as

$$\mathrm{IoU}(M, M') := \frac{|M \cap M'|}{|M \cup M'|} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

(2) **Dice Coefficient (Dice).** The *F1 or Dice coefficient* is defined as

$$\mathrm{Dice}(M, M') := \frac{2\,|M \cap M'|}{|M| + |M'|} = \frac{2 \cdot \mathrm{TP}}{2 \cdot \mathrm{TP} + \mathrm{FP} + \mathrm{FN}} = \frac{2\mathrm{IoU}(M, M')}{1 + \mathrm{IoU}(M, M')} \ .$$

(3) **Kullback-Leibler Divergence (KL).** Assume the masks $M, M' \in [0,1]^n$ are non-binary and normalized such that $|M| = |M'| = 1$. Then the masks can be interpreted as a distribution over pixels. A measure for the divergence of M from a ground truth

M' is the standard *Kullback-Leibler divergence* from statistics (note that this is not symmetric):

$$\mathrm{KL}(M, M') = \sum_{i \leq n} M_i \log\left(\frac{M_i}{M'_i}\right)$$

This measure is especially interesting if both prediction M and ground truth M' are non-binary.

It should be noted that the Dice coefficient and the sIoU are very closely related, and are qualitatively interchangeable, as the following proposition shows.

Proposition 3.1. *Let $M, M', N, N' \in [0,1]^n$ be masks. The sIoU and Dice coefficient are related via*

$$\tfrac{1}{2}\mathit{Dice}(M, M') \leq \mathit{IoU}(M, M') \leq \mathit{Dice}(M, M') \tag{3.1}$$

$$\mathit{Dice}(M, M') < \mathit{Dice}(N, N') \quad \Leftrightarrow \quad \mathit{IoU}(M, M') < \mathit{IoU}(N, N')\ . \tag{3.2}$$

Proof. Regarding point (3.1), the definition directly yields

$$\begin{aligned}\tfrac{1}{2}\mathrm{Dice}(M, M') &= \frac{\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \leq \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \\ &= \mathrm{IoU}(M, M') = \mathrm{Dice}(M, M') \cdot \tfrac{1}{2}(\mathrm{IoU}(M, M') + 1) \overset{\mathrm{IoU} \leq 1}{\leq} \mathrm{Dice}(M, M')\end{aligned}$$

For point (3.2), set $a = \mathrm{TP}_{M,M'} > 0$, $b = (\mathrm{TP}_{M,M'} + \mathrm{FP}_{M,M'} + \mathrm{FN}_{M,M'}) > 0$ and $c = \mathrm{TP}_{N,N'} > 0$, $d = (\mathrm{TP}_{N,N'} + \mathrm{FP}_{N,N'} + \mathrm{FN}_{N,N'}) > 0$. Then

$$\begin{aligned}&\mathrm{IoU}(M, M') = \frac{a}{b} < \frac{c}{d} = \mathrm{IoU}(N, N') \\ &\Leftrightarrow 0 < bc - ad = (bc + ac) - (ad + ac) = c(a + b) - a(c + d) \\ &\Leftrightarrow \mathrm{Dice}(M, M') = \frac{2a}{a + b} < \frac{2c}{c + d} = \mathrm{Dice}(N, N')\ .\end{aligned}$$

□

3.2.2. Loss Terms

The presented similarity coefficients for segmentation masks can also directly serve as optimization criterion. More precisely, the following loss formulations were considered throughout this work.

Definition 3.3 (Loss Terms). Let $M \in [0,1]^{b \times h \times w}$ be a batch of 2D mask predictions of batch size b, height h, and width w, and let $M' \in \{0,1\}^{b \times h \times w}$ be a batch of corresponding binary ground truth masks. Then the following terms may be used to calculate a loss value for the batch M.

(1) **Intersection Over Union Loss (IoU Loss).** This work suggests, and in [Th03] experimentally evaluates, the *smoothed intersection over union loss*. This directly optimizes for good sIoU. Given a smoothing summand δ to avoid division by zero (chosen to be 1 in [Th03]), this is defined as

$$\text{IoU Loss}_\delta(M, M') := 1 - \frac{|M \cap M'| + \delta}{|M \cup M'| + \delta} \,.$$

(2) **Tversky Loss and Dice Loss.** The *Tversky loss* aims to optimize a weighted version of the Dice coefficient from Def. 3.2 (2). The weight factors $\alpha, \beta \in [0, 1]$ for FP and FN allow to counteract the usual imbalance of the amount of positive and negative pixels in ground truth masks.

$$\text{Tversky Loss}_{\alpha,\beta}(M, M') := 1 - 2\frac{\text{TP}}{\text{TP} + \alpha\text{FP} + \beta\text{FN}}$$

If the balancing is omitted setting $\alpha = \beta = \frac{1}{2}$, this becomes $1 - \text{Dice}(M, M')$, also called *Dice loss*. And in case the balancing is omitted by setting $\alpha = \beta = 1$ this gives the (non-smooth) IoU loss from item (1).

(3) **Balanced Binary Cross-entropy (bBCE).** Instead of the image-wise penalties, one can also aggregate pixel-wise loss values, e.g., by using a pixel-wise binary cross-entropy. Balancing can here be achieved by using a *balanced binary cross-entropy* [24] with balancing factor $\beta \in [0, 1]$:

$$\text{bBCE}_\beta(M, M') := -\frac{1}{bhw} \sum_{i \leq bhw} (1 - \beta) M'_i \log M_i + \beta(1 - M'_i) \log(1 - M_i) \quad (3.3)$$

$$= -\frac{1}{bhw} \sum_{i \leq bhw} \left((1 - \beta)\delta_{M'_i} + \beta\delta_{1-M'_i}\right) \cdot \text{BCE}(M_i, M'_i) \quad (3.4)$$

The formulation in line (3.4) based on the standard binary cross-entropy (BCE) is used for implementation because it allows to directly use numerically stable BCE implementations. The following options are considered for β:

(i) *Non-balanced:* Setting $\beta = 0.5$ gives standard unbalanced binary cross-entropy.

(ii) *Global mean TP-rate [24, Th03, Th02]:* In [24] it is suggested to set β to the mean ratio of foreground pixels in an image throughout the training dataset T, i.e., $\beta = \frac{1}{|T|} \sum_{M' \in T} \frac{|M'|}{hw}$.

(iii) *Batch-wise TP-rate [Th01]:* Instead of the global mean of the foreground-to-background ratio, this can also be cheaply and dynamically be calculated batch-wise (bw) as $\beta = \frac{|M'|}{bhw}$. This is evaluated in [Th01].

(4) **Intersection Loss (Intersect Loss).** The baseline Net2Vec method implementation used a custom *balanced intersection loss*[5], which is essentially a simplified version of

[5] See the activation setting at line 61 and the loss definition at line 87 in https://github.com/ruthcfong/net2vec/blame/4d314fe/src/linearprobe_pytorch.py.

bBCE from (3) and IoU loss from (1):

$$\begin{aligned}\mathrm{Intersect\,Loss}_\beta(M, M') &:= -\frac{1}{b}\sum_{i\leq bhw}(1-\beta)M'_iM_i + \beta(1-M'_i)(1-M_i)\\ &= -\frac{1}{b}\left((1-\beta)\left|M\cap M'\right| + \beta\left|(\neg M)\cap(\neg M')\right|\right)\end{aligned}$$

(5) **Focal Loss [64] (fBCE).** The focal loss defined by Lin et al. in [64] adds a penalty reduction to bBCE. This means to reduce the loss of fairly well classified examples and instead concentrate on badly classified samples. Given floats γ, β it is defined as

$$\begin{aligned}\mathrm{fBCE}_{\gamma,\beta}(M, M') := -\frac{1}{bhw}\sum_{i\leq bhw}&(1-M_i)^\gamma(1-\beta)M'_i\log(M_i)\\ &+M_i^\gamma\beta(1-M'_i)\log(1-M_i)\ .\end{aligned}$$

(6) **Mean Squared Error (MSE).** The MSE is a standard pixel-wise loss for regression tasks, i.e., for a non-binary ground truth mask $M' \in [0,1]^{b\times h\times w}$. It is defined as

$$\mathrm{MSE}(M, M') := -\frac{1}{bhw}\sum_{i\leq bhw}(M_i - M'_i)^2\ .$$

(7) **Kullback-Leibler Divergence Loss.** The negative Kullback-Leibler divergence defined in Def. 3.2 (3) can directly be applied as optimization criterion, if the masks are interpreted as a distribution of positive values over pixels. For this, each mask $M_{i,\cdot,\cdot}$ in the batch must be normalized

$$\mathrm{KL\,Loss}(M, M') = \frac{1}{b}\sum_{l\leq b}\mathrm{KL}\left(\frac{M_{l,\cdot,\cdot}}{|M_{l,\cdot,\cdot}|}, \frac{M'_{l,\cdot,\cdot}}{|M'_{l,\cdot,\cdot}|}\right)$$

3.3. Datasets for Concept Analysis

As can be seen from the dataset overview in [Th08, Tab. 3], at the time of writing this thesis there are only few image datasets available that are large scale, feature real-world images, and provide fine-grained concept labels. Also, no simple dataset is available for initial benchmarking that both provides real-world images and concept segmentation annotations. Therefore, the contributions of this thesis contain two labeling strategies. These add concept segmentation labels to popular real-world image datasets related to urban traffic scenarios: For smaller benchmarks, the GTSRB dataset dataset labels are extended (Subsec. 3.3.1), and for more extensive studies the MS COCO dataset (Subsec. 3.3.2). These datasets were used for different benchmarks and demonstration of verification applications throughout my work. See [Th08, Tab. 3] and [Th08, Sec. 6] for a comparison with other datasets.

Table 3.1. Relative bounding box coordinates for exemplary digit concepts within GTSRB dataset samples. Signs are cropped to the sign bounding box annotation and of 48 × 48 pixels size. Given are x-y-coordinates of the digit upper left corner in the crop. Relative digit bounding boxes are all of size 12 × 18 pixels (width × height).

Digit	Sign	X	Y
0	30km/h	16	24
3	30km/h	16	14
5	50km/h	16	15
8	80km/h	16	14
1	100km/h	16	12

3.3.1. GTSRB: Traffic Sign Letters

The German Traffic Sign Recognition Benchmark dataset [93] (GTSRB dataset) is a small-resolution image classification dataset. It features 43 classes of traffic signs, distributed onto more than 50 000 real world image crops each containing one close-up frontal traffic sign. For each sample, annotations of the class and the traffic sign bounding box are provided (allowing to center the sign). The small resolutions of 15 × 15 to 250 × 250 pixels, and the fairly easy task make it well-suited for first and cheap evaluation of methods for real-world images. For example, a CNN with as few as four convolutional and two dense layers can already achieve 98.2 % classification accuracy [Th03].

Traffic signs give rise to a number of interesting concept types: contained symbols including letters [Th03, 61], overall sign shape [61], and sign colors [61]. For this work, in specific the symbols are of interest. These may be located within the image, and thus give rise to detection or segmentation concept labels.

Part of my work in [Th03] was to develop a simple automatic labeling algorithm for different letter types. The labeling utilizes the standardized format of the traffic signs. More precisely, the relative position and size of a letter is fairly static with respect to the bounding box of the sign. Thus, the letter bounding boxes could be statically placed using relative positioning rules that are specific to the traffic sign class. Some example labeling parameters are given in Table 3.1, and some examples are shown in Figure 3.4b.

3.3.2. MS COCO: Human Body Parts

The Microsoft Common Objects in COntext dataset [65] (MS COCO dataset) is a large scale real-world image dataset for object and keypoint detection, and object segmentation. It features more than 1.7 million keypoint labels for over 150 000 person instances in 118 287 training and 5000 validation images. The mean resolution is more than 560 px × 490 px (height × width), with a minimum of 51 px × 72 px, and a maximum of 640 px × 640 px. The images are manually selected from the image upload platform Flickr[6], and they depict various real-world indoor and outdoor scenarios. Annotation labels of the vanilla dataset include for object instances class labels, bounding boxes, instance segmentations; and keypoints for persons.

[6] https://www.flickr.com

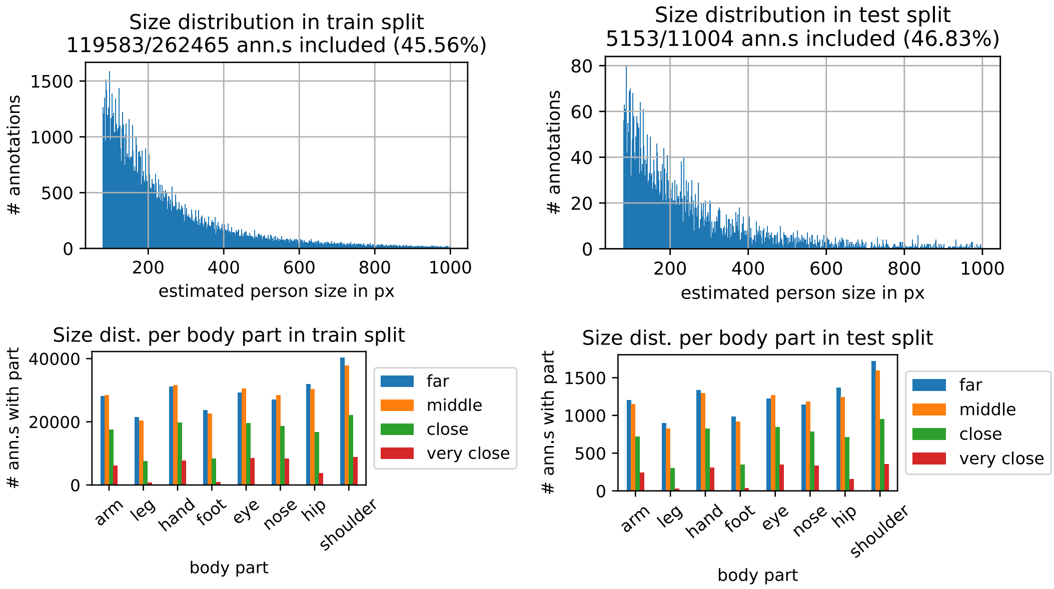


Figure 3.1. Distribution of person body sizes in pixels of annotations in the MS COCO 2017 training (*left*) and validation (*right*) dataset, with sizes estimated from keypoint annotations using the algorithm suggested in [Th01]. *Top:* Distributions of body sizes. *Bottom:* Number of annotations featuring a body part in each of the four size categories defined in [Th01].

My work in [Th01] includes an algorithm to estimate the following ground truth labels from person keypoint annotations: upright body size in pixels, used as an estimate for the distance of a person from the camera; and body part segmentations, using the aforementioned body size.

Body Part Segmentations Given the keypoints of a body part, a segmentation mask is obtained as follows: filled circles are placed at the keypoint positions. In case of more than one keypoint, solid straight lines are used to connect keypoint positions that are adjacent according to the annotations. The line-width is the same as the circle radius and chosen per-body part. To increase accuracy substantially, the circle radius and the line-width is *scaled with the size of the person* in the image (a close-by arm is thicker than a far-away one). Exemplary body parts defined by MS COCO dataset keypoints are: the single keypoints themselves (*wrist*, *ankle*), limbs (*arm*, *leg*), an approximation of *face* (combination of any of *eye*, *nose*), and and an approximation of *head* (combination of any of *face*, *ear*). This simple approach very roughly approximates the area of the body part in the image.

Person Body Sizes The idea of my approach is to estimate the body size of a person in pixels from the spatial relations of its skeletal keypoints. In praxis, two properties of standard keypoint annotations have to be tackled:

- *Incompleteness:* Not all keypoints of a person are labeled for each instance. So, only part of the bone structure can be recovered. Reasons may be that parts of the person protrude beyond the image boundaries, or are occluded and, thus, not labeled.

- *2D projection:* Usually, no 3D spatial information is available for the keypoints, but only the 2D projection of the skeletal information. This can drastically shorten distances between keypoints depending on the pose relative to the camera. For example, for an arm pointing to the camera, all keypoints fall into one position.

My method from [Th01] assumes "standard" skeletal proportions, and a "standard" original body size. Both are derived from various statistics (see [Th01] for details). The proportions describe how the in-image body size (in px) can be derived from one or several skeletal lengths in the image (in px), such as the shinbone length. Following literature, affine linear equations are used here. For details see [Th01, Fig. 2]. To tackle incompleteness of keypoints and skeletal lengths, several independent equations are considered. And to tackle the shortening of lengths due to the 2D projection, the largest of the size estimates for a person is taken. As an application, the approach chosen in [Th01] estimates the distance of a person from the camera via its downscaling factor in the image. The scaling factor is assumed to be proportional to the distance.

Limitations Obviously, the assumption of "standard" skeletal proportions and body sizes cannot generalize to all humans. For example, estimates for children are too tall, yielding comparatively large segmentation areas (e.g., large eyes). Thus, the approximation is inherently unfair, and should not be used for training of models meant for operation in critical applications. Also, the size estimation will only work if at least one indicative skeletal length is available. It will fail to be accurate in images featuring only unconnected keypoints, e.g., a single hand in the image. Similarly, in case that all available skeletal lengths are shortened by the 2D projection, the estimated size will be too short. And lastly, body parts consisting of several keypoints cannot be segmented properly if one or many of the keypoints are missing. In case they are not missing but occluded, the boundary of the occlusion cannot be determined from the keypoints.

Results Manual inspection of the method showed: The sizes are estimated correctly within a variance of $\pm 10\,\%$. A distribution analysis of the estimated person-to-camera distances over the MS COCO dataset (see Figure 3.1) showed a realistic tendency towards many small persons. Also, the distribution of body parts over different distance categories was measured. This well fits the intuition that close-up images of persons mostly contain upper body part concepts, but no lower-part concepts like feet. These findings were consistent between training and validation dataset.

3.4. Approach used for Concept Segmentation on Object Detection

As derived in Subsec. 3.1.1, a method is needed to obtain a (concept) model that predicts spatial information about object part concepts. The input to the concept model should be latent space representations of an image, i.e., intermediate output of a trained DNN. The method should be applicable to object detection CNNs (requirement (8.i)), common human body concepts (requirement (1)), and work for higher resolution concept datasets (requirement (8.ii)) like the MS COCO dataset presented in Subsec. 3.3.2. This section first introduces the main traits of the chosen baseline approach Net2Vec (Subsec. 3.4.1), before summarizing my modifications towards efficiency enabling application to object detection (Subsec. 3.4.2). To my knowledge, the here proposed method is the first CA method to be applied to object detection CNNs.

3.4.1. Baseline Method

The concept model to train shall accept for one input sample the activation map output of a CNN layer, and return a single channel segmentation mask with pixel values in $[0, 1]$. This continuous valued output can then be binarized using a threshold. The general concept model architecture considered here for achieving this, is derived from the Net2Vec method and depicted in Figure 3.2. The core idea is to apply a single convolution of stride 1 to the activation map, and upscale and normalize the so obtained continuous-valued mask.

In the following some details are given on the implementation of the steps from Figure 3.2.

1. *Preprocessing:* The original approach [24, 7] denoises the activation maps before applying the concept model. For this, values below a threshold are replaced by zero. The threshold is chosen per-filter such that only the highest $0.5\,\%$ of the respective filter output pixel values remain non-zero. This is not adopted here, as will be explained in Subsec. 3.4.2 (cf. [Th01]).

2. *Convolution:* The original approach chose a 1×1 kernel size for the convolution. Here also larger kernel sizes are investigated as discussed further in Sec. 3.5.

3. *Upscaling* For upscaling, Net2Vec suggests bilinear interpolation. This is adopted here, as it complements the simple linear character of the convolution.

4. *Normalization* Net2Vec suggests a sigmoid for normalization. In my work [Th03], also swapping normalization and upscaling is considered. The reason is that first sigmoid normalizing, then upscaling leads to slightly higher confidence values than the other way round. Swapping is dropped in my other work [Th01, Th02, Th06] for better comparability and smoothness.

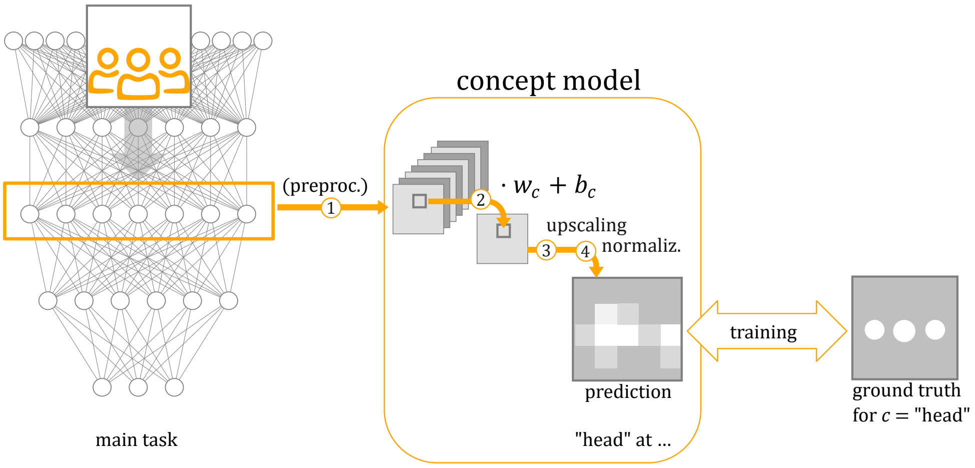


Figure 3.2. Visualization of steps conducted in a concept segmentation model of the Net2Vec type for concept *head*: (1) optional preprocessing of the CNN activation maps, (2) application of a linear convolution with CAV kernel weights w_c and bias b_c, (3) upscaling (here visualized: nearest neighbor), (4) normalization, e.g., using a sigmoid. The concept model is trained on ground truth binary segmentation masks. Graphic adapted from [Th01, Fig. 1].

The following choices apply to the training and evaluation:

5. *Training:* The original hyperparameter choices for the training in Net2Vec were: stochastic gradient descent (SGD) with a standard momentum of value 0.9, a learning rate (lr) of 10^{-4}, a batch size (bs) of 64, 30 epochs of training, and their implementation used an intersection loss (Def. 3.3 (4)). This proved quite inefficient for convergence on object part segmentation. In this work, these settings are replaced by ones found more efficient in comparative studies. In particular, several alternative loss formulations are evaluated including (cf. Def. 3.3): binary cross-entropy as well as global and bw bBCE, IoU loss, Tversky loss respectively Dice loss, and standard MSE. Details of the hyperparameter optimization are discussed in Sec. 3.5.

6. *Performance evaluation:* For evaluation of segmentation performance, sIoU is used (see Def. 3.2 (1)). sIoU is a standard metric for evaluation of binary segmentation masks that is robust to a low proportion of positive pixels. It is adopted from Net2-Vec to ensure comparability with one slight adaption: The final sIoU is averaged over the sIoU of the evaluation batches.

   At evaluation time, the continuous outputs are binarized using a threshold of 0.5. However, it is found that this is not always the optimal threshold, depending on the training objective (cf. Table 3.2, [Th06, Tab. 3]). One reason is the imbalance of positive and negative ground truth pixels, and the different effects of balancing losses.

Table 3.2. Comparison of sIoU results for concept models both at a threshold (thr) of 0.5, and at the optimal binarizing threshold (tuned on the validation set). The concept models were trained using BCE on the MS COCO concept dataset (Subsec. 3.3.2). For each concept (C) the results on the layer (L) with best sIoU at a threshold (thr) of 0.5 are given. The optimal threshold is determined by measuring sIoU values for different thresholds with a sampling rate of 0.1. Layer (L) indices represent the residual block count in the respective CNN. Table expanded from [Th06, Tab. 3b].

(a) Results for Mask R-CNN [38] model

C	L	sIoU @ thr	sIoU @ 0.5
ankle	3	0.210 @ 0.16	0.132
arm	4	0.330 @ 0.26	0.254
eye	3	0.436 @ 0.32	0.424
leg	3	0.326 @ 0.26	0.264
wrist	4	0.184 @ 0.21	0.119

(b) Results for EfficientDet D1 [96] model

C	L	sIoU @ thr	sIoU @ 0.5
ankle	4	0.142 @ 0.16	0.059
arm	6	0.305 @ 0.26	0.242
eye	5	0.340 @ 0.26	0.296
leg	5	0.310 @ 0.26	0.246
wrist	6	0.150 @ 0.16	0.069

3.4.2. Efficiency Improvements

The baseline method Net2Vec exhibits several limitations that impede practical application to object detection. In the following, a summary is given on countermeasures to Net2Vec limitations that are suggested and applied in this thesis.

Discarding of Expensive Denoising

Problem Net2Vec applies a per-filter low cut to the activation maps in a pre-processing step. The threshold is chosen to be a quantile of the training activation map values of each filter. The exact determination of this threshold requires to process all activation maps of one filter at once. This is infeasible for large training datasets, and for large sized activation maps. At least the latter is a blocking point for application to object detector CNNs, since these work on higher resolution images and, thus, have large activation maps in early layers.

Chosen Solution Solution approaches would be to forego CA results on these large dimensional layers, or to approximate the quantile threshold either on a subset of the data or by binning. Instead, I found in [Th01] that one can completely skip the additional and costly denoising step with a negligible change of less than ±0.01 % in the performance on body parts.

Experimental Evaluation (cf. [Th01, Sec. 4.1]) The results were found on the following experimental setting. Compared were an exact reimplementation of the Net2Vec method from [24] described in Subsec. 3.4.1, with and without the denoising step.

- *Concepts:* Evaluated were the body part concepts *hand*, *arm*, *foot*, *leg*, and *eye*.
- *Dataset:* To reproduce the settings in the original Net2Vec method evaluation in [24], also the BROad and DENsely labeled dataset [7] (BRODEN dataset) was used for training and testing. More precisely, as test data served each the training and test subsets of the BRODEN dataset, restricted to samples containing positive segmentation pixels for the respective concept.
- *Analyzed latent spaces:* Results were measured on the network architectures and layers for which experimental results are reported in the original work [24, Fig. 2 and 6]: all five convolutional layers of AlexNet [60], and the last two convolutional layers of VGG16 [91]. Weights trained on the ImageNet dataset [16] were taken from the freely available PyTorch modelzoo[7] for comparability.

The per-concept results for the formulation without denoising are included in the visualization in Figure 3.5. These are consistent with the results reported for Net2Vec by Fong and Vedaldi [24].

Non-global Balancing

The used global balancing of the loss (cf. Def. 3.3 (4)) requires to gather statistics of the ground truth annotations. This adds an additional development step, and adds a hyperparameter that needs tuning. Furthermore, the originally proposed tuning strategy expects the complete training data, or a representative subset, to be at hand.

Chosen Solution The loss formulation requiring a global balancing constant is replaced by a more efficient one. Depending on the setting, a Dice loss [Th02, Th01] or a non-balanced BCE (Def. 3.3 (3)) [Th06] is used. The Dice loss automatically pushes the optimal binarization threshold to 0.5, and hence eliminates the hyperparameter tuning. BCE requires to tune the binarization threshold if the outputs are to be binarized. This, however, can be done after training, and it is here conducted on the smaller validation instead of the training set [Th06].

Experimental Evaluation [Th01, Th03] Several comparative studies were conducted on choices of training settings and hyperparameters. Details are given in Sec. 3.5.

3.5. Performance Improvements to Concept Segmentation

The efficiency improvements summarized in 3.4.2 enable CA on object detection CNNs (cf. requirement (8.i) from Subsec. 3.1.1). However, the baseline Net2Vec method suffers from

[7] https://pytorch.org/vision/stable/models.html

substantial performance limitations, especially regarding object part concepts (cf. requirement (1)). This work introduces considerable performance improvements to the chosen CA method. An overview on these and the experimental evaluations is given in this section. The common goal is to increase effectivity of the training and ultimately the performance of the concept models. Against this background, the following modifications to the standard training procedure are suggested and evaluated:

- CA for activation maps with small receptive field or low resolution:
  - Adaptive kernel sizes ([Th03], Subsec. 3.5.1): Increased concept model kernel size in order to deal with small receptive fields
  - IoU and intersect encoding ([Th03, Th01, Th02], Subsec. 3.5.4): Penalty reduction of loss terms for imprecise boundary areas
- Better concept class balancing:
  - Concept model pre-training ([Th03], Subsec. 3.5.2): Class-balanced pre-training
  - IoU loss ([Th03], Subsec. 3.5.3): Automatic class balancing in the loss
- Hyperparameter optimization for performance ([Th03, Th01], Subsec. 3.5.5)
- Stabilization through ensembling ([Th02], Subsec. 3.5.6)

These aspects are summarized in the below subsections.

3.5.1. Adaptive Kernel Sizes

Problem The convolutional kernel size of the concept models imposes a strong inductive bias. This can be a fundamental issue for performance. The limitation here is the size of the theoretical and the effective receptive field [71] of neurons in a convolutional activation map. For a definition of those notions of receptive fields, consider a CNN, a convolutional layer of the CNN, and an activation map pixel p in that layer, i.e., a collection of neurons corresponding to the same spatial location in the image. The *theoretical receptive field* is the area of input pixels that are connected to p in the directed computational graph of the CNN. The *effective receptive field* is the area of input pixels with a sufficiently high (statistically) expected attribution to p. It can be seen as the “context” from the input that is used to derive the local information at p. The following was found about the sizes of the receptive fields by Luo et al. in [71]:

- *Theoretical receptive field:* For a CNN with windowed operations and square window shapes, the theoretical receptive field is a square window. The kernel size (width) of this window linearly increases with the depth of the layer, and the window sizes of prior windowed operations.

- *Effective receptive field:* Luo et al. investigated the size of the minimum square window that encloses the effective receptive field. They found that the kernel size of this window linearly increases only by the *square root* of the CNN depth, and even less if skip connections are present.

Practically, this means that the context used to produce the output of p is quite small for shallow networks, early layers, or middle layers of residual networks [39] with skip connections. The effect for concept analysis is that single activation map pixels may not hold enough context information to allow segmenting the respective concept accurately. Hence, increasing the effective receptive field of a pixel in the concept model output may be necessary for CA to work.

An example where this is the case is given in my work [Th03]. Here, the third convolutional layer of a small CNN is analyzed, and the GTSRB dataset digit concepts "3" and "5" were not differentiable from any single activation map pixel. The problem is visualized in Figure 3.3. The layer is preceded by two max-pooling layers with kernel size 2×2, and three convolutional layers with kernel size 3×3. This results in a theoretical receptive field of at most $18\,\mathrm{px} \times 18\,\mathrm{px}$ size per activation map pixel, and a considerably lower expected effective receptive field. The kernel size for the concept model was chosen 4×3 to ensure sufficient context for the model.

Suggested Solution As a countermeasure, I suggest and evaluate to increase the convolutional kernel size of the concept model. It is adapted from the very small size of 1×1 to a (not necessarily square) value that ensures a sufficient effective receptive field size. For this, an approximate minimum bounding box size and shape is chosen for the respective concept. The kernel size is then chosen to be the minimal one covering this bounding box size. Covering here means to be larger than the bounding box in width and height when upscaled by the size difference of activation map and original image. This simple rule ensures a sufficient context for the concept prediction.

It must be noted, however, that this concept-specific kernel size comes at several costs, inhibiting general applicability. All samples of a concept must be of approximately uniform size and give rise to a uniform approximate bounding box dimension. In real-world object detection with objects in different camera distances this requires to define separate concepts for different size categories (e.g., close-by *leg*s, middle distance *leg*s, far *leg*s), increasing analysis efforts. Another limitation is that the kernel size directly defines the CAVs dimension, making concepts of different size non-comparable. Hence, a uniform 1×1 kernel size remains preferable where possible.

Experimental Evaluation Experiments on the small traffic sign classifier from [Th03] required adaptive kernel sizes to achieve any meaningful results in the small-CNN-setting. I evaluated in [Th01], whether adaptive kernel sizes are also necessary for middle and later layer block outputs of larger object detector networks. As a setting, five body part concepts from the MS COCO dataset were split into each three size categories, which each gave rise to a concrete adaptive kernel size. Several experiments were conducted on the two standard classifier networks VGG16 and AlexNet. The results showed that training the size-specific concepts with adaptive kernel size does not bring significant performance benefits over

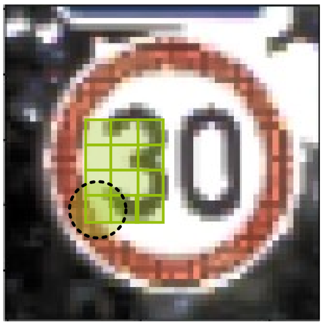


Figure 3.3. Visualization of the size and receptive field of activation map pixels for the third convolutional layer of the DNN from [Th03]. *Green:* Activation map pixel sizes upscaled by the scale difference between activation map and the original image. *Black circle:* Approximate effective receptive field of the lower left activation map pixel. *Green area:* Chosen concept model kernel size of 4 px × 3 px. (Image from GTSRB dataset)

training without adaptive kernel size or without any size restriction (see [Th01, Fig. 7]). Adaptive kernel sizes therefore can be skipped in larger model settings.

3.5.2. Pre-training of Concept Models

Problem Classes in concept classification and segmentation datasets usually are badly balanced. This means that few positive (pixel) examples are available defining a concept, but in relation to that many negative samples (or pixels). While bad balancing of the data can be counteracted in the loss, it is to be preferred to work on a balanced dataset. For classification tasks, a balanced training dataset can be obtained by merging equally sized subsets of a larger dataset, each randomly chosen from the training samples for one class. For segmentation tasks, such simple sample selection does not apply, as each image represents a set of pixel samples: (a) It may not be possible to select a subset of images that give rise to a desired pixel class balance (e.g., when *all* segmentation masks contain less than 50% positive pixels); (b) A sub-selection of images fulfilling a certain pixel-balance may be heavily biased (e.g., in real images only images with many, or close-by objects of interest are selected); and (c) The sub-selection becomes a non-trivial optimization problem.

Suggested Solution To avoid these problems of image selection, I developed a pre-training scheme for the chosen concept segmentation models. The idea is to view the pixel-wise output of the concept model as a classification of activation map crops, and to apply the simple sample selection balancing to those crops. Stride 1 convolutions can be described as a linear operation that is repeatedly applied to all crops in the input that are of the same size as the kernel of the convolution. The kernel weights are the weights of the linear operation, and the outputs for all crops make up the output mask of the convolution. Consider the intermediate mask output of the concept model before the upscaling (cf. Figure 3.2). The above interpretation makes the per-pixel output of this part of the concept model a binary classification of kernel-sized windows in the activation map into "concept" and "no concept". A ground truth for this classification can be obtained by downscaling the concept segmentation ground truth to the resolution of the activation map (see [Th03]).

Hence, the concept model kernel weights can be (pre-)trained on this classification task, picking only a subset of all available activation map crops. One can choose this subset to be class-balanced using the simple selection scheme described before.

Experimental Evaluation For the evaluation in [Th03], the following optimization settings were chosen: Standard SGD optimizer with Nesterov accelerated momentum of value 0.9 [94], a large batch size of 128, a small learning rate of 10^{-6}, standard binary cross-entropy for a loss, and maximum 15 epochs with early stopping at a loss improvement of lower than 0.04. The data samples with and without the concept were equally balanced. This setup achieved a classification accuracy of the pre-trained model of already 91 %. Pre-training significantly increased all used evaluated loss combinations (cf. [Th03, Tab. 1]).

3.5.3. Intersection Over Union Loss

Problem A high imbalance of positive and negative pixels can be a problem for optimization in binary semantic segmentation. One way to tackle this is by introducing balancing factors to the loss, as, e.g., in bBCE and Intersect Loss. These, however, require to fine-tune the balancing factors that determine the contribution of false positives and false negatives to the loss. This requires availability of sufficient and representative fine-tuning data, and the hyperparameter may not generalize [Th06].

Suggested Solution The IoU loss is a new loss formula which was suggested in my early work [Th03]. The loss directly optimizes for a good sIoU score. Other than the pixel-wise losses bBCE and Intersect Loss, and similar to the Dice loss, the IoU loss only respects the results for ground truth positive pixels, but weights them by the total amount of positive and false negative prediction pixels:

$$\text{IoU Loss}(M, M') = \frac{|M \cap M'|}{|M \cup M'|} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}} = \frac{1}{\text{TP} + \text{FP} + \text{FN}} \sum_{i \leq bhw} M_i M'_i \;.$$

This *automatically balances the contribution of false positives and false negatives.*

Experimental Evaluation The IoU loss is compared to three other standard losses for concepts of the GTSRB dataset in [Th03]. This comparative study showed that IoU loss strongly outperforms balanced and unbalanced bBCE and Intersect Loss if no balanced pre-training from the previous subsection is applied. It should be noted that the IoU loss objective (optimize sIoU) is interchangeable with the more common Dice loss (optimize Dice coefficient) according to Prop. 3.1. Thus, later work uses Dice loss [Th02, Th01]. For Dice loss, the comparative study in [Th06] also showed top results compared to unbalanced BCE and bBCE without tuning of a binarization threshold. Even when allowing tuning, Dice loss achieved acceptable results compared with BCE with threshold fine-tuning. Thus, it can be seen as a viable alternative, even though this comes at the cost of worse calibration (cf. [Th06, Tab. 3]).

3.5.4. Penalty Reduction

Problem A problem for optimization of the concept segmentation models is the low resolution of activation maps. Since the final prediction only is obtained via upscaling of low resolution intermediate masks, the predictions at the borders of segmentation areas are naturally imprecise. This leads to a high misleading contribution of these boundary areas to pixel-wise losses like bBCE and Intersect Loss. Pixel-wise losses are such that can be written as

$$L(M, M') = \operatorname*{mean}_{i \leq n} l(M_i, M'_i)$$

for predicted and ground truth masks $M \in [0,1]^n$, $M' \in \{0,1\}^n$ and a pixel-wise loss function $l\colon \mathbb{R} \times \mathbb{R} \to \mathbb{R}$.

Suggested Solution To reduce the contribution of such boundary areas to pixel-wise losses, my work [Th03] suggests an IoU based penalty reduction. The principle is related to the penalty reduction used in the definition of focal loss [64] (see Def. 3.3 (5)) (fBCE). The penalty reduction is specific to the pixel position i and the ground truth segmentation map M', and it is applied via a factor $f(\cdot)_i$ to the loss term for one pixel i:

$$L_{\text{reduced}}(M) \coloneqq \operatorname*{mean}_{i \leq n} f(M')_i \cdot l(M_i, M'_i) \ .$$

The proposed formula for the penalty reduction term relies on the assumption of a prototypical shape of the concept *C* (cf. [Th03, Th02]). For example, given faces of about the same size, the eyes will typically match a common oval shape. This is modeled as a prototype 2D binary mask $M^C \in \{0,1\}^{h^C \times w^C}$. The following two formulations are suggested: At each position $i = (i_1, i_2, i_3)$, either the *intersection* or the *intersection over union* is measured between the ground truth mask $M' \in \{0,1\}^{b \times h \times w}$ and the prototype mask *centered at* i. The according penalty reduction terms are defined as:

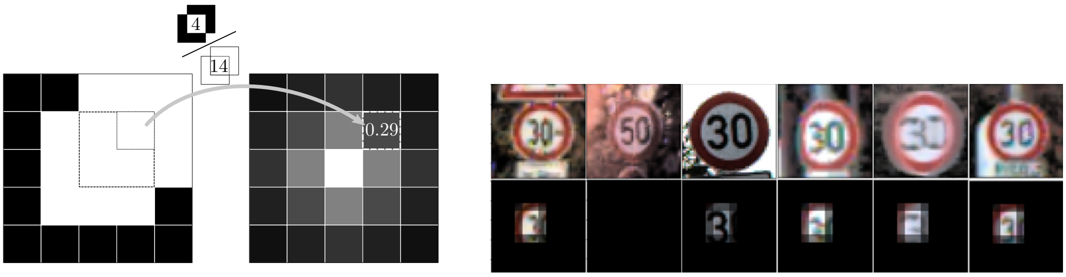
$$f^{\cdot}(M') \coloneqq f^{\cdot}_{+}(M') \cdot M' + (1 - f^{\cdot}_{+}(M')) \cdot (1 - M') \tag{3.5}$$

$$f^{\text{Intersect}}_{+}(M')_i \coloneqq \left| M'_{(i_1,\cdot,\cdot)} \cap (M^C \odot (i_2, i_3)) \right| \tag{3.6}$$

$$f^{\text{IoU}}_{+}(M')_i \coloneqq \text{IoU}\left(M'_{(i_1,\cdot,\cdot)}, (M^C \odot (i_2, i_3)) \right) \ .$$

For implementation, it is used that for bBCE and Intersect Loss this penalty reduction can be rewritten as

$$L_{\text{reduced}}(M, M') = L(M, f^{\cdot}_{+}(M') \cdot M') \ .$$



(a) IoU encoding

(b) *Top:* GTSRB dataset samples; *bottom:* IoU encoding of concept "*3*"

Figure 3.4. Visualization of IoU encoding from Equation 3.5. Figure 3.4a: Schematic for a quadratic ground truth (middle shape) and a same-sized quadratic prototype shape (top right shape). Figure 3.4b: IoU encoding applied to the concept "3" annotations for the GTSRB dataset (cf. Subsec. 3.3.1), adapted from [Th03, Fig. 2].

So, the binary ground truth masks are replaced by the continuous IoU penalty masks, which is called *intersection encoding* [Th02] respectively *IoU encoding* [Th03]. A schematic visualization for IoU encoding is give in Figure 3.4.

Experimental Evaluation IoU encoding is compared to standard ground truth encoding in my work [Th03]. It could not improve the pixel wise losses of bBCE and Intersect Loss in the simple considered setting (cf. [Th03, Tab. 1]). In [Th02] another valuable property of the encoded ground truth masks is utilized: Intersection encoding is used not as penalty reduction, but for restricting the ground truth segmentation masks to segmentation masks of (close-to-)center points. This turned out to be a cheap approach for turning a segmentation task into a fuzzy detection task. In this setting, the intersection encoding masks are binarized for training at concept specific thresholds.

3.5.5. Hyperparameter Optimization

In two experimental studies, the hyperparameter settings for training of the linear concept segmentation models is investigated. Compared were the following hyperparameter options:

Optimizer (cf. [Th01]): While the vanilla method uses standard SGD with momentum [80], this is compared to using Adaptive Moment Estimation [57] (Adam) gradient descent. Adam is another widely used optimization algorithm. Both gradient descent methods update the parameters batch-wise, subtracting an update vector that depends on the gradient with respect to the current batch of samples. While standard momentum uses for an update the current gradient and a fraction of the previous update vector, Adam additionally uses per-parameter learning rates, i.e., factors for the update vector. These factors depend on the squared gradients in previous steps [79]. The adaptive learning rate is well suited for sparse data, such as the activation maps with mostly small values which usually lead to sparse concept vectors [24, Th03]. Furthermore, Adam uses a bias correction that allows fast convergence independent of the update vector init value. These properties make it a good candidate for concept model optimization.

Learning rate (cf. [Th01]): Two larger learning rate (lr) values were evaluated for the better performing Adam optimizer. Besides the originally proposed one of 10^{-4} from Net2Vec, also values of 10^{-3} and 10^{-2} were considered (cf. [Th01]).

Batch size (cf. [Th01]): Given the best optimizer and learning rate, the two further batch size (bs) values of 8 and 128 were evaluated besides the originally proposed one of 64 (cf. [Th01]).

Loss (cf. [Th03, Th01]): The following losses from Def. 3.3 are compared in this work with publications [Th03, Th01, Th06]:

- the originally used intersection loss (see Def. 3.3 (4))
- the originally proposed globally balanced binary cross-entropy (see Def. 3.3 (3))
- batch-wise balanced binary cross-entropy (see Def. 3.3 (3))
- here proposed IoU loss [Th03] (see Def. 3.3 (1))
- Dice loss (see Def. 3.3 (2))
- MSE

Experimental Evaluation Two comparative studies were conducted. Overviews of the results can be found in Figure 3.5 and [Th03, Tab. 1]. The work in [Th03] (Intersect Loss, IoU loss, MSE, bBCE) uses the original learning rate and batch size settings. The later one in [Th01] further investigates the promising candidates (Dice loss as replacement of IoU loss, bBCE, bw bBCE) and uses the optimized lr and bs. The work in [Th06] (Dice loss, bBCE, BCE) extends upon [Th01] and compares not only the performance at 0.5 binarization threshold, but also at optimal binarization threshold and the expected calibration error [35]. All rely on the better Adam optimizer. The main findings were that Adam optimizer, with the slightly higher learning rate of 10^{-3}, and the significantly smaller batch size of 8 for all concepts much outperformed the original settings. Furthermore, Tversky loss, respectively here Dice loss or IoU loss, both outperformed balanced and unbalanced other losses in case no balanced pre-training is applied and the binarization threshold is not fine-tuned. With fine-tuning of the binarization threshold, BCE shows slight advantage over Dice loss (cf. [Th06, Tab. 3]).

3.5.6. Ensembling

Problem As was already found in [55], concept vectors of two training runs for the same concept can significantly deviate. While these differences may have no or little impact on

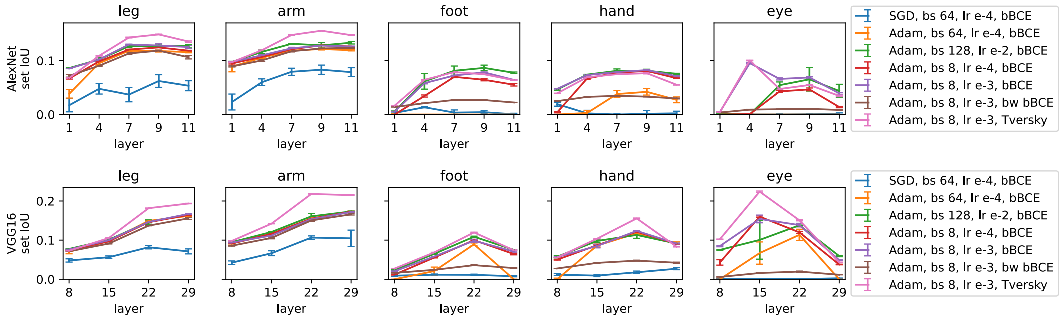


Figure 3.5. Comparison of the performance of different optimization methods for concept segmentation as suggested in [Th01] with the baseline from Net2Vec [24]. Comparison is shown for the two standard ImageNet classifiers AlexNet [60] (*top*) and VGG16 [91] (*bottom*) also analyzed in [24]. Concept models are each trained on a subset of the BRODEN dataset containing the respective body part concept. A reduced version of this visualization can be found in [Th01, Fig. 3].

the performance of the concept model, it makes comparison of CAVs unreliable. TCAV [55] simply discards CAVs with too much variance over training runs, assuming high variance indicates absence of a proper concept embedding. This, however, does not account for CAV variation originating from the optimization approach. Specifically, logistic regression and SVMs are only to a certain degree robust against outliers in the data, as they try to maximize the margin of the decision boundary with respect to *all* available training points. This can lead to different inclination (and thus CAV normal vectors) of the decision hyperplanes for two training subsets, one with and one without some outliers. Thus, instead of discarding CAVs, a "stabilized" concept model is needed that itself gives rise to a CAV. A standard approach for stabilization of approximately Gaussian distributed model results is model ensembling by taking the mean of several model outputs. Since a single "mean" linear concept model with a CAV should be obtained from ensembling, the mean of several hyperplanes must be defined. Before going into detail, some notation is introduced. Also, it is argued why simply taking the mean of the model parameters is not an option, as it gives too much weight to overly confident models.

Notation 3.2. Recall that an affine hyperplane $H \subset \mathbb{R}^n$ is defined by a distance measure of the orthogonal distance of a point to H:

$$\begin{aligned} &H := \{v \in \mathbb{R}^n \mid d_H(v) = 0\} \\ \text{for}\quad &d_H \colon \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R} \\ &\qquad v \mapsto w_H \circ (v - b_H \cdot w_H) = w_H \circ v - b_H \cdot \|w_H\|^2 \end{aligned}$$

where w_H is a normal vector of H, and $b_H \in \mathbb{R}$ is a scalar such that $b_H \cdot w_H \in H$ is the orthogonal support vector. A hyperplane is denoted $H = (w_H, b_H)$ and $s := \|w\|$ denotes the scaling factor of the distance function $d_{(w,b)}$. If not stated differently, all vectors and hyperplanes are in $\mathbb{R}^n$ for some $n \in \mathbb{N}$.

Note that

$$d_{(w,b)} = s \cdot d_{(\frac{1}{s}w, sb)} \qquad \text{and} \qquad (w,b) = \left(\frac{1}{s}w, sb\right) . \tag{3.7}$$

Further, note that for a hyperplane H the normal vector w, and hence b, are unique up to scaling of w. It can further be shown that any two pairs (w,b), (w',b') with $d_{(w,b)} = d_{(w',b')}$ are identical. The distance measure $d_(w,b)$ can be seen as a degree of confidence for the corresponding class, as which it is handled, e.g., for logistic regression. According to (3.7) above, scaling of the normal vector w also scales the distance (and inversely scales the bias b). Meanwhile, the hyperplane (w,b) is independent of the normal vector scaling. Hence, two concept models may provide the same classification or segmentation masks, but have weight vectors (i.e., normal vectors w) of different lengths. Thus, naively taking the mean of concept model parameters would overweight overly confident models, and thus is no option.

Suggested Solution Assume we are given a set of N hyperplanes $(H_i)_i = ((w_i, b_i))_{i=1}^N$, which we would like to reduce to a single hyperplane $H = (w,b)$. The following condition uniquely defines a mean hyperplane:

$$\begin{aligned}\forall v \in \mathbb{R}^n\colon \quad d_{(w,b)}(v) &:= \frac{1}{N}\sum_{i=1}^N d_{(w_i,b_i)}(v) = \frac{1}{N}\sum_{i=1}^N (w_i \circ v - b_i \cdot \|w_i\|^2) \\ &= \left(\frac{1}{N}\sum_{i=1}^N w_i\right) \circ v - \left(\frac{1}{N\|w\|^2}\sum_{i=1}^N (b_i \cdot \|w_i\|^2)\right) \cdot \|w\|^2 .\end{aligned}$$

This is uniquely given by

$$w = \frac{1}{N}\sum_{i=1}^N w_i \qquad \text{and} \qquad b = \frac{1}{N\|w\|^2}\sum_{i=1}^N (b_i \cdot \|w_i\|^2) .$$

Note that the contribution of the normal vectors w_i is weighted by their scaling factor $\|w_i\|$ (i.e., their length), as discussed before. To eliminate this dependence of the mean hyperplane on the scaling factors, I suggest to use the unique normalized normal vectors $\frac{w_i}{\|w_i\|}$ in the above definition. Then b simplifies to the mean of the b_i scaled by w (which is not necessarily normalized):

$$b = \frac{1}{N\|w\|^2}\sum_{i=1}^N b_i .$$

Experimental Evaluation The arithmetic mean definition is used in [Th02] to stabilize the concept vectors. It was found that the performance of the mean hyperplane concept model is at least as good as the mean performance of the concept models, and often even slightly better. In [Th01], the similarity of two concepts was determined as the mean cosine similarity between the normal vectors of several runs for these concepts. This is equivalent to the cosine distance of the arithmetic mean concept vectors of the concepts.

3.6. Concept Detection

Some applications require an allocation of concept instances to center points, i.e., spatial localization of concept instances. Examples are applications using spatial distances on semantic concept instances like [Th02]. The concept analysis approach discussed so far in Sec. 3.4 and Sec. 3.5 considers binary semantic segmentation of concepts, i.e., a classification of pixels into "belongs to concept" and "does not belong to concept". This representation of concept information is not well suited for localization of concept instances: (a) In case of overlap, single instances of concepts cannot be differentiated, as the pixel only contains information about the concept class, not the instance; and (b) the center points of concept instances must be estimated post-hoc from their segmentations which may be imprecise due to noisy boundary pixels.

General Idea This work suggests several approaches to train for center point segmentation instead of semantic segmentation. The general idea is to produce again binary segmentation masks using the same concept model architecture as before. Only the meaning of the per-pixel predictions changes: A pixel prediction now means not anymore "belongs/does not belong to an instance of the concept", but "is center of/is no center of an instance of the concept". While the model architecture stays the same, two aspects need to be reconsidered regarding the new training objective: The encoding of the ground truth as (fuzzy) center point heatmaps, and the loss. Of the here proposed implementation options, one was applied in the paper [Th02]. The idea of using heatmaps for center point prediction is generalized from the work of Zhou et al. in [116], which is explained as one of the available options below.

Suggestions for Ground Truth Encoding The ground truth masks should highlight center points of concept instances. I suggest the following three types of encodings for center points, the applicability of which depends on the available ground truth annotations:

Approximate, based on center point: In some cases, (approximate) center point coordinates are known for concept instances. This allows to generate heatmaps by adding a (fuzzy) peak for each center point onto a black mask. While such center point information is scarcely available [Th08], a common benefit of this approach is that the resolution of the created masks can be chosen independent of the original image resolution.

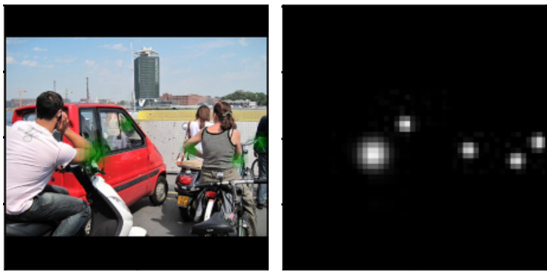


Figure 3.6. Example visualization of the keypoint-based center point heatmaps suggested here for an image from the MS COCO dataset and for the concept *elbow*. *Left:* The original image with keypoint heatmaps overlayed in green. *Right:* The heatmap mask in a resolution specific to one concept model.
For the original thanks to Harry -[The Travel]- Marmot, http://farm3.staticflickr.com/2444/3795350996_5711533f32_z.jpg © CC BY 2.0

- *Based on bounding box (from CenterNet approach [116]):* The simplest case is that there are for each concept instance annotations available that provide a bounding box (i.e., width and height) and a center point. This information can directly be used to define the fuzzy center point masks. For the CenterNet object detector [116], the following encoding was suggested by Zhou et al.: Each center point is used as the mean of a (unnormalized) Gaussian kernel. Overlapping kernels are united into one mask by taking their maximum (which is the Goedel fuzzy logic “or” operation). The standard variation, i.e., radius, of a Gaussian bump scales with the object width and height. An intuition behind the Gaussian bumps is that the exact position of a center point is normally distributed around a given mean. Practically, they simply provide a peak position.
- *Keypoints-based:* For body part concepts sometimes no bounding box information is available. This is for example the case for the MS COCO dataset where center points can only be estimated from keypoints. In such cases where keypoints are available, I suggest to use my size approximation scheme from [Th01] to first estimate the person size. This can then be used to approximate the concept size via a concept specific factor. Lastly, this can serve to produce Gaussian bumps for the center points as in the approach based on the bounding box. This can be done by linearly scaling the standard deviation of the bumps by the approximate concept size. An example output is shown in Figure 3.6. Note, however, that is only applicable for body part concepts where the center point can properly be estimated from keypoints, like *eye*. For example, *arm* and *leg* do not give rise to a unique notion of center point (*elbow* and *knee* may be too coarse).

Segmentation and prototype-based: These approaches assume that no precise center point information is available for concept instances, but instead only a semantic segmentation. This is the case for most available concept datasets with spatial concept information [Th08]. To estimate fuzzy center point information from these binary segmentation masks, a prototypical shape (mask) of the concept must be assumed. As discussed in Subsec. 3.4.2, given such a prototype shape gives rise to two types of continuous valued ground truth encodings:

- *IoU encoding (cf. Equation 3.5.4):* Here, those pixels are highlighted stronger which, if used as center point for the prototype shape, have a high IoU overlap score with the segmentation mask. However, the IoU score drops if the prototype shape and the mask have different amounts of positive pixels. This can be the case if, e.g., more than one concept occurs in the area considered for comparison, or the concept size naturally has high variance. Therefore, this encoding is only applicable if (a) exactly one concept is to be expected per comparison area; and (b) the concept instance segmentation area only little deviates from the prototype area.
- *Intersect encoding (cf. Equation 3.6):* IoU encoding normalizes by the united mask size, i.e., the total amount of positive pixels in the masks. The score used for intersection encoding instead only takes into account the overlap with the prototype. This makes it robust to slight changes in concept instance area size, and also to several concepts per image. However, it will still fail to separate concept instances that exhibit very high overlap.

Suggestions for Losses Given (continuous) ground truth heatmaps, the suggested training objective is to also predict continuous heatmaps. Local peaks in the predicted heatmap are then interpreted as concept center points, as suggested by Zhou et al. for training of the CenterNet architecture [116]. Thus, the loss can be one of the following:

- A regression loss like MSE or KL loss that directly encourages to mimic the pixel values respectively value distribution of the heatmaps;
- A pixel-wise loss accepting continuous ground truth as penalty reduction (cf. Subsec. 3.4.2), like bBCE, Intersect Loss, or a fBCE (this was used in [116]);
- A standard pixel-wise classification loss if the ground truth is binarized before usage.

Experimental Evaluation In this work, only the last loss option is thoroughly experimentally evaluated in [Th02]. There it showed good CA results (cf. [Th02, Fig. 3]) that proved suitable to create logically consistent surrogate models. A qualitative inspection of results using fBCE showed promising results and is a matter for future work.

3.7. Chapter Summary

This chapter (1) selects a respective baseline CA method, (2) provides concept labels for two AD related image datasets, and (3) substantially improves and extends the baseline CA method towards practical use with object detectors. The developed approach allows for the first time to practically apply supervised CA to large object detection CNNs and high resolution datasets, with meaningful results for object part concepts.

CA aims to associate semantic concepts with internal representations of DNNs. From requirements specific to the chosen use-case (cf. Sec. 1.2), a general approach for supervised CA is selected that builds upon the baseline Net2Vec [24]. It trains small, global linear models to predict presence of a concept in an image region from activation map information of a CNN. The baseline is substantially improved both regarding efficiency and performance in extensive comparative studies: Unnecessary memory bottlenecks are removed from the pre-processing; the choice of loss and other training hyperparameters is optimized towards better performance results; methods like pre-training are investigated for coping with high class imbalance; and an ensembling scheme is developed that increases stability while preserving desirable properties. Furthermore, the approach is extended to also apply to large sized concepts by adapting the receptive field of the concept models appropriately. And, finally, adaptations are suggested how to use the method not for segmentation of concepts in an image, but instead for detection of concept center points.

For evaluation of the concept model approaches, new concept labels are provided for two image datasets: the GTSRB dataset (traffic sign digits), and the MS COCO dataset (human body part segmentations). This is based on static labeling respectively existing keypoint annotations, and can be generalized to datasets with similar vanilla ground truth labels.

4. Concept Analysis Applications for Safety Assurance

In the visual domain, many types of symbolic requirements are available that capture domain knowledge. Generally, it is of interest to verify whether and in how far a trained DNN respects such requirements, e.g. for assuring safety or for judging the success of knowledge integration techniques. CA methods here provide the needed access to symbolic concepts within a CNN structure. However, related work on CA only considers simple dependencies between concepts, as discussed in Chap. 2. Concretely, work like TCAV [55] and CHAIN [102] allow to analyze the symbolic requirement "*The DNN output should only be sensitive to task-relevant concepts*".

In the course of this chapter, several further types of symbolic requirements for object detectors are collected and summarized in Sec. 4.1. Against the background of safety assessment, this thesis develops several CA-based applications to verify those requirements, and demonstrates these on use-cases related to pedestrian detection (cf. scope definition in Sec. 1.2). In sections 4.2 to 4.4 the applications are each discussed with respect to the identified requirements, and then presented in detail in the respective publications.

For the used notions of concepts and concept models the reader is referred to Sec. 2.1, and for details on the newly created evaluation datasets to Sec. 3.3. A broader categorization of the presented approaches into a taxonomy of safety evidence generation methods will be given later in Sec. 5.7.

4.1. Preliminaries: Symbolic Requirements for Object Detection

As discussed earlier, the opaqueness of DNNs together with non-symbolic image inputs make advanced assessment methods inevitable whenever background knowledge only is available in a symbolic format. This section collects, structures, and illustrates several types of symbolic requirements in the image domain, in specific for object detection from images. The ones presented here are all tackled using CA-based analysis methods in the remainder of the chapter.

The following notion of task-relevance is the key for some of the requirements. Notably, any task-relevant concept gives rise to input samples in which the concept information has a high positive or negative attribution to the output and is causally related to it.

Definition 4.1 (Task Invariance)**.** Consider a task that is to imitate a mapping f from inputs to ground-truth outputs (an oracle for ground-truth labels). Furthermore, consider a concept C according to Def. 2.1 with a value according to Def. 2.4. The task is said to be *not invariant* to C if there exist input samples where a change to the value of C in the inputs can lead to a change in the ground truth output of f. Concretely, there must exist pairs of samples such that: (1) the samples in a pair differ only by information about C, but (2) the associated ground truth outputs of f differ substantially.

Definition 4.2 (Relevance to a Task)**.** Consider a task regarding a ground truth mapping f, and a concept as in Def. 4.1. C is called *irrelevant or unrelated to the task* if the output of f is uncorrelated to the value of C in the inputs. C is considered *relevant to the task* if f is not invariant with respect to C as of Def. 4.1, and the output is causally related to values of C in the input.

This thesis provides evidence generation methods for the following *requirements* (illustrated with examples):

I. *The encoding of information is correct.*

I.1 *Task relevant concepts are encoded by the DNN.*
If no or little information about a concept is encoded in the network, the CNN must be considered invariant with respect to the concept. In case a function is invariant with respect to task-relevant concepts, it will fail on sample pairs where the ground truth is not invariant to the concept according to Def. 4.2.
Example: The concept *eye*—respectively facial features in general—are task-relevant for pedestrian detection in urban scenarios. Persons may be completely occluded, e.g., by a parking car, with only their face visible. Thus, just adding a face above of parked cars can turn a pedestrian-free situation into a safety-critical one.

I.2 *The encoding chosen by the network respects known semantic relations between concepts.*
Natural language semantics provide an efficient representation of typical real world situations and objects [4]. Semantic similarities in a certain context are based on application-specific considerations, and encode similarity either in the behavior or in the appearance of objects. While a CNN may be invariant to some semantic dimensions, similarity encoding of a CNN should not contradict any of those two similarity types:

- Appearance similarity: If this is broken, the CNN may use task-irrelevant (potentially even adversarial) visual features to characterize objects.
- Behavioral similarity: If this is broken, the CNN may be biased towards certain occurrences of an object.

> *Example:* A backpack is considered part of a person's back instead of a movable object that may move independently.

A weaker, qualitative version of the requirement may be formulated as follows: For concepts A, B, C, if A is more similar to B than to C, then this should also hold for the embeddings.
Example: Limb concepts like *arm*, *leg* share both appearance (limb) and behavioral aspects (may be folded/reach out, thereby de-/increasing the 3D bounding box of a person). They should be more similar than facial features like *eye*, since facial features are spatially more tightly coupled with the center point of a person. Assigning the same similarities between all the concepts indicates that the functional difference (potential sudden extension of the 3D bounding box) is not encoded in the CNN. For example, the CNN might be biased towards dotted patterns on clothes, resembling eyes.

I.3 *The encoding chosen by the network should not be biased, especially, it should be invariant with respect to task-unrelated features.*
This is the counterpart to Requirement (I.1). The CNN may falsely *not* be invariant to a task-unrelated concept in the following cases, each leaving room for constructing counterexamples:

- The concept is strongly correlated with the task output, e.g., shares a confounding factor: The underlying causal relations are not caught by the CNN.
  Example: In urban scenarios, pedestrians often co-occur with street lights, e.g., are waiting there. In case a pedestrian detection CNN adapts street lights as predictive concept, this may lead to hazardous false negatives at crossings without street lights.

- The concept is uncorrelated to the task output: The network may focus on spurious correlations.
  Example: The internal representations of a pedestrian detector, e.g., the concept encoding for *leg*, differentiates between different camera distances of persons. Consequently, different detection performance may be achieved in the different categories, e.g., close-by people are preferred over persons in middle and far range.

II. *The reasoning applied to encoded information is correct.*

II.1 *Abstract rules encoded by the DNN processing are logically correct.*
This plausibility requirement tackles the reasoning that can be found in the network, and can only be evidenced via manual inspection.
Example: Manual inspection of a pedestrian detection CNN reveals that it may be aware of typical features of a person, like body parts, but ignores spatial relations, or the number of entities for prediction. The CNN may, e.g., predict

just one bounding box for two persons, even though their features are not connected in the input image, and persons usually do not have more than two legs.

II.2 *(Fuzzy) logical rules on pre-defined semantic concepts are respected by the DNN behavior.*
This is the counterpart to Requirement (II.1): Given prior knowledge about causal relations and plausible reasoning for the task should be respected by the CNN. Such prior knowledge may origin from physics, or behavioral statistics respectively law. It must be noted that such prior knowledge usually is fuzzy—after all, the real world can be quite astonishing when it comes to providing counterexamples. For example, even though persons usually do not fly, one may on a very rare basis meet protesters hanging from bridges [76].
Example: Examples of physical prior knowledge are that "*persons usually do not fly, i.e., stay close above the ground or a solid object*", and "*eyes usually imply either presence of a person or of an animal*".

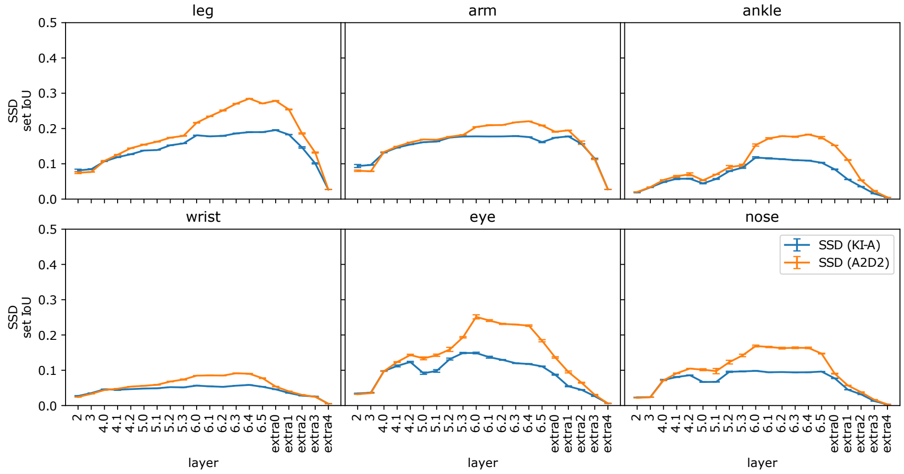
4.2. Verification of Information Encoding

This section presents three joint applications how to verify that the encoded information within a trained DNN is correct (Requirement (I)), using CA. First, an overview on the applications and experimental results is given in subsections 4.2.1 to 4.2.3. These include also some supportive results not available in my publications. After that, the detailed evaluation results are presented in my paper [Th01] (Subsec. 4.2.4).

4.2.1. Verification of Concept Embedding Quality

To be verified: Requirement (I.1)—Task relevant concepts are encoded by the DNN.
Example: eye for pedestrian detection

The goal here is to measure the quality of an embedding in a given latent space. This is solved by using the performance of trained concept models on a test set as quality measure ([Th03, Th01, Th02]). The best possible performance amongst all layer activation spaces then is considered the final embedding strength of a concept, which was proposed in my work in [Th02]. Since different layers encode concepts of different levels of complexity [41], the best performing layer is assumed to be the one with the most suitable representation of the concept within the network. Note that the inductive bias of simple linear concept models prevents overfitting during the search. However, underfitting due to bad convergence is possible when choosing sub-optimal training settings, as was shown in Sec. 3.5. A broadly well performing training scheme is developed in Sec. 3.4.



(a) Comparison of model performance per concept and layer. Curves show ensemble performance, bars indicate best and worst performance (mostly small difference).

	leg	*arm*	*ankle*	*wrist*	*eye*	*nose*
KI-A	0.28	0.22	0.18	0.09	0.25	0.17
A2D2	0.20	0.18	0.12	0.06	0.15	0.10
ratio	1.46	1.23	1.55	1.57	1.69	1.72

(b) Comparison of best ensemble model performance per concept.

Figure 4.1. Concept embedding qualities (cf. Sec. 3.5) for layers of two SSD [68] models with VGG16 backend trained on different datasets (rendered KI-A dataset, and Audi Autonomous Driving Dataset [28] (A2D2) with real images). The performance ratio of the concept models reflects the ca. 1.5 times better mean average precision of the A2D2-trained SSD.
Setup: Body part concept models were trained on ca. 19 000 samples from the MS COCO concept dataset (Sec. 3.3). Per concept and layer, 10 concept models were trained in two independent 5-fold cross-validation runs per concept. SSD performance is reported for the Pascal VOC challenge [21] (IoU threshold of 0.7).

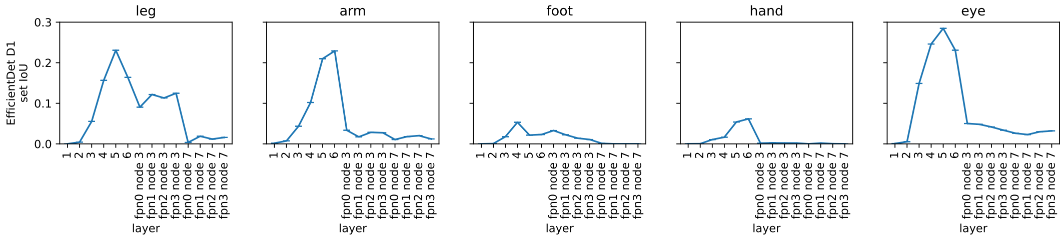


Figure 4.2. CA results for an EfficientDet D1 [96] model and body part concepts from the MS COCO concept dataset [Th01], as used in my work [Th06]. Model weights are taken from TensorFlow modelzoo, the used implementation is `https://github.com/rwightman/efficientdet-pytorch`. Results are with Laplace calibration [72].

Table 4.1. Overview on experiments conducted within this thesis that evaluate concept embedding quality. Compared are the used CNN and concept training and test data, CNN architecture (Arch.), and the number of layers (# L). The used CA methods are those described in Chap. 3. Typical object detector (backend) architectures are marked **bold**.

Concepts	Concept Data	Model Data	Model Arch.	# L	Ref.
digits	GTSRB [Th03]	GTSRB [93]	4-layer CNN [Th03]	1	[Th03, Tab. 1]
face features	Picasso [Th02]	ImageNet [16]	AlexNet [91]	3	[Th02, Fig. 3]
face features	Picasso [Th02]	ImageNet [16]	**VGG16 [91]**	9	[Th02, Fig. 3]
face features	Picasso [Th02]	ImageNet [16]	**ResNetXt50 [108]**	13	[Th02, Fig. 3]
body parts	BRODEN [7]	ImageNet [16]	AlexNet [91]	5	[Th01, Fig. 3], Fig. 3.5
body parts	BRODEN [7]	ImageNet [16]	**VGG16 [91]**	4	Fig. 3.5
body parts	MS COCO [Th01]	ImageNet [16]	AlexNet [60]	5	[Th01, Fig. 6]
body parts	MS COCO [Th01]	ImageNet [16]	**VGG16 [91]**	4	[Th01, Fig. 6]
body parts	MS COCO [Th01]	MS COCO [65]	**DeeplabV3 [11]**	2	Tab. 4.2
body parts	MS COCO [Th01]	MS COCO [65]	**Mask R-CNN [60]**	8	[Th01, Fig. 6], [Th06, Tab. 3], Tab. 3.2
body parts	MS COCO [Th01]	MS COCO [65]	**EfficientDet D1 [96]**	14	[Th06, Tab. 3], Fig. 4.2, Tab. 3.2
body parts	MS COCO [Th01]	A2D2 [28]	**SSD [68]**	20	Fig. 4.1
body parts	MS COCO [Th01]	KI-A [30]	**SSD [68]**	20	Fig. 4.1

Experimental Evaluation An overview over experimental evaluations of concept embedding quality is given in Table 4.1. Specifically, results for the object detectors EfficientDet D1 [96] and SSD [68] are added in Figures 4.2 respectively 4.1, and for the semantic segmentation network DeeplabV3 [11] in Table 4.2. Conducted experiments are diverse with respect to number of convolutions of the analyzed model; type of convolutions and connections (including residual connections); depth of analyzed layers; concept data; training data of the model; and task of the considered model (classification, object detection, instance segmentation, semantic segmentation). Using my suggested CA approach (cf. Chap. 3), some interesting general results that can be derived are:

- *Good embeddings can be found for task-relevant concepts:* Each considered task-relevant concept gave rise to a layer with well performing concept model.
- *Small objects tend to embed earlier than large ones:* The optimal embeddings of smaller object part concepts, like eye or wrist, tend to reside in layers earlier than the optimal ones for larger object part concepts, like arm or leg. So far, literature suggested that concepts encoded in layers of different depth mainly differ by their semantic complexity. Previous work already showed that CNNs perform abstraction steps from layer to layer (cf., e.g., [112]). Also, the prior work in [24] found that more locally distinct concepts like colors and textures are better embedded in earlier layers, whereas shapes like object parts in general are better embedded in later layers. However, my findings suggest that the scale of concept features influences which concepts are encoded in which depth, along with—or even instead of—the concept complexity.

Table 4.2. Comparison of the mean CA sIoU results for a late and an early layer of a DeeplabV3 [11] semantic segmentation CNN.
Setup: The CNN was trained on a generated dataset of urban street scenes from KI-A, the concept models on ca. 19 000 samples from the MS COCO concept dataset (Sec. 3.3).

Layer	*eye*	*leg*	*nose*	*shoulder*
3	0.35	0.42	0.28	0.35
1.1	0.10	0.12	0.08	0.07

- *Task-relevant concept embedding quality correlates with model performance:* In Figure 4.1 the concept analysis results are shown for two CNNs of the same architecture, but with weights of considerably different performance. On average, the embedding quality of the task-relevant body part concepts well approximated the performance ratio. This suggests that *human semantics within CNNs is not only helpful for verification, but is also beneficial for performance.* In other words, natural language gives rise to an efficient representation of real-world environments for standard perception tasks, as postulated in [4].

4.2.2. Verification of Concept Embedding Similarity

To be verified: Requirement (I.2)—The encoding chosen by the network respects known semantic relations between concepts.
Example: arm is more similar to leg than to eye.

This utilizes the vector space structure of latent spaces. Fong and Vedaldi found in [24] that latent spaces of CV CNN models behave like word vector spaces. This means that the standard cosine similarity measure between vectors encodes a form of semantic similarity. Therefore, it is a desirable property for DNN latent spaces, that the cosine similarity between CAVs corresponds to a semantic similarity measure between the concepts they encode.

Experimental Evaluation An exemplary qualitative analysis of an object detector is done in [Th01]. Figure 4.3 shows cosine similarities for the concepts *eye*, *arm*, *hand*, *leg*, and *foot*. According to the results, the considered networks are logically consistent with the following assumptions:

(1) Body parts may be independent, but are no semantic opposites (e.g., *leg* is not the strict antonym of *arm*). Hence, their cosine similarity score is non-negative.
(2) The limb concepts (*arm*, *leg*, *hand*, *foot*) are more similar to each other than to the facial concept *eye*.
(3) A sub-part relation of concepts implies high similarity: Of the investigated body parts, *arm* is most similar to *hand*, and *leg* is most similar to *foot*.

It was also found that low average cosine similarity between CAVs, i.e., more discriminative concepts, correlates with better concept embedding qualities in that layer. Concepts were

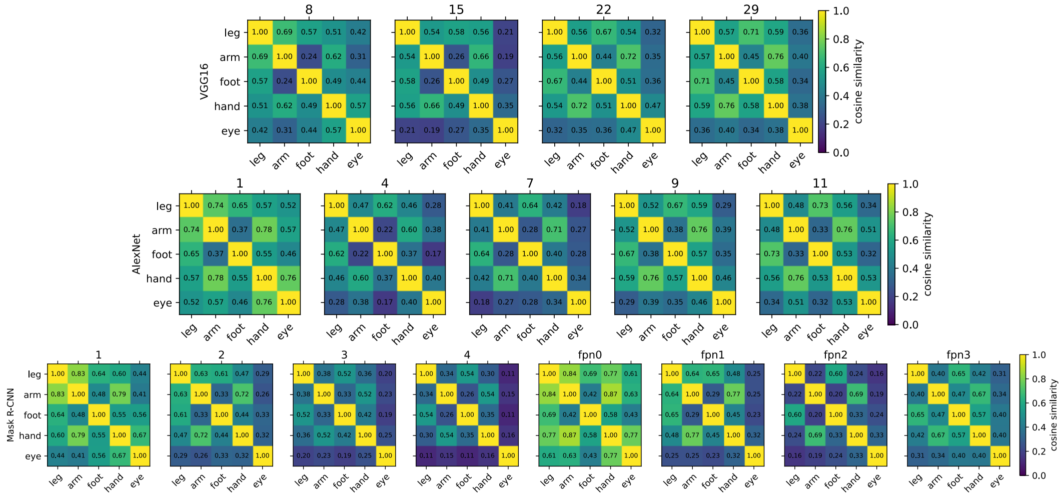


Figure 4.3. Cosine similarities between CAVs for *eye*, *arm*, *hand* (approximated via *wrist*), *leg*, and *foot* (approximated via *ankle*). Results are measured on the MS COCO dataset (see Subsec. 3.3.2), and given by model (*rows*) and by model layer (*columns*). Evaluated CNNs are the ImageNet trained classifiers AlexNet [60] and VGG16 [91], and and the MS COCO dataset trained state-of-the-art object detector Mask R-CNN [38]. (Figure adapted from [Th01, Fig. 4], AlexNet amended.)

less discriminative for the higher-resolution layers like the early convolutional and early Mask R-CNN feature pyramid network (fpn) layers.

4.2.3. Verification of Concept Bias and Invariances

To be verified: Requirement (I.3)—The encoding chosen by the network should not be biased, especially, it should be invariant with respect to task-unrelated features.
Example: A leg concept embedding similar for close persons and for far away ones.

As introduced in Chap. 2, CA grants access to additional concept outputs that directly give insights into latent space encodings. This allows to detect biases efficiently already in the internal representation before the final output. An important class of biases is a lack of invariance with respect to a task-unrelated semantic concept. For example, the distance of a person should not significantly influence detection accuracy of an object detector, in a desired distance range. Assume one wants to assess invariance of a semantic concept A with respect to another concept B. There are several options to do this:

- One can show that the concept embedding quality of B is low, i.e., that information on B cannot be properly localized in the CNN using the chosen CA approach (cf. Requirement (I.1)). This gives some confidence that A cannot depend on B. Note, however, that this depends on a stable convergence algorithm to guarantee that bad embedding strength truly is caused by absence of an encoding of B.

- In case a concept embedding of B is known, i.e., information about B can be localized in the CNN, one may be able to apply one of the following methods, assuming stable convergence of both the concept models of A and B:
  - If A and B have concept embeddings in the same intermediate output space: Check orthogonality of their CAVs, as detailed in Subsec. 4.2.2.
  - If A is a final output of the CNN: Assess the TCAV concept contribution of B towards A as originally suggested by Kim et al. in [55].
  - If a concept model for A can be attached to the CNN in differentiable way: Treat this as a final output and check the TCAV metric as before. This could be done using the CA method from Chap. 3.
- In [Th01] I consider the case that a concept model is available for A, but one wants or needs to be independent of a correct choice of concept model for B. This might be the case if B is not binary, but would require a custom concept model with several outputs, or if the influence of unstable or insufficient concept model convergence should be minimized. In [Th01], A was a body part, and B the body size consisting of three size categories (cf. Subsec. 3.3.2). For this case, the following approaches are suggested:
  - Measure performance discrepancies of the concept model for A for different test sub-sets, each restricted to one class of B (cf. [Th01, Fig. 6]). This answers, whether the output for A is biased towards one class of B.
  - Train concept models for A, each on a different training sub-set restricted to one class of B. To answer, whether one one class of B provides more information on A, these restricted models for A can be compared
    - with respect to their performance (cf. [Th01, Fig. 7] for an example); or
    - with respect to discrepancies of the models: Directly compare the different concept models by comparing their CAVs, e.g., using cosine similarity (see Figs. 4.5 and 4.4).

Experimental Evaluation In [Th01], the invariance of body part information (A) with respect to person size or person distance information (B) is investigated. Labels for the concepts were taken from the MS COCO dataset (see Subsec. 3.3.2). Five different body part concepts were considered, and the person size consisted of three different person distance categories.

Three of the described methods for invariance analysis were applied to each body part:

- The performance of the body part concept model was compared for test datasets that were each restricted to a person size (cf. [Th01, Fig. 6]). The results suggest that the encoding of the considered body part features in the CNN is independent of the person distance category.

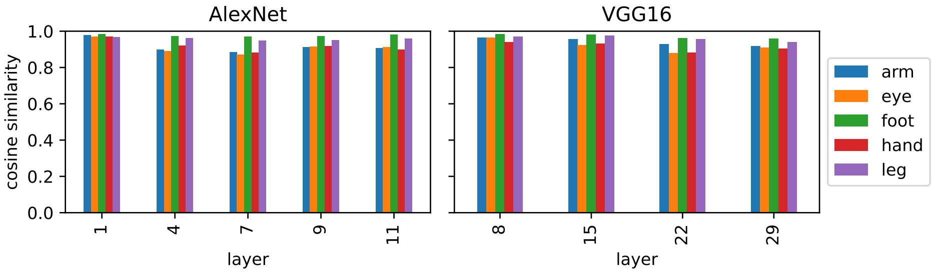


Figure 4.4. Cosine similarity between the CAV trained on the full MS COCO concept dataset (Subsec. 3.3.2), and the best approximation of it via a linear combination of restricted CAVs. Restricted CAVs are trained on subsets restricted to annotations in one person size category. Cosine similarities are shown for the two ImageNet trained classifiers, different layers and body part concepts.

- Body part models were trained for each person size category. The cosine similarities between CAVs of those restricted models are given in Figure 4.5. It shows that the cosine similarities between restricted CAVs of far and close distance categories are consistently the lowest ones, but still have a high value.
- Lastly, the loss of information is investigated between a concept model trained on the full dataset and models only trained on the subsets. Interestingly, the CAV of the full-data-model could not be represented exactly as a linear combination of the restricted CAVs[1], as shown in Figure 4.4. Two reasons might be that (1) the three size-specific data subsets do not cover the complete training data, and (2) that the training on more diverse concept data allows the concept model to capture more information.

[1] The standard least squares algorithm from the `numpy` library was used to obtain the linear parameters for the best approximation, https://numpy.org/

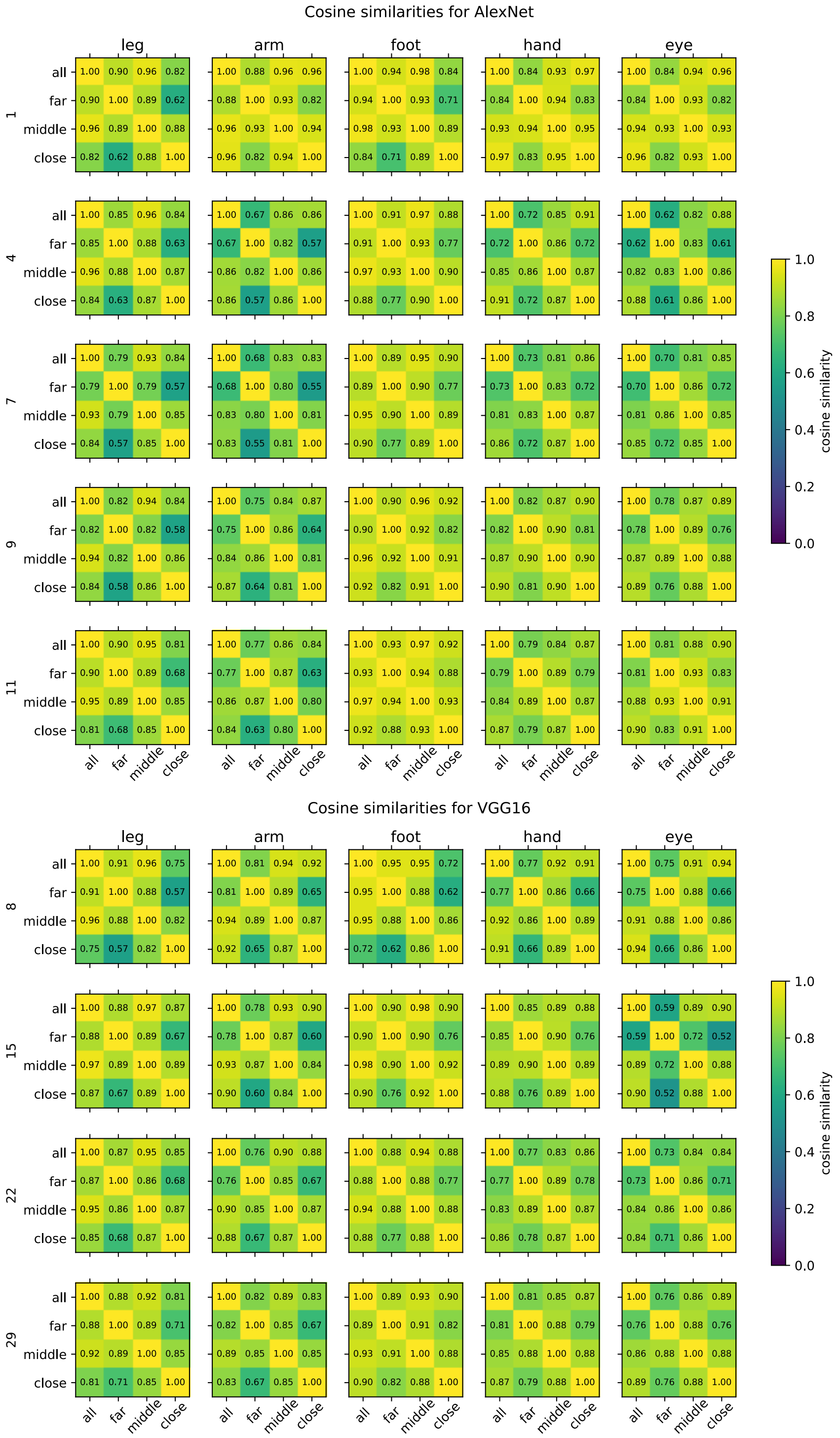


Figure 4.5. Cosine similarities between CAVs for the same body part (*columns*) and layer (*rows*) that were trained on different subsets of the MS COCO dataset. Each subset was restricted to annotations matching one person size category.

4.2.4. Publication: Verification of Size Invariance in DNN Activations Using Concept Embeddings

The next content is contained in the following paper:

[Th01] **Gesina Schwalbe**. “Verification of Size Invariance in DNN Activations Using Concept Embeddings”. In: *AIAI 2021: Artificial Intelligence Applications and Innovations*. IFIP Advances in Information and Communication Technology. Springer, 2021. DOI: 10.1007/978-3-030-79150-6_30. Reproduced in this work with permission from Springer Nature
Find the original full text attached on page 233.

Positioning in the Thesis

This paper contributes to this thesis in two main ways. First, it optimizes CA for object part concepts, the use with large networks, and higher resolution datasets as needed for object detectors in AD (cf. Sec. 3.5). And secondly, it demonstrates how CA can be used to verify three types of safety requirements for DNN-based perception discussed in the previous subsections: concept embedding quality from Subsec. 4.2.1, concept similarity constraints from Subsec. 4.2.2, and absence of concept bias from Subsec. 4.2.3.

My Scientific Contributions

- Formulation of three safety related requirements for DNN internal representations in the case of object detection: quality of concept embeddings, quality of semantics of embeddings, and and invariance to person body size;
- Substantial performance and stability improvements to the baseline CA approach Net2-Vec from [24], by optimized choices of training and pre-processing settings found in an evaluation study;
- First demonstration of CA on a state-of-the-art object detector with high resolution input images;
- Development and application of an automated labeling algorithm that derives approximate person body size labels and person body part segmentations from incomplete 2D skeletal annotations; this was applied to the MS COCO dataset for size distribution estimation and to obtain a new high resolution AD related concept dataset for supervised CA.

My Text and Content Contributions

I authored 100 % of the texts and visualizations. I was the only contributor to the content including review of related work, and conception, implementation, conduction and interpretation of experiments.

4.3. Verification of Rules via Interpretable Proxy Models

To be verified: Requirement (II.1)—Abstract rules encoded by the DNN processing are logically correct.
Example: Inspection may reveal that typical spatial relations are not captured by the network.

Manual inspection of an algorithm is an important tool to reveal potential logical problems, as will be detailed in Chap. 5. Thus, it is desirable to enable inspection of the internal reasoning of a DNN. To do this, [Th02] builds an expressive interpretable proxy model: Via CA, the latent space outputs are transformed into a pre-selected set of symbolic outputs; these are then used as inputs to an interpretable proxy model; and this is finally trained to predict the original DNN output. By now, proxy models based on decision trees (e.g., [14, 53]) or generalized linear models ([53, 12]) can be found in literature. In contrast, the work in this thesis uses highly expressive, first-order logic rules for a proxy. These are trained using ILP.

The demonstrated example relies on standard logical operations, and spatial predicates, i.e., predicates accepting spatial positions. The approach consists of four steps:

(1) *Select and define the logical predicates:* Choose the unary predicates that should be determined by concept outputs, e.g., $\texttt{Is}_{eye}$; and choose any further (spatial) relations that should be applicable to an image or image regions, e.g., spatial ones like `IsAbove`, `Contains`.

(2) *Add concept outputs, i.e., train the concept predicates:* Given the trained CNN, pre-defined concepts from the previous step, and concept segmentation data, apply CA to obtain concept models. More precisely, the CA approach proposed here uses:
   - *Stable concept models:* Concept models are stabilized using a simple ensembling approach (see Sec. 3.5). The ensemble again can be represented as a linear model.
   - *Optimal layer:* CA is applied to a set of candidate CNN layers. For each concept, the best performing concept model amongst these layers is finally chosen.
   - *Concept center prediction:* To better locate concept center points, the ground truth segmentation masks are shrunk to only highlight concept center points (see Sec. 3.6).

(3) *Select proxy model training samples:* To ensure high fidelity with the original CNN, inputs should outline the decision boundary of the original model. Thus, samples are selected that are close to the decision boundary of the original model.

(4) *Proxy model training:* In [Th02] the ILP framework Aleph [92] is used to obtain a set of first-order logic rules that approximate the CNN behavior with high fidelity.

Experimental Evaluation In [Th02] we showed that with this approach one can produce sets of rules that are inherently interpretable [Th07] and of high fidelity (cf. [Th02, Table 2]). This provides a good basis for human auditing of the CNN function.

4.3.1. Publication: Expressive Explanations of DNNs by Combining Concept Analysis with ILP

The next content is contained in the following paper:

[Th02] Johannes Rabold, **Gesina Schwalbe**, and Ute Schmid. "Expressive Explanations of DNNs by Combining Concept Analysis with ILP". in: *KI 2020: Advances in Artificial Intelligence*. Lecture Notes in Computer Science. Springer International Publishing, 2020, pp. 148–162. doi: `10.1007/978-3-030-58285-2_11`. Reproduced in this work with permission from Springer Nature
Find the original full text attached on page 246.

Positioning in the Thesis

In this paper it is evaluated whether CA can be used to modularize a trained CNN by providing interpretable inputs to a symbolic proxy model (here: trained using ILP). The concept analysis models are trained to mask the center points of their concept in order to ease the final localization. This adopts the concept segmentation task towards concept detection (cf. Sec. 3.6). Furthermore, ensembling for concept model stabilization is formulated and applied here (cf. Subsec. 3.5.6). The fidelity of such obtained proxies is evaluated on a face classification task with a generated dataset.

My Scientific Contributions

- Demonstration of using CA concept model outputs as inputs to a global symbolic proxy model with high fidelity results;
- Development and successful application of
  - a label preprocessing scheme for CA for improved concept instance localization;
  - an ensembling approach for concept models that preserves embedding properties.

My Text and Content Contributions

I contributed both content, results, texts and visualizations on concept analysis (sections *2.1*, *3.1*, *4.2*), as well as the used DNNs (section *4.1*). The ideas in paragraphs one to four pointed out in the conclusion section were proposed and authored by me. Experiments on CA, and the preparation of the CNNs were drafted, implemented, conducted, and evaluated by me.

4.4. Verification of Fuzzy Logic Rules

To be verified: Requirement (II.2)—(Fuzzy) logical rules on pre-defined semantic concepts are respected by the DNN behavior.
Example: The rule "eyes usually imply a pedestrian or an animal" should be respected.

A lot of complex prior knowledge about real world domains can be represented in the form of fuzzy logical rules [Th06]. Examples for the CV domain are types of occlusion robustness, such as "Bodyparts usually imply a pedestrian". It may be desirable, e.g., in the case of safety relevant applications, to verify that perception components comply with such rules and to measure in how far this is the case. The theory of t-norm fuzzy logics [77] provides the mathematical tools to formally represent such knowledge as fuzzy first-order logic rules. A simplified example for the occlusion robustness rule "An *eye* usually implies a *person*", would be

$$\begin{aligned} &\forall i \in I\colon F(i) \\ \text{with} \quad &F\colon I \to [0,1], \quad F(i) \coloneqq \forall \mathrm{p} \in \mathrm{P}\colon \mathtt{Is}_{\mathit{eye}}(\mathrm{p}, i) \to \mathtt{Is}_{\mathit{person}}(\mathrm{p}, i) \in [0,1] \end{aligned} \tag{4.1}$$

("In any image, a pixel belonging to an *eye* also belongs to a *person*") for a set I of images, P of pixel positions, and the predicates $\mathtt{Is}_{\mathit{eye}}, \mathtt{Is}_{\mathit{person}}\colon \mathrm{P} \times I \to [0,1]$ scoring the presence of concepts *eye* respectively *person* for a given pixel. Binary classification outputs of DNNs can serve to define such (fuzzy) predicates operating on the DNN input, as will be detailed in Subsec. 4.4.1.

Idea The idea in this thesis is to formulate the prior knowledge in the form of truth functions F as in (4.1) using the DNN-based predicates. This means F are well-formed formulas using at least one DNN-based predicate and have free variables for single instances or sets of DNN inputs. When grounded to an input sample i (or set of samples), the resulting truth value $F(i)$ measures the compliance of the DNN outputs with the rule, respectively serves as a *logical consistency score*. Importantly, once the predicates and F are defined, such logical consistency can be measured in a self-supervised manner, i.e., without further need for a ground truth information about whether the DNN outputs were correct.

Problem Prior to the work in this thesis, rules formulated on DNN tasks could only utilize final outputs for which the DNN was directly trained [13, 6, 44, 75]. This requires complex architectures and huge labeling effort. Also, adding or changing rules after the training was not possible without complete retraining in case new predicates, i.e., DNN outputs, were needed.

Approach This thesis proposes and demonstrates how to instead apply CA to a trained DNN in order to enrich the output accordingly after training. An attached concept mask output allows to define a predicate indicating per-pixel presence or absence of that concept. The post-hoc supervised CA from Chap. 3 requires few additional labels and training effort [54] compared to an architectural approach, allows to flexibly handle new needs for concept predicates post-training, and yields cheap-to-evaluate concept models. Altogether, the proposed approach allows to formulate complex, potentially fuzzy, symbolic requirements, and assess them by observing logical consistency over a dataset or by image. This can be done efficiently with little computational overhead, and without modifying the original network. My work [Th06] suggests a framework to obtain the logical consistency score, and evaluates several applications thereof with a focus on the research question: *Does logical consistency indicate erroneous behavior with a practically useful performance?*. Potential use-cases of logical consistency scores are:

- *Global logical consistency score:* Given a test set T of images, one can rate the logical consistency with F over T via the logical consistency score of the sentence $\forall i \in T\colon F(i)$. A low score means that the CNN (often) behaves inconsistently with background knowledge. The global score can serve for comparing networks, or even steer early stopping during training of a CNN.

- *Corner case analysis:* Another application is the selection of images for manual corner case analysis. Here, those images with suspiciously low truth-value of F are selected from a candidate set [Th06]. This can uncover error modes, and help in defining data augmentation strategies.

- *Online monitoring:* The small overhead for logical consistency evaluation allows for cheap online monitoring of logical consistency over single images or batches thereof, which is an important part of safety assurance (cf. Figure 1.1). Rules providing per-pixel or per-region consistency scores (e.g., $\texttt{Is}_{\mathit{eye}}(p) \rightarrow \texttt{Is}_{\mathit{person}}(p)$) even allow the monitor to locate sources of logical inconsistency in an image. The monitor will raise an alarm or increase uncertainty if the truth value of a formula becomes low for an image. Once logical inconsistencies are located, these may trigger alternate computational paths, or may decrease confidence in the correctness of the prediction. Such confidence information can later in the system be used by, e.g., a voter that can decide to discard or replace outputs [73].

Experimental Evaluation A detailed evaluation of the suggested use-cases was conducted in my work [Th06]. This was done on two state-of-the-art object detectors (Mask R-CNN [38] and EfficientDet D1 [96]), for two occlusion robustness rules, including "A human body part should belong to a person", and on the MS COCO dataset val2017 split. Several options for fuzzy modeling of such rules related to object detection were developed, discussed and experimentally compared. A main finding was that for both CNNs, a substantial amount of detection errors could be uncovered. Also, benefits of fuzziness and concept model calibration could be seen. More detailed results are discussed in the paper [Th06] and its supplemental material.

4.4.1. Definition and Notation for Fuzzy Predicates from DNN Outputs

This section details the idea and notation on how DNN outputs are interpreted as unary fuzzy logical predicates in this work. More generally, this applies to binary classification outputs of any function $d\colon I \to [0,1]^{\mathrm{P}_1\times\cdots\times\mathrm{P}_k}$ for a space of non-symbolic inputs I and indexing sets $\mathrm{P}_j, j \leq k \in \mathbb{N}$. We here assume that d is a DNN, accepts as inputs images from an image space I, and—potentially after post-processing—outputs a family of fuzzy masks $(M_j)_j$, with each $M_j \in [0,1]^{\mathrm{P}_j}$ masking a concept C_j (cf. Def. 2.1), and P_j being pixel positions. The pixel positions of the masks are assumed to correspond to positions or regions within the image, and pixel values represent a score for presence of C at that position respectively in that region. A predicate for one of the concepts is defined as follows.

Definition 4.3 (Concept Predicate)**.** In the following, consider d as above and a single concept output $c = d \circ \pi_j\colon I \to [0,1]^{\mathrm{P}}$ for the jth concept $C = C_j$, $\mathrm{P} = \mathrm{P}_j$. The predicate $\mathtt{Is}_C$ associated to c is defined via the natural bijection

$$\begin{pmatrix} c\colon I \to [0,1]^{\mathrm{P}} \\ i \mapsto c(i) \end{pmatrix} \longmapsto \begin{pmatrix} \mathtt{Is}_C\colon \mathrm{P} \times I \to [0,1] \\ (\mathrm{p}, i) \mapsto c(i)_{\mathrm{p}} \end{pmatrix} .$$

$\mathtt{Is}_C$ will give a score for the presence of concept C in an image $i \in I$ at a general pixel position $\mathrm{p} \in \mathrm{P}$.

Notation 4.1. For simplicity of notation, this work summarizes the space of pixels in an image $i \in I$ to $P = \mathrm{P} \times \{i\}$ with elements $p = (\mathrm{p}, i)$.

4.4.2. Publication: Enabling Verification of Deep Neural Networks in Perception Tasks Using Fuzzy Logic and Concept Embeddings

The next content is contained in the following paper:

[Th06] **Gesina Schwalbe**, Christian Wirth, and Ute Schmid. “Enabling Verification of Deep Neural Networks in Perception Tasks Using Fuzzy Logic and Concept Embeddings”. In: *CoRR* abs/2201.00572 (2022). URL: `http://arxiv.org/abs/2201.00572`. *Submitted to 2022 European Conference on Computer Vision (ECCV).* Find the original full text attached on page 261.

Positioning in the Thesis

This paper introduces a CA-based framework to verify or monitor whether a CNN complies with pre-defined fuzzy logic rules that may be of high complexity. Three applications are presented and demonstrated: a global score for compliance with a rule on a test set (*logical consistency*), corner case selection, and corner case detection. Symbolic concepts may be used in the rules as unary predicates, given that the concepts occur in inputs, outputs, or—for the first time—the internal representations of a CNN. Access to embedded concepts is provided via post-hoc supervised CA.

My Scientific Contributions

- Development of a novel, simple, extensible and scalable framework to construct a truth value monitor for fuzzy logic rules on task-related semantic concepts;
- Introduction and experimental evaluation of three methods for generation of safety evidences for pedestrian detection; evaluation on two state-of-the-art object detectors and two safety relevant rules showed that the framework can
  - uncover a substantial proportion of CNN detection errors during runtime,
  - support in comparing logical consistency of models, and
  - uncover error modes of a CNN by sample selection for manual inspection.
- Evaluation study comparing different modeling options showing, amongst others, benefits of fuzziness and calibration of concept outputs; investigated were options for:
  - concept analysis losses (with or without binarization threshold tuning),
  - calibration (with or without Laplace calibration [72] of concept model outputs), and
  - monitor modeling (different t-norm fuzzy logics [77], rule and predicate formulations).

My Text and Content Contributions

I contributed the texts and content in the main paper and the supplemental material, except for texts, experiments and results on model uncertainty calibration (parts of *Sec. 2*, *Sec. 3.3*, parts of *Sec. 4.1*). Visualizations and tables are authored by me.

4.5. Chapter Summary

This chapter presented a set of requirements for CNNs that are based on semantic prior knowledge. They are structured by the parent safety goals of *correct internal representation* (Requirement (I)) and *correct reasoning* (Requirement (II)). Evidence generation methods for the following requirements were outlined: task-relevant concepts are embedded (Requirement (I.1)); task-irrelevant ones not (Requirement (I.3)); the embeddings preserve semantic similarity (Requirement (I.2)); the CNN behaves plausible (Requirement (II.1)); and it respects prior knowledge in the form of fuzzy logical rules (Requirement (II.2)). The safety relevance of the formulated safety goals is demonstrated on AD related examples.

For each requirement, approaches to obtain safety evidence are outlined, with a summary of the respective experimental evaluations provided in this thesis. This includes efficient methods how to use CA in order to build expressive interpretable proxy models for manual inspection, and to verify and monitor logical consistency with respect to fuzzy logical rules. In both cases, the core idea is to enrich the output of a trained CNN by further semantically meaningful intermediate outputs. The concepts of interest are chosen to cover a set of pre-defined task-related concepts. The practical applicability to AD related settings is demonstrated for all of the discussed post-hoc methods.

5. Safety Argument Structure for DNNs

DNNs are an attractive modeling approach for hard-to-specify applications like CV perception tasks in HAD. This also holds for safety relevant applications, i.e., such that can cause harm to persons. Before putting such systems into operation, one needs to ensure with sufficient confidence that no harmful incidents will occur. Hence, the functional sufficiency of the system in terms of performance and safety must be assured and assessed. While there are several approaches to achieve and document the assurance, a popular method in the automotive industries is to provide a *safety case*, which is even mandatory when following the ISO 26262:2018 international standard [48] for automotive functional safety (ISO 26262). In short, a safety case is a structured argumentation with supportive evidence that all hazards that may cause harm, as well as their associated risks, are identified and sufficiently mitigated [8].

This chapter now aims to present my work towards a safety argument for CNNs in CV applications, and sort in the contribution of CA into the overall safety case. For this, Sec. 5.2 first carves out how traditional approaches to obtain a safety case, like the ISO 26262, are challenged by components relying on DNNs. Then, Sec. 5.3 positions my contributions amongst related work on safety assurance of DNNs in automotive perception applications. The remaining sections present the work from this thesis towards a safety argument structure of DNN-based perception: Sec. 5.4 tackles a *bottom-up* point of view on the argument by reviewing methods for safety evidence generation in [Th04] (an up-to-date update of the survey can be found in Appendix A). Sec. 5.5 summarizes a *top-down* argumentation strategy, i.e., it shows how to derive safety requirements, and where to allocate the symbolic requirements that were detailed in Sec. 4.1. The top-down and bottom-up approaches are joint in my core work [Th05] on safety argument structure, given in Sec. 5.6. This chapter finalizes with an overview in Sec. 5.7 on how the CA-based applications from this thesis contribute to a safety argument (including an initial evaluation study of CA applicability in [Th03]).

The next section, Sec. 5.1, provides a short revision of basic notions in automotive safety management used here. It is targeted at readers not familiar with the topic. Experienced readers may skip this part.

5.1. Basic Notions in Automotive Safety Assurance

Automotive safety efforts have a long history [100]. First international standards for embedded software in road vehicles were published in 1994 [15]. This section recalls basics and notions from standard safety assessment in automotive industries that are relevant to the later parts of this chapter. More precisely, some basics are recapitulated regarding safety terms (Subsec. 5.1.1), arguing safety using a safety case (Subsec. 5.1.2), and the later used goal structuring notation [89] (GSN) (Subsec. 5.1.3).

The definitions used throughout this thesis are based upon the ISO 26262, which is an adaptation of the IEC 61508:2010 standard [45] on functional safety of electrical, electronic, and programmable electronic systems (IEC 61508). Being first published in 2011, by now the ISO 26262 is widely adopted in automotive industries. Furthermore, the ISO/PAS 21448:2019 draft automotive standard [52] for safety of the intended functionality of a system (ISO/PAS 21448) is considered as a basis. It serves as an addition to the ISO 26262 for assistant systems, and is concerned with proper specification in case the system has to process complex environmental inputs or human interactions.

5.1.1. Risk and Safety

According to the two automotive safety standards ISO 26262 and ISO/PAS 21448, *safety* can be defined directly via the notion of *risk*.

Definition 5.1 (Hazard)**.** This thesis is concerned with mitigation of losses respectively *harm* to life or health of persons, i.e., physical injury or (longer-term) damage to health [48, Sec. 3.74]. In this context, a *hazard* is a potential source of harm [48, Sec. 3.75]. A hazard can substantiate into concrete harm given certain operational situations (*hazardous events* [48, Sec. 3.77]). A hazardous event can be, e.g., a (near) car crash.

Definition 5.2 (Risk)**.** *Risk* of a system is the probability that the system causes physical injury of a certain severity to persons [48, Sec. 3.128]. It is ranked *unreasonable* if it reaches an ethically unacceptable level according to valid societal moral concepts [48, Sec. 3.176], or if the appropriate acceptance authority is not willing to accept it without additional mitigation [97, Sec. 3.2.1]. A *risk acceptance principle* is a guideline on how to evaluate the acceptability of a risk that is introduced by a system.

The general goal of assuring safety is to reach an acceptable level of residual risk. So-called risk acceptance principles allow to define a threshold for this. Typical ones are the following [84] (including an exemplary interpretation for the case of HAD):

As Low As Reasonable Practicable (ALARP): If used for new technologies like DNNs this requires profound understanding of that technology, and some notion of best-practice development.

Globalement Au Moins Aussi Bon (GAMAB): The term comes from French “in general at least as good”. It assumes that the global level of risk of a new system must be at least as low as that of a comparable existing reference system. In the case of HAD, the reference system would be the manually steered car.

Minimum Endogenous Mortality (MEM): This principle considers the average risk for individuals. MEM demands that the natural mortality risk for an individual per year should not increase by more than a given percentage due to the new system. This relies on detailed statistics of the current mortality rates.

The notion of risk now gives rise to a definition of *safety*.

Definition 5.3 (Safety)**.** *Safety* of an automotive system is the absence of unreasonable risk imposed by the system.

According to the classification in [52, Tab. 1], safety infringements can be caused by one of *malfunction* of (parts of) the system with respect to a specified functionality [48, Sec. 3.132]; a *foreseeable misuse* of the originally intended functionality (cf. [52]; not considered here); inherent *performance limitations* of the system with respect to complex and diverse environmental sensory inputs, e.g., camera inputs (cf. [52]), or further *technology specific sources of hazards.*

Automotive Safety Integrity Level An important building block for safety assurance is the estimation of the current level of safety with respect to a given hazard. To ease this, the IEC 61508 introduced a discrete ranking system. For automotive purposes, the ISO 26262 refines that to the four so-called Automotive Safety Integrity Level [49, Sec. 6.3.4] (ASIL). The Automotive Safety Integrity Level [49, Sec. 6.3.4] (ASIL) ranking takes into account the *probability* that the hazard occurs and causes harm, the *severity* of potentially occurring harm, and the *controllability* in case of hazardous events, i.e., how well involved persons, e.g., the driver, are able to avoid the harm in a hazardous event, e.g., via timely braking. The practical meaning of the ASILs is that each level is associated with standard methods and measures for safety assurance. For example, non-life-threatening, not difficult to control hazardous events with a low probability may be treated just using quality management according to ISO 26262 guideline. Sub-functions that give rise to hazardous events of more critical ASIL require additional, and more careful, safety actions during development, verification, validation, and operation. The ASIL associated to a functional requirement, e.g., maximum failure rate of a component, can be reduced by adapting the system architecture. Some examples of system architectural measures like redundancy are given in Sec. A.3.

Modularization Another important concept for practical safety assessment is the *modularization* of the system, and finally a modular safety argument [33, Sec. 2:4]. Usually, starting from an overarching operational safety perspective, one derives (1) associated, implementation agnostic *functional safety requirements*, and thereof (2) implementation interface specific *technical safety requirements*. To achieve this, the system usually is modeled as system of functional or technical sub-elements (sub-system, hardware component, or software unit). Practically, this allows to break down high-level safety-related requirements into functional safety requirements for each functional sub-element, and trace down sources of system failures to single system elements or element (inter)actions [63]. This allows to treat and develop single elements without context of their parent system other than the functional safety requirements allocated to the element. This is called *Safety Element out of Context [48, Sec. 3.138] (SEooC)*.

5.1.2. Safety Case

Given a system that is potentially capable of causing harm, like a road vehicle, its manufacturer must provide sufficient proof of the safety of that system before bringing it into operation. In the automotive industries, this kind of proof often is conducted in the form of a *safety case* in accordance with ISO 26262.

Definition 5.4 (Safety Case)**.** A *safety case* for a system is a convincing and valid argument, supported by a documented body of evidence, that this system is adequately safe for a given environment and given application [8]. According to the ISO 26262, evidence is given by *work products*. These are pieces of documentation resulting from activities during the product lifecycle, and from one or several associated requirements described in the ISO 26262 standard [48, Sec. 3.136].

The arguments used in a safety case may be deterministic using proven causal (Boolean) relations, but also probabilistic, or merely qualitative [8]. There are several parts involved in constructing the safety case: the top-down, product-based derivation of requirements; a lifecycle process with best-practices, giving rise to a process-based argumentation and ensuring creation of all necessary work products; and the bottom-up evidence generation within the lifecycle. These are shortly recapitulated in the following.

Safety Goals and Top-down Argumentation

An important part towards building the safety argumentation line of a safety case is the *top-down* safety requirements derivation and risk analysis. This usually is done prior to the actual technical development, and is intended to systematically identify all necessary operational and functional requirements.

Definition 5.5 (Safety Goal)**.** The *safety goals* are the top-level safety requirements to be evidenced by the safety case.

The safety goals should directly originate from the hazards associated with the system of interest. For example, consider a car with the pedestrian detection assistant system from Sec. 1.2. In this case, associated hazards would be (1) unexpected braking, causing a rear crash, or (2) no braking, causing the car to hit a vulnerable road user, e.g., a pedestrian.

Allocation of Safety Requirements There are diverse approaches to identify hazardous events and hazards of a system, and trace them down to hazardous behavior of system elements. Some notable ones are [62, Sec. 8.1]:

Fault Tree Analysis [20] (FTA): A logical tree diagram is deduced top-to-bottom that causally relates hazardous system events with events of sub-systems. Due to its long history it is still widely adopted in industries [20].

Failure Mode and Effects Analysis (FMEA): The influence of single component errors on the system behavior is investigated in a bottom-up manner, trying to uncover new failure modes. It is widely used in combination with fault tree analysis.

System Theoretic Process Analysis (STPA): Previous approaches only consider errors of system components as sources of hazards. System Theoretic Process Analysis (STPA) instead also takes into account complex interactions of elements and external factors, e.g., human interaction and environmental conditions. The method relies on modeling the system as a controller-loop. Being introduced in 2011, STPA is a rather new concept specifically developed for highly complex sensor-actuator systems, and is currently finding its way into several industries.

Given a thorough causal analysis of hazardous event chains allows to allocate functional safety requirements to single system elements. These serve as basis for derivation of the technical safety requirements during system development [50, Sec. 6]. Finally, safety goals and functional safety requirements each can be associated with an ASIL. This can be used to judge the "importance" of the respective requirement, and deduce concrete ASIL-specific implementation, verification, and validation measures according to ISO 26262.

Lifecycle Process

The core structure given by the ISO 26262 is the suggested lifecycle procedure. This includes the top-down requirements derivation in the beginning, and ensures requirements application and evidence generation during development. Practically, a lifecycle procedure will generate at each stage a collection of *work products*. These are documentation providing evidence for requirements associated with the respective stage [48, p. 3.185]. See [78] for DNN specific work product examples. Following a lifecycle gives rise to a *process-based argumentation* for safety: One gains confidence in safety through application of current proven-in-use safety practice, which is known to prevent systematic faults in development.

The ISO 26262 follows a simple V-cycle [48, Fig. 1] (cf. [Th04]). It features the following steps (ones that are not considered here according to the scope definition from Sec. 1.2 are grayed out):

1) *Requirements derivation*: Hazard analysis, derivation of functional safety requirements
2) *Implementation* of software and hardware
3) *Verification* that all safety requirements are fulfilled on system elements (software, hardware)
4) *Integration* of software into embedded hardware

5) *Verification* of safety requirements on the integrated system
6) *Validation* of the integrated system, i.e., identification of yet unknown issues that may require adaptation/extension of the safety requirements
7) *Production*
8) *Maintenance and decommissioning*

Evidences

Following [8], some typical types of evidence for given safety requirements are given in the following. As examples serve the methods suggested in the ISO 26262 part on software level development [51] (for an overview see Figure 5.2).

- *"Proven-in-use argument"* [48, p. 3.115]: Field data has shown that the element in question is sufficiently safe in operation (with respect to its associated ASIL). Note that this—even for a large-scale mass-produced product—may require years of field operation [90].

- Proper *design* decisions both on
  - *System architectural level* via a technical safety concept for risk mitigation and control, including measures like redundancy or monitoring [51, Sec. 7.4.12]; and
  - *Unit level*, e.g., choice of algorithm. Concretely, [51, Sec. 8.4.5] suggests the following desirable properties for software units: simplicity and comprehensibility, robustness, suitability for software modification, i.e., maintainability, and verifiability.

- Application of a pre-defined *best-practice development process*, e.g., including strict review cycles and implementation measures known to improve quality;

- *Verification* of the implementation, the model or a simulation of the model (cf. [51, Tab. 4, Tab. 7]); as well as *validation* of the requirements thereon. Example methods for this are:
  - *Manual* walk-through or inspection;
  - *Testing* with test cases derived from (cf. [51, Tab. 8]) *requirements* and value *boundaries*, *equivalence classes*, or prior human *knowledge and experience*, e.g., on typically hard-to-process input perturbations;
  - *Formal* and *semi-formal* (automated) verification.

For new technologies, the relevant pillars of evidence are (cf. [Th05, Sec. 4]):
(1) *System architectural design* to reduce the safety load of an element;
(2) *Best-practice* in unit design and development; and
(3) *Verification* and *validation*.
This thesis concentrates on the latter two.

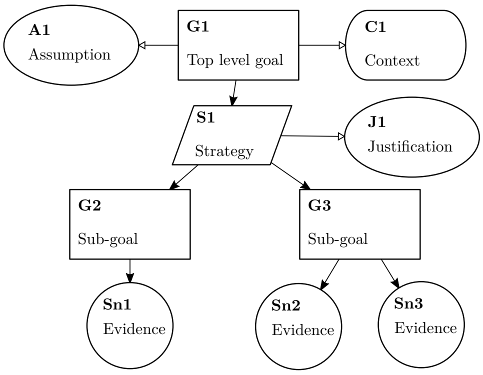


Figure 5.1. Basic elements of a argument visualization using GSN: goals (**G**), strategies (**S**), evidences (**Sn**), contexts (**C**), assumptions (**A**), and justifications (**J**)

5.1.3. Goal Structuring Notation

Goal structuring notation [89] (GSN) is a tree-like graphical notation for a safety argumentation. The standard for the notation is publicly available and widely used for documenting safety cases in industries [37, 26, 9]. The main elements of the notation are visualized in Figure 5.1: The *goals* formulate the claims or sub-claims to be proven, *strategies* can be used to break down goals into sub-goals, and one or more pieces of *evidence* are finally provided as leaves for the claims. Further rationales may (and should [36, 19, 9]) be given in the form of *context* for an element, *assumptions* for the validity of the claim or strategy, and *justifications* for making assumptions. The rationales allow to make so-called *assurance claim points* [36] more explicit. These are to-be-evidenced assertions. Assertions can be about the causal inference used for designing the argument (e.g., evidences or sub-goals truly contribute to proving a goal), as well as about validity of used contexts (e.g., the vehicle will not leave the specified operational design domain). This notion of an argument part about confidence in the validity of the argument has recently found its way into a standard (see [33, Sec. 2:5.2]). For alternatives on goal structuring notation have a look at [33, Sec. 2:4.8].

I use goal structuring notation [89] (GSN) in my works [Th05, Th03].

5.2. Challenges in Safety Assurance of DNNs

CNNs for perception tasks come with several downsides, originating from the application, the automated modeling, and inherent properties of DNNs. These are collected here to emphasize the necessity for new safety argument patterns and evidence generation methods.

Traditional Implicit Assumptions Recall the following assumptions that are inherent to many standard evidence generation methods and evidences:

The function to test is not chaotic. Testing on discrete samples of the input space inherently assumes that the behavior at one sample generalizes to a region around the sample. This validity range of the behavior at a sample is broken by unexpected discontinuity or chaotic behavior with strong value changes [Th05].

Influences on the output of the function to test are known. Testing on discrete samples of the input space also inherently assumes that the behavior at two samples generalizes to the region between the two samples. This means, all influencing factors are assumed to be known that differentiate these samples and change the behavior in-between them.

Given prior knowledge can be guaranteed to be respected in the function. In particular, there are measures available with which one can change the function to respect prior knowledge up to a given degree. And side-effects regarding compliance with other functional requirements are known and controllable. For example, in standard software one can introduce checks and separate routines for special cases. This will not break behavior on other cases.

The input space can be completely or sufficiently formally described. Such an understanding and description of the input space is necessary to sufficiently cover it with test cases, and to estimate the degree of coverage. An example is the identification of equivalence classes.

Adverse DNN Properties Given above assumptions and evidence approaches, the following properties directly infringe applicability of the mentioned evidence techniques. A graphical overview which DNN properties infringe what evidence generation methods is given in Figure 5.2.

Application: DNNs allow tackling hard-to-specify applications like environment perception. Such applications, however, come along with some direct safety issues:

- *Open world:* Due to the *open world context* of perception applications in HAD, the operational domain and the concrete algorithm are impossible to specify completely. In specific, the sub-space of realistic images in the space of all images (which mostly consists of meaningless noise) is currently unspecified.
- *Non-symbolic inputs:* Furthermore, expert knowledge is often only available on a semantic level. However, sensoric representations of the environment usually are *non-symbolic*, like pixels of images or points in point clouds. This makes it hard to specify verifiable requirements on the input.

Machine learning: Machine learning in specific means that the algorithmic model is not manually crafted based on expert knowledge. Instead, it is automatically derived from samples under constraints, like the model architecture or type, and the regularization.

- *Automated modeling:* Automated modeling means that prior expert knowledge (often) cannot be *guaranteed* to be included into the final model. This, however, is a basic assumption of many modeling best-practices suggested in the ISO 26262. Also, the learned model may be counterintuitive, invalidating the standard assumption for testing that influences on the output are sufficiently understood.

- *Monolithic:* ML models are usually *monolithic* and potentially large. This contradicts the modularity idea phrased in the ISO 26262.
- *Inherent uncertainty:* Statistical machine learned functions exhibit an inherent *uncertainty* which must be respected during integration, especially in architectural measures.

DNN technology: DNNs have a theoretically [42] and practically [111] very high capacity for storing information. This allows to fit arbitrary functions using a DNN architecture. However, the flexibility comes along with several properties that are unfavorable for safety assurance:

- *Black-box:* DNNs are usually *high-dimensional* and at the same time exhibit *distributed representations* of symbolic (and non-symbolic) knowledge. This makes them inherently *black-box*, i.e., hard to understand and to manually assess by humans. This again infringes understanding of influence factors, as well as formal specification of the model internals.
- *Brittleness:* DNNs are prone to *local instability* respectively unexpected chaotic behavior [95]. Changes to the input that are imperceptible or small for humans can rapidly change the output of a trained DNN. This is also known as existence of *adversarial examples*. Reasons can be task-unrelated predictive features that are contained in the data samples, like non-symbolic noise patterns [47]. The latent non-semantic influence factors and brittleness greatly infringe the validity range of test samples.
- *No local adaptivity:* Standard software models usually give rise to manual invention for adapting the model towards a new target. Due to the black-box property, for DNNs this can only be achieved by re-training or fine-tuning on new samples. However, the *influence of fine-tuning* on the original functionality on unrelated samples can hardly be controlled. Due to the distributed representations, keeping behavior on unrelated samples intact cannot be guaranteed. This drastically limits practical maintainability: Verification cannot be done incremental, but each re-training step would require a fully-fledged verification of the model.
- *Novelty:* Until now there is no or little agreed-on *best practice* for DNN development in safety relevant perception applications available. Standardization efforts are still ongoing (cf. Sec. 5.3).

Some further issues were not detailed here. These arise from the new hardware needs to do real-time inference on large DNNs, and from validation aspects that are heavily influenced by the complexity of the open world [Th05, Sec. 4.3]. Many of the ISO 26262 evidence generation methods are still applicable to DNNs [82], especially for the implementation part. But, as this list of issues shows, they loose their effectivity with respect to

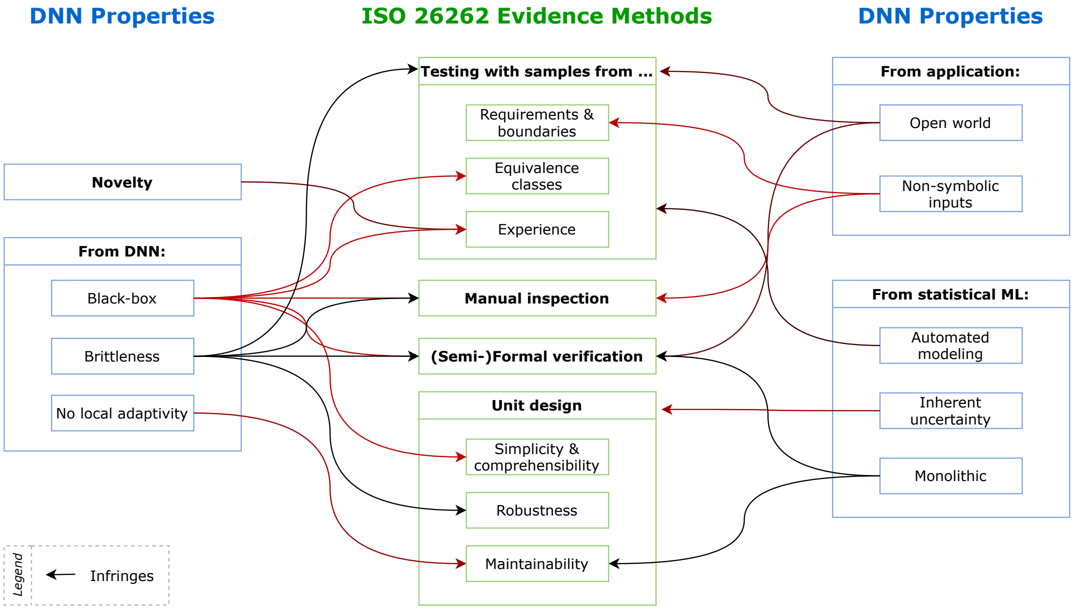


Figure 5.2. Overview on safety relevant DNN-specifics (*left* and *right*) and ISO 26262 evidence generation methods (*center*), and which DNN properties infringe what traditional evidences. Implementation independent system architectural evidence measures are skipped here.

DNN model safety assurance. New properties and property combinations, including specific failure modes like the brittleness, make new approaches towards safety argumentation (Sec. 5.5) and evidence acquisition (Sec. 5.4) inevitable for DNN-based perception applications.

5.3. Related Work

This section gives an overview on related standardization efforts, as well as research on DNN specific safety argument patterns and safety concerns. In the end, the work in this thesis is positioned within the related work.

Standards The base standard used here is the ISO 26262:2018 international standard [48] for automotive functional safety, which has been widely practically adopted throughout automotive industries since publication of its first version in 2011. It is adapted from the more general IEC 61508:2010 standard [45] on functional safety of electrical, electronic, and programmable electronic systems (IEC 61508). The IEC 61508 explicitly recommends not to use any AI-based functions in safety critical applications [46, Tab. A.2]. While the ISO 26262 only considers the safety of the directly intended functionality and treats hazards arising from malfunction, the ISO/PAS 21448:2019 draft automotive standard [52] for safety of the intended functionality of a system from 2019 extends this to also include safety of the intended functionality. Concretely, they also consider insufficiencies of the intended functionality, like foreseeable misuse or inherent performance limitations in case of unexpected inputs. This is especially relevant in case the system has to process complex environmental inputs or human interactions. However, ISO/PAS 21448 explicitly is not technology specific and does not cover DNN specific concerns. Similarly, the UL4600 standard for safety of autonomous products [59], also first published in 2019, does not detail safety specifics of DNNs, but already adopts some general considerations and recommendations for machine learned functions [59, p. 8.5]. This includes model capability and choice, model performance, data quality and representativity, general robustness, and considerations for post-deployment changes. The white paper [104], also published in 2019, concentrates on safety considerations for level 3 and 4 AD. Level 3 and 4 AD refers to automated driving road vehicles with backup-driver (possibly after take-over phase). The white paper was established by 11 automotive industry partners, and broadly collects relevant functions of automated driving, including associated safety-by-design considerations. While the covered system aspects are broad, they concentrate on best-practice recommendations. The discussed safety requirements and system-level safety argument patterns are very high-level. The paper is the base of a currently developed standard. The short German DIN SPEC 13266 [18] guideline for the development of deep learning image recognition systems from 2020 provides a list of aspects that need to be considered during development of DNN-based CV systems. This has a high overlap with the previously mentioned standards and does not go into detailed technical issues or methods for DNNs. Further more recent standardization initiatives are still ongoing, like

- ISO/IEC AWI TR 5469 Artificial intelligence — Functional safety and AI systems,
- ISO/TR 22100-5:2021: Safety of machinery — Relationship with ISO 12100 — Part 5: Implications of artificial intelligence/machine learning,
- ISO/IEC TR 24029-1:2021: Artificial Intelligence (AI) — Assessment of the robustness of neural networks, and
- ISO/PAS 8800: Road vehicles – Functional safety and AI.

They promise more in-depth considerations of safety argument patterns and evidence generation methods specific to DNNs in CV. Lastly and most recently, the SCSC-159 Assurance Case Guidance standard [33] that was published in August 2021 gathers some general state-of-the-art best-practices on building and structuring an assurance case. It aims to be applicable throughout many industries, and provides a broad collection of related standards.

Safety Concerns Specific to DNN Technology Besides the standards, some parallel work collecting concrete issues regarding safety assurance of DNNs were [103] and [83]. In [103] a comprehensive list of safety concerns is collected that may arise at any point in the lifecycle of a DNN-based perception function. Also, they shortly discuss mitigation methods for each of the identified concerns. While the list is detailed and a good starting point for considerations, it is more of an experience collection and no completeness argument is given, other than in my work [Th05, Th03]. Also, the discussed safety concerns explicitly do not mean to represent concrete failure types (cf. [103, Fig. 1]). Safety concerns introduced here were: data quality, representativity, and distributional shift over time; incomprehensible behavior; long tail distribution of open world data; unreliable uncertainty outputs; brittleness; training and test data separation; and safety-awareness in used performance metrics. Other than [103], [83] concentrates on direct *DNN insufficiencies*, i.e., performance limitations of the DNN that may directly cause errors. In this work, it is not yet differentiated between DNN insufficiencies representing failure modes (lack of generalization ability, robustness, or efficiency) and ones that may merely infringe safety assurance efforts (lack of plausibility or explainability). Picking up on this general scheme, my work [Th05] details the insufficiencies directly associated with failures and merges this with my prior work in [Th04, 86]. Besides the DNN insufficiencies, [83] introduces a rough categorization scheme for DNN related metrics. And, similar to my work [Th04], they suggest to allocate safety measures and regular metric measurements as evidences to each stage in the development lifecycle of a DNN model. They in detail discuss few methods and considerations for each of the selected lifecycle stages, but with a focus on an audience of DNN developers, not system and safety engineers.

Safety Argument Patterns Specific to DNN Technology The standards and papers discussed so far in this section all concentrate on providing evidence or work-products, and less consider the question for a top-down safety argument structure. An important consideration is: What assurance claim points [36] are involved when providing (performance) evidence to ML specific safety requirements, i.e., what assumptions exist and must be argued? In [9], six types of assurance claim points are introduced in detail. This covers proof for assumptions on surrounding system and operational design domain, as well as the assumed connection between the measured performance indicator and actual safety. Furthermore, the presented assurance claim points include general design and process evidences associated with the given requirement, namely correct model and training design choices, and training and test data sufficiency. Similarly, the work in [106] introduces detailed safety case patterns to elaborate on the assurance claim point branches for data appropriateness, model design, and training procedure. The break-down is rather shallow and concentrates on collecting associated goals instead of discussing technical details. As in the discussed standards, the AMLAS practical guideline [37] suggests a detailed development process for ML models. Using the lifecycle as a baseline, they suggest several detailed break-down patterns for assurance claim points associated with different lifecycle stages. Like for the other work, the focus here lies on further breaking down given ML specific or DNN specific safety requirements. The discussed work does not systematically derive the DNN specific performance requirements that are required as a starting point.

Positioning of this Thesis To summarize, the publications on safety argumentation in this thesis [Th04, Th03, Th05] constitute early and novel work towards

- concrete sources for DNN specific safety requirements, including systematic DNN insufficiency identification and association to failure modes [Th04, Th05],
- top-down safety argumentation patterns that formulate and break down concrete DNN specific safety requirements, as a basis for the work on assurance claim points [Th03, Th05], and
- bottom-up evidence generation method categorization [Th04, Th05].

5.4. Publication: A Survey on Methods for the Safety Assurance of Machine Learning Based Systems

The next content is contained in the following paper:

[Th04] **Gesina Schwalbe** and Martin Schels. “A Survey on Methods for the Safety Assurance of Machine Learning Based Systems”. In: *Proc. 10th European Congress Embedded Real Time Software and Systems*. Jan. 2020. URL: `https://hal.archives-ouvertes.fr/hal-02442819`
Find the original full text attached on page 293.

Positioning in the Thesis

This survey gives a first and systematic overview over methods and considerations that can assist in providing safety evidence for a ML-based component (cf. [Th04, Tab. 1]). It thus serves as a bottom-up approach towards building a safety argument. For structuring the review, a lifecycle-based approach is used, i.e., *when* a method can be applied, other than in the later work [Th05] which is based on *what* a method should achieve. Using the lifecycle-based overview, open challenges for the safety assessment of ML-based components are systematically derived. This includes the need for model introspection techniques with provable guarantees, which is further pursued in this thesis. A short update on the method review is attached in Appendix A.

My Scientific Contributions

- Suggestion and certification oriented overview of
  - Sources/types of ML specific safety requirements,
  - Safety desirables for ML development, which may serve as a basis for ML specific safety work products, and
  - Verification and validation goals for data-driven and machine learned components;
- Overview, categorization, and examples of ML safety evidence generation methods by
  - their portability to use-cases and technology, and
  - their use for the suggested safety desirables by lifecycle phase;
- Derivation of open challenges for safety assessment of ML-based components, including
  - model introspection with provable guarantees,
  - inclusion of (symbolic) expert knowledge,
  - validation of data representativity and model diversity.

My Text and Content Contributions

Structure, content, texts and visualizations were fully provided by me. Co-authors assisted in writing style and focus.

5.5. Top-down Safety Argument Structure

Parts of this thesis develop a top-down argument structure, with [Th05] representing the heart of this work. The top-down argument structure defines patterns for top-level DNN specific safety requirements, and argument patterns for breaking these further down. Here, the notion of DNN insufficiencies constitutes the core of the requirements derivation. This basis is introduced in detail in Subsec. 5.5.1. Then, the main ideas for the break-down patterns developed in my works [Th03, Th05] are summarized in Subsec. 5.5.2. In that course it is pointed out, how the requirements templates detailed in Sec. 4.1 fit into the overall argument. The top-down approach is later merged with a bottom-up approach in my publication [Th05] in Sec. 5.6.

5.5.1. DNN Insufficiencies

As discussed before in Sec. 5.2, there are several traits to the technology of DNNs which require special treatment in a safety case. Hence, in order to achieve completeness, the safety case must argue for each of these traits that it causes no unacceptable safety risk. This is also the guiding principle applied in my works [Th05, 86, Th03] to obtain a complete set of top-level, DNN technology specific safety requirements. What is now needed in a first step is a formal definition and systematic collection of such technology traits. This is provided by the notion of *DNN insufficiency* that is used in my thesis and is introduced and analyzed in this section.

Definitions

This thesis relies on the following definition of *DNN insufficiency* from my work [Th05].

Definition 5.6 (DNN Insufficiency)**.** A *DNN insufficiency* in general is defined as a property of a trained DNN model that has a negative impact for the use in safety relevant systems, and is inherent to the technology of DNNs. A *measurable* DNN insufficiency causes errors that decrease *associated performance measures*.

For example, this also includes the black-box property of DNNs which infringes verifiability. This work concentrates on measurable DNN insufficiencies, Specifically, the focus is on DNN insufficiencies that may directly cause *errors* of the DNN, i.e., DNN specific *performance limitations*. The ISO/PAS 21448 defines these quite general as insufficiencies in the implementation of the intended functionality [52, Sec. 3.9]. Not considered are issues that are specific to the integrated system implementation, e.g., memory consumption, inference time, or required transmission rates. Here, error is formally understood as follows.

Definition 5.7 (DNN Error)**.** An *error* of a DNN is an output that does not match or too much deviates from a human defined ground truth, with respect to a given distance measure on the output space.

As discussed in Sec. 1.2, error types of a general object detector can be false localization, i.e., shifted object center points or object dimensions, as well as false positives (invention of objects) and false negatives (ignoring of objects). While L_2 distance and overlap scores like those from Def. 3.2 provide continuous scores for localization errors, false positives and false negatives usually are considered to be discrete as follows (e.g., as used in my work [Th06]).

Definition 5.8. A detection is a *false positive* if it has no sufficient overlap with a ground truth object, and a ground truth object counts as *false negative* if there is no predicted detection with sufficient overlap. Sufficient overlap is defined by application specific user-defined thresholds with respect to a chosen overlap score (e.g., IoU).

A Categorization Scheme for DNN Insufficiencies

For new technologies like DNNs, a hard and new part in a safety argument is to (1) break the general black-box errors from Def. 5.7 and Def. 5.8 further down into a complete set of technology specific error types, and to (2) (best quantitatively) trace the errors back to root causes that can be controlled, i.e., measured and mitigated. As recapitulated in Sec. 5.2, DNNs rely on statistical optimization, i.e., are influenced in a hardly controllable manner by starting values [25], batch order, and other statistical aspects. Thus, direct causality between single error cases and training time decisions, e.g., single samples or statistical properties of the training data, usually cannot be established. To still grasp some main deficiencies of DNNs, [Th05] chose a categorization scheme based on error distributions.

Simply said, DNN insufficiencies here are differentiated by *how a test set must "look like"* that typically triggers the insufficiency to cause errors. For example, whether a semantic (day or night) or a non-semantic (level of Gaussian noise) feature in the data has to be changed to make the model fail more often. More precisely, the following modeling option is assumed to apply for the (measurable) DNN insufficiencies of interest.

Definition 5.9 (Reference Distribution)**.** A distribution of input samples on which the network is expected to perform good (with the *reference performance*) with respect to the associated performance measures is called the *reference distribution*.

Note that one can randomly sample a dataset from this distribution. A reference distribution must be chosen by the user and is application specific. By default, the training data distribution is considered and the performance on the training data serves as reference.

Definition 5.10 (Associated Distribution)**.** Consider a measurable DNN insufficiency. An input data distribution that *triggers* the insufficiency is called *associated*, where triggering means that (1) the DNN performance drops considerably on a test dataset sampled from this distribution, compared to the reference performance, and, and (2) the errors decreasing the performance are caused by the DNN insufficiency.

We here consider measurable DNN insufficiencies for which a reference and associated distribution can be chosen. Then, practically, DNN insufficiencies can be defined using an identified associated distribution. These again arise from test-subsets on which DNN unusually often fails. As an example, assume that an object detection DNN often fails on small objects on a test set. This gives rise to the specific DNN insufficiency that is associated to data with high occurrence rates of small objects. In this work, DNN insufficiencies are now differentiated by the *type of difference between their associated distribution and the reference distribution.*

Collection of DNN Insufficiencies

In my work [Th04, Th05] several types of DNN insufficiencies were identified, and then formalized and detailed in [Th05]. Those insufficiency types capable of directly causing errors are the following (cf. [Th05, Sec. 3.1]):

Issues with internal semantic representation and logic: These issues are assumed to arise from faulty internal semantic representation of environment features or logical steps of the task. The test cases triggering such an insufficiency can be constructed from symbolic logical rules. *Such issues are hard to find due to the black-box property of DNNs, which is tackled by the methods proposed in this thesis.* As an example, assume the logical insufficiency is that a DNN does not associate persons with person heads. Then any person that is mostly occluded except for the head will not be detected, even though the head might be well visible.

Non-semantic robustness issues: DNNs are known to exhibit locally instable behavior from a human point of view (cf. Sec. 5.2). This means addition of slight non-semantic artifacts to the input, like slight noise, causes the output of the DNN to change drastically, potentially to an erroneous prediction. Slight here means that the artifacts would not infringe or change a human's detection capability, or are even imperceptible for humans. These so-called adversarial examples may be caused by non-semantic predictive features in the training data [47], and may even exhibit a considerable validity range, i.e., be themselves robust against further input perturbations [40]. Examples of adversarial input perturbations are both artificial noise patterns [47], and naturally occurring contaminations like graffiti [22]. Altering the distribution of the respective non-semantic features may trigger the insufficiency.

Simple general performance issues: This encompasses issues caused by, e.g., simple dataset bias, over- or underfitting. Either the performance on the reference distribution already is unacceptably low (underfitting); or the associated distribution only differs slightly from the reference one, i.e., in few semantic dimensions and without more complex rules. Examples of such semantic dimensions can be that the model cannot cope with night-vision or rainy weather. Note that some instances of this insufficiency type may be closely related to the purely semantic logical issues.

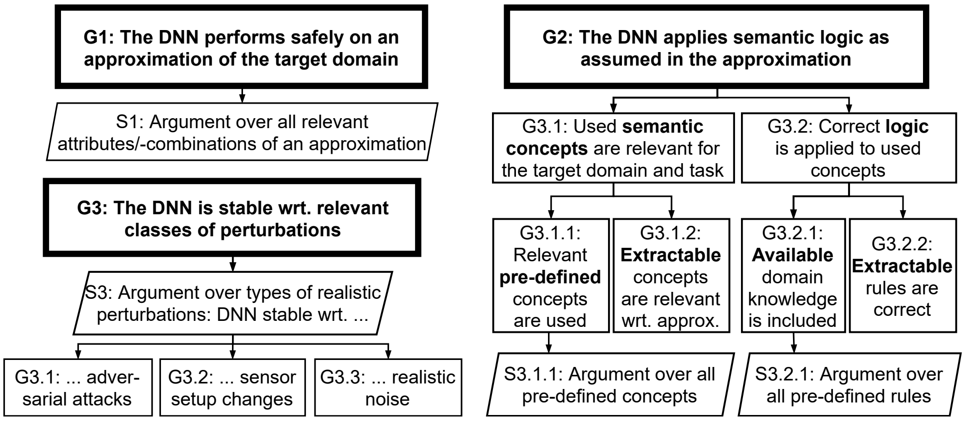

G1: The DNN performs safely on an approximation of the target domain
S1: Argument over all relevant attributes/-combinations of an approximation
G3: The DNN is stable wrt. relevant classes of perturbations
S3: Argument over types of realistic perturbations: DNN stable wrt. ...
G3.1: ... adver-sarial attacks
G3.2: ... sensor setup changes
G3.3: ... realistic noise
G2: The DNN applies semantic logic as assumed in the approximation
G3.1: Used semantic concepts are relevant for the target domain and task
G3.2: Correct logic is applied to used concepts
G3.1.1: Relevant pre-defined concepts are used
G3.1.2: Extractable concepts are relevant wrt. approx.
G3.2.1: Available domain knowledge is included
G3.2.2: Extractable rules are correct
S3.1.1: Argument over all pre-defined concepts
S3.2.1: Argument over all pre-defined rules

Figure 5.3. Overview on the top-level DNN specific safety requirements and their break-down strategies suggested in this thesis. Taken from [Th05, Fig. 1].

5.5.2. DNN Safety Argument Structure

This subsection summarizes the main ideas from [Th03, Th05] on how to systematically derive safety requirements and how to sub-structure them in the safety argument.

The basic idea is to make sufficient mitigation of the DNN insufficiencies from Subsec. 5.5.1 the top goal. This then results in one safety requirement per insufficiency. To achieve confidence in completeness, it must be ensured that existing insufficiencies are sufficiently well analyzed and understood. Experience, i.e., thorough literature analysis, and the systematic categorization scheme for DNN insufficiencies introduced before can help here. Note also that the DNN insufficiencies based on semantics (logical issues and simple performance issues) require proper semantic understanding and modeling of the input (termed *approximation of the target domain* in [Th05]).

The safety requirements for the previously defined insufficiency categories can be broken down using the following strategies. An overview is shown in Figure 5.3.

Mitigation of logical issues (G2): This requirement is the main focus of the methods developed in this thesis (cf. Chap. 4). It can practically be broken down into the requirements detailed in Sec. 4.1, namely mitigating issues with

- *Requirement (I) from Sec. 4.1* (**G3.1**)*:* the representation itself, and
- *Requirement (II) from Sec. 4.1* (**G3.2**)*:* the logical usage of embedded semantic concepts.

Also, it should be differentiated between (1) verification whether pre-defined prior knowledge is respected by the DNN (**G3.1.1**, **G3.2.1**), and (2) plausibility checks, i.e., manual validation of representations and logic learned by the DNN (**G3.1.2**, **G3.2.2**; cf. method example from Sec. 4.4).

Mitigation of robustness issues (G3): This can be broken down by the type of perturbation leading to the robustness issue. Super-types suggested in [Th05] are (**S3**) targeted adversarial attacks (**G3.1**), changes in the sensor setup (**G3.2**; e.g., degradation, position change), and realistic sources of noise (**G3.3**; e.g., fog, rain). Many examples of perturbations are collected in the CV-HAZOP database[1] [110].

Mitigation of simple performance issues (G1): By definition, these issues are characterized by few (or none) semantic features that, when changed, result in bad performance. Thus, the canonical break-down strategy is to iterate over the known semantic feature dimensions to check whether one exhibits a problem (**S1**).

[1] https://vitro-testing.com/cv-hazop/

5.6. Publication: Structuring the Safety Argumentation for Deep Neural Network Based Perception in Automotive Applications

The next content is contained in the following paper:

[Th05] **Gesina Schwalbe**, Bernhard Knie, Timo Sämann, Timo Dobberphul, Lydia Gauerhof, Shervin Raafatnia, and Vittorio Rocco. "Structuring the Safety Argumentation for Deep Neural Network Based Perception in Automotive Applications". In: *Computer Safety, Reliability, and Security. SAFECOMP 2020 Workshops*. Springer International Publishing, 2020, pp. 383–394. DOI: 10.1007/978-3-030-55583-2_29. Reproduced in this work with permission from Springer Nature
Find the original full text attached on page 303.

Positioning in the Thesis

This work establishes an approach to structure the safety argumentation for DNN-based perception components. The structure is based on a two-fold approach: (1) a top-down error-driven part (cf. Sec. 5.5), and (2) a bottom-up categorization scheme for evidence generation methods extending upon [Th04] (Sec. 5.4).

My Scientific Contributions

- Formulation of concrete DNN error types (DNN insufficiencies), merging own prior work, like [86], and that of co-authors (especially [83]);
- Development of a top-down argument structure formulated in GSN which breaks down safety requirements derived from DNN specific error types;
- Proposal of a bottom-up categorization of evidence generation methods that is aligned with practical notions from traditional safety assurance.

My Text and Content Contributions

I authored the texts except for sub-sections *4.1 Mechanisms During Creation*, *4.2 Mechanisms on Component and System Level*, and paragraph *Validation* of *4.3 Verification and Validation*. The paper visualizations and outline were provided by me.

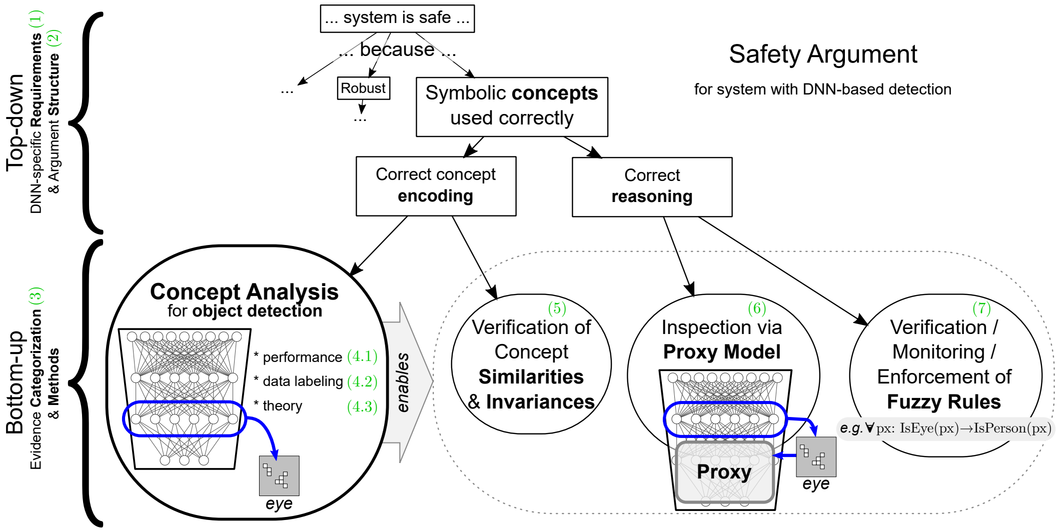

Top-down
DNN-specific Requirements (1)
& Argument Structure (2)
Bottom-up
Evidence Categorization (3)
& Methods
... system is safe ...
... because ...
...
Robust
...
Symbolic concepts used correctly
Safety Argument
for system with DNN-based detection
Correct concept encoding
Correct reasoning
Concept Analysis
for object detection
* performance (4.1)
* data labeling (4.2)
* theory (4.3)
eye
enables
(5) Verification of Concept Similarities & Invariances
(6) Inspection via Proxy Model
Proxy
eye
(7) Verification / Monitoring / Enforcement of Fuzzy Rules
e.g. ∀px: IsEye(px)→IsPerson(px)

Figure 5.4. Overview on the contributions of this thesis to the overall safety argument for a DNN-based system, notated in simplified GSN (cf. Subsec. 5.1.3). Contributions of this thesis by papers are: *(1–2):* [Th03, Th04, Th05], cf. Sec. 5.5; *(3):* [Th04, Th07, Th08], cf. Sec. 5.4, Chap. 2; *(4.1–2):* [Th01, Th02, Th03, Th06], cf. Chap. 3; *(4.3):* [Th03, Th07, Th08], cf. Chap. 2; *(5):* [Th01]; *(6):* [Th02]; *(7):* [Th06]; *(5–7):* cf. Chap. 4.

5.7. Contributions by Methods from this Thesis

This chapter gives an overview on how the CA-based methods for evidence generation from Chap. 4 contribute to the safety argument. This is tackled from two perspectives: Subsec. 5.7.1 sorts the CA-based verification applications from Chap. 4 into the different evidence generation categories that were identified previously in Sec. 5.6. This will show that all root categories are enriched by contributions from this thesis. Subsec. 5.7.2 gives with [Th03] an overall picture of where CA-based methods may be allocated in a safety argumentation. This merges the top-down perspectives, from Chap. 4, where methods are associated with types of safety requirements, and from Subsec. 5.5.2, which sorts these requirement types into the a safety argument template. A simplified overview from the second perspective is given in Figure 5.4.

5.7.1. Contributions by Evidence Generation Categories

In Sec. 5.6, three root categories of evidence methods were identified, based on *what* the methods should achieve: Improvements during development, verification and validation, or mitigation of safety issues via system architectural measures. The CA-based applications presented in Chap. 4 enrich each of these categories as follows.

Measures during development

(1) In [Th06] I developed a method for *corner case analysis* that allows to find, and manually generalize, samples in which the model behaves logically inconsistent with respect to pre-defined rules. This can be used, e.g., for augmentation of a training dataset.

(2) In [Th03, Sec. 3.2] an *interactive training scheme is proposed*[2] that allows to fine-tune a DNN towards better compliance with pre-defined logical rules. Other than similar work like logic tensor networks [6], this can be done after a pre-training, and for rules not only formulated on inputs and outputs, but also on embedded concepts.

(3) The output of a CNN can efficiently be augmented by concept models after the training using the developed *CA method* (see Chap. 3, [Th01], [Th03]).

Verification

(4) This thesis presents a supervised post-hoc *concept analysis* method suitable for object detection CNNs. This can serve to disentangle parts of the information contained in CNN latent spaces, which is needed for many verification approaches. Furthermore, many CA-based verification approaches are developed in this work as separately detailed in Chap. 4.

(5) The mentioned CA method is used to build *expressive rule-based proxy models* with high fidelity that are suitable for inspection or (semi-)formal property verification. Find details in Sec. 4.3 and [Th02].

Mechanisms on system level

(6) In [Th06] I developed a *fuzzy logic based monitoring mechanism* that allows to efficiently observe the compliance with fuzzy logic rules during runtime. Rules may again not only be formulated on inputs and outputs, but also on embedded concepts that are accessed using CA. Rule monitors may help in identifying and filtering potentially erroneous outputs.

[2] This is just a proposal and has not yet been experimentally verified.

5.7.2. Publication: Concept Enforcement and Modularization as Methods for the ISO 26262 Safety Argumentation of Neural Networks

The next content is contained in the following paper:

[Th03] **Gesina Schwalbe** and Martin Schels. "Concept Enforcement and Modularization as Methods for the ISO 26262 Safety Argumentation of Neural Networks". In: *Proc. 10th European Congress Embedded Real Time Software and Systems*. Jan. 2020. url: `https://hal.archives-ouvertes.fr/hal-02442796`
Find the original full text attached on page 315.

Positioning in the Thesis

This work details the potential value of CA for the safety argument of DNN-based components. For this, it points out three applications, which are sorted into an initial template for the safety argument of a DNN (cf. [Th03, Fig. 1]):
(1) verifying the inner representation of a DNN (cf. Sec. 4.2 and Sec. 4.4),
(2) post-training modularization of DNNs (cf. proxy model approach in Sec. 4.3),
(3) and fixing inner representations (cf. future work from [Th06]).
These applications are partly further investigated in later work in this thesis (cf. Chap. 4) and provide outlooks on possible future work. As an initial applicability study of CA, the CA method Net2Vec [24] is generalized to a different DNN model, training and concept dataset in a simple AD related setting. Some suggested improvements of the method are experimentally validated, and outperformed the baseline (cf. Sec. 3.5). The found improvements were used as a basis for the later application development presented in Chap. 4.

My Scientific Contributions

- Experimental validation and considerable improvement of the baseline Net2Vec [24] CA approach on a simple AD related traffic sign classification dataset; improvements regarding loss and input pre-processing were found in a broad experimental study.

- Development of a safety argument structure and formulation thereof in GSN; using this, identification of three gaps in existing evidence generation methods that may be filled using CA.

- Detailed proposal of two CA applications in safety assurance.

My Text and Content Contributions

Structure, content, texts and visualizations were fully provided by me. I drafted, implemented, conducted, and interpreted the experiments. Co-authors assisted in writing style and focus.

5.8. Chapter Summary

This chapter first recapitulated basic concepts of automotive safety assurance. This includes the widely used safety case which is a structured argument with evidences. Furthermore, it is argued why traditionally used evidences and requirements will not suffice for systems based on a safety relevant DNNs component.

The focus of this chapter lies on two main argumentation parts: the top-down argumentation which derives and breaks down safety requirements, and the bottom-up collection of various evidences. For the top-down part, a scheme for differentiating DNN error modes is presented. The three derived categories of issues (internal symbolic logic, robustness, simple performance issues like overfitting) are used to define basic safety requirements. These are further broken down, giving rise to the requirements considered in Sec. 4.1. For the bottom-up part, a categorization scheme for evidence generation methods is introduced which is aligned with the traditional main pillars of evidences. These are concretely: measures during development, verification and validation, and mechanisms on system level.

Finally, the contribution of the CA-based evidence generation methods that are developed in this thesis is clarified: For the first time, it is possible to give quantitative evidence for safety requirements that are associated with the internal semantics and internal symbolic logic of a DNN. With respect to this, all categories of evidence generation methods are covered by approaches developed in this thesis.

6. Conclusion and Outlook

Efficient Concept Analysis for Object Detection This thesis defines and demarcates CA as a new sub-field of XAI research with interesting applications. Further, a post-hoc supervised CA method for concept segmentation is developed and extensively evaluated. On object parts, the proposed approach showed much better performance and more stable convergence than the baseline. Combined with the efficiency improvements developed here, CA is, for the first time, practically applicable to object detection and high resolution datasets. Finally, it is shown that the method can easily be adapted towards detection of object center points instead of concept segmentation. For evaluations, an automated labeling scheme is used. This allows to easily add concept labels to existing human body keypoint datasets. Altogether, a corner stone is provided for CA applications on object detection. I hope this will foster further research in that direction.

Safety Assessment of DNN-based Systems The formal safety assurance argument for DNN-based systems faces many practical challenges that are pinpointed in this thesis. Here, some concrete steps are taken to solve these. First, the notion of DNN insufficiencies that potentially lead to errors is formalized. Three main insufficiency types are identified: Simple generalization issues (e.g., from overfitting), robustness issues, and issues with internal semantics and symbolic logic. It is shown how these can be used to define concrete top-level safety requirements for the DNN specific part of a safety argument. Furthermore, breakdown patterns are presented that reduce those requirements to a practically manageable level.

In complement to this top-down argument, bottom-up evidence driven patterns are also provided. These are given in the form of a structured overview and categorization scheme for evidence generation methods. The scheme is based on the main evidence pillars from traditional safety assessment: measures during development, verification and validation, and mechanisms on system level. Hence, the categorization of evidence methods is compatible with the standard argument part of the non-DNN parts of the system. The overview may serve as a look-up table during safety assessment efforts. And lastly, the combination of method categorization and safety requirements allows to identify concrete gaps in existing evidence generation methods for DNN safety assurance. One example of a gap is further tackled in this thesis: methods to ensure compliance with symbolic constraints, and to ensure the validity of internal representations.

Safety Evidence for Symbolic Constraints on CNNs This thesis presents several methods for ensuring compliance of CNNs with symbolic logical constraints of different types. The developed verification and monitoring applications all are based on CA for associating semantic concepts with CNN internals.

First, concept embeddings are used to verify the quality of CNN internal semantics (i.e., embedding of semantic concepts and similarities between them). This allows to assess invariances with respect to task unrelated concepts. For example, experiments showed that internal representations of CV CNN are mostly invariant to human body sizes. In several experiments, intuitive results were achieved showing that CV CNNs generally seem to be well aligned with human semantics. Furthermore, a correlation was observed between the concept embedding quality metric and CNN performance. This gives rise to the conjecture that good alignment of the CNN representations with human semantics is not only needed for verifiability, but also beneficial for performance.

Another verification application developed in this thesis, is to use CA as input to a symbolic proxy model. This allows to obtain symbolic interpretable proxy models that can be used, e.g., for manual inspection, even for non-symbolic inputs of the CNN. It was found that such can be realized with high fidelity.

Lastly, a CA-based verification framework is developed that can verify and monitor compliance with any symbolic fuzzy logic rules. The main idea utilized here is that a DNN together with concept models can be interpreted as a family of fuzzy logical predicates, which then may be used for rule formulation. It is demonstrated how the self-supervised framework enables comparison of models with respect to logical consistency and detection of corner cases, both offline and during runtime. The derived runtime monitor was found to uncover a considerable amount of detection errors, even for a single simple logical rule. The offline corner case detection efficiently points out interesting error cases. These may be generalized to symbolic rules according to which the DNN fails.

Altogether, this thesis identified and patched several gaps for practical safety assessment of CNNs in CV tasks. Specifically, diverse evidence generation methods are developed to assure compliance with symbolic background knowledge.

Future Work Some proposals from this thesis are waiting for further practical evaluation, such as the methods for concept center point detection. Also, it would be interesting to see the fuzzy logic monitoring for more rules, and for direct comparison of more networks. Furthermore, the scope of this thesis is limited to AD perception. Hence, it would be interesting to see how well the approaches generalize to other domains, or different tasks, like trajectory planning, and different input, such as videos. A very promising field of further applications is hybrid neuro-symbolic learning: The developed verification methods allow to define differentiable training constraints. These can be used as (additional) loss terms during training, as demonstrated for fuzzy logical rules in the logic tensor networks framework [6]. But also, they may serve valuable and efficient for multi-task training in a fine-tuning step. Alternatively, the found symbolic error cases may help in defining data augmentation strategies for fine-tuning. Such hybrid learning promises benefits not only with respect to compliance with safety requirements, but also with respect to general performance improvements of CNNs. In general, it should be noted that the methods and applications developed here may be suitable to support other responsibility aspects like fairness. Evaluation of such contributions also is a matter of future work.

Still a considerable amount of gaps need to be filled towards convincing safety assurance of safety relevant DNNs. The solutions from this thesis tackle some of the pain points, namely compliance with symbolic background knowledge and argument structuring. This leaves many wounds open. To just name a few: DNN robustness issues; data representativity and coverage assurance; test case selection and generation; problems with domain shifts or gaps; issues introduced by pruning and quantization and other measures necessary for ML deployment; and confidence and safety standard compliance of new ML tooling like the used ML libraries. And this has not yet touched issues for HAD functions other than perception components, or for advanced human-machine-interactions and vehicle-to-vehicle or vehicle-to-infrastructure communication settings. This illustrates clearly that the cross-cutting concern of safety demands for many more interdisciplinary research activities.

Coming back to the issue of compliance with symbolic prior knowledge, XAI—and in specific CA—promise great advances. Many further practically applicable XAI methods are required to assist developers and auditors in providing responsible AI [5]. Hence, I want to close with a call for further research in his direction, in order to approach feasible and responsible AI applications in safety relevant settings.

Bibliography

[1] Amina Adadi and Mohammed Berrada. "Peeking inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)". In: *IEEE Access*. Vol. 6. 2018, pp. 52138–52160. DOI: 10.1109/ACCESS.2018.2870052.

[2] Fouzia Altaf, Syed M. S. Islam, Naveed Akhtar, and Naeem Khalid Janjua. "Going Deep in Medical Image Analysis: Concepts, Methods, Challenges, and Future Directions". In: *IEEE Access* 7 (2019), pp. 99540–99572. DOI: 10.1109/ACCESS.2019.2929365.

[3] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul F. Christiano, John Schulman, and Dan Mané. "Concrete Problems in AI Safety". In: *CoRR* abs/1606.06565 (2016). URL: http://arxiv.org/abs/1606.06565.

[4] Jacob Andreas, Dan Klein, and Sergey Levine. "Learning with Latent Language". In: *Proc. Conf. North Amer. Chapter of the Assoc. for Computational Linguistics: Human Language Technologies*. Vol. 1. Association for Computational Linguistics, 2018, pp. 2166–2179. ISBN: 978-1-948087-27-8. DOI: 10.18653/v1/N18-1197.

[5] Alejandro Barredo Arrieta, Natalia Díaz Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. "Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI". In: *Inf. Fusion* 58 (2020), pp. 82–115. ISSN: 1566-2535. DOI: 10.1016/j.inffus.2019.12.012.

[6] Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. "Logic Tensor Networks". In: *CoRR* abs/2012.13635 (Jan. 2021). URL: https://arxiv.org/abs/2012.13635.

[7] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. "Network Dissection: Quantifying Interpretability of Deep Visual Representations". In: *Proc. 2017 IEEE Conf. Comput. Vision and Pattern Recognition*. IEEE Computer Society, 2017, pp. 3319–3327. ISBN: 978-1-5386-0457-1. DOI: 10.1109/CVPR.2017.354.

[8] P. G. Bishop and R. E. Bloomfield. "A Methodology for Safety Case Development". In: *Industrial Perspectives of Safety-Critical Systems: Proc. Safety-Critical Systems Symp.* Springer London, 1998. ISBN: 978-3-540-76189-1. URL: http://openaccess.city.ac.uk/549/.

[9] Simon Burton, Lydia Gauerhof, Bibhuti Bhusan Sethy, Ibrahim Habli, and Richard Hawkins. “Confidence Arguments for Evidence of Performance in Machine Learning for Highly Automated Driving Functions”. In: *Computer Safety, Reliability, and Security*. Vol. 11699. Lecture Notes in Computer Science. Springer International Publishing, June 2019, pp. 365–377. ISBN: 978-3-030-26250-1. DOI: 10.1007/978-3-030-26250-1_30.

[10] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan Su. “This Looks like That: Deep Learning for Interpretable Image Recognition”. In: *Advances in Neural Information Processing Systems 32*. Vol. 32. 2019, pp. 8928–8939.

[11] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. “Rethinking Atrous Convolution for Semantic Image Segmentation”. In: *CoRR* abs/1706.05587 (Dec. 2017). URL: http://arxiv.org/abs/1706.05587.

[12] Runjin Chen, Hao Chen, Ge Huang, Jie Ren, and Quanshi Zhang. “Explaining Neural Networks Semantically and Quantitatively”. In: *Proc. 2019 IEEE/CVF International Conference on Computer Vision*. IEEE, Oct. 2019, pp. 9186–9195. ISBN: 978-1-72814-803-8. DOI: 10.1109/ICCV.2019.00928.

[13] Chih-Hong Cheng, Frederik Diehl, Yassine Hamza, Gereon Hinz, Georg Nührenberg, Markus Rickert, Harald Ruess, and Michael Troung-Le. “Neural Networks for Safety-Critical Applications — Challenges, Experiments and Perspectives”. In: *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*. Sept. 2017, pp. 1005–1006. DOI: 10.23919/DATE.2018.8342158.

[14] Conner Chyung, Michael Tsang, and Yan Liu. “Extracting Interpretable Concept-Based Decision Trees from CNNs”. In: *Proc. 2019 ICML Workshop Human in the Loop Learning*. Vol. 1906.04664. CoRR, June 2019. URL: http://arxiv.org/abs/1906.04664.

[15] MISRA Consortium. *A Brief History of MISRA*. 2021. URL: https://www.misra.org.uk/a-brief-history-of-misra/ (visited on 10/04/2021).

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. “ImageNet: A Large-Scale Hierarchical Image Database”. In: *Proc. 2009 IEEE Conf. Comput. Vision and Pattern Recognition*. IEEE, June 2009, pp. 248–255. ISBN: 978-1-4244-3992-8. DOI: 10.1109/CVPR.2009.5206848.

[17] DIN Deutsches Institut für Normung e. V. *Artificial Intelligence - Life Cycle Processes and Quality Requirements - Part 1: Quality Meta Model*. 2019-04. Vol. 1. DIN Spec 92001. Beuth Verlag, Apr. 2019. DOI: 10.31030/3050203.

[18] DIN Deutsches Institut für Normung e. V. *DIN SPEC 13266:2020-04: Guideline for the development of deep learning image recognition systems*. 2020-04. Beuth Verlag, Apr. 2020. DOI: 10.31030/3134557.

[19] Clément Duffau, Thomas Polacsek, and Mireille Blay-Fornarino. "Support of Justification Elicitation: Two Industrial Reports". In: *Advanced Information Systems Engineering*. Vol. 10816. Lecture Notes in Computer Science. Springer International Publishing, June 2018, pp. 71–86. ISBN: 978-3-319-91562-3. DOI: `10.1007/978-3-319-91563-0_5`.

[20] Clifton Ericson. "Fault Tree Analysis - a History". In: *Proc. 17th Int. Systems Safety Conf.* 1999.

[21] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. "The Pascal Visual Object Classes (VOC) Challenge". In: *Int J Comput Vis* 88.2 (June 2010), pp. 303–338. ISSN: 1573-1405. DOI: `10.1007/s11263-009-0275-4`.

[22] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. "Robust Physical-World Attacks on Deep Learning Visual Classification". In: *Proc. 2018 IEEE Conf. Computer Vision and Pattern Recognition.* IEEE Computer Society, 2018, pp. 1625–1634. DOI: `10.1109/CVPR.2018.00175`.

[23] Patrick Feifel, Frank Bonarens, and Frank Koster. "Reevaluating the Safety Impact of Inherent Interpretability on Deep Neural Networks for Pedestrian Detection". In: *Proc. 2021 IEEE/CVF Conf. Comput. Vision and Pattern Recognition.* 2021, pp. 29–37.

[24] Ruth Fong and Andrea Vedaldi. "Net2Vec: Quantifying and Explaining How Concepts Are Encoded by Filters in Deep Neural Networks". In: *Proc. 2018 IEEE Conf. Comput. Vision and Pattern Recognition.* IEEE Computer Society, 2018, pp. 8730–8738. DOI: `10.1109/CVPR.2018.00910`.

[25] Jonathan Frankle and Michael Carbin. "The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks". In: *Proc. 7th Int. Conf. Learning Representations.* OpenReview.net, 2019. URL: `https://openreview.net/forum?id=rJl-b3RcF7`.

[26] Lydia Gauerhof, Peter Munk, and Simon Burton. "Structuring Validation Targets of a Machine Learning Function Applied to Automated Driving". In: *Computer Safety, Reliability, and Security.* Lecture Notes in Computer Science. Springer International Publishing, 2018, pp. 45–58. ISBN: 978-3-319-99130-6.

[27] Yunhao Ge, Yao Xiao, Zhi Xu, Meng Zheng, Srikrishna Karanam, Terrence Chen, Laurent Itti, and Ziyan Wu. "A Peek into the Reasoning of Neural Networks: Interpreting with Structural Visual Concepts". In: *Proc. 2021 IEEE/CVF Conf. Computer Vision and Pattern Recognition.* 2021, pp. 2195–2204.

[28] Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S. Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, Tiffany Fernandez, Martin Jänicke, Sudesh Mirashi, Chiragkumar Savani, Martin Sturm, Oleksandr Vorobiov, Martin Oelker, Sebastian Garreis, and Peter Schuberth. "A2D2: Audi Autonomous Driving Dataset". In: *CoRR* (Apr. 2020). URL: https://www.a2d2.audi.

[29] Amirata Ghorbani, James Wexler, James Y. Zou, and Been Kim. "Towards Automatic Concept-Based Explanations". In: *Advances in Neural Information Processing Systems 32*. 2019, pp. 9273–9282. URL: http://papers.nips.cc/paper/9126.pdf.

[30] EICT GmbH. *KI Absicherung*. 2021. URL: https://www.ki-absicherung-projekt.de/ (visited on 09/05/2021).

[31] M. Graziani, V. Andrearczyk, S. Marchand-Maillet, and H. Müller. "Concept Attribution: Explaining CNN Decisions to Physicians". In: *Computers in Biology and Medicine* 123 (Aug. 2020), p. 103865. ISSN: 0010-4825. DOI: 10.1016/j.compbiomed.2020.103865.

[32] Mara Graziani, Vincent Andrearczyk, and Henning Müller. "Regression Concept Vectors for Bidirectional Explanations in Histopathology". In: *Understanding and Interpreting Machine Learning in Medical Image Computing Applications*. Lecture Notes in Computer Science. Springer International Publishing, 2018, pp. 124–132. ISBN: 978-3-030-02628-8. DOI: 10.1007/978-3-030-02628-8_14.

[33] SCSC Assurance Case Working Group. *SCSC-159: Assurance Case Guidance — Challenges, Common Issues and Good Practice*. Version 1. SCSC SCSC-159. SCSC Assurance Case Working Group, Aug. 2021. URL: https://scsc.uk/r159:1.

[34] Jindong Gu and Volker Tresp. "Semantics for Global and Local Interpretation of Deep Neural Networks". In: *CoRR* abs/1910.09085 (Oct. 2019). URL: http://arxiv.org/abs/1910.09085.

[35] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. "On Calibration of Modern Neural Networks". In: *Proc. 2017 Int. Conf. Machine Learning*. PMLR, July 2017, pp. 1321–1330. URL: http://proceedings.mlr.press/v70/guo17a.html.

[36] Richard Hawkins, Tim Kelly, John Knight, and Patrick Graydon. "A New Approach to Creating Clear Safety Arguments". In: *Advances in Systems Safety*. Springer, 2011, pp. 3–23. ISBN: 978-0-85729-133-2. DOI: 10.1007/978-0-85729-133-2_1.

[37] Richard Hawkins, Colin Paterson, Chiara Picardi, Yan Jia, Radu Calinescu, and Ibrahim Habli. "Guidance on the Assurance of Machine Learning in Autonomous Systems (Amlas)". In: *CoRR* abs/2102.01564 (Feb. 2021). URL: http://arxiv.org/abs/2102.01564.

[38] K. He, G. Gkioxari, P. Dollár, and R. Girshick. “Mask R-CNN”. In: *Proc. 2017 IEEE Int. Conf. Comput. Vision.* Oct. 2017, pp. 2980–2988. DOI: 10.1109/ICCV.2017.322.

[39] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep Residual Learning for Image Recognition”. In: *Proc. 2016 IEEE Conf. Comput. Vision and Pattern Recognition.* June 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[40] Warren He, Bo Li, and Dawn Song. “Decision Boundary Analysis of Adversarial Examples”. In: *Proc. 6th Int. Conf. Learning Representations.* OpenReview.net, 2018. URL: https://openreview.net/forum?id=BkpiPMbA-.

[41] Fred Hohman, Haekyu Park, Caleb Robinson, and Duen Horng Polo Chau. “Summit: Scaling Deep Learning Interpretability by Visualizing Activation and Attribution Summarizations”. In: *IEEE Trans. Visual. Comput. Graphics* 26.1 (Jan. 2020), pp. 1096–1106. ISSN: 1941-0506. DOI: 10.1109/TVCG.2019.2934659.

[42] Kurt Hornik. “Approximation Capabilities of Multilayer Feedforward Networks”. In: *Neural Networks* 4.2 (Jan. 1991), pp. 251–257. ISSN: 0893-6080. DOI: 10.1016/0893-6080(91)90009-T.

[43] Sebastian Houben, Stephanie Abrecht, Maram Akila, Andreas Bär, Felix Brockherde, Patrick Feifel, Tim Fingscheidt, Sujan Sai Gannamaneni, Seyed Eghbal Ghobadi, Ahmed Hammam, Anselm Haselhoff, Felix Hauser, Christian Heinzemann, Marco Hoffmann, Nikhil Kapoor, Falk Kappel, Marvin Klingner, Jan Kronenberger, Fabian Küppers, Jonas Löhdefink, Michael Mlynarski, Michael Mock, Firas Mualla, Svetlana Pavlitskaya, Maximilian Poretschkin, Alexander Pohl, Varun Ravi-Kumar, Julia Rosenzweig, Matthias Rottmann, Stefan Rüping, Timo Sämann, Jan David Schneider, Elena Schulz, **Gesina Schwalbe**, Joachim Sicking, Toshika Srivastava, Serin Varghese, Michael Weber, Sebastian Wirkert, Tim Wirtz, and Matthias Woehrle. “Inspect, Understand, Overcome: A Survey of Practical Methods for AI Safety”. In: *CoRR* abs/2104.14235 (Apr. 2021). URL: http://arxiv.org/abs/2104.14235.

[44] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard H. Hovy, and Eric P. Xing. “Harnessing Deep Neural Networks with Logic Rules”. In: *Proc. 54th Annu. Meeting of the Association for Computational Linguistics.* Vol. 1: Long Papers. The Association for Computer Linguistics, 2016. ISBN: 978-1-945626-00-5. URL: http://aclweb.org/anthology/P/P16/P16-1228.pdf.

[45] International Electrotechnical Commission IEC. *Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems - Part 1: General Requirements.* 2.0. Vol. 1. IEC 61508:2010. VDE Verlag, Apr. 2010. ISBN: 978-2-88910-524-3. URL: https://www.vde-verlag.de/iec-normen/preview-pdf/info_iec61508-1%7Bed2.0%7Db.pdf.

[46] International Electrotechnical Commission IEC. *Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems - Part 3: Software Requirements*. 2.0. Vol. 3. IEC 61508:2010. VDE Verlag, Apr. 2010. ISBN: 978-2-88910-524-3.

[47] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. "Adversarial Examples Are Not Bugs, They Are Features". In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 125–136. URL: http://papers.nips.cc/paper/8307.pdf.

[48] ISO/TC 22/SC 32. *ISO 26262-1:2018(En): Road Vehicles — Functional Safety — Part 1: Vocabulary*. Second. Vol. 1. ISO 26262:2018(En). International Organization for Standardization, Dec. 2018. URL: https://www.iso.org/standard/68383.html.

[49] ISO/TC 22/SC 32. *ISO 26262-3:2018(En): Road Vehicles — Functional Safety — Part 3: Concept Phase*. Second. Vol. 3. ISO 26262:2018(En). International Organization for Standardization, Dec. 2018. URL: https://www.iso.org/standard/68385.html.

[50] ISO/TC 22/SC 32. *ISO 26262-4:2018(En): Road Vehicles — Functional Safety — Part 4: Product Development at the System Level*. Second. Vol. 4. ISO 26262:2018(En). International Organization for Standardization, Dec. 2018. URL: https://www.iso.org/standard/68383.html.

[51] ISO/TC 22/SC 32. *ISO 26262-6:2018(En): Road Vehicles — Functional Safety — Part 6: Product Development at the Software Level*. Second. Vol. 6. ISO 26262:2018(En). International Organization for Standardization, Dec. 2018. URL: https://www.iso.org/standard/68388.html.

[52] ISO/TC 22/SC 32/WG 8. *ISO/PAS 21448:2019(En): Road Vehicles — Safety of the Intended Functionality*. Two thousand, nineteenth. International Organization for Standardization, Jan. 2019. URL: https://www.iso.org/standard/70939.html.

[53] Dmitry Kazhdan, Botty Dimanov, Mateja Jamnik, Pietro Liò, and Adrian Weller. "Now You See Me (CME): Concept-Based Model Extraction". In: *Proc. 29th ACM Int. Conf. Information and Knowledge Management Workshops*. Vol. 2699. CEUR Workshop Proceedings. CEUR-WS.org, 2020. URL: http://ceur-ws.org/Vol-2699/paper02.pdf.

[54] Dmitry Kazhdan, Botty Dimanov, Helena Andres Terre, Mateja Jamnik, Pietro Liò, and Adrian Weller. "Is Disentanglement All You Need? Comparing Concept-Based & Disentanglement Approaches". In: *CoRR* abs/2104.06917 (Apr. 2021). URL: http://arxiv.org/abs/2104.06917.

[55] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres. "Interpretability beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV)". In: *Proc. 35th Int. Conf. Machine Learning*. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 2668–2677. URL: http://proceedings.mlr.press/v80/kim18d.html.

[56] Jinkyu Kim and John F. Canny. "Interpretable Learning for Self-Driving Cars by Visualizing Causal Attention". In: *Proc. 2017 IEEE Int. Conf. Comput. Vision*. IEEE Computer Society, 2017, pp. 2961–2969. DOI: 10.1109/ICCV.2017.320.

[57] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *Proc. 3rd Int. Conf. Learning Representations*. 2015. URL: http://arxiv.org/abs/1412.6980.

[58] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. "Concept Bottleneck Models". In: *Proc. 2020 Int. Conf. Machine Learning*. PMLR, Nov. 2020, pp. 5338–5348. URL: http://proceedings.mlr.press/v119/koh20a.html.

[59] Philip Koopman, Edge Case Research, and Underwriters Laboratories. *UL4600: Standard for Safety of Autonomous Products*. Edge Case Research, Dec. 2019. URL: https://edge-case-research.com/ul4600/.

[60] Alex Krizhevsky. "One Weird Trick for Parallelizing Convolutional Neural Networks". In: *CoRR* abs/1404.5997 (2014). URL: http://arxiv.org/abs/1404.5997.

[61] Jan Kronenberger and Anselm Haselhoff. "Dependency Decomposition and a Reject Option for Explainable Models". In: *Proc. 2019 IEEE Conf. Comput. Vision and Pattern Recognition Workshops*. 2019. URL: http://arxiv.org/abs/2012.06523.

[62] Nancy Leveson. *Engineering a Safer World: Systems Thinking Applied to Safety*. Engineering Systems. MIT Press, 2012. ISBN: 978-0-262-53369-0. URL: https://mitpress.mit.edu/books/engineering-safer-world.

[63] Nancy Leveson and John Thomas. *STPA Handbook*. Mar. 2018. URL: https://psas.scripts.mit.edu/home/get_file.php?name=STPA_handbook.pdf (visited on 03/27/2020).

[64] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. "Focal Loss for Dense Object Detection". In: *Proc. 2017 IEEE Int. Conf. Comput. Vision*. IEEE Computer Society, Oct. 2017, pp. 2999–3007. DOI: 10.1109/ICCV.2017.324.

[65] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft COCO: Common Objects in Context". In: *Proc. 13th European Conf. Computer Vision - Part V*. Vol. 8693. Lecture Notes in Computer Science. Springer International Publishing, 2014, pp. 740–755. ISBN: 978-3-319-10602-1. DOI: 10.1007/978-3-319-10602-1_48.

[66] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. "Explainable AI: A Review of Machine Learning Interpretability Methods". In: *Entropy* 23.1 (Jan. 2021), p. 18. DOI: 10.3390/e23010018.

[67] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. "Deep Learning for Generic Object Detection: A Survey". In: *International Journal of Computer Vision* 128.2 (Feb. 2020), pp. 261–318. DOI: 10.1007/s11263-019-01247-4.

[68] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. "SSD: Single Shot MultiBox Detector". In: *Computer Vision – ECCV 2016*. Lecture Notes in Computer Science. Springer International Publishing, 2016, pp. 21–37. ISBN: 978-3-319-46448-0. DOI: 10.1007/978-3-319-46448-0_2.

[69] Max Losch, Mario Fritz, and Bernt Schiele. "Interpretability beyond Classification Output: Semantic Bottleneck Networks". In: *Proc. 3rd ACM Computer Science in Cars Symp. Extended Abstracts*. Oct. 2019. URL: https://arxiv.org/abs/1907.10882.

[70] Adriano Lucieri, Muhammad Naseer Bajwa, Andreas Dengel, and Sheraz Ahmed. "Explaining AI-Based Decision Support Systems Using Concept Localization Maps". In: *Neural Information Processing*. Communications in Computer and Information Science. Springer International Publishing, 2020, pp. 185–193. ISBN: 978-3-030-63820-7. DOI: 10.1007/978-3-030-63820-7_21.

[71] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. "Understanding the Effective Receptive Field in Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 29. Curran Associates, Inc., 2016, pp. 4898–4906.

[72] David J. C. MacKay. "The Evidence Framework Applied to Classification Networks". In: *Neural Comput.* 4.5 (Sept. 1992), pp. 720–736. ISSN: 0899-7667. DOI: 10.1162/neco.1992.4.5.720.

[73] Rowan McAllister, Yarin Gal, Alex Kendall, Mark van der Wilk, Amar Shah, Roberto Cipolla, and Adrian Weller. "Concrete Problems for Autonomous Vehicle Safety: Advantages of Bayesian Deep Learning". In: *Proc. 26th Int. Joint Conf. Artificial Intelligence*. 2017, pp. 4745–4753. DOI: 10.24963/ijcai.2017/661.

[74] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. "Linguistic Regularities in Continuous Space Word Representations". In: *Proc. 2013 Conf. North American Chapter Association for Computational Linguistics: Human Language Technologies.* Association for Computational Linguistics, June 2013, pp. 746–751. URL: https://www.aclweb.org/anthology/N13-1090.

[75] Yatin Nandwani, Abhishek Pathak, Mausam, and Parag Singla. "A Primal Dual Formulation for Deep Learning with Constraints". In: *Advances in Neural Information Processing Systems 32.* Curran Associates, Inc., 2019, pp. 12157–12168. URL: http://papers.nips.cc/paper/9385.pdf.

[76] NDR. "Proteste von Umweltschützern: Polizeieinsätze auf A2 und A7". In: *ndr.de* (Nov. 2020). URL: https://www.ndr.de/nachrichten/niedersachsen/braunschweig_harz_goettingen/Proteste-von-Umweltschuetzern-Polizeieinsaetze-auf-A2-und-A7,protestaktion138.html (visited on 09/06/2021).

[77] Vilém Novák, Irina Perfilieva, and J. Mockor. *Mathematical Principles of Fuzzy Logic.* The Springer International Series in Engineering and Computer Science. Springer US, Jan. 1999. ISBN: 978-0-7923-8595-0. DOI: 10.1007/978-1-4615-5217-8.

[78] Krystian Radlak, Piotr Serwa, Michal Szczepankiewicz, and Tim Jones. "Organization of ML-Based Product Development as per ISO 26262". In: *CoRR* abs/1910.05112 (Oct. 2019). URL: http://arxiv.org/abs/1910.05112.

[79] Sebastian Ruder. "An Overview of Gradient Descent Optimization Algorithms". In: *CoRR* abs/1609.04747 (June 2017). URL: http://arxiv.org/abs/1609.04747.

[80] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning Representations by Back-Propagating Errors". In: *Nature* 323.6088 (Oct. 1986), pp. 533–536. ISSN: 1476-4687. DOI: 10.1038/323533a0.

[81] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision* 115.3 (Dec. 2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[82] Rick Salay, Rodrigo Queiroz, and Krzysztof Czarnecki. "An Analysis of ISO 26262: Using Machine Learning Safely in Automotive Software". In: *CoRR* abs/1709.02435 (2017). URL: https://arxiv.org/abs/1709.02435.

[83] Timo Sämann, Peter Schlicht, and Fabian Hüger. "Strategy to Increase the Safety of a DNN-Based Perception for HAD Systems". In: *CoRR* abs/2002.08935 (2020). URL: https://arxiv.org/abs/2002.08935.

[84] Hendrik Schäbe. “Different Principles Used for Determination of Tolerable Hazard Rates”. In: *Materials of the Worrld Congress on Railway Research.* 2001, pp. 25–29. URL: http://www.railway-research.org/IMG/pdf/041.pdf.

[85] Gesina Schwalbe. “Concept Embedding Analysis: A Review”. In: (2021).

[86] **Gesina Schwalbe** and Martin Schels. “Strategies for Safety Goal Decomposition for Neural Networks”. In: *Extended Abstracts of 3rd ACM Comput. Science in Cars Symp.* Oct. 2019. DOI: 10.13140/RG.2.2.29392.74244.

[87] **Gesina Schwalbe** and Ute Schmid. “Concept Enforcement and Modularization for the ISO26262 Safety Case of Neural Networks”. In: *Proc. ECML PhD Forum 2019.* Sept. 2019. URL: https://ecmlpkdd2019.org/submissions/phdforum/.

[88] **Gesina Schwalbe** and Christian Wirth. *Hybrid Learning for DNNs.* https://github.com/continental/hybrid_learning. 2021. *Source code of developed framework for concept embedding analysis.*

[89] SCSC Assurance Case Working Group. *SCSC-141B: Goal Structuring Notation Community Standard.* Jan 2018. SCSC Assurance Case Working Group, Dec. 2018. URL: https://scsc.uk/scsc-141B.

[90] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. “On a Formal Model of Safe and Scalable Self-Driving Cars”. In: *CoRR* abs/1708.06374 (2017). URL: http://arxiv.org/abs/1708.06374.

[91] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *Proc. 3rd Int. Conf. Learning Representations.* 2015. URL: http://arxiv.org/abs/1409.1556.

[92] A. Srinivasan. *The Aleph Manual.* 2004. URL: https://www.cs.ox.ac.uk/activities/programinduction/Aleph/ (visited on 09/04/2021).

[93] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. “The German Traffic Sign Recognition Benchmark: A Multi-Class Classification Competition”. In: *Proc. 2011 Int. Joint Conf. Neural Networks.* Y. IEEE, July 2011, pp. 1453–1460. ISBN: 978-1-4244-9635-8. DOI: 10.1109/IJCNN.2011.6033395.

[94] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. “On the Importance of Initialization and Momentum in Deep Learning”. In: *Proc. 30th Int. Conf. Machine Learning.* PMLR, May 2013, pp. 1139–1147. URL: https://proceedings.mlr.press/v28/sutskever13.html.

[95] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. “Intriguing Properties of Neural Networks”. In: *Proc. 2nd Int. Conf. Learning Representations.* 2014. URL: http://arxiv.org/abs/1312.6199.

[96] Mingxing Tan, Ruoming Pang, and Quoc V. Le. “EfficientDet: Scalable and Efficient Object Detection”. In: *Proc. 2020 IEEE/CVF Conf. Comput. Vision and Pattern Recognition.* 2020, pp. 10781–10790.

[97] US Department of Defence. *MIL-STD-882E: Department of Defense Standard Practice: System Safety.* E. May 2021. URL: http://everyspec.com/MIL-STD/MIL-STD-0800-0899/MIL-STD-882E_41682/.

[98] Giulia Vilone and Luca Longo. “Explainable Artificial Intelligence: A Systematic Review”. In: *CoRR* abs/2006.00093 (Oct. 2020). URL: http://arxiv.org/abs/2006.00093.

[99] Laura von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Michal Walczak, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, Jochen Garcke, Christian Bauckhage, and Jannis Schuecker. “Informed Machine Learning - A Taxonomy and Survey of Integrating Prior Knowledge into Learning Systems”. In: *IEEE Trans. Knowl. Data Eng.* (2021), pp. 1–1. ISSN: 1558-2191. DOI: 10.1109/TKDE.2021.3079836.

[100] George H. Jr. Waltz. “Making the Death Seat Safer”. In: *Popular Science* (July 1950).

[101] Alvin Wan, Lisa Dunlap, Daniel Ho, Jihan Yin, Scott Lee, Suzanne Petryk, Sarah Adel Bargal, and Joseph E. Gonzalez. “NBDT: Neural-Backed Decision Tree”. In: *Posters 2021 Int. Conf. Learning Representations.* Sept. 2020. URL: https://openreview.net/forum?id=mCLVeEpplNE.

[102] Dan Wang, Xinrui Cui, and Z. Jane Wang. “CHAIN: Concept-Harmonized Hierarchical Inference Interpretation of Deep Convolutional Neural Networks”. In: *CoRR* abs/2002.01660 (2020). URL: https://arxiv.org/abs/2002.01660.

[103] Oliver Willers, Sebastian Sudholt, Shervin Raafatnia, and Stephanie Abrecht. “Safety Concerns and Mitigation Approaches Regarding the Use of Deep Learning in Safety-Critical Perception Tasks”. In: *Computer Safety, Reliability, and Security. SAFECOMP 2020 Workshops.* Lecture Notes in Computer Science. Springer International Publishing, 2020, pp. 336–350. ISBN: 978-3-030-55583-2. DOI: 10.1007/978-3-030-55583-2_25.

[104] Matthew Wood, Philipp Robbel, David Wittmann, Siyuan Liu, Yali Wang, Christian Knobel, David Boymanns, Sandro Syguda, Thomas Wiltschko, Neil Garbacik, Michael O‘Brien, Udo Dannebaum, Jack Weast, Bernd Dornieden, Michael Maass, Radboud Duintjer Tebbens, Marc Meijs, Mohamed Harb, Jonathon Reach, Karl Robinson, Toshika Srivastava, Mohamed Essayed Bouzouraa, Matthias Löhning, Bernhard Dehlink, Dirk Kaule, Richard Krüger, Jelena Frtunikj, Florian Raisch, Miriam Gruber, Jessica Steck, Julia Mejia-Hernandez, Pierre Blüher, Kamil Klonecki, Pierre Schnarz, Stefan Pukallus, Kai Sedlaczek, David Smerza, Dalong Li, Adam Timmons, Marco Bellotti, Michael Schöllhorn, Alan Tatourian, Philipp Schnet-

ter, Philipp Themann, Thomas Weidner, and Peter Schlicht. *Safety First for Automated Driving*. 2019. URL: https://www.aptiv.com/docs/default-source/white-papers/safety-first-for-automated-driving-aptiv-white-paper.pdf (visited on 08/29/2019).

[105] *WordNet: An Electronic Lexical Database*. Language, Speech, and Communication. A Bradford Book, May 1998. ISBN: 978-0-262-06197-1.

[106] Ernest Wozniak, Carmen Cârlan, Esra Acar-Celik, and Henrik J. Putzer. "A Safety Case Pattern for Systems with Machine Learning Components". In: *Computer Safety, Reliability, and Security. SAFECOMP 2020 Workshops*. Lecture Notes in Computer Science. Springer International Publishing, 2020, pp. 370–382. ISBN: 978-3-030-55583-2. DOI: 10.1007/978-3-030-55583-2_28.

[107] Weibin Wu, Yuxin Su, Xixian Chen, Shenglin Zhao, Irwin King, Michael R. Lyu, and Yu-Wing Tai. "Towards Global Explanations of Convolutional Neural Networks with Concept Attribution". In: *Proc. 2020 IEEE/CVF Conf. Comput. Vision and Pattern Recognition*. June 2020, pp. 8649–8658. DOI: 10.1109/CVPR42600.2020.00868.

[108] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. "Aggregated Residual Transformations for Deep Neural Networks". In: *Proc. 2017 IEEE Conf. Comput. Vision and Pattern Recognition*. IEEE Computer Society, 2017, pp. 5987–5995. DOI: 10.1109/CVPR.2017.634.

[109] Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. "On Completeness-Aware Concept-Based Explanations in Deep Neural Networks". In: *Advances in Neural Information Processing Systems 33*. Vol. 33. 2020, pp. 20554–20565.

[110] O. Zendel, M. Murschitz, M. Humenberger, and W. Herzner. "CV-HAZOP: Introducing Test Data Validation for Computer Vision". In: *Proc. 2015 IEEE Int. Conf. Comput. Vision*. Dec. 2015, pp. 2066–2074. DOI: 10.1109/ICCV.2015.239.

[111] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. "Understanding Deep Learning Requires Rethinking Generalization". In: *Proc. 5th Int. Conf. Learning Representations*. OpenReview.net, 2017. URL: https://openreview.net/forum?id=Sy8gdB9xx.

[112] Quanshi Zhang, Ruiming Cao, Feng Shi, Ying Nian Wu, and Song-Chun Zhu. "Interpreting CNN Knowledge via an Explanatory Graph". In: *Proc. 32nd AAAI Conf. Artificial Intelligence*. AAAI Press, 2018, pp. 4454–4463. URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17354.

[113] Quanshi Zhang, Wenguan Wang, and Song-Chun Zhu. “Examining CNN Representations with Respect to Dataset Bias”. In: *Proc. 32nd AAAI Conf. Artificial Intelligence.* AAAI Press, 2018, pp. 4464–4473. URL: `https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17429`.

[114] Ruihan Zhang, Prashan Madumal, Tim Miller, Krista A. Ehinger, and Benjamin I. P. Rubinstein. “Invertible Concept-Based Explanations for CNN Models with Non-Negative Concept Activation Vectors”. In: *Proc. 35th AAAI Conf. Artificial Intelligence.* Vol. 35. AAAI Press, 2021, pp. 11682–11690. URL: `https://ojs.aaai.org/index.php/AAAI/article/view/17389`.

[115] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. “Learning Deep Features for Discriminative Localization”. In: *Proc. 2016 IEEE Conf. Comput. Vision and Pattern Recognition.* IEEE Computer Society, 2016, pp. 2921–2929. DOI: `10.1109/CVPR.2016.319`.

[116] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. “Objects as Points”. In: *CoRR* abs/1904.07850 (Apr. 2019). URL: `http://arxiv.org/abs/1904.07850`.

[Th01] **Gesina Schwalbe**. “Verification of Size Invariance in DNN Activations Using Concept Embeddings”. In: *AIAI 2021: Artificial Intelligence Applications and Innovations.* IFIP Advances in Information and Communication Technology. Springer, 2021. DOI: `10.1007/978-3-030-79150-6_30`. Reproduced in this work with permission from Springer Nature.
(*details of contribution on p. 60; full text on pp. 233*)

[Th02] Johannes Rabold, **Gesina Schwalbe**, and Ute Schmid. “Expressive Explanations of DNNs by Combining Concept Analysis with ILP”. In: *KI 2020: Advances in Artificial Intelligence.* Lecture Notes in Computer Science. Springer International Publishing, 2020, pp. 148–162. DOI: `10.1007/978-3-030-58285-2_11`. Reproduced in this work with permission from Springer Nature.
(*details of contribution on p. 63; full text on pp. 246*)

[Th03] **Gesina Schwalbe** and Martin Schels. “Concept Enforcement and Modularization as Methods for the ISO 26262 Safety Argumentation of Neural Networks”. In: *Proc. 10th European Congress Embedded Real Time Software and Systems.* Jan. 2020. URL: `https://hal.archives-ouvertes.fr/hal-02442796`.
(*details of contribution on p. 91; full text on pp. 315*)

[Th04] **Gesina Schwalbe** and Martin Schels. “A Survey on Methods for the Safety Assurance of Machine Learning Based Systems”. In: *Proc. 10th European Congress Embedded Real Time Software and Systems.* Jan. 2020. URL: `https://hal.archives-ouvertes.fr/hal-02442819`.
(*details of contribution on p. 82; full text on pp. 293*)

[Th05] **Gesina Schwalbe**, Bernhard Knie, Timo Sämann, Timo Dobberphul, Lydia Gauerhof, Shervin Raafatnia, and Vittorio Rocco. "Structuring the Safety Argumentation for Deep Neural Network Based Perception in Automotive Applications". In: *Computer Safety, Reliability, and Security. SAFECOMP 2020 Workshops*. Springer International Publishing, 2020, pp. 383–394. DOI: 10.1007/978-3-030-55583-2_29. Reproduced in this work with permission from Springer Nature.
(*details of contribution on p. 88; full text on pp. 303*)

[Th06] **Gesina Schwalbe**, Christian Wirth, and Ute Schmid. "Enabling Verification of Deep Neural Networks in Perception Tasks Using Fuzzy Logic and Concept Embeddings". In: *CoRR* abs/2201.00572 (2022). URL: http://arxiv.org/abs/2201.00572. *Submitted to 2022 European Conference on Computer Vision (ECCV).*
(*details of contribution on p. 67; full text on pp. 261*)

[Th07] **Gesina Schwalbe** and Bettina Finzel. "A Comprehensive Taxonomy for Explainable Artificial Intelligence: A Systematic Survey of Surveys on Methods and Concepts". In: *CoRR* abs/2105.07190 (2021). URL: http://arxiv.org/abs/2105.07190. *Accepted with minor revisions at Special Issue on Explainable and Interpretable Machine Learning and Data Mining of Springer Nature Journal* Data Mining and Knowledge Discovery*, ISSN 1573-756X.*
(*details of contribution on p. 16; full text on pp. 128*)

[Th08] **Gesina Schwalbe**. "Concept Embedding Analysis: A Review". In: *CoRR* abs/2203.13909 (2022). URL: https://arxiv.org/abs/2203.13909. *Submitted to Springer Nature Journal Artificial Ingelligence Review Journal, ISSN 1573-7462.*
(*details of contribution on p. 17; full text on pp. 186*)

Part II.

Appendix

A. Update: Evidence Generation Methods for Safety Assurance of DNNs

The research area on methods for DNN model verification and improvement (direct or on system architectural level) has seen many interesting developments in the past years. In order to convey the up-to-date breadth of the topic, this supplemental material provides an update regarding the method surveys from my works [Th03, Th05] in Chap. 5.

To structure methods for the generation of safety evidence, one approach is to align with the traditional lifecycle and work products of the ISO 26262, i.e., by *when* evidence generation methods can be applied during development. As shown in [A32], many work product definitions must be adapted towards ML, but can still serve as a basis. This lifecycle-based sorting has been used in my survey [Th03], which gives the starting point for considerations about evidence methods in Sec. 5.4. The lifecycle-based approach is amended in my work [Th05] (cf. Sec. 5.6). In this chapter, the categorization from [Th05] is used that is instead based on the traditional pillars of evidence discussed in Subsec. 5.1.2. This corresponds to sorting evidence generation methods by *what* they shall achieve (or, in other words, *why* they are used): Improving the model (Sec. A.1), verifying or validating properties (Sec. A.2), or reducing the safety load of the ML model by applying system architectural measures (Sec. A.3). In the following, a shallow taxonomy based on these categories is highlighted by up-to-date method examples. A graphical overview is given in Figure A.1.

A.1. Measures During Development

To structure development methods, the main ingredients for DNN model creation are derived from a ML-based system model [A41]. This consists of: the learning content, especially the training data, the choice of prior model architecture, the training objective and routine, and—if applicable—any post-processing steps like quantization (not considered here).

Dataset Optimization The automated modeling of DNNs relies on training samples, training constraints, and the priors on the model architecture. Besides the labeling quality for the data, the data representativity including sufficient variance has a considerable influence on the final model quality. While careful data collection is mandatory to come by the data quality requirements, results can be improved further by targeted data augmentation.

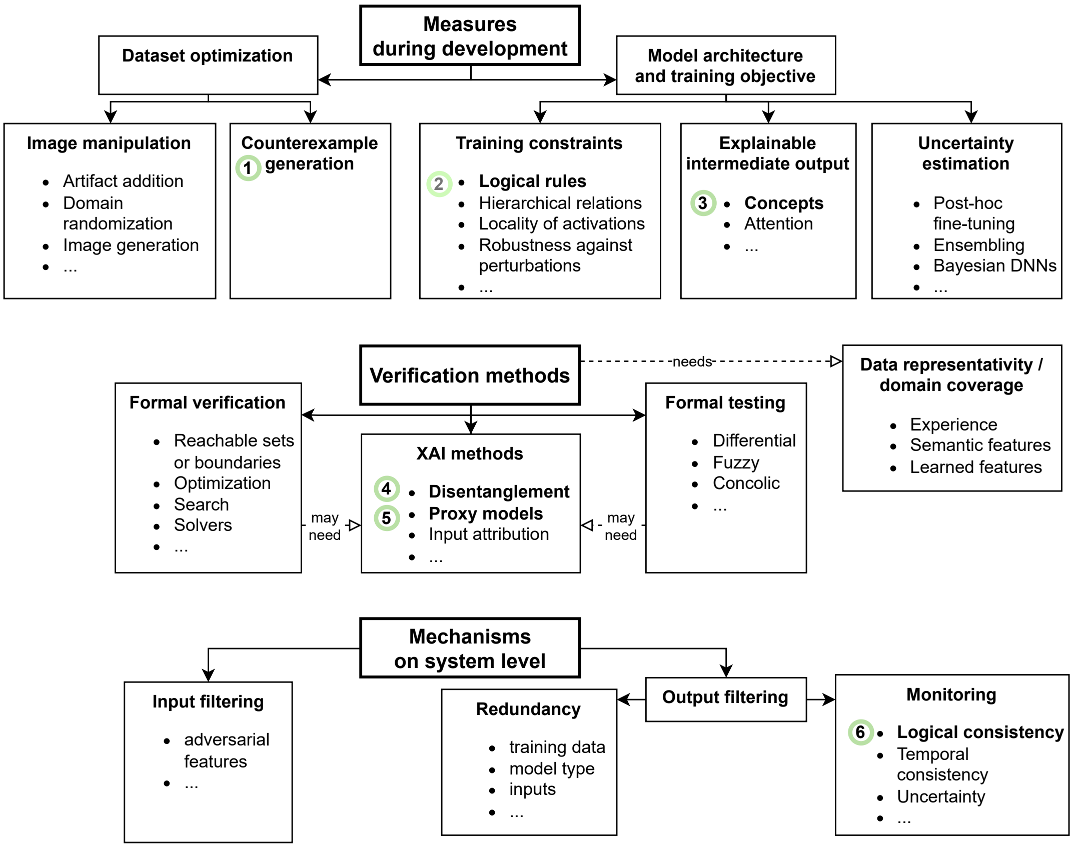


Figure A.1. Overview on categories of evidence generation methods summarized in Appendix A. Indicators (*circled numbers*) show where contributions of this thesis are allocated. For details on the contributions see Sec. 5.7. Depicted is the sub-structuring of the three pillars of evidence: Measures during development (*top*; Sec. A.1), offline verification methods (*middle*; Sec. A.2), and mechanisms on system level (*bottom*; Sec. A.3).

An extensive collection of data augmentation methods is given in [A37]. Approaches to enrich an image base are

- *Image manipulation* without access to a pretrained target model. Examples are
  - Addition of task specific realistic *artifacts* that can be caused by the environment (e.g., fog), the sensor (e.g., Gaussian noise), or the processing pipeline (e.g., compression artifacts). The CV-HAZOP database [A44] publicly collects relevant realistic perturbations for real world image CV applications.
  - *Domain randomization* [A39], i.e., random variations of input parts that are unrelated to the tasks like the background for object detection, or the texture of objects to avoid texture bias [A11];
  - *Image generation* via simulation can be used to augment a dataset by specific, manually crafted situations. This requires a suitable rendering engine such as [A6, A34], and a suitable ontology to construct the scenarios. An exemplary ontology for AD images was developed in the PEGASUS project [A4].
- *Counterexample generation* requires access to a pre-trained model and aims to find yet wrongly treated examples for re-training. As shown in [A8], targeted attacks can be designed using realistic perturbations. A white-box counterexample generation framework is, e.g., presented in [A7] that selects counterexamples from a semantic feature space by singular value decomposition and search. The method for corner case selection developed in my work [Th06] allows to find symbolic rules describing error modes. These can be used for synthesis or selection of corresponding counterexamples. Note that counterexample generation in general can also be used for verification whenever the absence of counterexamples should be proven.

Model Architecture and Training Objective As scribed in Subsec. 5.1.2, some highly desirable properties of the final model from safety perspective are inclusion of prior knowledge like a symbolic specification (cf. traditional assumptions collected in Sec. 5.2); verifiability [A19, Sec. 8.4.5]; and monitoring capabilities including proper uncertainty outputs of the DNN function [A19, Sec. 7.4.12]. These are discussed in the following:

- The training objective can help to include prior knowledge in the form of soft *training constraints*. Such prior knowledge can be, e.g.,
  - *Hierarchical relations*, like "cars are vehicles", which were used in the loss formulation in [A35]. They add super-class neurons to a classification DNN and use fuzzy logic to formulate the regularization terms enforcing the super-class relationship.
  - *Logical inter-dependencies*—or, more general, rules—formulated on symbolic intermediate outputs. This is used in [A17, A42]. Similar to [A35], [A17] distills complex Goedel fuzzy logic rules into DNNs using a student-teacher approach. In general, this could also be applied for complex spatial or temporal constraints.

  - *Locality of activations* for object-based tasks as suggested in [A45]. Single filters in a CNN are penalized to only activate locally in a spatial region of an image. This is based on the assumption that an object can be anchored to a spatial relation.
  - *Robustness against perturbations, temporal consistency.* To achieve this, [A21], e.g., add a self-supervision loss that penalizes deviation of activations when a sample is perturbed slightly. Similarly, [A40] applies a self-supervised temporal consistency loss to segmentation of videos. This penalizes deviation between a prediction and an estimate thereof that is smoothly obtained from the predictions on the previous frames.

- Verification of the model can substantially be eased by *explainable intermediate outputs*. Examples of such partly explainable models are (see also [Th07]):
  - Attention heatmap intermediate outputs as used in [A24]. They add a multiplicative masking module that learns to represent attention.
  - Concept predictions of concept bottleneck models as detailed in my survey [Th08]. Post-hoc attached concept outputs are used in my works [Th02, Th06].

- *Proper output uncertainty estimation* is a relevant property for online monitoring of the model outputs [A27]. This is needed to detect samples with high probability of incorrectness due to some lack of knowledge inherent to the DNN. Such may be samples that reside close to a decision boundary, or such that are in the long tail of the training distribution. The latter ones are "unknown" or "out-of-training-data-distribution" for the DNN model. Both types count as so-called epistemic uncertainties, i.e., uncertainty whether (locally) the DNN model itself is correct [A23]. In [A27] the authors emphasize the importance of propagating such uncertainty information through the system. In contrast to epistemic uncertainties, uncertainty in the DNN outputs can also indicate aleatoric uncertainty, i.e., noise inherent to the training data [A23]. This may, e.g., originate from inconsistent labels. Standard DNN confidence outputs are often badly calibrated without additional training measures [A12]. Concretely, they are often overly confident about their predictions. Such uncalibrated confidence outputs cannot be used to estimate the probability of correctness of the output. Some standard measures to ensure proper calibration are compared, e.g., in [A15]. Example measures include:
  - *Post-hoc fine-tuning* of weights, like done in the temperature scaling approach [A12] which linearly scales the last-layer weights towards better calibration metrics;
  - *Ensembling* of several models, e.g., via Monte-Carlo Dropout [A23], or the concept model ensembling suggested and used by me in [Th02] (see Sec. 3.5);
  - *Bayesian DNNs* [A10], such as the architecture suggested in [A10] where weights are replaced by Gaussian distributions over weights.

A.2. Offline Verification

One issue for sample-based verification of perception applications is providing representative test data and a sufficient test coverage. Traditional coverage criteria are experience and semantic feature coverage. In addition, in [Th05] I suggest also to consider coverage of the learned latent space as general targets.

Just as for traditional software modeling and development, formal and semi-formal methods have been developed to verify properties of DNNs. Due to the black-box property, a considerable effort is needed to access symbolic properties of DNN internal workings, either for (semi-)formal verification or inspection purposes. The research field of XAI aims at providing such methods.

Quantitative and Qualitative XAI Methods from XAI can help to uncover or access properties of DNNs like encoded concepts. When used qualitatively for manual inspection, XAI methods can assist in corner case and error mode identification. Practically, this is only useful if either further analysis steps, or mitigation methods can be derived from the corner cases. Successive analysis steps may be given, e.g., by rules for constructing test cases. And mitigation methods can be, e.g., data augmentation with corner cases, or an online monitoring mechanism for detecting the error case during runtime. Some interesting classes of XAI methods are (for details see my work [Th07]):

- *Input attribution methods* generally try to identify input features (e.g., pixels) with high influence on a single prediction. In the image domain, this often is modeled as heatmaps. These can be used for manual plausibility checks, or automatic monitoring of simple properties like locality of the attribution to detected objects. Examples are
  - *Gradient-based* approaches, like sensitivity analysis where the gradient is directly taken as attribution mask [A3];
  - *Back-propagation-based* methods that back-trace neuron activations from the output to the input, like LRP [A2] or PatternAttribution [A25];
  - *Perturbation-based* methods that record the influence of input changes via sampling, like LIME [A33], or D-RISE for object detection [A31].
- *Interpretable proxy models* aim to approximate a local or global behavior of the DNN in a human interpretable manner (cf. [A16, Sec. 5.4]). Examples of such interpretable models are logical rules as obtained in rule extraction [A14], or graphs of mined internal features as used in [A46]. This thesis introduces a rule extraction method that obtains approximate expressive rules from a DNN (see Sec. 4.3 and [Th02]).
- Safety assessment may require to give strong evidence about complex semantic properties. Proxy models may not be sufficiently faithful for strong evidence; and solely input

and output features may not be rich enough to check more complex properties. Thus, *disentanglement* of the distributed internal semantics of a DNN into human understandable concepts may be necessary. More precisely, one needs to localize human understandable concepts within the DNN representations, which then can be used to verify semantic properties. This is for example done in my works [Th01, Th06]. A core contribution of this thesis is to develop supervised CA-based methods for the automatic verification of semantic prior knowledge (cf. Chap. 3, Chap. 4).

Formal Verification Formal verification of DNNs is challenged by the large size of typical networks, but can also benefit from standard structures. For example, the layer structure of feedforward DNNs and the piecewise linearity of ReLU activation networks can be leveraged. An extensive survey on different types of formal verification methods specific for DNNs can be found in [A26]. Formal verification can be used, e.g., to find counterexamples to formal properties, find the validity range of a property, or find a reachable set. Most common approaches towards formal verification are:

- Estimation of a *reachable set or a boundary*, which can be done, e.g., using layer-by-layer induction, or by leveraging Lipschitz continuity of standard layers like done in DeepGo [A36];
- *Optimization* towards a solution (e.g., a counterexample for the DNN) possibly using linear constraints arising from piecewise-linear linear activations;
- Structured (exhaustive) *search* like the VeriVis approach [A30]; and
- Standard equation *solvers* that interpret piecewise-linear DNN as set of linear constraints that can be solved for like the Reluplex solvability modulo theory solver [A22].

Formal Testing Semi-formal methods for DNNs include *formal testing*. This refers to testing methods that include formalized and formally verifiable steps [A16, Sec. 8.1]. A survey on current methods with a focus on AD perception is provided in [A43]. Two typical coverage criteria that are used when formally deriving test cases are coverage of semantic dimensions, and coverage of DNN latent space dimensions. Semantic dimensions can be defined, e.g., using a scene description language such as SCENIC [A9] or OpenSCENARIO [A5, A1]. For reaching semantic coverage, samples can be used that accept semantic scene descriptions. For latent space coverage the sampling must take into account that usually not all points in a DNN latent space can be obtained from valid inputs. Therefore, sampling here usually is done locally around valid seeds. Examples of fuzzy testing methods are:

- *Differential testing* that tries to maximize the output difference between several models like used in [A29];

- *Fuzzy testing*, i.e., explorative sampling starting at valid seeds, as done, e.g., in TensorFuzz [A28]. TensorFuzz adds noise to existing samples and selects them in case they reveal errors or reside in yet unexplored latent space regions. DLFuzz [A13] tries to find local adversarial perturbations.

- *Concolic testing* [A38], i.e., exploration of the input space using symbolic constraints for test sample selection.

A.3. Mechanisms on System Level

Due to the complexity of the input space and DNNs themselves, offline verification will not be able to sufficiently guarantee safety for a high safety integrity level of the component, or more general, to guarantee an acceptable residual risk. Therefore, system architectural measures are inevitable that reduce risk from DNN errors, i.e., reduce the safety load and ASIL. While many standard system architectural measures are still applicable, some new methods can be tailored to specific DNN error modes and internals. Some examples are given below, categorized by filtering of inputs or outputs.

- *Filtering of inputs* or input features that may confuse the DNN, like filtering of adversarial features as done in [A18] and [A20];
- *Filtering of (parts of) the output* via additional control mechanisms, for example
  - *Redundancy*, e.g., with respect to training datasets, model architectures or model types, and sensor inputs;
  - *Monitoring*, i.e., meta-classification of the input regarding its potential of containing an error. This can be monitoring of uncertainty, temporal consistency between video frames, and logical consistency of outputs with intermediate outputs and other sensor outputs. The last is implemented by the logical consistency monitor developed in my work [Th06].

Bibliography

[A1] ASAM e.V. *ASAM OpenSCENARIO*. 1.0.0. OpenSCENARIO. ASAM e.V., 2020. URL: `https://releases.asam.net/OpenSCENARIO/1.0.0`.

[A2] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. “On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation”. In: *PLOS ONE* 10.7 (July 2015), e0130140. ISSN: 1932-6203. DOI: `10.1371/journal.pone.0130140`.

[A3] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. “How to Explain Individual Classification Decisions”. In: *J. Mach. Learn. Res.* 11 (Aug. 2010), pp. 1803–1831. ISSN: 1532-4435. URL: `http://portal.acm.org/citation.cfm?id=1859912`.

[A4] G. Bagschik, T. Menzel, and M. Maurer. “Ontology Based Scene Creation for the Development of Automated Vehicles”. In: *Proc. 2018 IEEE Intelligent Vehicles Symp.* IEEE, 2018, pp. 1813–1820. ISBN: 978-1-5386-4452-2. DOI: 10.1109/IVS.2018.8500632.

[A5] Werner Damm, Stephanie Kemper, Eike Möhlmann, Thomas Peikenkamp, and Astrid Rakow. “Using Traffic Sequence Charts for the Development of HAVs”. In: *Proc. 9th European Congress Embedded Real Time Systems.* Feb. 2018.

[A6] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. “CARLA: An Open Urban Driving Simulator”. In: *Conference on Robot Learning*. Oct. 2017, pp. 1–16. URL: http://proceedings.mlr.press/v78/dosovitskiy17a.html.

[A7] Tommaso Dreossi, Shromona Ghosh, Xiangyu Yue, Kurt Keutzer, Alberto L. Sangiovanni-Vincentelli, and Sanjit A. Seshia. “Counterexample-Guided Data Augmentation”. In: *Proc. 27th Int. Joint Conf. Artificial Intelligence.* ijcai.org, 2018, pp. 2071–2078. ISBN: 978-0-9992411-2-7. DOI: 10.24963/ijcai.2018/286.

[A8] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. “Robust Physical-World Attacks on Deep Learning Visual Classification”. In: *Proc. 2018 IEEE Conf. Computer Vision and Pattern Recognition.* IEEE Computer Society, 2018, pp. 1625–1634. DOI: 10.1109/CVPR.2018.00175.

[A9] Daniel J. Fremont, Tommaso Dreossi, Shromona Ghosh, Xiangyu Yue, Alberto L. Sangiovanni-Vincentelli, and Sanjit A. Seshia. “Scenic: A Language for Scenario Specification and Scene Generation”. In: *Proc. 40th ACM SIGPLAN Conf. Programming Language Design and Implementation.* PLDI 2019. Association for Computing Machinery, June 2019, pp. 63–78. ISBN: 978-1-4503-6712-7. DOI: 10.1145/3314221.3314633.

[A10] Jochen Gast and Stefan Roth. “Lightweight Probabilistic Deep Networks”. In: *Proc. 2018 IEEE Conf. Comput. Vision and Pattern Recognition.* IEEE Computer Society, 2018, pp. 3369–3378. DOI: 10.1109/CVPR.2018.00355.

[A11] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. “ImageNet-Trained CNNs Are Biased towards Texture; Increasing Shape Bias Improves Accuracy and Robustness”. In: *Proc. 7th Int. Conf. Learning Representations.* OpenReview.net, 2019. URL: https://openreview.net/forum?id=Bygh9j09KX.

[A12] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. “On Calibration of Modern Neural Networks”. In: *Proc. 2017 Int. Conf. Machine Learning*. PMLR, July 2017, pp. 1321–1330. URL: http://proceedings.mlr.press/v70/guo17a.html.

[A13] Jianmin Guo, Yu Jiang, Yue Zhao, Quan Chen, and Jiaguang Sun. "DLFuzz: Differential Fuzzing Testing of Deep Learning Systems". In: *Proc. ACM Joint Meeting on European Software Engineering Conf. and Symp. Foundations of Software Engineering*. ACM, 2018, pp. 739–743. DOI: 10.1145/3236024.3264835.

[A14] Tameru Hailesilassie. "Rule Extraction Algorithm for Deep Neural Networks: A Review". In: *CoRR* abs/1610.05267 (2016). URL: http://arxiv.org/abs/1610.05267.

[A15] Maximilian Henne, Adrian Schwaiger, Karsten Roscher, and Gereon Weiss. "Benchmarking Uncertainty Estimation Methods for Deep Learning with Safety-Related Metrics". In: *Proc. Workshop Artificial Intelligence Safety*. Vol. 2560. CEUR Workshop Proceedings. CEUR-WS.org, 2020, pp. 83–90. URL: http://ceur-ws.org/Vol-2560/paper35.pdf.

[A16] Sebastian Houben, Stephanie Abrecht, Maram Akila, Andreas Bär, Felix Brockherde, Patrick Feifel, Tim Fingscheidt, Sujan Sai Gannamaneni, Seyed Eghbal Ghobadi, Ahmed Hammam, Anselm Haselhoff, Felix Hauser, Christian Heinzemann, Marco Hoffmann, Nikhil Kapoor, Falk Kappel, Marvin Klingner, Jan Kronenberger, Fabian Küppers, Jonas Löhdefink, Michael Mlynarski, Michael Mock, Firas Mualla, Svetlana Pavlitskaya, Maximilian Poretschkin, Alexander Pohl, Varun Ravi-Kumar, Julia Rosenzweig, Matthias Rottmann, Stefan Rüping, Timo Sämann, Jan David Schneider, Elena Schulz, **Gesina Schwalbe**, Joachim Sicking, Toshika Srivastava, Serin Varghese, Michael Weber, Sebastian Wirkert, Tim Wirtz, and Matthias Woehrle. "Inspect, Understand, Overcome: A Survey of Practical Methods for AI Safety". In: *CoRR* abs/2104.14235 (Apr. 2021). URL: http://arxiv.org/abs/2104.14235.

[A17] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard H. Hovy, and Eric P. Xing. "Harnessing Deep Neural Networks with Logic Rules". In: *Proc. 54th Annu. Meeting of the Association for Computational Linguistics*. Vol. 1: Long Papers. The Association for Computer Linguistics, 2016. ISBN: 978-1-945626-00-5. URL: http://aclweb.org/anthology/P/P16/P16-1228.pdf.

[A18] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. "Adversarial Examples Are Not Bugs, They Are Features". In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 125–136. URL: http://papers.nips.cc/paper/8307.pdf.

[A19] ISO/TC 22/SC 32. *ISO 26262-6:2018(En): Road Vehicles — Functional Safety — Part 6: Product Development at the Software Level*. Second. Vol. 6. ISO 26262:2018(En). International Organization for Standardization, Dec. 2018. URL: https://www.iso.org/standard/68388.html.

[A20] Nikhil Kapoor, Andreas Bär, Serin Varghese, Jan David Schneider, Fabian Hüger, Peter Schlicht, and Tim Fingscheidt. "From a Fourier-Domain Perspective on Adversarial Examples to a Wiener Filter Defense for Semantic Segmentation". In: *Proc. 2021 Int. Joint Conf. Neural Networks*. July 2021, pp. 1–8. DOI: 10.1109/IJCNN52387.2021.9534145.

[A21] Nikhil Kapoor, Chun Yuan, Jonas Löhdefink, Roland Zimmerman, Serin Varghese, Fabian Hüger, Nico Schmidt, Peter Schlicht, and Tim Fingscheidt. "A Self-Supervised Feature Map Augmentation (FMA) Loss and Combined Augmentations Finetuning to Efficiently Improve the Robustness of CNNs". In: *Computer Science in Cars Symposium*. ACM, Dec. 2020, pp. 1–8. ISBN: 978-1-4503-7621-1. DOI: 10.1145/3385958.3430477.

[A22] Guy Katz, Clark Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. "Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks". In: *Proc. 29th Int. Conf. Comput. Aided Verification*. Lecture Notes in Computer Science. Springer International Publishing, 2017, pp. 97–117. ISBN: 978-3-319-63387-9. DOI: 10.1007/978-3-319-63387-9_5.

[A23] Alex Kendall and Yarin Gal. "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?" In: *Advances in Neural Information Processing Systems 30*. 2017, pp. 5580–5590. URL: http://papers.nips.cc/paper/7141.pdf.

[A24] Jinkyu Kim and John F. Canny. "Interpretable Learning for Self-Driving Cars by Visualizing Causal Attention". In: *Proc. 2017 IEEE Int. Conf. Comput. Vision*. IEEE Computer Society, 2017, pp. 2961–2969. DOI: 10.1109/ICCV.2017.320.

[A25] Pieter-Jan Kindermans, Kristof T. Schütt, Maximilian Alber, Klaus-Robert Müller, Dumitru Erhan, Been Kim, and Sven Dähne. "Learning How to Explain Neural Networks: PatternNet and PatternAttribution". In: *Proc. 6th Int. Conf. on Learning Representations*. Feb. 2018. URL: https://openreview.net/forum?id=Hkn7CBaTW.

[A26] Changliu Liu, Tomer Arnon, Christopher Lazarus, Christopher Strong, Clark Barrett, and Mykel J. Kochenderfer. "Algorithms for Verifying Deep Neural Networks". In: *Foundations and Trends® in Optimization* 4.3-4 (Feb. 2021), pp. 244–404. ISSN: 2167-3888, 2167-3918. DOI: 10.1561/2400000035.

[A27] Rowan McAllister, Yarin Gal, Alex Kendall, Mark van der Wilk, Amar Shah, Roberto Cipolla, and Adrian Weller. "Concrete Problems for Autonomous Vehicle Safety: Advantages of Bayesian Deep Learning". In: *Proc. 26th Int. Joint Conf. Artificial Intelligence*. 2017, pp. 4745–4753. DOI: 10.24963/ijcai.2017/661.

[A28] Augustus Odena, Catherine Olsson, David Andersen, and Ian Goodfellow. "TensorFuzz: Debugging Neural Networks with Coverage-Guided Fuzzing". In: *Proc. 36th Int. Conf. on Machine Learning*. Proceedings of Machine Learning Research.

May 2019, pp. 4901–4911. URL: http://proceedings.mlr.press/v97/odena19a.html.

[A29] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. “DeepXplore: Automated Whitebox Testing of Deep Learning Systems”. In: *Proc. 26th Symp. Operating Systems Principles*. Vol. abs/1705.06640. ACM, 2017, pp. 1–18. ISBN: 978-1-4503-5085-3. DOI: 10.1145/3132747.3132785.

[A30] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. “Towards Practical Verification of Machine Learning: The Case of Computer Vision Systems”. In: *CoRR* abs/1712.01785 (2017). URL: http://arxiv.org/abs/1712.01785.

[A31] Vitali Petsiuk, Rajiv Jain, Varun Manjunatha, Vlad I. Morariu, Ashutosh Mehra, Vicente Ordonez, and Kate Saenko. “Black-Box Explanation of Object Detectors via Saliency Maps”. In: *Proc. 2021 IEEE/CVF Conf. Comput. Vision and Pattern Recognition*. 2021, pp. 11443–11452.

[A32] Krystian Radlak, Piotr Serwa, Michal Szczepankiewicz, and Tim Jones. “Organization of ML-Based Product Development as per ISO 26262”. In: *CoRR* abs/1910.05112 (Oct. 2019). URL: http://arxiv.org/abs/1910.05112.

[A33] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. “”Why Should I Trust You?”: Explaining the Predictions of Any Classifier”. In: *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. KDD ’16. ACM, 2016, pp. 1135–1144. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939778.

[A34] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. “Playing for Data: Ground Truth from Computer Games”. In: *Computer Vision – ECCV 2016*. Lecture Notes in Computer Science. Springer International Publishing, 2016, pp. 102–118. ISBN: 978-3-319-46475-6. DOI: 10.1007/978-3-319-46475-6_7.

[A35] Soumali Roychowdhury, Michelangelo Diligenti, and Marco Gori. “Image Classification Using Deep Learning and Prior Knowledge”. In: *Workshops of the 32nd AAAI Conf. Artificial Intelligence*. Vol. WS-18. AAAI Workshops. AAAI Press, June 2018, pp. 336–343. ISBN: 978-1-57735-801-5. URL: https://aaai.org/ocs/index.php/WS/AAAIW18/paper/view/16575.

[A36] Wenjie Ruan, Xiaowei Huang, and Marta Kwiatkowska. “Reachability Analysis of Deep Neural Networks with Provable Guarantees”. In: *Proc. 27th Int. Joint Conf. Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, July 2018, pp. 2651–2659. DOI: 10.24963/ijcai.2018/368.

[A37] Connor Shorten and Taghi M. Khoshgoftaar. “A Survey on Image Data Augmentation for Deep Learning”. In: *Journal of Big Data* 6.1 (July 2019), p. 60. ISSN: 2196-1115. DOI: 10.1186/s40537-019-0197-0.

[A38] Youcheng Sun, Min Wu, Wenjie Ruan, Xiaowei Huang, Marta Kwiatkowska, and Daniel Kroening. "Concolic Testing for Deep Neural Networks". In: *Proc. 33rd ACM/IEEE Int. Conf. Automated Software Engineering*. ACM, 2018, pp. 109–119. DOI: 10.1145/3238147.3238172.

[A39] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. "Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World". In: *Proc. 2017 IEEE/RSJ Int. Conf. Intelligent Robots and Systems*. Sept. 2017, pp. 23–30. DOI: 10.1109/IROS.2017.8202133.

[A40] Serin Varghese, Sharat Gujamagadi, Marvin Klingner, Nikhil Kapoor, Andreas Bär, Jan David Schneider, Kira Maag, Peter Schlicht, Fabian Hüger, and Tim Fingscheidt. "An Unsupervised Temporal Consistency (TC) Loss to Improve the Performance of Semantic Segmentation Networks". In: *2021 IEEE/CVF Conf. Comput. Vision and Pattern Recognition Workshops*. June 2021, pp. 12–20. DOI: 10.1109/CVPRW53098.2021.00010.

[A41] Stefan Voget, Alexander Rudolph, and Jürgen Mottok. "A Consistent Safety Case Argumentation for Artificial Intelligence in Safety Related Automotive Systems". In: *Proc. 9th European Congress Embedded Real Time Systems*. Jan. 2018. URL: https://hal.archives-ouvertes.fr/hal-02156048.

[A42] Hu Wang. "ReNN: Rule-Embedded Neural Networks". In: *Proc. 24th Int. Conf. Pattern Recognition*. IEEE Computer Society, 2018, pp. 824–829. ISBN: 978-1-5386-3788-3. DOI: 10.1109/ICPR.2018.8545379.

[A43] Matthias Woehrle, Christoph Gladisch, and Christian Heinzemann. "Open Questions in Testing of Learned Computer Vision Functions for Automated Driving". In: *Computer Safety, Reliability, and Security*. Lecture Notes in Computer Science. Springer International Publishing, 2019, pp. 333–345. ISBN: 978-3-030-26250-1. DOI: 10.1007/978-3-030-26250-1_27.

[A44] O. Zendel, M. Murschitz, M. Humenberger, and W. Herzner. "CV-HAZOP: Introducing Test Data Validation for Computer Vision". In: *Proc. 2015 IEEE Int. Conf. Comput. Vision*. Dec. 2015, pp. 2066–2074. DOI: 10.1109/ICCV.2015.239.

[A45] Q. Zhang, Y. N. Wu, and S. Zhu. "Interpretable Convolutional Neural Networks". In: *Proc. 2018 IEEE/CVF Conf. Comput. Vision and Pattern Recognition*. IEEE, June 2018, pp. 8827–8836. ISBN: 978-1-5386-6420-9. DOI: 10.1109/CVPR.2018.00920.

[A46] Quanshi Zhang, Ruiming Cao, Feng Shi, Ying Nian Wu, and Song-Chun Zhu. "Interpreting CNN Knowledge via an Explanatory Graph". In: *Proc. 32nd AAAI Conf. Artificial Intelligence*. AAAI Press, 2018, pp. 4454–4463. URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17354.

[Th01] **Gesina Schwalbe**. "Verification of Size Invariance in DNN Activations Using Concept Embeddings". In: *AIAI 2021: Artificial Intelligence Applications and Innovations*. IFIP Advances in Information and Communication Technology. Springer, 2021. DOI: 10.1007/978-3-030-79150-6_30. Reproduced in this work with permission from Springer Nature.
(*details of contribution on p. 60; full text on pp. 233*)

[Th02] Johannes Rabold, **Gesina Schwalbe**, and Ute Schmid. "Expressive Explanations of DNNs by Combining Concept Analysis with ILP". In: *KI 2020: Advances in Artificial Intelligence*. Lecture Notes in Computer Science. Springer International Publishing, 2020, pp. 148–162. DOI: 10.1007/978-3-030-58285-2_11. Reproduced in this work with permission from Springer Nature.
(*details of contribution on p. 63; full text on pp. 246*)

[Th03] **Gesina Schwalbe** and Martin Schels. "Concept Enforcement and Modularization as Methods for the ISO 26262 Safety Argumentation of Neural Networks". In: *Proc. 10th European Congress Embedded Real Time Software and Systems*. Jan. 2020. URL: https://hal.archives-ouvertes.fr/hal-02442796.
(*details of contribution on p. 91; full text on pp. 315*)

[Th05] **Gesina Schwalbe**, Bernhard Knie, Timo Sämann, Timo Dobberphul, Lydia Gauerhof, Shervin Raafatnia, and Vittorio Rocco. "Structuring the Safety Argumentation for Deep Neural Network Based Perception in Automotive Applications". In: *Computer Safety, Reliability, and Security. SAFECOMP 2020 Workshops*. Springer International Publishing, 2020, pp. 383–394. DOI: 10.1007/978-3-030-55583-2_29. Reproduced in this work with permission from Springer Nature.
(*details of contribution on p. 88; full text on pp. 303*)

[Th06] **Gesina Schwalbe**, Christian Wirth, and Ute Schmid. "Enabling Verification of Deep Neural Networks in Perception Tasks Using Fuzzy Logic and Concept Embeddings". In: *CoRR* abs/2201.00572 (2022). URL: http://arxiv.org/abs/2201.00572. *Submitted to 2022 European Conference on Computer Vision (ECCV).*
(*details of contribution on p. 67; full text on pp. 261*)

[Th07] **Gesina Schwalbe** and Bettina Finzel. "A Comprehensive Taxonomy for Explainable Artificial Intelligence: A Systematic Survey of Surveys on Methods and Concepts". In: *CoRR* abs/2105.07190 (2021). URL: http://arxiv.org/abs/2105.07190. *Accepted with minor revisions at Special Issue on Explainable and Interpretable Machine Learning and Data Mining of Springer Nature Journal* Data Mining and Knowledge Discovery, *ISSN 1573-756X.*
(*details of contribution on p. 16; full text on pp. 128*)

[Th08] **Gesina Schwalbe**. "Concept Embedding Analysis: A Review". In: *CoRR* abs/2203.13909 (2022). URL: https://arxiv.org/abs/2203.13909. *Submitted to Springer*

Nature Journal Artificial Ingelligence Review Journal, ISSN 1573-7462.
(details of contribution on p. 17; full text on pp. 186)

B. Full Texts of Papers

In the following, the original full text versions are given of publications that are part of this work. Find an overview on the appended papers in the publications overview on pp. xv.

Attached Full Texts

A Comprehensive Taxonomy for Explainable Artificial Intelligence: A Systematic Survey of Surveys on Methods and Concepts

Gesina Schwalbe[1,2][0000-0003-2690-2478] and Bettina Finzel[2][0000-0002-9415-6254]

[1] Continental AG, Regensburg, Germany
gesina.schwalbe@continental-corporation.com
[2] Cognitive Systems Group, University of Bamberg, Germany
bettina.finzel@uni-bamberg.de

Abstract. In the meantime, a wide variety of terminologies, motivations, approaches and evaluation criteria have been developed within the research field of explainable artificial intelligence (XAI). With the amount of XAI methods vastly growing, a taxonomy of methods is needed by researchers as well as practitioners: To grasp the breadth of the topic, compare methods, and to select the right XAI method based on traits required by a specific use-case context. In the literature many taxonomies for XAI methods of varying level of detail and depth can be found. While they often have a different focus, they also exhibit many points of overlap. This paper unifies these efforts, and provides a taxonomy of XAI methods that is complete with respect to notions present in the current state-of-research. In a structured literature analysis and meta-study we identified and reviewed more than 50 of the most cited and current surveys on XAI methods, metrics, and method traits. After summarizing them in a survey of surveys, we merge terminologies and concepts of the articles into a unified structured taxonomy. Single concepts therein are illustrated by in total more than 50 diverse selected example methods, which we categorize accordingly. The taxonomy may serve both beginners, researchers, and practitioners as a reference and wide-ranging overview on XAI method traits and aspects. Hence, it provides foundations for targeted, use-case-oriented, and context-sensitive future research.

Keywords: Explainable Artificial Intelligence · Interpretability · Taxonomy · Meta-Analysis · Survey-of-surveys · Review

1 Introduction

Machine learning models offer the great benefit that they can deal with hardly specifiable problems as long as these can be exemplified by data samples. This has opened up a lot of opportunities for promising automation and assistance systems, like highly automated driving, medical assistance systems, text summaries and question-answer systems, just to name a few. However, many types of models that are automatically learned from data will not only exhibit high performance, but also be black-box—*i.e.*, they hide information on the learning progress, internal representation, and final processing in a format not or hardly interpretable by humans.

There are now diverse use-case specific motivations for allowing humans to *understand* a given software component, *i.e.*, to build up a mental model approximating the algorithm in a certain way. This starts with legal reasons, like the General Data Protection Regulation [56] adopted by the European Union in recent years. Another example are domain specific standards, like the functional safety standard ISO 26262 [69] requiring assessibility of software components in safety critical systems. This is even detailed to an explicit requirement for explainability of machine learning based components in the ISO/TR 4804 [68] draft standard. Many further reasons of public interest, like fairness or security, as well as business interests like ease of debugging, knowledge retrieval, or appropriate user trust have been identified [9,82]. This need to translate behavioral or internal aspects of black-box algorithms into a human interpretable form gives rise to the broad research field of explainable artificial intelligence (XAI).

In recent years, the topic of XAI methods has received an exponential boost in research interest [9,85,1,151]. For practical application of XAI in human-AI interaction systems, it is important to ensure a choice of XAI method(s) appropriate for the corresponding use-case. Without question, thorough use-case analysis including the main goal and derived requirements is one essential ingredient here [82]. Nevertheless, we argue that a necessary foundation for choosing correct requirements is a complete knowledge of the different *aspects* (traits, properties) of XAI methods that may influence their applicability. Well-known aspects are, *e.g.*, portability, *i.e.*, whether the method requires access to the model internals or not, or locality, *i.e.*, whether single predictions of a model are explained or some global properties. As will become clear from our literature analysis in section 5, this only just scratches the surface of aspects of XAI methods that are relevant for practical application.

This paper aims to (1) help beginners in gaining a good initial overview and starting point for a deeper dive, (2) support practitioners seeking a categorization scheme for choosing an appropriate XAI method for their use-case, and (3) to assist researchers in identifying desired combination of aspects that have not or little been considered so far. For this, we provide a complete collection and a structured overview in the form of a taxonomy of XAI method aspects in section 5, together with method examples for each aspect. The method aspects are obtained from an extensive literature survey on categorization schemes for explainability and interpretability methods, resulting in a meta-study on XAI surveys presented in section 4. Other than similar work, we do not aim to provide a survey on XAI methods. Hence, our taxonomy is not constructed as a means of a sufficient chapter scheme. Instead, we try to compile a taxonomy that is complete with respect to existing valuable work. We believe that this gives a good starting point for in-depth understanding of sub-topics of XAI research, and research on XAI methods themselves.

Our main contributions are:

- *Complete XAI Method Taxonomy:* A structured, detailed, and deep taxonomy of XAI method aspects (see Figure 7); In particular, the taxonomy is complete with respect to application relevant XAI method and evaluation aspects that have so far been considered in literature, according to our structured literature search.
- *Extensive XAI Method Meta-study:* A survey-of-surveys of more than 50 works on XAI related topics that may serve to pick the right starting point for a deeper dive into XAI and XAI sub-fields. To our knowledge, this is the most extensive and detailed meta-study specifically on XAI and XAI methods available so far.
- *Broad XAI Method Review:* A large collection, review, and categorization of more than 50 hand-picked diverse XAI methods (see Table 6). This evidences practical applicability of our taxonomy structure and the identified method aspects.

The rest of the paper reads as follows: In section 2 we reference some related work, recapitulate major milestones in the history of XAI and provide important definitions of terms. This is meant for those readers less familiar with the topic of XAI. After that, we detail our systematic review approach in section 3. The results of the review are split into the following chapters: The detailed review of the selected surveys is presented in section 4, including their value for different audiences and research focus; and section 5 details collected XAI method aspects and our proposal of a taxonomy thereof. Each considered aspect is accompanied by illustrative example methods, a summary of which can be found in Table 6. We conclude our work in section 6.

2 Background

This chapter gives some background regarding related work (subsection 2.1), and the history of XAI including its main milestones (subsection 2.2). Lastly, subsection 2.3 introduces some basic terms and definitions used throughout this work. Experienced readers may skip the respective subsections.

2.1 Related Work

Meta-studies on XAI Methods Short meta-studies collecting surveys of XAI methods are ofthen contained in the related work section of XAI reviews like [85, Sec. 2.3]. These are, however, by nature restricted in length and usually concentrate on most relevant and cited reference works like [52,1,9]. We here instead consider a very broad and extensive collection of surveys. By now, also few dedicated meta-studies can be found in literature. One is the comprehensive and recent systematic literature survey conducted by Vilone et al. [134, Chap. 4]. This reviews 53 survey articles related to XAI topics, which are classified differentially according to their focus. Their literature analysis reaches further back in time and targets even more general topics in XAI than ours. Hence, several of their included surveys are very specific, quite short, and do not provide any structured notions of taxonomies. Also, the focus of their review lies in research topics, while the slightly more detailed survey-of-surveys presented here also takes into account aspects that relate to suitability for beginners and practitioners. Another dedicated survey-of-surveys is the one by Chatzimparmpas et al. [25]. With 18 surveys on visual XAI this is smaller compared to ours, and has quite a different focus: They review the intersection of XAI with visual analytics. Lastly, due to their publication date they only include surveys from 2018 or earlier, which misses on important recent work as will be seen in Figure 2. The same holds for the very detailed DARPA report [95] from 2019. Hence, both papers miss many recent works covered in our meta-study (cf. Figure 2). Further, the DARPA report has a very broad focus, also including sibbling research fields related to XAI. And, similar to Vilone et al., they target researchers, hence give no estimation of what work is especially suitable for beginners or practitioners.

(Meta-)Studies on XAI Method Traits Related work on taxonomies is mostly shallow, focused on a sub-field, or is purely functional in the sense that taxonomies merely serve as a survey chapter structure. A quite detailed, but less structured collection of XAI method traits can be found in the courses of discussion in [98,23,20,95,84]. The focus of each will be discussed later in section 4. But despite their detail, we found that each article features unique aspects that are not included in the others. The only systematic meta-review on XAI method traits known to us is the mentioned survey by Vilone et al. [134]. They, however, generally review notions related to XAI, not concretely XAI method traits. Only a shallow taxonomy is derived from parts of the retrieved notions (cf. [134, Sec. 5]).

Use-case Analysis A related, wide and important field which is out-of-scope of this paper is that of use-case and requirements analysis. This, *e.g.*, includes analysis of the background of the explanation receiver [92]. Instead, we here concentrate on finding aspects of the XAI methods themselves that may be used for requirements analysis and formulation, *e.g.*, whether the explainability method must be model-agnostic or the amount of information conveyed to the receiver must be small. For further detail on use-case specific aspects the reader may be referred to one of the following surveys further discussed in section 4 [92,57,53,82,9].

2.2 History of XAI

XAI is not a new topic, although the number of papers published in recent years might suggest it. The abbreviation XAI for the term explainable artificial intelligence was first used in 2004 in [132] (see [23, Sec. 2.3]). According to [16], the first mention of the underlying concept already dates back to the year 1958, when McCarthy described his idea of how to realize AI systems. In his seminal paper he promoted a declarative approach, based on a formal language and a problem-solving algorithm that operates on data represented in the given formal language. Such an approach could be understood by humans and the system's behaviour could be adapted, if necessary [91]. Basically, McCarthy described the idea of inherently transparent systems that would be explainable by design.

Inherently transparent approaches paved the way for expert systems that were designed to support human decision-makers (see [70]). Due to the big challenge to integrate, often implicit, expert knowledge, these systems lost their popularity in the 90's of the last century. Meanwhile, neural networks have become a new and powerful approach to solve sub-symbolic problems. The first approaches aiming at making neural network decisions transparent for debugging purposes date back to the mid 90's. For example in 1992, Craven and Shavlik presented several methods to visualize numerical data, such as decision surfaces [31].

Due to the introduction of the new European General Data Protection Regulation (GDPR) in May 2018, transparency of complex and opaque approaches, such as neural networks, took on a new meaning. Researchers and companies started to develop new AI frameworks, putting more emphasis on the aspect of accountability and the "right of explanation" [56,30]. Besides debugging and making decisions transparent to developers or end-users, decisions of a system now also had to be comprehensible for further stakeholders. According to [1] the term XAI gained popularity in the research community after the Defense Advanced Research Projects Agency (DARPA) published its paper about explainable artificial intelligence, see [58].

In their definition, XAI efforts aim for two main goals. The first one is to create machine learning techniques that produce models that can be explained (their decision-making process as well as the output), while maintaining a high level of learning performance. The second goal is to convey a user-centric approach, in order to enable humans to understand their artificial counterparts. As a consequence, XAI aims for increasing the trust in learned models and to allow for an efficient partnership between human and artificial agents [58]. In order to reach the first goal, DARPA proposes three strategies: deep explanation, interpretable models and model induction, which are defined in subsection 2.3. Among the most prominent XAI methods that implement this goal for deep learning, especially in computer vision, are for example LIME [112], LRP [12] and Grad-CAM [121]. The second, more user-centric, goal defined by DARPA requires a highly inter-disciplinary perspective. This is based on fields such as computer science, social sciences as well as psychology in order to produce more explainable models, suitable explanation interfaces, and to communicate explanations effectively under consideration of psychological aspects.

In 2019 Miller [92], *e.g.*, made an important contribution to the implementation of the user-centric perspective with his paper on artificial intelligence from the viewpoint of the social sciences. He considers philosophical, psychological, and interaction-relevant aspects of explanation, examining different frameworks and requirements. An important turn toward a user-centric focus of XAI was also supported by Rudin [114] in her paper from 2019, where she argues for using inherently interpretable models instead of opaque models, motivated by the right to explanation and the societal impact of intransparent models.

Another milestone in the development of XAI is the turn toward evaluation metrics for explanations [96]. The XAI community now acknowledges more in depth that it is not enough to generate explanations, but that it is also crucial to evaluate how good they are with respect to some formalized measure.

2.3 Basic Definitions

This section introduces some basic terms related to XAI which are used throughout this paper. Detailed definitions of the identified XAI method aspects will be given in section 5.

Important work that is concerned with definitions for XAI can be found in, *e.g.*, Lipton's paper [86], Adadi & Berrada [1], and the work of Doshi-Velez & Kim from 2017 [39] who are often cited as base work for formalizing XAI terms (cf. *e.g.* [85]). Some definitions are taken from Bruckert et al. [19], who present explanation as a process involving recognition, understanding and explicability respectively explainability, as well as interpretability. An overview is given in Fig. 1.

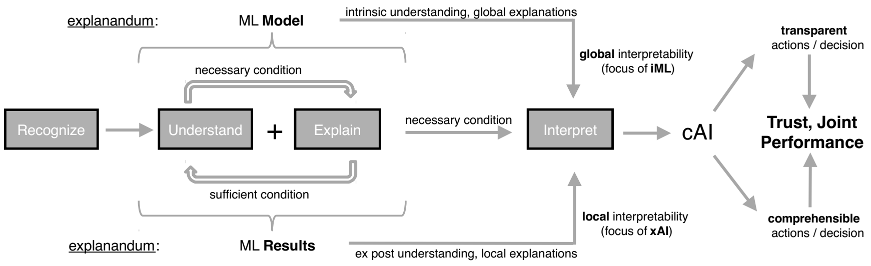

explanandum:
ML Model
intrinsic understanding, global explanations
necessary condition
global interpretability (focus of iML)
transparent actions / decision
Recognize
Understand
+
Explain
necessary condition
Interpret
cAI
Trust, Joint Performance
sufficient condition
local interpretability (focus of xAI)
comprehensible actions / decision
explanandum:
ML Results
ex post understanding, local explanations

Fig. 1. A framework for comprehensible artificial intelligence [19].

In the following we will assume we are given a system that should (partly) be explained to a human. We use the following nomenclature:

Understanding is described as the human ability to recognize correlations, as well as the context of a problem and is a necessary precondition for explanations [19]. The concept of understanding can be divided into mechanistic understanding (*"How does something work?"*) and functional understanding (*"What is its purpose?"*) [103].

Explicability refers to making properties of an AI system inspectable [19].

Explainability goes one step further than *explicability* and aims for making (a) the context of an AI system's reasoning, (b) the model, or (c) the evidence for a decision output accessible, such that they can be *understood* by a human [19].

Transparency is fulfilled by an AI system, if its algorithmic behaviour with respect to decision outputs or processes can be *understood* by a human *mechanistically* [103]. Transparency will be discussed more closely in section 5.1.

Explaining means utilizing *explicability* or *explainability* to allow a human to *understand* a model and its purpose [19,103].

Global explanations *explain* the model and its logic as a whole ("How was the conclusion derived?").

Local explanations *explain* individual decisions or predictions ("Why was this example classified as a car?").

Interpretability means that an AI system's decision can be *explained globally* or *locally* (with respect to *mechanistic understanding*), and that the system's purpose can be *understood* by a human actor [103](*i.e. functional understanding*).

Correctability means that an AI system can be adapted by a human actor in a targeted manner in order to ensure correct decisions [80,129,119].

Interactivity applies if it is possible to incrementally explore the internal working of a model, and to adapt it (*correctability*). This is a contrast to *local* and *global interpretability*, which refers to presenting decision outputs and paths.

Comprehensibility relies, similar to *interpretability*, on local and global *explanations* and *functional understanding.* Additionally, *comprehensible* artificial intelligence fulfills *interactivity* [19,119]. Both, *interpretable* presentation and intervention are considered as important aspects for in depth *understanding* and therefore preconditions to *comprehensibility* (see also [53]).

Human-AI system is a system that contains both algorithmic components and a human actor, which have to cooperate for achieving a goal. We here consider in specific **explanation systems**, *i.e.*, such human-AI systems in which the cooperation involves *explanations* about an algorithmic part of the system (the *explanandum*) by an *explanator* component, to the human interaction partner (the *explainee*) resulting in an action of the human.

Explanandum (*what is to be explained*, cf. subsection 5.1) means the complete oracle to be *explained* in an *explanation system.* This usually encompasses a model (*e.g.*, a deep neural network), which may or may not encompass the actual object of explanation.

Explanator (*the one who explains*, cf. subsection 5.2) is the *explanation system* component providing *explanations.*

Explainee *(the one to whom is explained)* is the receiver of the *explanations* in the *explanation system.* Note that this often but not necessarily is a human. *Explanations* may also be used *e.g.*, in multi-agent systems for communication between the agents and without a human in the loop in most of the information exchange scenarios.

Interpretable models are defined as machine learning techniques that learn more structured representations, or that allow for tracing causal relationships. They are *inherently interpretable* (cf. definition in subsection 5.2), *i.e.*, no additional methods need to be applied to *explain them*, unless the strcutured representations or relationship are too complex to be processed by a human actor at hand.

Interpretable machine learning (iML) is the area of research concerned with the creation of *interpretable* AI systems (*interpretable models*).

Model induction (also called model distillation, student-teacher approach, or reprojection [53]) is a strategy that summarizes techniques which are used to infer an approximate *explainable* model—the (*explainable*) *proxy* or *surrogate model*—by observing the input-output behaviour of a model that is *explained*.

Deep explanation refers to combining deep learning with other methods in order to create hybrid systems that produce richer representations of what a deep neural network has learned, and that enable extraction of underlying semantic concepts [58].

Comprehensible artificial intelligence (cAI) is the result of a process that unites *local interpretability* based on *XAI* methods and *global interpretability* with the help of *iML* [19]. The ultimate goal of such systems would be to reach *ultra-strong machine learning*, where machine learning helps humans to improve in their tasks. For example, [97] examined the *comprehensibility* of programs learned with Inductive Logic Programming, and [120,119] showed that *comprehensibility* of such programs could help laymen to *understand* how and why a certain prediction was made.

Explainable artificial intelligence (XAI) is the area of research concerned with *explaining* an AI system's decision.

3 Approach to Literature Search

The goals of this paper are to (1) provide a complete overview of relevant aspects or properties of XAI methods, and (2) ease finding the right survey providing further details. In order to achieve this, a systematic and broad literature analysis was conducted on papers in the time range of 2010 to 2021. The target of this meta-survey are works that either contained reviews on XAI methods, or considerations on XAI metrics and taxonomy aspects.

3.1 Search

Our search consisted of two iterations, one directly searching for work on XAI taxonomies, and one more general search for general XAI surveys.

Work on XAI Taxonomies The iteration for work on XAI taxonomies started with an initial pilot phase. Here we identified common terms associated directly with XAI taxonomies (for abbreviations both the abbreviation and the full expression must be considered):

- machine learning terms: AI, DNN, Deep Learning, ML

- explainability terms: XAI, explain, interpret
- terms associated with taxonomies: taxonomy, framework, toolbox, guide

In the second search phase, we collected Google Scholar[3] search results for combinations of these terms. The main considered search terms are summarized in Table 1. For each search, the first 300 search results were scanned by the title for relevance to our search target. Promising ones then were scanned by abstract. Only work that we could access was finally included. In a last phase, we scanned the works recursively for references to further relevant reviews.

General XAI Surveys The second iteration collected search results for XAI surveys that do not necessarily propose a taxonomy, but possibly implicitly use one. For this, we again conducted the mentioned three search phases. The search terms now were the more general ones "XAI" and "XAI survey". These again were scanned first by title, then by abstract. This resulted in a similar number of finally chosen and in-depth assessed papers as the taxonomy search (not counting duplicate search results).

Table 1. Main used search phrases for the search for XAI taxonomies in the Google Scholar database with approximate number of matches

Matches	Search phrase
> 300	explain AI taxonomy
ca. 80	XAI taxonomy toolbox guide
> 300	explainable AI taxonomy toolbox guide
ca. 20	explain interpret AI artificial intelligence DNN Deep Learning ML machine learning taxonomy framework toolbox guide XAI

3.2 Categorization

For the sub-selection and categorization, we considered the general focus, the length, level of detail, target audience, citation count per year, and recency.

General focus Regarding the *general focus*, we sorted the obtained reviews into three categories:

General XAI method collections (subsection 4.2): Reviews that contain a broad collection of XAI methods without an explicit focus on a specific subfield;

Domain specific XAI method collections (subsection 4.3): Reviews that also contain a collection of XAI methods, but with a concrete focus on application domain (*e.g.* medicine), method technologies (*e.g.* inductive logic programming), or method traits (*e.g.* black-box methods);

[3] https://scholar.google.com

Conceptual reviews (subsection 4.1): Reviews that do not contain or not focus on XAI method examples, but on conceptual aspects for the field of XAI like taxonomy or metrics;

Toolboxes (subsection 4.4): Summary of a complete toolbox with implementation.

Length For the *length* we considered the number of pages up to the bibliography, resulting in three categories (cf. Figure 2): short (up to 6 pages), medium (up to 15 pages), long (up to 50 pages), and very long (more than 50 pages).

Target audience As *target audiences* we considered three types: beginners in the field of XAI (potentially coming from a different domain), practitioners, and researchers. A review was associated with one or several target audiences, if it specifically targeted that audience, or if it was judged practically suitable or interesting for readers of that audience.

Citation count per year The *citation count per year* of the surveys was used as a proxy for reception. It was collected from the popular citation databases Google Scholar, Semantic Scholar[4], OpenCitations[5], and NASA ADS[6]. The highest result (mostly google scholar) was chosen for comparison.

Recency was compared via the publication year.

Using these categorizations, some further works were ruled out for inclusion into the survey of surveys. Excluded were reviews which: would not sufficiently match our search target; were too specific (*e.g.* comparison of only very few specific methods); are very short (up to 4 pages) and are covered by prior or successive work of the authors, or are not often cited or not sufficiently focused.

3.3 Results

We finally reviewed over 70 surveys on XAI with publication date up to beginning of 2021. Most of them are from the years 2019 to 2021 (cf. Figure 2), which well fits the exponential increase in XAI methods that was observed so far [9,85,1,151]. These were analysed for XAI method aspects, taxonomy structuring proposals, and suitable example methods for each aspect. A selection of surveys and some examples toolboxes are detailed in the following.

To exemplify the aspects of our proposed taxonomy, we selected again more than 50 concrete XAI methods that are reviewed in example sections for the corresponding XAI aspects. The selection focused on high diversity and recency of the methods, in order to establish a broad view on the XAI topic. Finally, each of the methods were analysed on main taxonomy aspects, which is summarized in Table 6.

4 A Survey of Surveys on XAI Methods and Aspects

Along with the vastly growing demand and interest in XAI methods came an abundance of helpful reviews on the topic. This encompasses overviews on all

[4] https://www.semanticscholar.org/

[5] https://opencitations.net/

[6] https://ui.adsabs.harvard.edu/

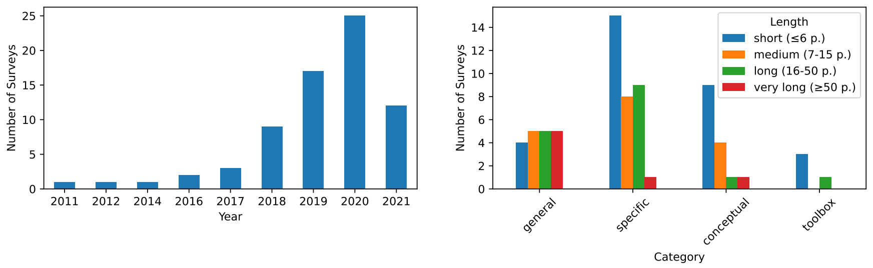


Fig. 2. Distribution of the reviewed surveys over the years 2010 to 2021 (*left*), and distribution of the survey lengths by general focus category (*right*)

kinds of aspects, methods, and metrics, each with custom focus. Given this battering amount of literature, it may be hard, especially for beginners and practitioners, to find a survey suited to their needs. The needs can encompass a specific focus, a desired level of detail and length, or others. This chapter provides a short survey of XAI surveys that shall help a reader in finding appropriate material to dig further into the topic of XAI.

The focus of the literature search lay on works that use, propose, or deduce a structured view on XAI methods and metrics. Search and categorization criteria are detailed in section 3. In the rest of this section, more than 50 selected surveys are reviewed with respect to their key focus points, level of detail, and their target audience. While this survey of surveys for sure will not be complete, it should both give a good overview on the breadth of XAI, as well as serve as a good starting when looking for a deeper dive into topics of the field.

Our results so far promise that many more helpful surveys on the topic of XAI will come up within the next years (cf. Figure 2). A key result of this structured meta-study will be presented later in section 5: The works are analyzed for taxonomy aspects of XAI methods, resulting in the—to our knowledge—most complete taxonomy of XAI methods available so far.

4.1 Conceptual Reviews

By now, many works have gathered and formulated important concepts related to XAI research. We here roughly divided the available literature by their main focus: broadly focused surveys, surveys from the perspectives of stakeholders and human-computer interaction, and finally surveys with an explicit focus on XAI metrics. An overview can be found in Table 3.

Broad Conceptual Surveys A very short, high-level, and beginner-friendly introduction to XAI can be found in the work of Gunning et al. from 2019 [59]. They derive four open challenges for the field: user-centric explanations, tradeoff between accuracy and interpretability, automated abstraction, and appropriate end-user trust. Regarding an XAI method taxonomy, a base for many later

Table 3. Overview on clusters of reviewed conceptual surveys on XAI

Focus	
Broad	Gunning et al. 2019 [59], Lipton 2018 [86], Müller et al. 2019 [95]
Stakeholder perspective	Gleicher 2016 [53], Langer et al. 2021 [82]
HCI perspective	Miller 2019 [92], Chromik & Schüßler 2020 [29], Ferreira et al. 2020 [45], Müller et al. 2021 [96]
XAI method evaluation	Doshi-Velez and Kim 2017 [39], Zhou et al. 2021 [151]

surveys was [86] by Lipton from 2018. In this medium-length work, Lipton provides a small and high-level XAI taxonomy. The focus, namely motivation and desiderata for interpretability, are held broad and suitable for beginners and interdisciplinary discussion. In contrast, the very broad and long DARPA report [95] is targeted at researchers of the field, and was provided by Müller et al. later in 2019. The goal of the report is to broadly capture all relevant developments and topics in the field of XAI. This resulted in a detailed meta-study on state-of-the-literature, and a detailed list of XAI method aspects and metrics (chapters 7 and 8).

XAI from a Stakeholder Perspective The early work [53] by Gleicher from 2016 highlights the stakeholder perspective on XAI problems. In the medium-length survey, Gleicher suggests a framework of general considerations for practically tackling an explainability problem. A similar focus is set by Langer et al. in their long recent work [82] from 2021. They review in detail XAI from the point of view of satisfying stakeholder desiderata. For this they collect standard goals of XAI problems and propagate that XAI design choices should take into account all stakeholder in an interdisciplinary manner.

XAI from a HCI Perspective When humans interact with AI driven machines, this human-machine-system can benefit from explanations obtained by XAI. Hence, there are by now several surveys concentrating on XAI against the background of human-computer-interaction (HCI). An important and well-received base work in this direction is that of Miller from 2019 [92]. He conducts a long, detailed, and extensive survey of work from philosophy, psychology, and cognitive science regarding explanations and explainability. The main conclusions for XAI research are: (1) (local) explanations should be understood contrastive, *i.e.*, they should clarify why an action was taken instead of another; (2) explanations are selected in a biased manner, *i.e.*, do not represent the complete causal chain but few selected causes; (3) causal links are more helpful to humans than probabilities and statistics; and (4) explanations are social in the sense that the background of the explanation receiver matters. A much shorter paper was provided by Chromik and Schüßler [29] in 2020. They rigorously develop a taxonomy for evaluating black-box XAI methods with the help of human subjects. Furthermore, they make some concrete suggestions for study design. The related survey [45] by Ferreira et al. from 2020 is slightly longer, and may serve as an entry point to the topic for researchers. They presents a taxonomy of XAI that links with computer science and HCI communities, with a structured and dense collection of many related reviews. Very recent work on XAI metrics is provided by Müller et al. in their 2021 paper [96]. They collect concrete and practical design principles for XAI in human-machine-systems. Several relevant XAI metrics are recapitulated, and their broad collection of related surveys may serve as entry point to the research field of human-AI-systems.

Surveys with a Focus on Evaluation Continuing on the HCI perspective, a hot topic in the field of XAI are metrics for measuring the quality of explanations for human receivers. An early and base work on XAI metric categorization is [39] by Doshi-Velez and Kim, 2017. The medium-length survey collects latent dimensions of interpretability with recommendations on how to choose and evaluate an XAI method. Their taxonomy for XAI metrics is adapted by us (cf. subsection 5.3). A full focus on metrics for evaluating XAI methods is set in the recent work [151] by Zhou et al. from 2021. This medium-length, detailed meta-study reviews diverse XAI metrics, aligned with shallow taxonomies both for methods and metrics.

4.2 Broad Method Collections

There by now exists an abundance of surveys containing broad collections of XAI methods, serving as helpful starting points for finding the right method. The following shortly highlights surveys that feature a generally broad focus (for specifically focused surveys see subsection 4.3). This summary shall help to find a good starting point for diving deeper into methods. Hence, the surveys are first sorted by their length (as a proxy for the amount of information), and then by their reception (citations per year). The latter was found to strongly correlate with the age.

Short Surveys A short overview mostly on explanations of supervised learning approaches was provided by Došilović et al. in 2018 [40]. This quite short but broad and beginner-friendly introductory survey shortly covers diverse approaches and open challenges towards XAI. Slightly earlier in 2017, Biran and Cotton published their XAI survey [18]. This is a short and early collection of explainability concepts. Their general focus lies on explanations of single predictions, and diverse model types like rule-based systems and Bayesian networks, which are each shortly discussed. Most recently, in 2020 Benchekroun et al. collected and presented XAI methods from an industry point of view [17]. They present a preliminary taxonomy that includes pre-modelling explainability as an approach to link knowledge about data with knowledge about the used model and its results. Regarding the industry perspective, they specifically motivate standardization.

Medium-length Surveys An important and very well received earlier work on XAI method collections was provided by Gilpin et al. in 2018 [52]. Their extensive survey includes very different kinds of XAI methods, including, e.g., rule extraction. In addition, for researchers they provide references to further more specialized surveys in the field. It is very similar to the long survey [1], only shortened by skipping detail on the methods. More detail, but a slightly more specialized focus, is provided by Du et al. in their 2019 survey [41]. The beginner-friendly high-level introduction to XAI features few, in detail discussed examples. These concentrate on perturbation based attention visualization. Similarly, the examples in [98] also mostly focus on visual domains and explanations. This review by Murdoch et al. in 2019 is also beginner-friendly, and prefers detail over number of methods. The examples are embedded into a comprehensive short introductory review of key categories and directions in XAI. Their main focus is on the proposal of three simple practical desiderata for explanations: The model to explain should be predictive (predictive accuracy), the explanations should be faithful to the model (descriptive accuracy), and the information presented by the explanations should be relevant to the receiver. Slightly earlier, Goebel et al. concentrate in their work [54] from 2018 more on multimodal explanations and question-answering systems. This survey contains a high-level review of the need for XAI, and discussion of some exemplary state-of-the-art methods. A very recent and beginner-friendly XAI method review is [67] by Islam et al. from 2021. Their in-depth discussion of example methods is aligned with a shallow taxonomy, and many examples are practically demonstrated in a common simple case study. Additionally, they present a short meta-study of XAI surveys, and a collection of future perspectives for XAI. The latter includes formalization, XAI applications for fair and accountable ML, XAI for human-machine-systems, and more interdisciplinary cooperation in XAI research.

Long Surveys A well-received lengthy and extensive XAI literature survey was conducted by Adadi and Berrada in 2018 [1]. They reviewed and shortly discuss 381 papers related to XAI to provide a holistic view on the XAI research landscape at that time. This includes methods, metrics, cognitive aspects, and aspects related to human-machine-systems. Another well-received, but more recent, slightly less extensive and more beginner-friendly survey is [23] by Carvalho et al. in 2019. They collect and discuss in detail different aspects of XAI, especially motivation, properties, desirables, and metrics. Each aspect is accompanied by some examples. Similarly beginner-friendly is the introductory work of Xie et al. from 2020 [141]. Their field guide explicitly targets newcomers to the field with a general introduction to XAI, and a wide variety of examples of standard methods, mostly for explanations of DNNs. A more formal introduction is provided by Das and Rad in their 2020 survey [35]. They collect formal definitions of XAI related terms, and develop a shallow taxonomy. The focus of the examples is on visual local and global explanations of DNNs based on (model-specific) backpropagation or (model-agnostic) pertubation-based methods. More recent and much broader is the survey [85] of Linardatos et al. from 2021. It provides an extensive technical collection and review of XAI methods with code and toolbox references, which makes it specifically interesting for practitioners.

Very Long Surveys By now there are a couple of surveys available that aim to give a broad, rigorous, and in-depth introduction to XAI. A first work in this direction is the regularly updated book on XAI by Molnar [93], first published in 2017[7]. This book is targeted at beginners, and gives a basic and detailed introduction on interpretability methods, including many transparent and many model-agnostic ones. The focus lies more on fundamental concepts of interpretability and detail on standard methods than on the amount of discussed methods. Meanwhile, Arrieta et al. put up a very long and broad collection of XAI methods in their 2020 review [9]. The well-received survey can also be considered a base work of state-of-the-art XAI, as they introduce the notion of *responsible AI*, *i.e.*, development of AI models respecting fairness, model explainability, and accountability. This is also the focus of their work, in which they provide terminology, a broad but less detailed selection of example methods, and practical discussion for responsible AI. A very recent extensive XAI method survey is that of Burkart and Huber from 2021 [20]. They review in moderate detail explainability methods, primarily for classification and regression in supervised machine learning. Specifically, they include many rule-based and decision-tree based explanation methods, as well as aspects on data analysis and ontologies for formalizing input domains. This is preceeded by a deep collection of many general XAI aspects. Finally and a little earlier, Vilone and Longo published in 2020 an equally extensive systematic literature survey on XAI in general [134]. The systematic literature search was similar to this one, only with a different focus. They include a broad meta-study of reviews, as well as reviews and discussion of works on XAI theory, methods, and method evaluations.

4.3 Method Collections with Specific Focus

Besides the many broad method collections, there by now are numerous ones specifically concentrating on an application domain, specific input or task types, certain surrogate model types, or other traits of XAI methods. We here manually clustered surveys by similarity of their main focus points. An overview on the resulting clusters is given in Table 5.

XAI for Specific Application Domains Some method surveys focus on concrete practical application domains. One is [34] by Danilevsky et al. from 2020, who survey XAI methods for *natural language processing*. This includes a taxonomy, a review of several metrics, and a dense collection of XAI methods. Another important application domain is the *medical domain*. For example, Singh et al. provided in 2020 a survey and taxonomy of XAI methods for image classification with focus on medical applications [124]. A slightly broader focus on general XAI methods for medical applications was selected by Tjoa and Guan in the same

[7] https://github.com/christophM/interpretable-ml-book/tree/v0.1

Table 4. Overview on focus points/specialties of discussed general XAI method collections. Sorted first by length then reception (citations per year). Detail is manually ranked in values from 1 (short discussion per method) to 5 (detailed discussion per method). Audience can be beginner (B), practitioner (P), or researcher (R).

Authors	Year	Audience	Focus / Specialty	Detail
Short				
Došilović et al.	2018 [40]	B, R	XAI for supervised learning	1
Biran & Cotton	2017 [18]	R	Local explanations and interpretable models	1
Benchekroun et al.	2020 [17]	B, P	Industry point of view	5
Medium				
Gilpin et al.	2018 [52]	R	XAI research aspects	1
Du et al.	2019 [41]	B	Perturbation based attention visualization	3
Murdoch et al.	2019 [98]	B, P	Predictive, descriptive, relevant desiderata for explanations, practical examples	5
Goebel et al.	2018 [54]		Need for XAI	3
Islam et al.	2021 [67]	B, R	Demonstration of XAI methods in case-study (credit scoring)	5
Long				
Adadi & Berrada	2018 [1]	P	XAI research landscape (methods, metrics, cognitive aspects, human-machine-systems)	1
Carvalho et al.	2019 [23]	B, R	Aspects associated with XAI (esp. motivation, properties, desirables)	2
Xie et al.	2020 [141]	B	Explanation of DNNs	3
Das & Rad	2020 [35]	R	Visualization methods for visual DNNs	3
Linardatos et al.	2021 [85]	P, R	Technical, with source code	3
Very long				
Molnar	2020 [93]	B	Fundamental concepts of interpretability	5
Arrieta et al.	2020 [9]	P, R	Notion of responsible AI: Terminology, broad collection of examples	1
Burkart & Huber	2021 [20]	R	Supervised machine learning; linear, rule-based methods and decision trees, data analytics and ontologies	3
Vilone & Longo	2020 [134]	R	Systematic and broad review of XAI surveys, theory, methods, and method evaluation	3

Table 5. Overview on clusters of reviewed surveys with a restricted focus

Focus restricted by:	**Focus restricted to:**	
Application domain	natural language processing	[34]
	medicine	[124,131]
	recommendation systems	[101,148]
Application type	interactive machine learning	[8,14,6]
Task	visual tasks	[117,118,99,7,2,84,147]
	reinforcement learning	[107,65]
Technology	rule-based XAI	[33,133,60,22]
Other XAI method traits	model-agnostic methods	[57]
	counterfactual explanations	[21]

year [131]. Their long review shortly discusses more than sixty methods, and sorts them into a shallow taxonomy. Another domain sparking needs for XAI is that of *recommendation systems*, *e.g.*, in online shops. An example here is the long, detailed, and practically oriented survey by Nunes and Jannach from 2017 [101]. Amongst others, they present a detailed taxonomy of XAI methods for recommendation systems (cf. [101, Fig. 11]). A similar, but even more extensive and lengthy survey was provided by Zhang and Chen in 2020 [148]. They generally review recommendation systems also in a practically oriented manner, and provide a good overview on models that are deemed explainable.

XAI for Interactive ML Applications Several studies concentrate on XAI to realize interactive machine learning. For example, Anjomshoae et al. in 2019 reviewed explanation generation, communication and evaluation for autonomous agents and human-robot interaction [8]. Baniecki and Biecek instead directly concentrated on interactive machine learning in their 2020 medium-length survey on the topic [14]. They present challenges in explanation, traits to overcome these, as well as a taxonomy for interactive explanatory model analysis. The longer but earlier review [6] by Amershi et al. from 2014 more concentrates on practical case studies and research challenges. They motivate incremental, interactive and practical human-centered XAI methods.

XAI for Visual Tasks Some of the earlier milestones for the current field of XAI were methods to explain input importance for models with image inputs [35], such as LIME [112] and LRP [12]. Research on interpretability and explainability of models for visual tasks is still very active, as several surveys with this focus show. One collection of both methods and method surveys with a focus on visual explainability is the book [117] edited by Samek et al. in 2019. This includes the following surveys:

- Samek et al. [118]: A short introductory survey on visual explainable AI for researchers, giving an overview on important developments;

- Nguyen et al. [99]: A survey specifically on feature visualization methods. These are methods to find prototypical input patterns that most activate parts of a DNN. The survey includes a mathematical perspective on the topic and a practical overview on applications.
- Ancona et al. [7]: A detailed survey on gradient-based methods to find attribution of inputs to outputs; and
- Alber [2]: A detailed collection of implementation considerations regarding different methods for highlighting input attribution. The review includes code snippets for the TensorFlow deep learning framework.

Similar to [7], Li et al. focus in [84] from 2020 on XAI methods to obtain heatmaps as visual explanations. They in detail discuss seven examples of methods, and conduct an experimental comparative study with respect to five specialized metrics. Another survey focusing on visual interpretability is the earlier work by Zhang and Zhu from 2018 [147]. This well-received medium-length survey specializes on visual explanation methods for convolutional neural networks.

XAI for Reinforcement Learning Tasks Just as for visual tasks, there are some studies specifically focusing on explanations in tasks solved by reinforcement learning. One is [107] by Puiutta and Veith from 2020. This medium to lengthy review provides a short taxonomy on XAI methods for reinforcement learning. It reviews more than 16 methods specific to reinforcement learning in a beginner-friendly way. A comparable and more recent, but slightly longer, more extensive, and more technical survey on the topic is [65] by Heuillet et al. from 2021.

XAI Methods Based on Rules One type of explanation outputs is that of (formal) symbolic rules. Both generation of interpretable rule-based models, as well as extraction of approximate rule sets from less interpretable models have a long history. We here collect some more recent reviews on these topics. One is the historical review [33] on the developments in inductive logic programming, by Cropper et al. in 2020. Inductive logic programming summarizes methods to automatically construct rule-sets for solving a task given some formal background knowledge and few examples. The mentioned short survey aims to look back the last 30 years of development in the field, and serve as a good starting point for beginners. On the side of inherently interpretable rule-based models, Vassiliades et al. recently in 2021 reviewed Argumentation Frameworks in detail [133]. An Argumentation Framework provides an interpretable logical argumentation line that formally deduces a statement from (potentially incomplete) logical background knowledge. This can, *e.g.*, be used to find the most promising statement from some choices, and, hence, as explainable (potentially interactive) model on symbolic data. The long, detailed, and extensive survey formally introduces standard Argumentation Frameworks, reviews existing methods and applications, and promotes Argumentation Frameworks as promising interpretable models. While the previous surveys concentrate on directly training inherently interpretable models consisting of rules, Hailesilassie in 2016 shortly reviewed rule extraction methods [60]. Rule extraction aims to approximate a trained model with symbolic rule sets or decision trees. These methods have a long history but faced their limits when applied to large-sized models like state-of-the-art neural networks, as discussed in the survey. A more recent survey that covers both rule extraction as well as integration of symbolic knowledge into learning processes is the review [22] by Calegari et al. from 2020. Their long and in-depth overview covers the main symbolic/sub-symbolic integration techniques for XAI, including rule extraction methods for some steps.

XAI Reviews Specialized on Other XAI Method Traits While the previous surveys were mostly concentrated on a specific type of task, there are also some that are restricted to certain types of explainability methods. One is the short survey [21] from 2019 by Byrne. This reviews specifically counterfactual explanations with respect to evidence from human reasoning. The focus here lies on additive and subtractive counterfactual scenarios. A very well received, long and extensive survey for model-agnostic XAI methods on tabular data is [57] by Guidotti et al. from 2018. Besides the method review, they also develop a formal approach to define XAI use-cases that is especially useful for practitioners.

4.4 Toolboxes

A single XAI method often does not the full job of making all relevant model aspects clear to the explainee. Hence, toolboxes have become usual that implement more than one explainability method in a single library with a common interface. For detailed lists of available toolboxes the reader is referred to, e.g., the related work in [10, Tab. 1], the repository links in [85, Tabs. A.1, A.2]. For implementation considerations in the case of visual interpretability to the review [2] is a suitable read. We here provide some examples of publications presenting toolboxes that we analyzed for taxonomy aspects.

The beginner-friendly Microsoft toolbox InterpretML [100] from 2019 implements five model-agnostic and four transparent XAI methods that are shortly introduced in the paper. Another toolbox from that year is iNNvestigate by Alber et al. [3]. They specifically concentrate on some standard post-hoc heatmapping methods for visual explanation. Arya et al. presented the IBM AI explainability 360 toolbox [10] with 8 diverse XAI methods in 2019. The implemented workflow follows a proposed practical, tree-like taxonomy of XAI methods. Implemented methods cover a broad range of explainability needs, including explainability of data, inherently interpretable models, and post-hoc explainability both globally and locally. More recently, Spinner et al. presented in 2020 their toolbox explAIner [126]. This is realized as a plugin to the existing TensorBoard[8] toolkit for the TensorFlow deep learning framework. The many post-hoc DNN explanation and analytics tools are aligned with the suggested pipeline phases of model understanding, diagnosis, and refinement. Target users are both researchers, practitioners, and beginners.

[8] TensorBoard toolkit: https://www.tensorflow.org/tensorboard

5 Taxonomy

In this section, a taxonomy of XAI methods is established that unites terms and notions from literature that allow to differentiate and evaluate XAI methods. For this, in total more than 70 surveys selected in the course of our literature search (cf. section 3), including the ones discussed in section 4, were analyzed for such terms. We then identified synonymous terms. Finally, we sub-structured the notions systematically according to practical considerations, in order to provide a complete picture of state-of-the-art XAI method and evaluation aspects. This section details the found notions and synonyms, and our proposed structure thereof, which defines the outline. An overview of the root-level structure is provided in Figure 3, and a complete overview of the final structure in Figure 7. The respective sub-trees for each of the first-level aspect categories can be found at the beginning of each section.

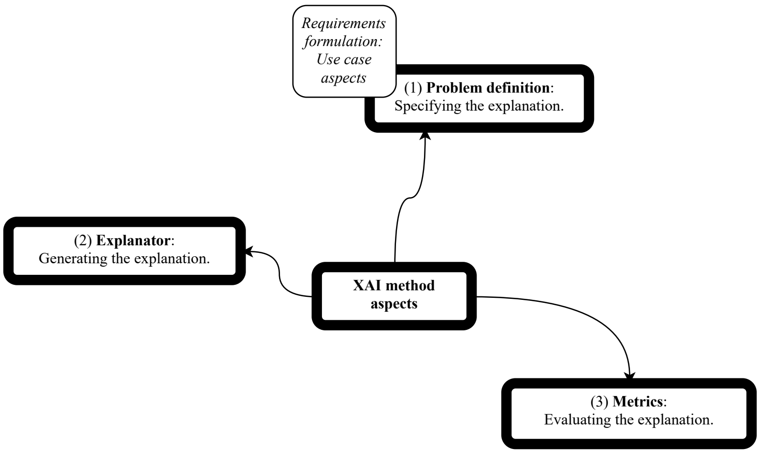

Requirements formulation: Use case aspects
(1) Problem definition: Specifying the explanation.
(2) Explanator: Generating the explanation.
XAI method aspects
(3) Metrics: Evaluating the explanation.

Fig. 3. Overview on top-level categorization of taxonomy aspects explained in section 5. Find a visualization of the complete taxonomy in Figure 7 on page 44.

At the root level, we propose to categorize in a *procedural manner* according to the steps for building an explanation system. The following gives a high-level overview on the first two levels of the taxonomy structure and the outline of this chapter (cf. overview in Figure 3):

1. **Problem definition** (subsection 5.1): One usually should start with the *problem definition.* This encompasses
   - traits of the *task*, and
   - the *explanandum* (precisely: the *interpretability* of the explanandum).
2. **Explanator properties** (subsection 5.2): Then, we detail the *explanator properties* (subsection 5.2), which we functionally divided into properties of

   - *input*,
   - *output*,
   - *interactivity* with the user, and
   - any further *formal constraints* posed on the explanator.
3. **Metrics** (subsection 5.3): Lastly, we discuss different *metrics* (subsection 5.3) that can be applied to explanation systems in order to evaluate their qualities. Following Doshi-Velez & Kim [39], these are substructered by their dependence on subjective human evaluation and the application into:
   - *functionally grounded* metrics (independent of human judgement),
   - *human grounded* metrics (subjective judgement required), and
   - *application grounded* (full human-AI-system required).

The presented aspects are illustrated by selected example methods (marked in gray). The selection of example methods is by no means complete. Rather it intends to give an impression about the wide range of the topic, and how to apply our taxonomy to both some well-known, and less known but interesting methods.

On Requirements Derivation. Note that from a procedural perspective, the first step when designing an explanation system should be to determine the use-case specific *requirements* (as part of the problem definition). The requirements can be derived from all the taxonomy aspects that are collected in this review. This gives rise to a similar sub-structuring as the procedual one shown above:

- Both explanandum and explanator must match the *task*,
- the explanandum should be chosen to match the (inherent) *interpretability* needs, and
- the explanator must fulfill any other (functional and architectural) *explanator constraints* (see subsection 5.2), as well as
- any *metric target values.*

This should be motivated by the actual goal or desiderata of the explanation, which can be, *e.g.*, verifiability of properties like fairness, safety, and security, knowledge discovery, promotion of user adoption respectively trust, or many more. An extensive list of desiderata can be found in [82]. As detailed in subsection 2.1, detailed collection of XAI needs, desiderata, and typical use-cases is out-of-scope of this work. Instead, our collection of taxonomy aspects shall serve as a starting point for effective and complete use-case analysis and requirements derivation.

5.1 Problem Definition

The following aspects consider the concretion of the explainability problem. Apart from the use-case analysis which is skipped in this work, details on the following two aspects must be clear:

- **Task**: the *task* that is to be explained must be clear, and

– **Model Interpretability**: the solution used for the task, meaning the *type of explanandum.* For explainability purposes the level of *interpretability* of the explanandum model is the relevant point here.

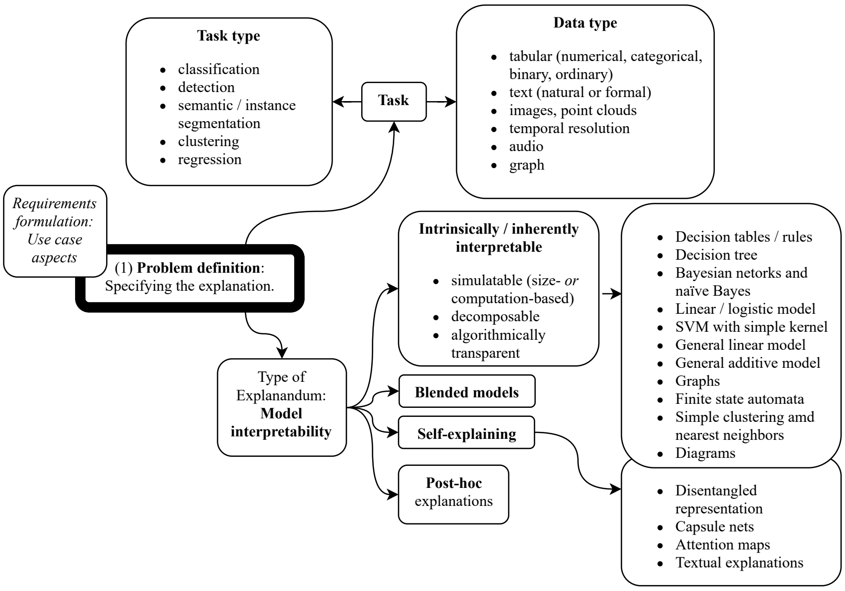

Task type
classification
detection
semantic / instance segmentation
clustering
regression
Task
Data type
tabular (numerical, categorical, binary, ordinary)
text (natural or formal)
images, point clouds
temporal resolution
audio
graph
Requirements formulation: Use case aspects
(1) Problem definition: Specifying the explanation.
Type of Explanandum: Model interpretability
Intrinsically / inherently interpretable
simulatable (size- or computation-based)
decomposable
algorithmically transparent
Blended models
Self-explaining
Post-hoc explanations
Decision tables / rules
Decision tree
Bayesian netorks and naïve Bayes
Linear / logistic model
SVM with simple kernel
General linear model
General additive model
Graphs
Finite state automata
Simple clustering amd nearest neighbors
Diagrams
Disentangled representation
Capsule nets
Attention maps
Textual explanations

Fig. 4. Overview on the taxonomy aspects related to the problem definition that are detailed in subsection 5.1. Find a visualization of the complete taxonomy in Figure 7 on page 44.

Task. Out-of-the-box, XAI methods usually only apply to a specific set of

– *task types* of the to-be-explained model, and
– *input data types.*

For white-box methods that access model internals, additional constraints may hold for the *architecture* of the model (cf. portability aspect in [143]).

Task Type. Typical task categories are unsupervised clustering (clu), regression, classification (cls), detection (det), segmentation (seg) either semantic, which is pixel-wise classification, or segmentation of instances. Many XAI methods targeting a question for classification, *e.g.*, "Why this class?", can be extended to det, seg, and temporal resolution. This can be achieved via snippeting of the new dimensions: "Why this class in this spatial/temporal snippet?". It must be noted that XAI methods working on classifiers often require access to the prediction of a continuous classification score instead of the final discrete classification. Such methods can also be used on regression tasks to answer questions about local trends, *i.e.*, “Why does the prediction tend into this direction?”. Examples of regression predictions are bounding box dimensions in object detection.

RISE [104]

D-RISE [105]

ILP [33]

CA-ILP [109]

Examples. RISE [104] (Randomized Input Sampling for Explanation) is a model-agnostic attribution analysis method that is specialized on image classifiers. For an input image, it produces a heatmap that highlights those super-pixels in the image, which, when deleted, have the greatest influence on the class confidence. High-attribution super-pixels are found by randomly dimming super-pixels of the input image. The method D-RISE [105] extends this to object detection. It considers not a one-dimensional class confidence but the total prediction vector of a detection. Influence of a dimming is measured as the distance between the prediction vectors. The mentioned image-specific (*i.e.* local) explanation methods use the continuous class or prediction scores of the explanandum, and, hence, are in principle also applicable to regressors. In contrast, surrogate models produced using inductive logic programming (ILP) [33] require the binary classification output of a model. ILP frameworks require as input background knowledge (logical theory), together with positive and negative examples. From this, a logic program in the form of first-order rules is learned that covers as many of the samples as possible. An example of an ILP-based XAI method for convolutional image classifier is CA-ILP [109] (Concept Analysis for ILP). In order to explain parts of the classifier with logical rules, they first train global small models that extract symbolic features from the DNN intermediate outputs. These feature outputs are then used to train an ILP surrogate model. Lastly, clustering tasks can often be explained by providing examples or prototypes of the final clusters, which will be discussed in subsection 5.2.

Input Data Type. Not every XAI method supports every input and output *signal type*, also called data type [57]. One input type is tabular (symbolic) data, which encompasses numerical, categorical, binary, and ordinary (ordered) data. Other symbolic input types are natural language or graphs, and non-symbolic types are images and point clouds (with or without temporal resolution), as well as audio.

LIME [112]

Examples. Typical examples for image explanations are methods producing heatmaps. These highlight parts of the image that were relevant for the decision or a part thereof. This highlighting of input snippets can also be applied to textual inputs where single words or sentence parts may serve as snippets. A prominent example of heatmapping that is both applicable to images and text inputs is the model-agnostic LIME [112] method (Local Interpretable Model-agnostic Explanations). It locally approximates the explanandum model by a linear model on feature snippets of the input. For training of that linear model, randomly selected snippets are removed. In case of textual inputs, the words are considered as snippets, and for images super-pixels. The removal of super-pixels is here realized by blackening them. While LIME is suitable for image or textual input data, [57] provides a broad overview on model-agnostic XAI methods for tabular data.

Model Interpretability. By model interpretability we here refer to the level of interpretability of the explanandum, *i.e.* the model used to solve the original task of the system. A model is interpretable, if it gives rise not only to mechanistic understanding (transparency), but also to a functional understanding by a human [103]. Explainability of (aspects of) the explanation system can be achieved by one of the following choices:

- start from the beginning with an *intrinsically interpretable* explanandum model (also called *ante-hoc interpretable* [20] or *intrinsically interpretable*) or a
- *blended*, *i.e.* partly interpretable, model (also called *interpretable by design* [20]);
- design the explanandum model to include *self-explanations* as additional output; or
- *post-hoc* find an interpretable helper model without changing the trained explanandum model.

Intrisic or Inherent Interpretability. As introduced in [86], one can further differentiate between different levels of model transparency: The model can directly be understood as a whole, *i.e.* adapted as mental model by a human (*simulatable* [9]); or it can be split up into parts each of which is simulatable (*decomposable* [9]). Simulatability can either be measured based on the size of the model, or the needed length of computation (cf. discussion of metrics in subsection 5.3). As a third category, *algorithmic transparency* is considered, which means the model is mathematically understood, *e.g.*, the shape of the error surface is known. This is considered the weakest form of transparency. The following models are considered inherently transparent in the literature (cf. [93, Chap. 4], [57, Sec. 5], [100]):

decision tables and rules as experimentally evaluated in [66,4,49]; This encompasses boolean rules as can be extracted from decision trees, or fuzzy or first-order logic rules. For further insights in inductive logic programming approaches to find the latter kind of rules see, *e.g.*, the recent survey [33].

decision trees as empirically evaluated in [66,49,4];

bayesian networks and naïve Bayes models as of [20]; interpretability of Bayesian network classifiers was, *e.g.*, experimentally evaluated in [49].

linear and logistic models as of, *e.g.*, [93];

support vector machines as of [124]; as long as the used kernel function is not too complex, non-linear SVMs give interesting insights to the decision boundary.

general linear models (GLM) to inherently provide weights for the importance of input features; Here, it is assumed that there is a linear relationship between the input features and the expected output value when this is transformed by a given transformation. For example, in logistic regression, the transformation is the logit. See, *e.g.*, [93, Sec. 4.3] for a basic introduction and further references.

general additive models (GAM) also inherently coming with feature importance weights; Here it is assumed that the expected output value is the sum of transformed features. See the survey [24] for more details and further references.

Additive Model Explainer [27]

Examples. One concrete example of general additive models is the Additive Model Explainer [27]. They train predictors for a given set of features, and another small DNN predicting the additive weights for the feature predictors. They use this setup to learn a GAM surrogate models for a DNN, which also provides a prior to the weights: They should correspond to the sensitivity of the DNN with respect to the features.

graphs as of, *e.g.*, [141];

finite state automata as of [139];

simple clustering and nearest neighbors approaches as of [20];

k-means clustering [61]

k-NN [5]

Examples. Examples are k-nearest neighbors (supervised), or k-means clustering (unsupervised). The standard k-means clustering method [61] works with an intuitive model, simply consisting of k prototypes and a proximity measure, with inference associating new samples to the closest prototype representing a cluster. k-nearest neighbors (k-NN) determines for a new input the k samples from a labeled database that are most similar to the new input sample. The majority vote of the nearest labels is then used to assign a label to the new instance. As long as the proximity measure is not too complex, these methods can be regarded as unsupervised respectively supervised inherently interpretable models. k-NN was experimentally evaluated for interpretability in [49].

diagrams as of [65].

Blended Models. Blended models (also called *interpretable by design* [20]) consist partly of intrinsically transparent, symbolic models, that are integrated in sub-symbolic non-transparent ones. These kind of hybrid models are especially interesting for neuro-symbolic computing and similar fields combining symbolic with sub-symbolic models [22].

Logic Tensor Nets [38]

Examples. An example of a blended model are Logic Tensor Networks [38]. Their idea is to use fuzzy logic to encode logical constraints on

DNN outputs, with a DNN acting as fuzzy logic predicate. The framework in [38] allows additionally to learn semantic relations subject to symbolic fuzzy logic constraints. The relations are represented by simple linear models. Unsupervised deep learning can be made interpretable by approaches such as combining autoencoders with visualization approaches or by explaining choices of "neuralized" clustering methods [73] (*i.e.*, clustering models translated to a DNN) with saliency maps. Enhancing an autoencoder was applied for example in the FoldingNet [142] architecture on point clouds. There, a folding-based decoder allows to view the reconstruction of point clouds, namely the warping from a 2D grid into the point cloud surface. A saliency based solution can be produced by algorithms such as layer-wise relevance propagation which will be discussed in later examples.

FoldingNet [142]

Neuralized clustering [73]

Self-explaining Models. Self-explaining models provide additional outputs that explain the output of a single prediction. According to [52], there are three standard types of outputs of explanation generating models: *attention maps*, *disentangled representations*, and *textual or multi-modal explanations*.

attention maps These are heatmaps that highlight relevant parts of a given single input for the respective output.

Examples. The work in [76] adds an attention module to a DNN that is processed in parallel to, and later multiplied with, convolutional outputs. Furthermore, they suggest a clustering-based post-processing of the attention maps to highlight most meaningful parts.

disentangled representations Representations in the intermediate output of the explanandum are called disentangled if single or groups of dimensions therein directly represent symbolic (also called semantic) concepts.

Examples. One can by design force one layer of a DNN to exhibit a disentangled representation. One example are capsule networks [115], that structure the network not by neurons but into groups of neurons, the capsules, that characterize each one entity, *e.g.*, an object or object part. The length of a capsule vector is interpreted as the probability that the corresponding object is present, while the rotation encodes properties of the object (*e.g.*, rotation or color). Later capsules get as input the weighted sum of transformed previous capsule outputs, with the transformations learned and the weights obtained in an iterative routing process. A simpler disentanglement than alignment of semantic concepts with groups of neurons is alignment of single dimensions. This is done, *e.g.*, in the ReNN [137] architecture. They explicitly modularize their DNN to ensure semantically meaningful intermediate outputs. Other methods rather follow a post-hoc approach that fine-tunes a trained DNN towards more disentangled representations, like it is suggested for Semantic Bottleneck Networks [87]. These consist of the pretrained backbone

Capsule Nets [115]

ReNN [137]

Semantic Bottlenecks [87]

of a DNN, proceeded by a layer in which each dimension corresponds to a semantic concept, called semantic bottleneck, and finalized by a newly trained front DNN part. During fine-tuning, first the connections from the backend to the semantic bottleneck are trained, then the parameters of the front DNN. Another interesting fine-tuning approach is that of concept whitening [28], which supplements batch-normalization layers with a linear transformation that learns to align semantic concepts with unit vectors of an activation space.

Concept Whitening [28]

textual or multi-model explanations These provide the explainee with a direct verbal or combined explanation that as part of the model output.

Examples. An example are the explanations provided by [77] for the application of end-to-end steering control in autonomous driving. Their approach is two-fold: They add a custom layer that produces attention heatmaps similar to [76], and these are used by a second custom part to generate textual explanations of the decision which are (weakly) aligned with the model processing. ProtoPNet [26] for image classification provides visual examples rather than text. The network architecture is based on first selecting prototypical image patches, and then inserting a prototype layer that predicts similarity scores for patches of an instance with prototypes. These can then be used for explanation of the final result in the manner of "This is a sparrow as its beak looks like that of other sparrow examples". A truly multi-modal example is [62], which trains alongside a classifier a long-short term memory DNN (LSTM) to generate natural language justifications of the classification. The LSTM uses both the intermediate features and predictions of the image classifier, and is trained towards high class discriminativeness of the justifications. The explanations can optionally encompass bounding boxes for features that were important for the classification decision, making it multi-modal.

[77]

ProtoPNet [26]

[62]

Post-hoc. Post-hoc methods use a (local or global) helper model from which to derive an explanation. This explainable helper model can either aim to

- fully mimic the behavior of the explanandum, or
- only approximate sub-aspects like input attribution.

Helper models that fully approximate the explanandum are often called *surrogate* or *proxy* models, and the process of training them is termed *model distillation*, *student-teacher* approach, or *model induction*. However, it is often hard to differentiate between the two types. Hence, for consistency, we here use the terms proxy and surrogate model for any types of helper models. Many examples for post-hoc methods are given in the course of the upcoming taxonomy aspects.

5.2 Explanator

One can consider an explanator simply as an implemented function that outputs explanations. This allows to structure aspects of the explanator into the main defining aspects of a function:

- the *input*,
- the *output*,
- the function class described by *mathematical properties or constraints*, and
- the actual processing of the explanation function, like its *interactivity.*

An overview on the aspects discussed in this section is given in Figure 5.

Input. The following explanator characteristics are related to the explanator input:

- What are the *required inputs* (the explanandum model, data samples, or even user feedback)?
- In how far is the method *portable* to other input types, *e.g.* explanandum model types?
- In how far are explanations *local to an input instance or global for the complete input*?

Required Input. The necessary inputs to the explanator may differ amongst methods [126]. While the explanandum, the *model* to explain, must usually be provided to the explanator, many methods do also require valid *data* samples, or even *user feedback* (cf. section 5.2) or further situational *context* (cf. [36] for a more detailed definition of context).

Portability. An important practical aspect for post-hoc explanations is whether or in how far the explanation method is dependent on access to internals of the explanandum model. This level of dependency is called portability, translucency, or transferability. In the following, we will not further differentiate between the strictness of requirements of model-specific methods. Transparent and self-explaining models are always model-specific, as the interpretability requires a special model type or model architecture (modification). Higher levels of dependency are:

model-agnostic also called *pedagogical* [143] or black-box: This means that only access to model input and output is required.

SHAP [88]

Examples. A prominent example of model-agnostic methods is the previously discussed LIME [112] method for local approximation via a linear model. Another method to find feature importance weights without any access to model internals is SHAP [88] (SHapley Additive exPlanation). Their idea is to axiomatically ensure: local fidelity; features missing from the original input have no effect; an increase of a weight also means an increased attribution of the feature to the

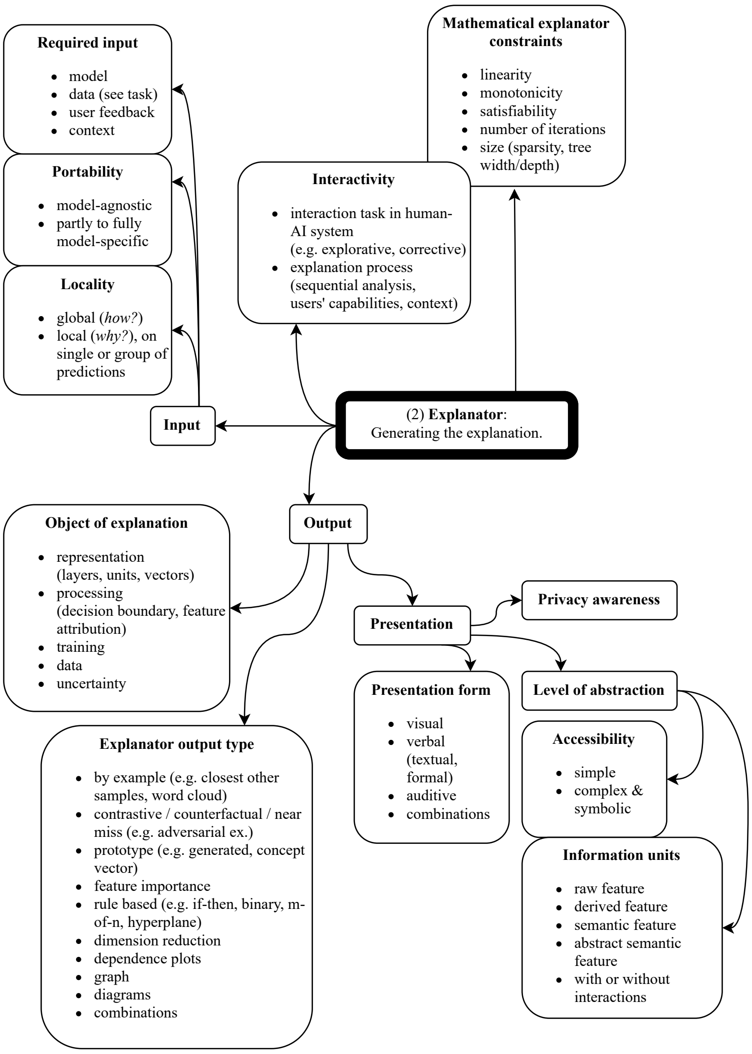

Required input
model
data (see task)
user feedback
context
Portability
model-agnostic
partly to fully model-specific
Locality
global (*how?*)
local (*why?*), on single or group of predictions
Input
Mathematical explanator constraints
linearity
monotonicity
satisfiability
number of iterations
size (sparsity, tree width/depth)
Interactivity
interaction task in human-AI system (e.g. explorative, corrective)
explanation process (sequential analysis, users' capabilities, context)
(2) Explanator: Generating the explanation.
Output
Object of explanation
representation (layers, units, vectors)
processing (decision boundary, feature attribution)
training
data
uncertainty
Presentation
Privacy awareness
Presentation form
visual
verbal (textual, formal)
auditive
combinations
Level of abstraction
Accessibility
simple
complex & symbolic
Information units
raw feature
derived feature
semantic feature
abstract semantic feature
with or without interactions
Explanator output type
by example (e.g. closest other samples, word cloud)
contrastive / counterfactual / near miss (e.g. adversarial ex.)
prototype (e.g. generated, concept vector)
feature importance
rule based (e.g. if-then, binary, m-of-n, hyperplane)
dimension reduction
dependence plots
graph
diagrams
combinations

Fig. 5. Overview on the taxonomy aspects related to the explanator that are detailed in subsection 5.3. Find a visualization of the complete taxonomy in Figure 7 on page 44.

final output; and uniqueness of the weights. Just as LIME, SHAP just requires a definition of "feature" or snippet on the input in order to be applicable.

model-specific also called *decompositional* [143] or white-box: This means that access is needed to the internal processing or architecture of the explanandum model, or even constraints apply.

Sensitivity Analysis [13]

Deconvnet [145]

Backprop [123]

Guided Backprop [127]

LRP [12]

PatternAttribution [78]

Examples. Methods relying on gradient or relevance information for generation of visual attention maps are strictly model-specific. A gradient-based method is Sensitivity Analysis [13]. They pick the vector representing the steepest ascend in the gradient tangential plane of a sample point. This method is independent of the type of input features, but can only analyse one one-dimensional output at once. Output-type-agnostic but dependent on a convolutional architecture and image inputs is Deconvnet [145] and its successors Backpropagation [123] and Guided Backpropagation [127]. They approximate a reconstruction of an input by defining inverses of pool and convolution operations, which allows to backpropagate the activation of single filters back to input image pixels (see [140] for a good overview). The idea of Backpropagation is generalized axiomatically by LRP [12] (Layer-wise Relevance Propagation): They require that the sum of linear relevance weights for each neuron in a layer should be constant throughout the layers (relevance is neither created nor extinguished from layer to layer). Methods that achieve this are, *e.g.*, Taylor decomposition or the back-propagation of relevance weighted by the forward-pass weights. The advancement PatternAttribution [78] fulfills the additional constraint to be sound on linear models.

hybrid also called *eclectic* [143] or *gray-box*: This means the explanator only depends on access to parts of the model intermediate output, but not the full architecture.

DeepRED [152]

Examples. The rule extraction technique DeepRED [152] (Deep Rule Extraction with Decision tree induction) is an example of an eclectic method, so neither fully model-agnostic nor totally reliant on access to model internals. The approach conducts a backwards induction over the layer outputs of a DNN, between each two applying a decision tree extraction. While they enable rule extraction for arbitrarily deep DNNs, only small networks will result in rules of decent length for explanations.

Explanation Locality. Literature differentiates between different ranges of validity of an explanation, respectively surrogate model. A surrogate model is valid in the ranges where high fidelity can be expected (see subsection 5.3). The range of input required by the explanator depends on the targeted validity range, so whether the input must representing a *local* or the *global* behavior of the explanandum. The general locality types are:

local An explanation is considered local if the explanator is valid in a neighborhood of one or a group of given (valid) input samples. Local explanations tackle the question of *why* a given decision for one or a group of examples was made.

Examples. Heatmapping methods are typical examples for local-only explanators, such as the discussed perturbation-based model-agnostic methods RISE [104], D-RISE [105], LIME [112], SHAP [88], as well as the model-specific sensitivity and backpropagation based methods LRP [12], PatternAttribution [78], Sensitivity Analysis [13], and Deconvnet and its successors [145,123,127].

global An explanation is considered global if the explanator is valid in the complete (valid) input space. Other than the *why* of local explanations, global interpretability can also be described as answering *how* a decision is made.

Explanatory Graphs [146]

Examples. A graph-based global explanator is generated by [146]. Their idea is that semantic concepts in an image usually consist of sub-objects to which they have a constant relative spatial relation (*e.g.*, a face has a nose in the middle and two eyes next to each other), and that the localization of concepts should not only rely on high filter activation patterns, but also on their sub-part arrangement. To achieve this, they translate the convolutional layers of a DNN into a tree of nodes (concepts), the *explanatory graph*. Each node belongs to one filter, is anchored at a fixed spatial position in the image, and represents a spatial arrangement of its child notes. The graph can also be used for local explanations via heatmaps: To localize a node in one input image, it is assigned the position closest to its anchor for which its filter activation is highest and for which the expected spatial relation to its children is best fulfilled. While most visualization based methods provide only local visualizations, a global, prototype-based, visual explanation is provided by Feature Visualizations [102]. The goal here is to visualize the function of a part of a DNN by finding prototypical input examples that strongly activate that part. These can be found via picking, search, or optimization. Other than visualizations, rule extraction methods usually only provide global approximations. An example is the well-known model-agnostic rule extractor VIA [130] (Validity Interval Analysis), which iteratively refines or generalizes pairs of input- and output-intervals. An example for getting from local to global explanations is SpRAy [83] (Spectral Relevance Analysis). They suggest to apply spectral clustering [135] to local feature attribution heatmaps of a data samples in order to find spuriously distinct global behavioral patterns. The heatmaps were generated via LRP [12].

Feature Visualization [102]

VIA [130]

SpRAy [83]

Output. The output is characterized by several aspects:

– what is explained (the *object of explanation*),

- how it is explained (the actual *output type*, also called *explanation content type*), and
- how it is *presented.*

Object of Explanation. The object (or scope [93]) of an explanation describes which item of the development process should be explained. Items we identified in literature:

processing The objective is to understand the (symbolic) processing pipeline of the model, *i.e.*, to answer parts of the question "How does the model work?". This is the usual case for model-agnostic analysis methods. Types of processing to describe are, *e.g.*, the *decision boundary*, and *feature attribution* (or feature importance). Note that these are closely related, as highly important features usually locally point out the direction to the decision boundary. In case a symbolic explanator is targeted, one may need to first find a symbolic representation of input, output, or the model internal representation. Note that model-agnostic methods that do not investigate the input data, usually target explanations of the model processing.

Examples. Feature attribution methods encompass all the discussed attribution heatmapping methods (*e.g.*, RISE [104], LIME [112], LRP [12]). LIME can be considered a corner case, as it both explains feature importance but also tries to approximate the decision boundary using a linear model on super-pixels, which can itself serve directly as an explanation. A typical way to describe decision boundaries are decision trees or sets of rules, like extracted by the discussed VIA [130], and DeepRED [152]. Standard candidates for model-agnostic decision tree extraction are TREPAN [32] for M-of-N rules at the split points, and the C4.5 [108] decision tree generator for shallower but wider trees with interval-based splitting points. Concept tree [111] is a recent extension of TREPAN that adds automatic grouping of correlated features into the candidate concepts to use for the tree nodes.

TREPAN [32]

C4.5 [108]

Concept Tree [111]

inner representation Machine learning models learn new representations of the input space, like the latent spaces representations found by DNNs. Explaining these inner representations answers "How does the model see the world?". A more fine-grained differentiation considers whether *layers*, *units*, or *vectors* in the feature space are explained.

Examples.

- units: One example of unit analysis is the discussed Feature Visualization [102]. In contrast to this unsupervised assignment of convolutional filters to prototypes, NetDissect [15] (Network Dissection) assigns filters to pre-defined semantic concepts in a supervised manner: For a filter, that semantic concept (color, texture, material, object, or object part) is selected for which the ground truth segmentation masks have the highest overlap with the upsampled filter's activations. The authors also suggest

NetDissect [15]

that concepts that are less entangled, so less distributed over filters, are more interpretable, which is measurable with their filter-to-concept-alignment technique.

– vectors: Other than NetDissect, Net2Vec [47] also wants to assign concepts to their possibly entangled representations in the latent space. For a concept, they learn a linear 1×1-convolution on the output of a layer, which segments the concept in an image. The weight vector of the linear model for a concept can be understood as a prototypical representation (embedding) for that concept in the DNN intermediate output. They found that such embeddings behave like vectors in a word vector space: Concepts that are semantically similar feature embeddings with high cosine similarity. Similar to Net2Vec, TCAV [75] (Testing Concept Activation Vectors) also aims to find embeddings of NetDissect concepts. They are interested in embeddings that are represented as a linear combination of convolutional filters, but in embedding vectors lying in the space of the complete layer output. In other words, they do not segment concepts but make an image-level classification whether the concept is present. These are found by using an SVM model instead of the 1×1-convolution. Additionally, they suggest to use partial derivatives along those concept vectors to find the local attribution of a semantic concept to a certain output. Other than the previous supervised methods, ACE [51] (Automatic Concept-based Explanations) does not learn a linear classifier but does an unsupervised clustering of concept candidates in the latent space. The cluster center then is selected as embedding vector. A super-pixeling approach together with outlier removal are used to obtain concept candidates.

Net2Vec [47]

TCAV [75]

ACE [51]

– layers: The works of [144] and IIN [43] (invertible interpretation networks) extend on the previous approaches and analyse a complete layer output space at once. For this, they find a subspace with a basis of concept embeddings, which allows an invertible transformation to a disentangled representation space. While IIN use invertible DNNs for the bijection of concept to latent space, [144] linear maps in their experiments. These approaches can be seen as a post-hoc version of the Semantic Bottleneck [87] architecture, only not replacing the complete later part of the model, but just learning connections from the bottleneck to the succeeding trained layer. [144] additionally introduces the notion of completeness of a set of concepts as the maximum performance of the model intercepted by the semantic bottleneck.

Concept completeness [144]

IIN [43]

development (during training) Some methods focus on assessing effects during training [93, Sec. 2.3]: "How does the model evolve during the training? What effects do new samples have?"

Examples. One example is the work of [122], who inspect the model

[122]

during training to investigate the role of depth for neural networks. Their findings indicate that depth actually is of computational benefit. An example which can be used to provide, *e.g.*, prototypical explanations are Influence Functions [79]. They gather the influence of training samples during the training to later assess the total impact of samples to the training. They also suggest to use this information as a proxy to estimate the influence of the samples to model decisions.

Influence Functions [79]

uncertainty In [93] it is suggested to capture and explain (*e.g.*, visualize) the uncertainty of a prediction of the model. This encompasses the broad field of Bayesian deep learning [74] and uncertainty estimation [64]. It is argued in, *e.g.*, [106] for medical applications and in [90] for autonomous driving, why it is important to make the uncertainty of model decisions accessible to users.

data Pre-model interpretability [23] is the point where explainability touches the large research area of data analysis and feature mining.

Examples. Typical examples for projecting high-dimensional data into easy-to-visualize 2D space are component analysis methods like PCA (Principal Component Analysis) [72]. A slightly more sophisticated approach is t-SNE [89] (t-Distributed Stochastic Neighbor Embedding). In order to visualize a set of high-dimensional data points, they try to find a map from these points into a 2D or 3D space that is faithful on pairwise similarities. And also clustering methods can be used to generate prototype or example based explanations of typical features in the data. Examples here are k-means clustering [61] and the graph-based spectral clustering [135].

PCA [72]

t-SNE [89]

spectral clustering [135]

Output Type. The output type, also considered the actual explanator [57], describes the type of information presented to the explainee. Note that this ("what" is shown) is mostly independent of the presentation form ("how" it is shown). Typical types are:

by example instance *e.g.*, closest other samples, word cloud;

Examples. The discussed ProtoPNet [26] is based on selecting and comparing relevant example snippets from the input image data.

contrastive / counterfactual / near miss including adversarial examples;

Examples. The perturbation-based feature importance heatmapping approach of RISE is extended in CEM [37] (Contrastive, Blackbox Explanations Model). The do not only find positively contributing features, but also the features that must minimally be absent to not change the output.

CEM [37]

prototype *e.g.*, generated, concept vector;

Examples. A typical prototype generator is used in the discussed Feature Visualization [102] method: images are generated, *e.g.*, via gradient descent, that represent the prototypical pattern for activating a filter. While this considers prototypical inputs, concept embeddings as collected in TCAV [75] and Net2Vec [47] describe prototypical activation patterns for a given semantic concept. The concept mining approach ACE [51] combines prototypes with examples: They search a concept embedding as prototype for an automatically collected set of example patches, that can be used to explain the prototype.

feature importance that will highlight features with high attribution or influence on the output;

Examples. A lot of feature importance methods producing heatmaps have been discussed before (*e.g.*, RISE [104], D-RISE [105], CEM [37], LIME [112], SHAP [88], LRP [12], PatternAttribution [78], Sensitivity Analysis [13], Deconvnet and successors [145,123,127]). One further example is the work in [48], which follows a perturbation-based approach. Similar to RISE, their idea is to find a minimal occlusion mask that if used to perturb the image (*e.g.*, blur, noise, or blacken) maximally changes the outcome. To find the mask, backpropagation is used, making it a model-specific method. Some older but popular and simpler example methods are Grad-CAM [121] and its predecessor CAM [149] (Class Activation Mapping). While Deconvnet and its successors can only consider the feature importance with respect to intermediate outputs, (Grad-)CAM produces class-specific heatmaps, which are the weighted sum of the filter activation maps for one (usually the last) convolutional layer. For CAM, it is assumed the convolutional backend is finalized by a global average pooling layer that densely connects to the final classification output. Here, the weights in the sum are the weights connecting the neurons of the global average pooling layer to the class outputs. For Grad-CAM, the weights in the sum are the averaged derivation of the class output by each activation map pixel. This is also used in the more recent [150], who do not apply Grad-CAM directly to the output but to each of a minimal set of projections from a convolutional intermediate output of a DNN that predict semantic concepts. Similar to Grad-CAM, SIDU [94] (Similarity Distance and Uniqueness) also adds up the filter-wise weighted activations of the last convolutional layer. The weights encompass a combination of a similarity score and a uniqueness score for the prediction output under each filter activation mask. The scores aim for high similarity of a masked predictions with the original one and low similarity to the other masked prediction, leading to masks capturing more complete and interesting object regions.

[48]

CAM [149]

Grad-CAM [121]

Concept-wise Grad-CAM [150]

SIDU [94]

rule based *e.g.*, decision tree; or if-then, binary, m-of-n, or hyperplane rules (cf. [60]);

Examples. The mentioned exemplary rule-extraction methods DeepRED [152] and VIA [130], as well as decision tree extractors TRE-

PAN [32], Concept Tree [111] and C4.5 [108] all provide global, rule-based output. For further rule extraction examples we refer the reader to the comprehensive surveys [60,138,11] on the topic, and

LIME-Aleph [110]

the survey [139] for recurrent DNNs. An example of a local rule-extractor is the recent LIME-Aleph [110] approach, which generates a local explanation in the form of first-order logic rules. This is learned using inductive logic programming (ILP) [33] trained on the symbolic knowledge about a set of semantically similar examples. Due to the use of ILP, the approach is limited to tabular input data

NBDT [136]

and classification outputs, but just as LIME it is model-agnostic. A similar approach is followed by NBDT [136] (Neural-Backed Decision Trees). They assume that the concept embeddings of super-categories are represented by the mean of their sub-category vectors (*e.g.*, the mean of "cat" and "dog" should be "animal with four legs"). This is used to infer from bottom-to-top a decision tree where the nodes are super-categories and the leaves are the classification classes. At each node it is decided which of the sub-nodes best applies to the image. As embedding for a leaf concept (an output classes) they suggest to take the weights connecting the penultimate layer to a class output, and as similarity measure for the categories they use dot-product (cf. Net2Vec and TCAV).

dimension reduction *i.e.*, sample points are projected to a sub-space;

Examples. Typical dimensionality reduction methods mentioned previously are PCA [72] and t-SNE [89].

dependence plots which plot the effect of an input feature on the final output of a model;

PDP [50]

Examples. PDP [50] (Partial Dependency Plots, cf. [93, sec. 5.1]) calculate for one input feature and each vallue of this feature the expected model outcome averaged over the dataset. This results in a plot (for elach output) that indicates the global influence of the

ICE [55]

respective feature on the model. The local equivalent, ICE [55] (Individual Conditional Expectation, cf. [93, sec. 5.2]) plots, obtain the PDP for generated data samples locally around a given sample.

graphs as of *e.g.* [95,96,85];

Examples. The previously discussed Explanatory Graph [146] method provides amongst others a graph-based explanation output.

combinations of mentioned model types.

Presentation. The presentation of information can be characterized by two categories of properties: the used *presentation form*, and the *level of abstraction* used to present available information. The presentation form simply summarizes the human sensory input channels utilized by the explanation, which can be: visual (the most common one including diagrams, graphs, and heatmaps), textual in either natural language or formal form, auditive, and combinations thereof. In the following the aspects influencing the level of abstraction are elaborated.

These can be split up into (1) aspects of the smallest building blocks of the explanation, the *information units*, and (2) the *accessibility* or level of *complexity* of their combinations (the information units). Lastly, further filtering may be applied before finally presenting the explanation, including privacy filters.

information units The basic units of the explanation, cognitive chunks [39], or information unit, may differ in the level or processing applied to them. The simplest form are unprocessed *raw features*, as used in explanations by example. *Derived features* capture some indirect information contained in the raw inputs, like super-pixels or attention heatmaps. These need not necessarily have a semantic meaning to the explainee, other than explicitly *semantic features*, *e.g.*, concept activation vector attributions. The last type of information units are *abstract semantic features* not directly grounded in any input, *e.g.*, prototypes. *Feature interactions* may occur as information units or be left unconsidered for the explanation.

Examples. Some further notable examples of heatmapping methods for feature attribution are SmoothGrad [125] and Integrated Gradients [128]. One drawback of the methods described so far is that the considered loss surfaces that are linearly approximated tend to be "rough", *i.e.*, exhibit significant variation in the point-wise values, gradients, and thus feature importance [116]. SmoothGrad [125] aims to mitigate this by averaging the gradient from random samples within a ball around the sample to investigate. Integrated gradients [128] do the averaging (to be precise: integration) along a path between two points in the input space. A technically similar approach but with a different goal is Integrated Hessians [71]. They intend not to grasp and visualize the sensitivity of the model for one feature (as derived feature), but their information units are interactions of features, *i.e.*, how much the change of one feature changes the influence of the other on the output. This is done by having a look at the Hessian matrix, which is obtained by two subsequent integrated gradient calculations.

SmoothGrad [125]

Integrated Gradients [128]

Integrated Hessians [71]

accessibility The accessibility, level of detail, or level of complexity describes how much intellectual effort the explainee has to bring up in order to understand the simulatable parts of the explanation. Thus, the perception of complexity heavily depends on the end-user, which is mirrored in the human-grounded complexity / interpretability metric discussed later in subsection 5.3. In general, one can differentiate between representations that are considered *simpler*, and such that are more *expressive but complex.* Because accessibility refers to the simulatable parts, this differs from the decomposable transparency level: For example, very large decision trees or very high-dimensional (general) linear models may be perceived as globally complex by the end-user. However, when looking at the simulatable parts of the explanator, like small groups of features or nodes, they are easy to grasp.

Examples. Accessibility can indirectly be assessed by the complexity and expressivity of the explanation (see subsection 5.3). To give some examples: Simple presentations are, *e.g.*, linear models, general additive models, decision trees and Boolean decision rules, Bayesian models, or clusters of examples (cf. subsection 5.1); generally more complex are, *e.g.*, first-order or fuzzy logical decision rules.

privacy awareness Sensible information like names may be contained in parts of the explanation, even though they are not necessary for understanding the actual decision. In such cases, an important point is privacy awareness [22]: Is sensible information removed if unnecessary, or properly anonymized if needed?

Interactivity. The interaction of the user with the explanator may either be static, so the explainee is once presented with an explanation, or interactive, meaning an iterative process accepting user feedback as explanation input. Interactivity is characterized by the *interaction task* and the *explanation process.*

interaction task The user can either inspect explanations or *correct* them. Inspecting takes place through *exploration* of different parts of one explanation or through consideration of various alternatives and complementing explanations, such as implemented in the *iNNvestigate* toolbox [3]. Besides, the user can be empowered within the human-AI partnership to provide corrective feedback to the system via an explanation interface, in order to adapt the explanator and thus the explanandum.

Examples. State-of-the-art systems

- CAIPI [129] enable the user to perform *corrections on labels* and to act upon wrong explanations through interactive machine learning (intML), such as implemented in the CAIPI approach [129],
- EluciDebug [81] they allow for *re-weighting of features* for explanatory debugging, like the EluciDebug system [81],
- Crayons [44] *adaption of features* as provided by Crayons [44], and
- LearnWithME [119] correcting generated verbal explanations through user-defined constraints, such as implemented in the medical-decision support system LearnWithME [119].

explanation process As mentioned above explanation usually takes place in an iterative fashion. Sequential analysis allows the user to query further information in an iterative manner and to understand the model and its decisions over time, in accordance with the users' capabilities and the given context [42,46].

Multi-modal explanations [63]

Examples. This includes combining different methods to create multi-modal explanations and involving the user into a dialogue, such as realized through a phrase-critic model as presented in [63] or with the help of an explanatory dialogue such as proposed by [46].

Mathematical Constraints. Mathematical constraints encode some formal properties of the explanator that were found to be helpful for explanation receival. Constraints mentioned in literature are:

linearity Considering a concrete proxy model as explanator output, linearity is often considered as a desirable form of simplicity [75,23,93].

monotonicity Similar to linearity, one here again considers a concrete proxy model as output of the explanator. It is then considered a desirable level of simplicity if the dependency of that model's output on one input feature is monotonous.

satisfiability This is the case if the explantor outputs readily allow application of formal methods like solvers.

number of iterations While some XAI methods require a one-shot inference of the explanandum model (*e.g.*, gradient-based methods), others require several iterations of queries to the explanandum. Since these might be costly or even restricted in some use cases a limited number of iterations needed by the explanator may be desirable in some cases. Such restrictions may arise from non-gameability [82] constraints on the explanandum model, *i.e.*, the number of queries is restricted in order to guard against systematic optmization of outputs by users (*e.g.*, searching for adversaries).

size constraints Many explanator types respectively surrogate model types allow for architectural constraints, like size or disentanglement, that correlate with reduced complexity. See also the respective complexity / interpretability metrics in subsection 5.3.

> *Examples.* For linear models, sparsity can reduce complexity [53], for decision tree depth and width can be constrained. One common way to achieve sparsity, is to add regularization terms to the training of linear explanators and interpretable models.

5.3 Metrics

By now, there is a considerable amount of metrics suggested to assess the quality of XAI methods with respect to different goals. This section details the types of metrics considered in literature. Following the original suggestion in [39], we categorize metrics by their level of human involvement required to measure them:

- *functionally-grounded* metrics,
- *human-grounded* metrics, and
- *appplication-grounded* metrics.

An overview is given in Figure 6. For approaches to measure the below described metrics we refer the reader to [84]. They provide a good starting point with an in-depth analysis of metrics measurement for visual feature attribution methods.

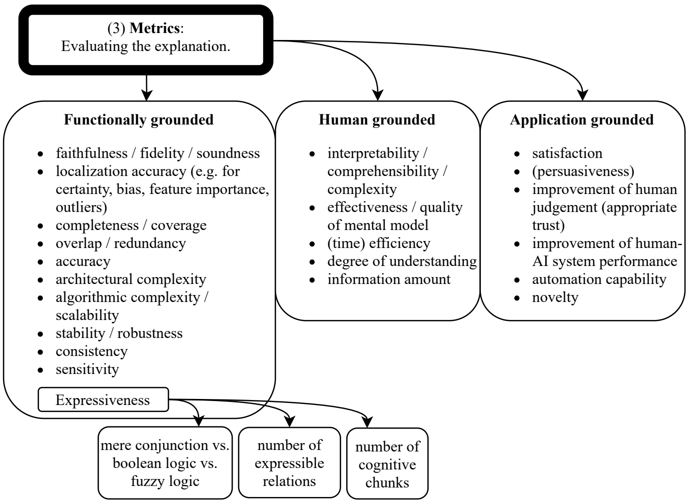

(3) **Metrics**:
Evaluating the explanation.
Functionally grounded
faithfulness / fidelity / soundness
localization accuracy (e.g. for certainty, bias, feature importance, outliers)
completeness / coverage
overlap / redundancy
accuracy
architectural complexity
algorithmic complexity / scalability
stability / robustness
consistency
sensitivity
Expressiveness
mere conjunction vs. boolean logic vs. fuzzy logic
number of expressible relations
number of cognitive chunks
Human grounded
interpretability / comprehensibility / complexity
effectiveness / quality of mental model
(time) efficiency
degree of understanding
information amount
Application grounded
satisfaction
(persuasiveness)
improvement of human judgement (appropriate trust)
improvement of human-AI system performance
automation capability
novelty

Fig. 6. Overview on the XAI metrics detailed in subsection 5.3. Find a visualization of the complete taxonomy in Figure 7 on page 44.

Functionally Grounded Metrics. Metrics are considered functionally grounded if the do not requiring any human feedback but instead measure formal properties of the explanator. This applies to the following metrics:

Faithfulness [84] or fidelity [23], soundness [143], causality [22], measures how accurately the behavior of the explantor conforms with that of the actual object of explanation. If a full surrogate model is used, this is the accuracy of the surrogate model outputs with respect to the explanandum outputs. And the fidelity of inherently interpretable models serving also as explanator is naturally 100%. Note that fidelity is more often used if the explanator consists of a complete surrogate model (*e.g.*, a linear proxy like LIME [112]), and faithfulness in more general contexts [20]. Other works use the terms interchangeably like [41,23], which we adopt here. More simplification usually comes along with less faithfulness, since corner cases are not captured anymore, also called the *fidelity-interpretability trade-off.*

Localization accuracy with respect to some ground truth (cf. [84] for visual feature importance maps) means how well an explanation points out respectively correctly localizes certain points of interest. These points of interest must be given by a ground truth that is preferrably provided by mathematical properties, such as certainty, bias, feature importance, and outliers (cf. [23]). How such points are highlighted for the explainee depends on the explanation type. For example, they could be highlighted in a feature importance heatmap [84] or expressed by the aggregated relevance within regions of interest [113]. Note that localization capability is closely related to faithfulness, but refers to specific properties of interest.

Completeness or coverage measures how large the validity range of an explanation is, so in which subset of the input space high fidelity can be expected. It can be seen as a generalization of fidelity to the distribution of fidelity. Note that coverage can also be calculated for parts of an explanation such as single rules of a rule set, as considered in [20].

Overlap is considered in [20] for rule-based explanations. It measures the number of data samples that satisfy more than one rule of the rule set, *i.e.*, measures the size of areas where the validity ranges of different rules overlap. This can be seen as a measure for redundancy in the rule set, and may sometimes correlate with perceived complexity [20].

Accuracy ignores the prediction quality of the original model and only considers the prediction quality of the surrogate model for the original task. This only applies to post-hoc explanations.

Architectural complexity can be measured using metrics specific to the explanator type. The goal is to approximate the subjective, human-grounded complexity that a human perceives, and do this using purely architectural properties like size measures. Such architectural metrics can be, e.g., the number of used input features, the sparsity of linear models [53], the width or depth of decision trees [20], or, in case of rules, the number of defined rules and the number of unique used predicates [20].

Algorithmic complexity and scalability measure the information theoretic complexity of the algorithm used to derive the explanator. This includes the time to convergence (to an acceptable solution), and is especially interesting for complex approximation schemes like rule extraction.

Stability or robustness [22] measures the change of explanator (output) given a change on the input samples. This corresponds to (adversarial) robustness of deep neural networks and a stable algorithm is usually also better comprehensible and desirable. Stability makes most sense for local methods.

Consistency measures the change of the explanator (output) given a change on the model to explain. The idea behind consistency is that functionally equivalent models should produce the same explanation. This assumption is important for model-agnostic approaches, while for model-specific ones a dependency on the model architecture may even be desirable. (*e.g.*, for architecture visualization).

Sensitivity measures whether local explanations change if the model output changes strongly. The intuition behind this is that a strong change in the model output usually comes along with a change in the discrimination strategy of the model between the differing samples [84]. Such changes should be reflected in the explanations. Note that this may be in conflict with stability goals for regions in which the explanandum model behaves chaotically.

Expressiveness or the level of detail refers to the level of detail of the formal language used by the explanator. It is interested in approximating the expected information density perceived by the user. It is closely related to the level of abstraction of the presentation. Several functionally grounded proxies were suggested to obtain comparable measures for expressivity:

- the depth or amount of *added information*, also measured as the mean number of used information units per explanation;
- *number of relations* that can be expressed; and
- the *expressiveness category* of used rules, namely mere conjunction, boolean logic, first-order logic, or fuzzy rules (cf. [143]).

Human Grounded Metrics. Other than functionally grounded metrics, human grounded metrics require to involve a human directly on proxy tasks for their measurement. Human involvement can be observation of a person's reactions, but also direct human feedback. Often, proxy tasks are considered instead of the final application to avoid a need for expensive experts or application runtime (think of medical domains). The goal of an explanation always is that the receiver of the explanation can build a *mental model* of (aspects of) the object of explanation. Human grounded metrics aim to measure some fundamental psychological properties of the XAI methods, namely quality of the *mental model.* The following are counted as such in literature:

Interpretability or comprehensibility or complexity measures how accurately the mental model approximates the explanator model. This measure mostly relies on subjective user feedback whether they "could make sense" of the presented information. It depends on background knowledge, biases, and cognition of the subject and can reveal use of vocabulary inappropriate to the user [52].

Effectiveness measures how accurately the mental model approximates the object of explanation. In other words, one is interested in how well a human can simulate the (aspects of interest of the) object after being presented with the explanations. Proxies for the effectiveness can be fidelity and accessibility [93, Sec. 2.4]. This may serve as a proxy for interpretability.

(Time) efficiency measures how time efficient an explanation is, *i.e.*, how long it takes a user to build up a viable mental model. This is especially of interest in applications with a limited time frame for user reaction, like product recommendation systems [101] or automated driving applications [77].

Degree of understanding measures in interactive contexts the current status of understanding. It helps to estimate the remaining time or measures needed to reach the desired extend of the explainee's mental model.

Information amount measures the total subjective amount of information conveyed by one explanation. Even though this may be measured on an information theoretic basis, it usually is subjective and thus requires human feedback. Functionally grounded related metrics are the (architectural) complexity of the object of explanation, together with fidelity, and coverage. For example, more complex models have a tendency to contain more information, and thus require more complex explanations if to be approximated widely and accurately.

Application Grounded Metrics. Other than human grounded metrics, application grounded ones work on human feedback for the final application. The following metrics are considered application grounded:

Satisfaction measures the direct content of the explainee with the system, so implicitly measures the benefit of explanations for the explanation system user.

Persuasiveness assesses the capability of the explanations to nudge an explainee into a certain direction. This is foremostly considered in recommendation systems [101], but has high importance when it comes to analysis tasks, where false positives and false negatives of the human-AI system are undesirable. In this context, a high persuasiveness may indicate a miscalibration of indicativeness.

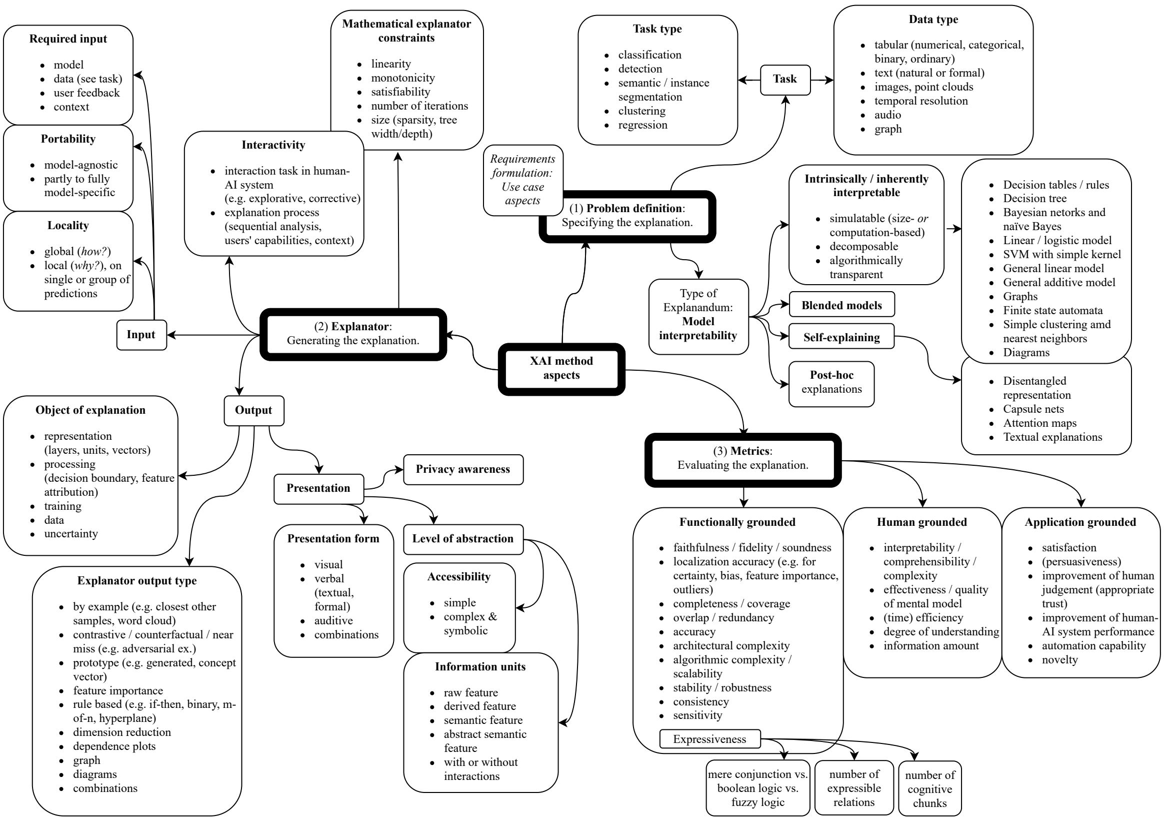
Improvement of human judgement [96] measures whether the explanation system user develops an appropriate level of trust in the decisions of the explained model. Correct decisions should be trusted more than wrong decisions, *e.g.*, because explanations of wrong decisions are illogical.

Improvement of human-AI system performance considers the end-to-end task to be achieved by all of explanandum, explainee, and explanator. This can, *e.g.*, be the diagnosis quality of doctors assisted by a recommendation system.

Improvement of human-AI-system performance [96] directly measures the increase of performance of the given human-machine system when adding explanations. For example, explanations of a medical assistant system may help the doctor to better find and correct errors of the assistant, leading to fewer wrong diagnoses.

Automation capability gives an estimate on how much of the manual work conducted by the human in the human-AI system can be be automatized. Especially for local explanation techniques automation may be an important factor for feasibility if the number of samples a human needs to scan can be drastically reduced.

Novelty estimates the subjective degree of novelty of information provided to the explainee [82]. This is closely related to efficiency and satisfaction: Especially in exploratory use cases, high novelty can drastically increase efficiency (no repetitive work for the explainee), and keep satisfaction high (decrease the possibility for boredom of the explainee).



XAI method aspects
(1) Problem definition: Specifying the explanation.
Requirements formulation: Use case aspects
Task
Task type
classification
detection
semantic / instance segmentation
clustering
regression
Data type
tabular (numerical, categorical, binary, ordinary)
text (natural or formal)
images, point clouds
temporal resolution
audio
graph
Type of Explanandum: Model interpretability
Intrinsically / inherently interpretable
simulatable (size- or computation-based)
decomposable
algorithmically transparent
Decision tables / rules
Decision tree
Bayesian netorks and naïve Bayes
Linear / logistic model
SVM with simple kernel
General linear model
General additive model
Graphs
Finite state automata
Simple clustering amd nearest neighbors
Diagrams
Blended models
Self-explaining
Disentangled representation
Capsule nets
Attention maps
Textual explanations
Post-hoc explanations
(2) Explanator: Generating the explanation.
Mathematical explanator constraints
linearity
monotonicity
satisfiability
number of iterations
size (sparsity, tree width/depth)
Interactivity
interaction task in human-AI system (e.g. explorative, corrective)
explanation process (sequential analysis, users' capabilities, context)
Input
Required input
model
data (see task)
user feedback
context
Portability
model-agnostic
partly to fully model-specific
Locality
global (how?)
local (why?), on single or group of predictions
Output
Object of explanation
representation (layers, units, vectors)
processing (decision boundary, feature attribution)
training
data
uncertainty
Explanator output type
by example (e.g. closest other samples, word cloud)
contrastive / counterfactual / near miss (e.g. adversarial ex.)
prototype (e.g. generated, concept vector)
feature importance
rule based (e.g. if-then, binary, m-of-n, hyperplane)
dimension reduction
dependence plots
graph
diagrams
combinations
Presentation
Privacy awareness
Presentation form
visual
verbal (textual, formal)
auditive
combinations
Level of abstraction
Accessibility
simple
complex & symbolic
Information units
raw feature
derived feature
semantic feature
abstract semantic feature
with or without interactions
(3) Metrics: Evaluating the explanation.
Functionally grounded
faithfulness / fidelity / soundness
localization accuracy (e.g. for certainty, bias, feature importance, outliers)
completeness / coverage
overlap / redundancy
accuracy
architectural complexity
algorithmic complexity / scalability
stability / robustness
consistency
sensitivity
Expressiveness
mere conjunction vs. boolean logic vs. fuzzy logic
number of expressible relations
number of cognitive chunks
Human grounded
interpretability / comprehensibility / complexity
effectiveness / quality of mental model
(time) efficiency
degree of understanding
information amount
Application grounded
satisfaction
(persuasiveness)
improvement of human judgement (appropriate trust)
improvement of human-AI system performance
automation capability
novelty

Fig. 7. Overview of the complete taxonomy that is detailed in section 5

Table 6: Review of an exemplary selection of XAI techniques according to the defined taxonomy aspects (without inherently transparent models from section 5.1). Abbreviations by column: *image data*=img, *point cloud data*=pcl; *Trans.*=transparency, *post-hoc*=p, *transparent*=t, *self-explaining*=s, *blended*=b; *processing*=p, *representation*=r, *development during training*=t *data*=d; *visual*=vis, *symbolic*=sym, *plot*=plt; *feature importance*=fi, *contrastive*=con, *prototypical*=proto, *decision tree*=tree, *distribution*=dist

Name	**Cite**	**Task**	**Model-agnostic?**	**Transp.**	**Global?**	**Obj. Expl.**	**Form**	**Type**
Self-explaining and blended models								
-	[62]	cls		s		p	sym/vis	rules/fi
-	[77]	any		s		p	sym/vis	rules/fi
ProtoPNet	[26]	cls,img		s		p/r	vis	proto/fi
Capsule Nets	[115]	cls		s		r	sym	fi
Semantic Bottlenecks, ReNN, Concept Whitening	[87,137,28]	any		s		r	sym	fi
Logic Tensor Nets	[38]	any		b	✓	p/r	sym	rule
FoldingNet	[142]	any,pcl		b		p	vis	fi/red
Neuralized clustering	[73]	any		b		p	vis	fi
Black-box heatmapping								
LIME, SHAP	[112,88]	cls	✓	p		p	vis	fi/con
RISE	[104]	cls,img	✓	p		p	vis	fi
D-RISE	[105]	det,img	✓	p		p	vis	fi
CEM	[37]	cls,img	✓	p		p	vis	fi/con
White-box heatmapping								
Sensitivity analysis	[13]	cls		p		p	vis	fi
Deconvnet, (Guided) Backprop.	[145,123,127]	img		p		p	vis	fi
CAM, Grad-CAM	[149,121]	cls,img		p		p	vis	fi
SIDU	[94]	cls,img		p		p	vis	fi
Concept-wise Grad-CAM	[150]	cls,img		p		p/r	vis	fi
SIDU	[94]	cls,img		p		p	vis	fi
LRP	[12]	cls		p		p	vis	fi
Pattern Attribution	[78]	cls		p		p	vis	fi
-	[48]	cls		p		p	vis	fi
SmoothGrad, Integrated Gradients	[125,128]	cls		p		p	vis	fi
Integrated Hessians	[71]	cls		p		p	vis	fi
Global representation analysis								
Feature Visualization	[102]	img		p	✓	r	vis	proto
NetDissect	[15]	img		p	✓	r	vis	proto/fi
Net2Vec	[47]	img		p	(✓)	r	vis	fi
TCAV	[75]	any		p	✓	r	vis	fi
ACE	[51]	any		p	✓	r	vis	fi
-	[144]	any		p	✓	r	vis	proto

Table 6: continued from page 45

Name	**Cite**	**Task**	**Model-agnostic?**	**Transp.**	**Global?**	**Obj. Expl.**	**Form**	**Type**
IIN	[43]	any		p	(✓)	r	vis/sym	fi
Explanatory Graph	[146]	img		p	(✓)	p/r	vis	graph
Dependency plots								
PDP	[50]	any	✓	p		p	vis	plt
ICE	[55]	any	✓	p	✓	p	vis	plt
Rule extraction								
TREPAN, C4.5, Concept Tree	[32,108,111]	cls	✓	p	✓	p	sym	tree
VIA	[130]	cls	✓	p	✓	p	sym	rules
DeepRED	[152]	cls		p	✓	p	sym	rules
LIME-Aleph	[110]	cls	✓	p		p	sym	rules
CA-ILP	[109]	cls		p	✓	p	sym	rules
NBDT	[136]	cls		p	✓	p	sym	tree
Interactivity								
CAIPI	[129]	cls,img	✓	p		r	vis	fi/con
EluciDebug	[81]	cls	✓	p		r	vis	fi,plt
Crayons	[44]	cls,img	✓	t		p	vis	plt
LearnWithME	[119]	cls	✓	t	✓	p, r	sym	rules
Multi-modal phrase-critic model	[63]	cls,img		p	✓	p	vis,sym	plt,rules
Inspection of the training								
-	[122]	any		p	✓	t	vis	dist
Influence functions	[79]	cls		p	✓	t	vis	fi/dist
Data analysis methods								
t-SNE, PCA	[89,72]	any	✓	p	✓	d	vis	red
k-means, spectral clustering	[61,135]	any	✓	p	✓	d	vis	proto

6 Discussion and Conclusion

The abundance of existing literature on structuring the field and methods of XAI has by now reached an overwhelming level for beginners and practitioners. To help in finding good starting points for a deeper dive, we here presented a rich and structured survey of surveys on XAI topics. This showed an increasing breadth of application fields and method types investigated in order to provide explainable yet accurate learned models. Especially, the exponentially increasing number of both XAI methods and method surveys suggests an increasing interest in XAI unrestrained growth of the research field in the upcoming years. As some persisting trends we found the application domains of medicine and recommendation systems, as well as (visual) explanations for visual models. Some upcoming or re-awakening trends seem to be the field of natural language processing, and rule-based explanation methods.

With the ever-growing amount of methods for achieving explainability and interpretability, XAI method surveys developed many approaches for method categorization. Early foundational concepts have been extended and sub-structured to increasingly detailed collections of aspects for differentiating XAI methods. This paper systematically united aspects scattered over existing literature into an overarching taxonomy structure. Starting from the definition of the problem of XAI, we found the following three main parts of an explanation procedure suitable for a top-level taxonomy structure: the task, the explainer, and evaluation metrics. This paper in detail defines and sub-structures each of these parts. The applicability of those taxonomy aspects for differentiating methods is evidenced practically: Numerous example methods from the most relevant as well as the most recent research literature are discussed and categorized according to the taxonomy aspects. An overview on the examples is given in the end, concentrating on the seven classification criteria that are most significant in the literature. These concretely are the *task*, the form of *interpretability* (*e.g.* inherently interpretable), whether the method is *model-agnostic or model-specific*, whether it generates *global or local* explanations, what the *object of explanation* is, in what form explanations are *presented*, and the type of explanation.

As highlighted in the course of the paper, the creation of an explanation system should be tailored tightly to the use-case: This holds for all of development, application, and evaluation of XAI the system. Concretely, the different stakeholders and their contexts should be taken into account [82,53]. Our proposed taxonomy may serve as a profound basis to analyze stakeholder needs and formulate concrete, use-case specific requirements upon the explanation system. The provided survey of surveys then provides the entry point when looking for XAI methods fulfilling the derived requirements.

Altogether, our proposed unified taxonomy and our survey allow (a) beginners to gain an easy and targeted overview and entry point to the field of XAI; (b) practitioners to formulate sufficient requirements for their explanation use–case and to find accordingly suitable XAI methods; and lastly (c) researchers to properly position their research efforts on XAI methods and identify potential gaps. We hope this work fosters research and fruitful application of XAI.

Finally, it must be mentioned that our survey is and will not be or stay complete: Despite our aim for a broad representation of the XAI field, the used structured literature search is naturally biased by the current interests in specific sub-fields of AI and the search terms. On the other hand, we hope this ensures relevance to a large part of the research community. Furthermore, we concentrated on a broadly applicable taxonomy for XAI methods, whilst subfields of XAI may prefer or come up with more detailed differentiation aspects or a different taxonomy structure.

We are looking forward to see future updates of the state-of-research captured in this work, and practical guides for choosing the right XAI method according to a use-case definition.

Acknowledgments

We would like to thank Christian Hellert as well as the anonymous reviewers for their detailed and valuable feedback. The research leading to these results is partly funded by the BMBF ML-3 project Transparent Medical Expert Companion (TraMeExCo), FKZ 01IS18056 B, 2018–2021, and by the German Federal Ministry for Economic Affairs and Energy within the project "KI Wissen – Automotive AI powered by Knowledge". We would like to thank the consortium for the successful cooperation.

References

1. Adadi, A., Berrada, M.: Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). In: IEEE Access. vol. 6, pp. 52138–52160 (2018). https://doi.org/10.1109/ACCESS.2018.2870052
2. Alber, M.: Software and Application Patterns for Explanation Methods. In: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, pp. 399–433. Lecture Notes in Computer Science, Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-28954-6_22
3. Alber, M., Lapuschkin, S., Seegerer, P., Hägele, M., Schütt, K.T., Montavon, G., Samek, W., Müller, K.R., Dähne, S., Kindermans, P.J.: iNNvestigate Neural Networks! J. Mach. Learn. Res. **20**(93), 1–8 (2019), http://jmlr.org/papers/v20/18-540.html
4. Allahyari, H., Lavesson, N.: User-oriented Assessment of Classification Model Understandability. In: 11th Scandinavian Conference on Artificial Intelligence. IOS Press (2011), http://urn.kb.se/resolve?urn=urn:nbn:se:bth-7559
5. Altman, N.S.: An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician **46**(3), 175–185 (Aug 1992). https://doi.org/10.1080/00031305.1992.10475879, https://www.tandfonline.com/doi/abs/10.1080/00031305.1992.10475879
6. Amershi, S., Cakmak, M., Knox, W.B., Kulesza, T.: Power to the People: The Role of Humans in Interactive Machine Learning. AI Mag. **35**(4), 105–120 (Dec 2014). https://doi.org/10.1609/aimag.v35i4.2513
7. Ancona, M., Ceolini, E., Öztireli, C., Gross, M.: Gradient-Based Attribution Methods. In: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, pp. 169–191. Lecture Notes in Computer Science, Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-28954-6_9
8. Anjomshoae, S., Najjar, A., Calvaresi, D., Främling, K.: Explainable Agents and Robots : Results from a Systematic Literature Review. In: 18th Int. Conf. Autonomous Agents and Multiagent Systems (AAMAS 2019). pp. 1078–1088. International Foundation for Autonomous Agents and MultiAgent Systems (2019), http://urn.kb.se/resolve?urn=urn:nbn:se:umu:diva-158024
9. Arrieta, A.B., Rodríguez, N.D., Ser, J.D., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., Herrera, F.: Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Information Fusion **58**, 82–115 (2020). https://doi.org/10.1016/j.inffus.2019.12.012, http://www.sciencedirect.com/science/article/pii/S1566253519308103

10. Arya, V., Bellamy, R.K.E., Chen, P.Y., Dhurandhar, A., Hind, M., Hoffman, S.C., Houde, S., Liao, Q.V., Luss, R., Mojsilovic, A., Mourad, S., Pedemonte, P., Raghavendra, R., Richards, J.T., Sattigeri, P., Shanmugam, K., Singh, M., Varshney, K.R., Wei, D., Zhang, Y.: One explanation does not fit all: A toolkit and taxonomy of AI explainability techniques. CoRR **abs/1909.03012** (2019), http://arxiv.org/abs/1909.03012
11. Augasta, M.G., Kathirvalavakumar, T.: Rule extraction from neural networks — A comparative study. In: Proc. 2012 Int. Conf. Pattern Recognition, Informatics and Medical Engineering. pp. 404–408 (Mar 2012). https://doi.org/10.1109/ICPRIME.2012.6208380, https://ieeexplore.ieee.org/document/6208380
12. Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.R., Samek, W.: On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLOS ONE **10**(7), e0130140 (Jul 2015). https://doi.org/10.1371/journal.pone.0130140, https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0130140
13. Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., Müller, K.R.: How to explain individual classification decisions. Journal of Machine Learning Research **11**, 1803–1831 (Aug 2010), http://portal.acm.org/citation.cfm?id=1859912
14. Baniecki, H., Biecek, P.: The Grammar of Interactive Explanatory Model Analysis. ArXiv200500497 Cs Stat (Sep 2020), http://arxiv.org/abs/2005.00497
15. Bau, D., Zhou, B., Khosla, A., Oliva, A., Torralba, A.: Network dissection: Quantifying interpretability of deep visual representations. In: Proc. 2017 IEEE Conf. Comput. Vision and Pattern Recognition. pp. 3319–3327. IEEE Computer Society (2017). https://doi.org/10.1109/CVPR.2017.354
16. Belle, V.: Logic meets probability: Towards explainable ai systems for uncertain worlds. In: 26th International Joint Conference on Artificial Intelligence. pp. 5116–5120 (2017)
17. Benchekroun, O., Rahimi, A., Zhang, Q., Kodliuk, T.: The Need for Standardized Explainability. ArXiv201011273 Cs (Oct 2020), http://arxiv.org/abs/2010.11273
18. Biran, O., Cotton, C.V.: Explanation and justification in machine learning: A survey. In: Proc. IJCAI 2017 Workshop Explainable Artificial Intelligence (XAI) (2017), http://www.cs.columbia.edu/~orb/papers/xai_survey_paper_2017.pdf
19. Bruckert, S., Finzel, B., Schmid, U.: The next generation of medical decision support: A roadmap toward transparent expert companions. Frontiers in Artificial Intelligence **3**, 75 (2020)
20. Burkart, N., Huber, M.F.: A survey on the explainability of supervised machine learning. J. Artif. Intell. Res. **70**, 245–317 (Jan 2021). https://doi.org/10.1613/jair.1.12228
21. Byrne, R.M.J.: Counterfactuals in Explainable Artificial Intelligence (XAI): Evidence from Human Reasoning. In: Proc. 2019 Int. Joint Conf. Artificial Intelligence. pp. 6276–6282 (2019), https://www.ijcai.org/proceedings/2019/876
22. Calegari, R., Ciatto, G., Omicini, A.: On the integration of symbolic and sub-symbolic techniques for XAI: A survey. Intell. Artif. **14**(1), 7–32 (Jan 2020). https://doi.org/10.3233/IA-190036
23. Carvalho, D.V., Pereira, E.M., Cardoso, J.S.: Machine learning interpretability: A survey on methods and metrics. Electronics **8**(8), 832 (Aug 2019). https://doi.org/10.3390/electronics8080832

24. Chang, C.H., Tan, S., Lengerich, B., Goldenberg, A., Caruana, R.: How interpretable and trustworthy are GAMs? CoRR **abs/2006.06466** (Jun 2020), https://arxiv.org/abs/2006.06466
25. Chatzimparmpas, A., Martins, R.M., Jusufi, I., Kerren, A.: A survey of surveys on the use of visualization for interpreting machine learning models. Information Visualization **19**(3), 207–233 (Jul 2020). https://doi.org/10.1177/1473871620904671
26. Chen, C., Li, O., Tao, D., Barnett, A., Rudin, C., Su, J.: This looks like that: Deep learning for interpretable image recognition. In: Advances in Neural Information Processing Systems 32. vol. 32, pp. 8928–8939 (2019), https://proceedings.neurips.cc/paper/2019/hash/adf7ee2dcf142b0e11888e72b43fcb75-Abstract.html
27. Chen, R., Chen, H., Huang, G., Ren, J., Zhang, Q.: Explaining neural networks semantically and quantitatively. In: Proc. 2019 IEEE/CVF International Conference on Computer Vision. pp. 9186–9195. IEEE (Oct 2019). https://doi.org/10.1109/ICCV.2019.00928
28. Chen, Z., Bei, Y., Rudin, C.: Concept whitening for interpretable image recognition. CoRR **abs/2002.01650** (Feb 2020), https://arxiv.org/abs/2002.01650
29. Chromik, M., Schuessler, M.: A taxonomy for human subject evaluation of black-box explanations in XAI. In: Proc. Workshop Explainable Smart Systems for Algorithmic Transparency in Emerging Technologies. vol. Vol-2582, p. 7. CEUR-WS.org (2020)
30. Council, A.U.P.P.: Statement on algorithmic transparency and accountability. Commun. ACM (2017)
31. Craven, M.W., Shavlik, J.W.: Visualizing learning and computation in artificial neural networks. International journal on artificial intelligence tools **1**(03), 399–425 (1992)
32. Craven, M.W., Shavlik, J.W.: Extracting tree-structured representations of trained networks. In: Advances in Neural Information Processing Systems 8, NIPS, Denver, CO, USA, November 27-30, 1995. pp. 24–30. MIT Press (1995), http://papers.nips.cc/paper/1152-extracting-tree-structured-representations-of-trained-networks
33. Cropper, A., Dumancic, S., Muggleton, S.H.: Turning 30: New ideas in inductive logic programming. CoRR **abs/2002.11002** (2020), https://arxiv.org/abs/2002.11002
34. Danilevsky, M., Qian, K., Aharonov, R., Katsis, Y., Kawas, B., Sen, P.: A survey of the state of explainable ai for natural language processing. ArXiv201000711 Cs (Oct 2020), http://arxiv.org/abs/2010.00711
35. Das, A., Rad, P.: Opportunities and Challenges in Explainable Artificial Intelligence (XAI): A Survey. ArXiv200611371 Cs (Jun 2020), http://arxiv.org/abs/2006.11371
36. Dey, A.K.: Understanding and using context. Personal Ubiquitous Comput. **5**(1), 4–7 (Jan 2001). https://doi.org/10.1007/s007790170019
37. Dhurandhar, A., Chen, P.Y., Luss, R., Tu, C.C., Ting, P., Shanmugam, K., Das, P.: Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In: Advances in Neural Information Processing Systems 31. pp. 592–603. Curran Associates, Inc. (2018), http://papers.nips.cc/paper/7340-explanations-based-on-the-missing-towards-contrastive-explanations-with-pertinent-negatives.pdf
38. Donadello, I., Serafini, L., d'Avila Garcez, A.S.: Logic tensor networks for semantic image interpretation. In: Proc. 26th Int. Joint Conf. Artificial Intelligence. pp. 1596–1602. ijcai.org (2017). https://doi.org/10.24963/ijcai.2017/221

39. Doshi-Velez, F., Kim, B.: Towards a rigorous science of interpretable machine learning. arXiv e-prints **abs/1702.08608** (Feb 2017), http://adsabs.harvard.edu/abs/2017arXiv170208608D
40. Došilović, F.K., Brčić, M., Hlupić, N.: Explainable artificial intelligence: A survey. In: 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). pp. 0210–0215 (May 2018). https://doi.org/10.23919/MIPRO.2018.8400040
41. Du, M., Liu, N., Hu, X.: Techniques for interpretable machine learning. Commun. ACM **63**(1), 68–77 (Dec 2019). https://doi.org/10.1145/3359786
42. El-Assady, M., Jentner, W., Kehlbeck, R., Schlegel, U., Sevastjanova, R., Sperrle, F., Spinner, T., Keim, D.: Towards xai: Structuring the processes of explanations. In: ACM Workshop on Human-Centered Machine Learning (2019)
43. Esser, P., Rombach, R., Ommer, B.: A disentangling invertible interpretation network for explaining latent representations. In: Proc. 2020 IEEE Conf. Comput. Vision and Pattern Recognition. pp. 9220–9229. IEEE (Jun 2020). https://doi.org/10.1109/CVPR42600.2020.00924, https://openaccess.thecvf.com/content_CVPR_2020/papers/Esser_A_Disentangling_Invertible_Interpretation_Network_for_Explaining_Latent_Representations_CVPR_2020_paper.pdf
44. Fails, J.A., Olsen Jr, D.R.: Interactive machine learning. In: Proceedings of the 8th international conference on Intelligent user interfaces. pp. 39–45 (2003)
45. Ferreira, J.J., Monteiro, M.S.: What Are People Doing About XAI User Experience? a Survey on AI Explainability Research and Practice. In: Design, User Experience, and Usability. Design for Contemporary Interactive Environments. pp. 56–73. Lecture Notes in Computer Science, Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-49760-6_4
46. Finzel, B., Tafler, D.E., Scheele, S., Schmid, U.: Explanation as a process: User-centric construction of multi-level and multi-modal explanations. In: Edelkamp, S., Möller, R., Rueckert, E. (eds.) KI 2021: Advances in Artificial Intelligence. pp. 80–94. Springer International Publishing, Cham (2021)
47. Fong, R., Vedaldi, A.: Net2Vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. In: Proc. 2018 IEEE Conf. Comput. Vision and Pattern Recognition. pp. 8730–8738. IEEE Computer Society (2018). https://doi.org/10.1109/CVPR.2018.00910
48. Fong, R.C., Vedaldi, A.: Interpretable explanations of black boxes by meaningful perturbation. In: Proc. 2017 IEEE Int. Conf. on Comput. Vision. pp. 3449–3457. IEEE Computer Society (2017). https://doi.org/10.1109/ICCV.2017.371, http://arxiv.org/abs/1704.03296
49. Freitas, A.A.: Comprehensible classification models: A position paper. ACM SIGKDD Explorations Newsletter **15**(1), 1–10 (Mar 2014). https://doi.org/10.1145/2594473.2594475, https://doi.org/10.1145/2594473.2594475
50. Friedman, J.H.: Greedy function approximation: A gradient boosting machine. Ann. Stat. **29**(5), 1189–1232 (Oct 2001). https://doi.org/10.1214/aos/1013203451
51. Ghorbani, A., Wexler, J., Zou, J.Y., Kim, B.: Towards automatic concept-based explanations. In: Advances in Neural Information Processing Systems 32. pp. 9273–9282 (2019), http://papers.nips.cc/paper/9126-towards-automatic-concept-based-explanations
52. Gilpin, L.H., Bau, D., Yuan, B.Z., Bajwa, A., Specter, M., Kagal, L.: Explaining explanations: An overview of interpretability of machine learning. In: Proc. 5th

IEEE Int. Conf. Data Science and Advanced Analytics. pp. 80–89. IEEE (2018). https://doi.org/10.1109/DSAA.2018.00018

53. Gleicher, M.: A Framework for Considering Comprehensibility in Modeling. Big Data **4**(2), 75–88 (Jun 2016). https://doi.org/10.1089/big.2016.0007
54. Goebel, R., Chander, A., Holzinger, K., Lecue, F., Akata, Z., Stumpf, S., Kieseberg, P., Holzinger, A.: Explainable AI: The New 42? In: Machine Learning and Knowledge Extraction. pp. 295–303. Lecture Notes in Computer Science, Springer International Publishing (2018). https://doi.org/10.1007/978-3-319-99740-7_21
55. Goldstein, A., Kapelner, A., Bleich, J., Pitkin, E.: Peeking Inside the Black Box: Visualizing Statistical Learning With Plots of Individual Conditional Expectation. J. Comput. Graph. Stat. **24**(1), 44–65 (Jan 2015). https://doi.org/10.1080/10618600.2014.907095
56. Goodman, B., Flaxman, S.: European union regulations on algorithmic decision-making and a “right to explanation”. AIMag **38**(3), 50–57 (Oct 2017). https://doi.org/10.1609/aimag.v38i3.2741
57. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM Comput. Surv. **51**(5), 93:1–93:42 (Aug 2018). https://doi.org/10.1145/3236009
58. Gunning, D., Aha, D.: Darpa’s explainable artificial intelligence (xai) program. AI Magazine **40**(2), 44–58 (2019)
59. Gunning, D., Stefik, M., Choi, J., Miller, T., Stumpf, S., Yang, G.Z.: XAI—Explainable artificial intelligence. Sci. Robot. **4**(37) (Dec 2019). https://doi.org/10.1126/scirobotics.aay7120
60. Hailesilassie, T.: Rule extraction algorithm for deep neural networks: A review. CoRR **abs/1610.05267** (2016), http://arxiv.org/abs/1610.05267
61. Hartigan, J.A., Wong, M.A.: Algorithm AS 136: A k-means clustering algorithm. J. R. Stat. Soc. Ser. C Appl. Stat. **28**(1), 100–108 (1979). https://doi.org/10.2307/2346830
62. Hendricks, L.A., Akata, Z., Rohrbach, M., Donahue, J., Schiele, B., Darrell, T.: Generating visual explanations. In: Computer Vision – ECCV 2016. pp. 3–19. Lecture Notes in Computer Science, Springer International Publishing (2016). https://doi.org/10.1007/978-3-319-46493-0_1
63. Hendricks, L.A., Hu, R., Darrell, T., Akata, Z.: Grounding visual explanations. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 264–279 (2018)
64. Henne, M., Schwaiger, A., Roscher, K., Weiss, G.: Benchmarking uncertainty estimation methods for deep learning with safety-related metrics. In: Proc. Workshop Artificial Intelligence Safety. CEUR Workshop Proceedings, vol. 2560, pp. 83–90. CEUR-WS.org (2020), http://ceur-ws.org/Vol-2560/paper35.pdf
65. Heuillet, A., Couthouis, F., Díaz-Rodríguez, N.: Explainability in deep reinforcement learning. Knowledge-Based Systems **214**, 106685 (Feb 2021). https://doi.org/10.1016/j.knosys.2020.106685
66. Huysmans, J., Dejaeger, K., Mues, C., Vanthienen, J., Baesens, B.: An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. Decision Support Systems **51**(1), 141–154 (Apr 2011). https://doi.org/10.1016/j.dss.2010.12.003, https://www.sciencedirect.com/science/article/pii/S0167923610002368
67. Islam, S.R., Eberle, W., Ghafoor, S.K., Ahmed, M.: Explainable artificial intelligence approaches: A survey. ArXiv210109429 Cs (Jan 2021), http://arxiv.org/abs/2101.09429

68. ISO/TC 22 Road vehicles: ISO/TR 4804:2020: Road Vehicles — Safety and Cybersecurity for Automated Driving Systems — Design, Verification and Validation. International Organization for Standardization, first edn. (Dec 2020), https://www.iso.org/standard/80363.html
69. ISO/TC 22/SC 32: ISO 26262-6:2018(En): Road Vehicles — Functional Safety — Part 6: Product Development at the Software Level, ISO 26262:2018(En), vol. 6. International Organization for Standardization, second edn. (Dec 2018), https://www.iso.org/standard/68388.html
70. Jackson, P.: Introduction to Expert Systems. Addison-Wesley Longman Publishing Co., Inc., USA, 3rd edn. (1998)
71. Janizek, J.D., Sturmfels, P., Lee, S.I.: Explaining explanations: Axiomatic feature interactions for deep networks. CoRR **abs/2002.04138** (2020), https://arxiv.org/abs/2002.04138
72. Jolliffe, I.T.: Principal Component Analysis. Springer Series in Statistics, Springer-Verlag, second edn. (2002). https://doi.org/10.1007/b98835
73. Kauffmann, J., Esders, M., Montavon, G., Samek, W., Müller, K.R.: From Clustering to Cluster Explanations via Neural Networks. arXiv:1906.07633 [cs, stat] (Jun 2019), http://arxiv.org/abs/1906.07633, arXiv: 1906.07633
74. Kendall, A., Gal, Y.: What uncertainties do we need in Bayesian deep learning for computer vision? In: Advances in Neural Information Processing Systems 30. pp. 5580–5590 (2017), http://papers.nips.cc/paper/7141-what-uncertainties-do-we-need-in-bayesian-deep-learning-for-computer-vision
75. Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., Sayres, R.: Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In: Proc. 35th Int. Conf. Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 2668–2677. PMLR (Jul 2018), http://proceedings.mlr.press/v80/kim18d.html
76. Kim, J., Canny, J.F.: Interpretable learning for self-driving cars by visualizing causal attention. In: Proc. 2017 IEEE Int. Conf. Comput. Vision. pp. 2961–2969. IEEE Computer Society (2017). https://doi.org/10.1109/ICCV.2017.320
77. Kim, J., Rohrbach, A., Darrell, T., Canny, J.F., Akata, Z.: Textual explanations for self-driving vehicles. In: Proc. 15th European Conf. Comput. Vision, Part II. Lecture Notes in Computer Science, vol. 11206, pp. 577–593. Springer (2018). https://doi.org/10.1007/978-3-030-01216-8_35, http://arxiv.org/abs/1807.11546
78. Kindermans, P.J., Schütt, K.T., Alber, M., Müller, K.R., Erhan, D., Kim, B., Dähne, S.: Learning how to explain neural networks: PatternNet and PatternAttribution. In: Proc. 6th Int. Conf. on Learning Representations (Feb 2018), https://openreview.net/forum?id=Hkn7CBaTW
79. Koh, P.W., Liang, P.: Understanding Black-box Predictions via Influence Functions. In: Proc. 34th Int. Conf. Machine Learning. pp. 1885–1894. PMLR (Jul 2017), http://proceedings.mlr.press/v70/koh17a.html
80. Kulesza, T., Burnett, M., Wong, W.K., Stumpf, S.: Principles of explanatory debugging to personalize interactive machine learning. In: Proceedings of the 20th international conference on intelligent user interfaces. pp. 126–137 (2015)
81. Kulesza, T., Stumpf, S., Burnett, M., Wong, W.K., Riche, Y., Moore, T., Oberst, I., Shinsel, A., McIntosh, K.: Explanatory debugging: Supporting end-user debugging of machine-learned programs. In: 2010 IEEE Symposium on Visual Languages and Human-Centric Computing. pp. 41–48. IEEE (2010)
82. Langer, M., Oster, D., Speith, T., Hermanns, H., Kästner, L., Schmidt, E., Sesing, A., Baum, K.: What Do We Want From Explainable Artificial Intelli-

gence (XAI)? – A Stakeholder Perspective on XAI and a Conceptual Model Guiding Interdisciplinary XAI Research. Artificial Intelligence p. 103473 (Feb 2021). https://doi.org/10.1016/j.artint.2021.103473

83. Lapuschkin, S., Wäldchen, S., Binder, A., Montavon, G., Samek, W., Müller, K.R.: Unmasking Clever Hans predictors and assessing what machines really learn. Nat. Commun. **10**(1), 1096 (Mar 2019). https://doi.org/10.1038/s41467-019-08987-4
84. Li, X.H., Shi, Y., Li, H., Bai, W., Song, Y., Cao, C.C., Chen, L.: Quantitative Evaluations on Saliency Methods: An Experimental Study. ArXiv201215616 Cs (Dec 2020), http://arxiv.org/abs/2012.15616
85. Linardatos, P., Papastefanopoulos, V., Kotsiantis, S.: Explainable AI: A review of machine learning interpretability methods. Entropy **23**(1), 18 (Jan 2021). https://doi.org/10.3390/e23010018
86. Lipton, Z.C.: The mythos of model interpretability. Queue **16**(3), 31–57 (Jun 2018). https://doi.org/10.1145/3236386.3241340
87. Losch, M., Fritz, M., Schiele, B.: Interpretability beyond classification output: Semantic bottleneck networks. In: Proc. 3rd ACM Computer Science in Cars Symp. Extended Abstracts (Oct 2019), https://arxiv.org/pdf/1907.10882.pdf
88. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: Advances in Neural Information Processing Systems 30. pp. 4765–4774. Curran Associates, Inc. (2017), http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf
89. van der Maaten, L., Hinton, G.: Visualizing Data using t-SNE. J. Mach. Learn. Res. **9**(86), 2579–2605 (2008), http://jmlr.org/papers/v9/vandermaaten08a.html
90. McAllister, R., Gal, Y., Kendall, A., van der Wilk, M., Shah, A., Cipolla, R., Weller, A.: Concrete problems for autonomous vehicle safety: Advantages of Bayesian deep learning. In: Proc. 26th Int. Joint Conf. Artificial Intelligence. pp. 4745–4753 (2017), https://doi.org/10.24963/ijcai.2017/661
91. McCarthy, J.: Programs with Common Sense. In: Proceedings of the Teddington Conference on the Mechanisation of Thought Processes. pp. 77–84 (1958)
92. Miller, T.: Explanation in artificial intelligence: Insights from the social sciences. Artificial Intelligence **267**, 1–38 (Feb 2019). https://doi.org/10.1016/j.artint.2018.07.007
93. Molnar, C.: Interpretable Machine Learning. Lulu.com (2020), https://christophm.github.io/interpretable-ml-book/
94. Muddamsetty, S.M., Jahromi, M.N.S., Ciontos, A.E., Fenoy, L.M., Moeslund, T.B.: Introducing and assessing the explainable AI (XAI) method: SIDU. ArXiv210110710 Cs (Jan 2021), http://arxiv.org/abs/2101.10710
95. Mueller, S.T., Hoffman, R.R., Clancey, W., Emrey, A., Klein, G.: Explanation in human-AI systems: A literature meta-review, synopsis of key ideas and publications, and bibliography for explainable AI. ArXiv190201876 Cs (Feb 2019), https://arxiv.org/abs/1902.01876v1
96. Mueller, S.T., Veinott, E.S., Hoffman, R.R., Klein, G., Alam, L., Mamun, T., Clancey, W.J.: Principles of explanation in human-AI systems. CoRR **abs/2102.04972** (Feb 2021), https://arxiv.org/abs/2102.04972
97. Muggleton, S.H., Schmid, U., Zeller, C., Tamaddoni-Nezhad, A., Besold, T.: Ultra-strong machine learning: comprehensibility of programs learned with ilp. Machine Learning **107**(7), 1119–1140 (2018)
98. Murdoch, W.J., Singh, C., Kumbier, K., Abbasi-Asl, R., Yu, B.: Definitions, methods, and applications in interpretable machine learning. PNAS **116**(44), 22071–22080 (Oct 2019). https://doi.org/10.1073/pnas.1900654116

99. Nguyen, A., Yosinski, J., Clune, J.: Understanding Neural Networks via Feature Visualization: A Survey. In: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, pp. 55–76. Lecture Notes in Computer Science, Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-28954-6_4
100. Nori, H., Jenkins, S., Koch, P., Caruana, R.: InterpretML: A unified framework for machine learning interpretability. CoRR **abs/1909.09223** (Sep 2019), http://arxiv.org/abs/1909.09223
101. Nunes, I., Jannach, D.: A systematic review and taxonomy of explanations in decision support and recommender systems. User Modeling and User-Adapted Interaction **27**(3-5), 393–444 (Dec 2017). https://doi.org/10.1007/s11257-017-9195-0
102. Olah, C., Mordvintsev, A., Schubert, L.: Feature visualization. Distill **2**(11), e7 (Nov 2017). https://doi.org/10.23915/distill.00007
103. Páez, A.: The pragmatic turn in explainable artificial intelligence (xai). Minds and Machines **29**(3), 441–459 (2019)
104. Petsiuk, V., Das, A., Saenko, K.: RISE: Randomized input sampling for explanation of black-box models. In: Proc. British Machine Vision Conf. p. 151. BMVA Press (2018), http://bmvc2018.org/contents/papers/1064.pdf
105. Petsiuk, V., Jain, R., Manjunatha, V., Morariu, V.I., Mehra, A., Ordonez, V., Saenko, K.: Black-box explanation of object detectors via saliency maps. In: Proc. 2021 IEEE/CVF Conf. Comput. Vision and Pattern Recognition. pp. 11443–11452 (2021), https://openaccess.thecvf.com/content/CVPR2021/html/Petsiuk_Black-Box_Explanation_of_Object_Detectors_via_Saliency_Maps_CVPR_2021_paper.html
106. Pocevičiūtė, M., Eilertsen, G., Lundström, C.: Survey of XAI in digital pathology. Lect. Notes Comput. Sci. **2020**, 56–88 (2020). https://doi.org/10.1007/978-3-030-50402-1_4
107. Puiutta, E., Veith, E.M.S.P.: Explainable reinforcement learning: A survey. In: Machine Learning and Knowledge Extraction - 4th IFIP TC 5, TC 12, WG 8.4, WG 8.9, WG 12.9 International Cross-Domain Conference, CD-MAKE 2020, Dublin, Ireland, August 25-28, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12279, pp. 77–95. Springer (2020). https://doi.org/10.1007/978-3-030-57321-8_5
108. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Series in Machine Learning, Morgan Kaufmann (1993), https://kupdf.net/download/j-ross-quinlan-c4-5-programs-for-machine-learning-1993_5b095daee2b6f5024deefc30_pdf
109. Rabold, J., Schwalbe, G., Schmid, U.: Expressive explanations of DNNs by combining concept analysis with ILP. In: KI 2020: Advances in Artificial Intelligence. pp. 148–162. Lecture Notes in Computer Science, Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-58285-2_11
110. Rabold, J., Siebers, M., Schmid, U.: Explaining black-box classifiers with ILP – empowering LIME with Aleph to approximate non-linear decisions with relational rules. In: Proc. Int. Conf. Inductive Logic Programming. pp. 105–117. Lecture Notes in Computer Science, Springer International Publishing (2018). https://doi.org/10.1007/978-3-319-99960-9_7, https://link.springer.com/chapter/10.1007/978-3-319-99960-9_7
111. Renard, X., Woloszko, N., Aigrain, J., Detyniecki, M.: Concept tree: High-level representation of variables for more interpretable surrogate decision trees. In: Proc. 2019 ICML Workshop Human in the Loop Learning. arXiv.org (2019), http://arxiv.org/abs/1906.01297

112. Ribeiro, M.T., Singh, S., Guestrin, C.: "Why should I trust you?": Explaining the predictions of any classifier. In: Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining. pp. 1135–1144. KDD '16, ACM (2016), https://arxiv.org/abs/1602.04938
113. Rieger, I., Kollmann, R., Finzel, B., Seuss, D., Schmid, U.: Verifying deep learning-based decisions for facial expression recognition. In: Proceedings of the ESANN Conference 2020 (2020)
114. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. **1**(5), 206–215 (May 2019). https://doi.org/10.1038/s42256-019-0048-x
115. Sabour, S., Frosst, N., Hinton, G.E.: Dynamic routing between capsules. In: Advances in Neural Information Processing Systems 30. pp. 3856–3866. Curran Associates, Inc. (2017), http://papers.nips.cc/paper/6975-dynamic-routing-between-capsules
116. Samek, W., Montavon, G., Lapuschkin, S., Anders, C.J., Müller, K.R.: Toward interpretable machine learning: Transparent deep neural networks and beyond. CoRR **abs/2003.07631** (2020), https://arxiv.org/abs/2003.07631
117. Samek, W., Montavon, G., Vedaldi, A., Hansen, L.K., Müller, K.R. (eds.): Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, Lecture Notes in Computer Science, vol. 11700. Springer (2019). https://doi.org/10.1007/978-3-030-28954-6
118. Samek, W., Müller, K.R.: Towards explainable artificial intelligence. In: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, Lecture Notes in Computer Science, vol. 11700, pp. 5–22. Springer (2019). https://doi.org/10.1007/978-3-030-28954-6_1
119. Schmid, U., Finzel, B.: Mutual explanations for cooperative decision making in medicine. KI-Künstliche Intelligenz pp. 1–7 (2020)
120. Schmid, U., Zeller, C., Besold, T., Tamaddoni-Nezhad, A., Muggleton, S.: How does predicate invention affect human comprehensibility? In: International Conference on Inductive Logic Programming. pp. 52–67. Springer (2016)
121. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-CAM: Visual explanations from deep networks via gradient-based localization. In: Proc. 2017 IEEE Int. Conf. Computer Vision. pp. 618–626. IEEE (Oct 2017). https://doi.org/10.1109/ICCV.2017.74, https://arxiv.org/abs/1610.02391
122. Shwartz-Ziv, R., Tishby, N.: Opening the black box of deep neural networks via information. CoRR **abs/1703.00810** (2017), http://arxiv.org/abs/1703.00810
123. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: Visualising image classification models and saliency maps. In: Proc. 2nd Int. Conf. Learning Representations, Workshop Track Proceedings. vol. abs/1312.6034. CoRR (2014), http://arxiv.org/abs/1312.6034
124. Singh, A., Sengupta, S., Lakshminarayanan, V.: Explainable deep learning models in medical image analysis. J. Imaging **6**(6), 52 (Jun 2020). https://doi.org/10.3390/jimaging6060052
125. Smilkov, D., Thorat, N., Kim, B., Viégas, F.B., Wattenberg, M.: SmoothGrad: Removing noise by adding noise. CoRR **abs/1706.03825** (2017), http://arxiv.org/abs/1706.03825
126. Spinner, T., Schlegel, U., Schafer, H., El-Assady, M.: explAIner: A visual analytics framework for interactive and explainable machine learning. IEEE Trans. Visual. Comput. Graphics **26**, 1064–1074 (2020). https://doi.org/10.1109/TVCG.2019.2934629

127. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.A.: Striving for simplicity: The all convolutional net. In: Proc. 3rd Int. Conf. Learning Representations, ICLR 2015, Workshop Track Proceedings. vol. abs/1412.6806. CoRR (2015), http://arxiv.org/abs/1412.6806
128. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: Proc. 34th Int. Conf. Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 3319–3328. PMLR (2017), http://proceedings.mlr.press/v70/sundararajan17a.html
129. Teso, S., Kersting, K.: Explanatory interactive machine learning. In: Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society. pp. 239–245 (2019)
130. Thrun, S.: Extracting rules from artificial neural networks with distributed representations. In: Advances in Neural Information Processing Systems 7. pp. 505–512. MIT Press (1995), http://papers.nips.cc/paper/924-extracting-rules-from-artificial-neural-networks-with-distributed-representations.pdf
131. Tjoa, E., Guan, C.: A survey on explainable artificial intelligence (XAI): Toward medical XAI. IEEE Trans. Neural Netw. Learning Syst. pp. 1–21 (2020). https://doi.org/10.1109/TNNLS.2020.3027314
132. van Lent, M., Fisher, W., Mancuso, M.: An explainable artificial intelligence system for small-unit tactical behavior. In: Proc. 2004 Nat. Conf. Artificial Intelligence. pp. 900–907. AAAI Press; MIT Press; 1999 (2004)
133. Vassiliades, A., Bassiliades, N., Patkos, T.: Argumentation and explainable artificial intelligence: A survey. Knowl. Eng. Rev. **36** (2021/ed). https://doi.org/10.1017/S0269888921000011
134. Vilone, G., Longo, L.: Explainable artificial intelligence: A systematic review. ArXiv200600093 Cs (Oct 2020), http://arxiv.org/abs/2006.00093
135. von Luxburg, U.: A tutorial on spectral clustering. Stat Comput **17**(4), 395–416 (Dec 2007). https://doi.org/10.1007/s11222-007-9033-z
136. Wan, A., Dunlap, L., Ho, D., Yin, J., Lee, S., Petryk, S., Bargal, S.A., Gonzalez, J.E.: NBDT: Neural-backed decision tree. In: Posters 2021 Int. Conf. Learning Representations (Sep 2020), https://openreview.net/forum?id=mCLVeEpplNE
137. Wang, H.: ReNN: Rule-embedded neural networks. In: Proc. 24th Int. Conf. Pattern Recognition. pp. 824–829. IEEE Computer Society (2018). https://doi.org/10.1109/ICPR.2018.8545379, http://arxiv.org/abs/1801.09856
138. Wang, Q., Zhang, K., II, A.G.O., Xing, X., Liu, X., Giles, C.L.: A comparative study of rule extraction for recurrent neural networks. CoRR **abs/1801.05420** (2018), http://arxiv.org/abs/1801.05420
139. Wang, Q., Zhang, K., Ororbia II, A.G., Xing, X., Liu, X., Giles, C.L.: An empirical evaluation of rule extraction from recurrent neural networks. Neural Computation **30**(9), 2568–2591 (Jul 2018). https://doi.org/10.1162/neco_a_01111, http://arxiv.org/abs/1709.10380
140. Weitz, K.: Applying Explainable Artificial Intelligence for Deep Learning Networks to Decode Facial Expressions of Pain and Emotions. Ph.D. thesis, Otto-Friedrich-University Bamberg (Aug 2018), http://www.cogsys.wiai.uni-bamberg.de/theses/weitz/Masterarbeit_Weitz.pdf
141. Xie, N., Ras, G., van Gerven, M., Doran, D.: Explainable deep learning: A field guide for the uninitiated. CoRR **abs/2004.14545** (2020), https://arxiv.org/abs/2004.14545

142. Yang, Y., Feng, C., Shen, Y., Tian, D.: Foldingnet: Interpretable unsupervised learning on 3d point clouds. CoRR **abs/1712.07262** (2017), http://arxiv.org/abs/1712.07262
143. Yao, J.: Knowledge extracted from trained neural networks: What's next? In: Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security 2005, Orlando, Florida, USA, March 28-29, 2005. SPIE Proceedings, vol. 5812, pp. 151–157. SPIE (2005). https://doi.org/10.1117/12.604463
144. Yeh, C.K., Kim, B., Arik, S., Li, C.L., Pfister, T., Ravikumar, P.: On completeness-aware concept-based explanations in deep neural networks. In: Advances in Neural Information Processing Systems 33. vol. 33, pp. 20554–20565 (2020), https://proceedings.neurips.cc/paper/2020/hash/ecb287ff763c169694f682af52c1f309-Abstract.html
145. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Proc. 13th European Conf. Computer Vision - Part I. Lecture Notes in Computer Science, vol. 8689, pp. 818–833. Springer International Publishing (2014). https://doi.org/10.1007/978-3-319-10590-1_53, https://doi.org/10.1007/978-3-319-10590-1_53
146. Zhang, Q., Cao, R., Shi, F., Wu, Y.N., Zhu, S.C.: Interpreting CNN knowledge via an explanatory graph. In: Proc. 32nd AAAI Conf. Artificial Intelligence. pp. 4454–4463. AAAI Press (2018), https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17354
147. Zhang, Q., Zhu, S.C.: Visual interpretability for deep learning: A survey. Front. IT EE **19**(1), 27–39 (2018). https://doi.org/10.1631/FITEE.1700808
148. Zhang, Y., Chen, X.: Explainable recommendation: A survey and new perspectives. FNT in Information Retrieval **14**(1), 1–101 (2020). https://doi.org/10.1561/1500000066
149. Zhou, B., Khosla, A., Lapedriza, À., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: Proc. 2016 IEEE Conf. Comput. Vision and Pattern Recognition. pp. 2921–2929. IEEE Computer Society (2016). https://doi.org/10.1109/CVPR.2016.319, https://arxiv.org/abs/1512.04150
150. Zhou, B., Sun, Y., Bau, D., Torralba, A.: Interpretable basis decomposition for visual explanation. In: Computer Vision – ECCV 2018. pp. 122–138. Lecture Notes in Computer Science, Springer International Publishing (2018). https://doi.org/10.1007/978-3-030-01237-3_8, https://link.springer.com/chapter/10.1007%2F978-3-030-01237-3_8
151. Zhou, J., Gandomi, A.H., Chen, F., Holzinger, A.: Evaluating the quality of machine learning explanations: A survey on methods and metrics. Electronics **10**(5), 593 (Jan 2021). https://doi.org/10.3390/electronics10050593
152. Zilke, J.R., Loza Mencía, E., Janssen, F.: DeepRED – rule extraction from deep neural networks. In: Proc. 19th Int. Conf. Discovery Science. pp. 457–473. Lecture Notes in Computer Science, Springer International Publishing (2016). https://doi.org/10.1007/978-3-319-46307-0_29, https://link.springer.com/chapter/10.1007%2F978-3-319-46307-0_29

Concept Embedding Analysis: A Review

Gesina Schwalbe (ORCID: 0000-0003-2690-2478)[1]

[1*] Continental AG, Regensburg, Germany.

Contributing authors:
gesina.schwalbe@continental-corporation.com;

Abstract

Deep neural networks (DNNs) have found their way into many applications with potential impact on the safety, security, and fairness of human-machine-systems. Such require basic understanding and sufficient trust by the users. This motivated the research field of explainable artificial intelligence (XAI), i.e. finding methods for opening the "black-boxes" DNNs represent. For the computer vision domain in specific, practical assessment of DNNs requires a globally valid association of human interpretable concepts with internals of the model. The research field of *concept (embedding) analysis (CA)* tackles this problem: CA aims to find global, assessable associations of humanly interpretable semantic concepts (e.g., eye, bearded) with internal representations of a DNN. This work establishes a general definition of CA and a taxonomy for CA methods, uniting several ideas from literature. That allows to easy position and compare CA approaches. Guided by the defined notions, the current state-of-the-art research regarding CA methods and interesting applications are reviewed. More than thirty relevant methods are discussed, compared, and categorized. Finally, for practitioners, a survey of fifteen datasets is provided that have been used for supervised concept analysis. Open challenges and research directions are pointed out at the end.

Acknowledgments. The research leading to these results was partly funded by by the German Federal Ministry for Economic Affairs and Energy within the projects "KI Wissen – Automotive AI powered by Knowledge" and "KI Absicherung – Safe AI for automated driving". Thanks to the consortia for the successful cooperation.

1 Introduction

Deep neural networks have become a key to many application fields. However, with their popularity, their potential impact to safety and fairness increased to an extend that the need for responsible artificial intelligence (Arrieta et al, 2020) became apparent. For example, the European Union General Data Protection Regulation enacted in 2016 demands algorithms not only to be efficient, but also transparent and fair (Goodman and Flaxman, 2017). Similarly, the automotive functional safety standard ISO 26262 (ISO/TC 22/SC 32, 2018) considers manual inspection of algorithms an important measure of safety assurance. These requirements gave rise to the field of explainable artificial intelligence which was boosted in 2016 by the United States Defense Advanced Research Projects Agency (DARPA) (DARPA, 2016; Adadi and Berrada, 2018). XAI in general tries to translate behavioral or internal aspects of black-box algorithms like DNNs into a human interpretable form (Schwalbe and Finzel, 2021).

While many different aspects of a DNN can be the subject of a human understandable explanation, their safety assessment requires interpretable access to the *internal representation*. This requirement holds even for hardly specifiable tasks with non-symbolic inputs like object detection from images. This access allows to debug or audit the DNN function with respect to failure modes that violate symbolic prior knowledge (e.g., "eyes usually belong to a movable object"); and this access potentially points out how to fix inappropriate internal representations or directly enforce sufficient ones. Besides the subject of explanation, another relevant property of explainability methods is their locality or scope (Adadi and Berrada, 2018): Many methods for analyzing computer vision DNNs produce local explanations specific to one or few samples (Zhang and Zhu, 2018). In contrast, *global* explanations are valid for many more examples than would be possible with manual inspection of many local explanations. This makes global explanations interesting for practical assessment tasks. Another practical constraint imposed by assessment use-cases is that explanations must provide high *fidelity* and that they must be *simple* (e.g., linear transformations of the object of explanation, as suggested by Kim et al (2018)) to avoid introducing further complexity. Lastly, assessments also require analysis of feature interdependencies and interaction. This is another drawback of solely using the popular heatmapping methods (Zhang et al, 2018b; Janizek et al, 2020) that highlight local attention of a DNN, like relevance (Bach et al, 2015), saliency (Sundararajan et al, 2017), or perturbation (Fong and Vedaldi, 2017) based ones.

A promising class of methods fulfilling the mentioned desires is that of *concept (embedding) analysis (CA)* methods. The main idea of concept embedding analysis is to associate (in a simple way) semantic concepts taken from natural human language, like eye, with vectors—the concept embedding or *concept activation vectors (CAV)* (Kim et al, 2018)—, or sub-spaces in the intermediate output space of one (or several) layer(s) in a DNN, the latent space(s). These approaches allow global access to latent space representations of semantic concepts in an interpretable way, without introducing further complexity. Thus, they fulfill the mentioned desires for assessment tasks and model improvement.

Early work on concept analysis started with unsupervised visualization of single features in convolutional DNNs (CNNs), i.e. neurons, filters or groups thereof (Olah et al, 2017). Later on, supervised approaches were based on the NetDissect method developed in Bau et al (2017), where single filters of CNNs were associated with predefined visual semantic concepts. Two cornerstones towards concept analysis research were established by Fong and Vedaldi (2018) and Kim et al (2018) in 2018. Both methods associate predefined semantic concepts with vectors in the latent spaces, thus post-hoc disentangling the representation of such semantic concepts within DNN internal representations. On this basis, distributed representations in unsupervised settings were first considered in the ACE approach (Ghorbani et al, 2019) proposed in 2019. Another notable development was that of concept bottleneck models (Losch et al, 2019; Koh et al, 2020): Stepping away from post-hoc explainability, these include prior knowledge about task-relevant concepts directly in the architecture of a DNN. This is done with a bottleneck layer in which single nodes are trained to correspond to semantic concepts. These baseline approaches were extended and generalized in many interesting ways in recent years. Thus, time seems to be ready to take a closer look on the developments in this young research area, as will be done in this review.

Contributions

This paper shall shed light onto concept analysis as a practically interesting and emerging sub-field of XAI, in order to foster research in this direction. Its main contributions towards this are as follows:

- A formal, general *definition* is provided capturing the core ideas uniting concept analysis methods, and differentiating it from similar research fields (cf. Section 3).
- In order to help researchers to find further research directions and position their work, a general *taxonomy* for concept analysis approaches is developed and applied to state-of-the-art research methods (cf. Section 4 and Figure 1).
- The breadth of the topic is demonstrated by an in-depth and systematic review of more than 30 diverse *concept analysis approaches*. These are discussed, compared, and categorized according to the taxonomy (cf. Section 4 and Tables 1–4).
- An overview of more than 15 image *datasets* used for concept analysis is compiled (cf. Section 6 and Table 6).
- Based on the taxonomy and the method reviews, open challenges and promising *research directions* are uncovered and discussed (cf. Section 7).

Scope

This review goes along with the baseline work from Kim et al (2018); Fong and Vedaldi (2018); Bau et al (2017); Ghorbani et al (2019). The main focus lies on visual interpretability, considering computer vision tasks as guiding examples. The literature review combined a two-level recursive reference search starting with the mentioned baselines, and a keyword search based on the keywords “Concept Analysis” and “Concept Embedding Analysis” on the search engine Google Scholar[1]. Papers were included by a two-stage review approach, first filtering by title and then by the abstract. Included were global interpretability methods that pursue the mentioned general goal of associating human interpretable concepts with DNN latent space vectors or sub-spaces.

Outline

This paper is structured as follows: In Section 2, related work and research fields are recapitulated to demarcate the scope of concept analysis as considered here. A precise definition for concept analysis is then developed in Section 3, and a detailed taxonomy for concept analysis is prescribed. The breadth of the topic is demonstrated in Section 4, in which interesting concept analysis methods are outlined, compared, and classified according to the derived taxonomy (cf. Tables 1–4). Applications of concept analysis are then discussed in Section 5, including model distillation (cf. Table 5) and qualitative as well as quantitative DNN assessment methods. Accompanying these collections, Section 6 compiles an overview on image datasets for supervised concept analysis that are used in methods presented throughout this paper. This may serve as a starting point and reference for researchers looking for typical evaluation datasets. Section 7 provides an outline of current challenges in the area of concept analysis, and an outlook on potential research directions.

2 Related work

The field of XAI has seen an exponential rise in research interest in the last decade (Vilone and Longo, 2020; Arrieta et al, 2020; Linardatos et al, 2021; Zhou et al, 2021). By now, good introductory work is available, like Molnar (2020), and many extensive method surveys, including general ones like Linardatos et al (2021); Vilone and Longo (2020); Arrieta et al (2020); Carvalho et al (2019); Adadi and Berrada (2018); Gilpin et al (2018), and ones specific to sub-topics like reinforcement learning (Heuillet et al, 2021), visual interpretability (Tjoa and Guan, 2020; Zhang and Zhu, 2018), or rule extraction (Hailesilassie, 2016). A meta-review for XAI methods can be found in Schwalbe and Finzel (2021). We also consider the typical XAI method classification aspects of black- versus white-box, post-hoc versus inherent interpretability, and simplicity. These considerations are further refined and tailored to concept analysis in specific, and extended by implementation specific properties of CA methods. A first review on concept analysis is provided

[1] https://scholar.google.com/

by Kazhdan et al (2021), with a two-class scheme for classifying concept analysis methods. In contrast to our broad review, the focus in that work lies on the general practical comparison of two concept analysis approaches with latent space disentanglement. Especially, our review for the first time gives a broad overview of the methods and applications of concept analysis, providing a common point of view of methods usually sorted into very different categories (e.g., post-hoc and inherent interpretability). In the following, related XAI research fields are distinguished from concept analysis.

Visual attribution methods

A very popular subfield of XAI is about providing heatmaps that point out what input region (e.g., pixels in an image or words in a text) the DNN mostly attended to make a concrete decision, i.e. which region had the highest *attribution*. Notable types are perturbation based methods (Ribeiro et al, 2016; Fong and Vedaldi, 2017), relevance back-propagation such as LRP (Bach et al, 2015), activation map based approaches (Zhou et al, 2016), and gradient respectively sensitivity based approaches (Sundararajan et al, 2017). All these approaches find their use in some of the concept analysis applications presented here, e.g., for training (Wu et al, 2020), more precise localization of concepts (Lucieri et al, 2020; Fong and Vedaldi, 2018), or dependency analysis (Kim et al, 2018). While the heatmaps essentially provide a simple linear approximation of the DNN behavior (Ribeiro et al, 2016), these explanations are inherently local and do not provide insights into internal encodings of the DNN. This also prohibits deeper analysis of relations between concepts (Rabold et al, 2020). In contrast, the underlying mapping of concepts to internal features of concept analysis methods is inherently global.

Latent space disentanglement

Concept analysis is very closely related to latent space disentanglement (Kazhdan et al, 2021) methods like VAEs (Kingma and Welling, 2014; Higgins et al, 2016), as both approaches try to represent (parts of) the latent space as interpretable features. Similar to so-called concept bottleneck models (Koh et al, 2020), disentanglement methods have a bottleneck layer in which single units are trained to encode a feature. These features, however, are not necessarily aligned with human semantics (especially when trained unsupervised), but instead should encode the most prevalent independent factors of variation causally related to the DNN output. The target features of concept bottleneck models instead are *interpretable* factors of variation (Kazhdan et al, 2021) perceived by humans. These are not necessarily—and often are not—independent, but allow to assess the dependencies of semantic concepts encoded by the DNN.

3 Concept embedding analysis

To get a clear understanding of the scope of concept analysis considered here, related terms are defined in Subsection 3.1, and optional but desirable properties are collected from literature. Moreover, a taxonomy is provided in Subsection 3.2, that allows a detailed analysis and classification of a use-case or method for CA. The taxonomy is accompanied by illustrative examples, and later applied to the sample methods highlighted in Section 4 (cf. Tables 1–4).

3.1 Definitions

The output of a neural network layer spans a vector space. In this paper, this is called *latent space* of the layer, or *activation maps* if collections of units spatially correspond to patches in the input as in the case of convolutional layers. In this work the term *concept (embedding) analysis* generally refers to investigation of the questions *whether*, *how well*, *how*, and with *which properties* information about a *semantic concept* is represented within the latent spaces or sub-spaces thereof.

Semantic concepts

A *(semantic) concept*, also called *attribute* (attr.) in Kronenberger and Haselhoff (2019), is a concept occurring in a natural language, i.e. a notion that can be described using natural language.[2] Examples are the synonym sets established in WordNet (Fellbaum, 1998). The semantic concepts of interest here are those that can be assigned to (patches of) a DNN input sample in the form of additional labels. Hence, the selection of concepts depends on the context given by the task. In Kazhdan et al (2021) this dependency is emphasized in the definition of concepts as interpretable factors of variations for a given training dataset. The paper by Bau et al (2017) introduced the following classes of semantic concepts for images: sample-level attributes like scene (e.g., rainy) and image qualities (Abid and Zou, 2021) (e.g., contrast); full objects (e.g., person); part objects (e.g., head, leg); and object-level attributes like material (e.g., wooden), texture (e.g., striped), and color (e.g., green). Other examples of object-level attributes are gender, and facial attributes like bearded as considered in Kim et al (2018). For supervised concept analysis, it is assumed that labeled examples of the concepts of interest are available, and different approaches require different label formats. Formats considered here are the most common binary labels (e.g., Kim et al (2018)) respectively binary segmentation masks (e.g., Fong and Vedaldi (2018); Schwalbe (2021)), multi-class concepts (Kazhdan et al, 2020), and regression concepts (Graziani et al, 2018, 2020). Other possible types are multi-valued concepts with independent values (e.g., weather with raininess and cloudiness), and spatial allocation of concepts, e.g., via bounding boxes or x-y-positions. Note that the direct output of the DNN usually also relates to a semantic concept, like objects for object detectors. The primary interests here are concepts that are not included as features in the DNN final output, e.g., body parts in the case of a pedestrian detector.

[2] Concept terms are marked like this throughout this work.

Concept analysis (CA)

By basic *concept analysis* or concept embedding analysis we define any activity that tries to associate semantic concepts with vectors or sub-spaces in one preselected latent space of a DNN. Further, we assume that this is done via a simple, global model that allows to predict information about the concept from the latent space activations of an input. The model is called the *concept model* of the concept, and its output depends on the label type of the concept (e.g., binary presence of the concept, localization, regression, classification, etc.). *Simple* means transparent or human interpretable, or not introducing much more non-interpretable complexity, like linear models (Kim et al, 2018). The pair of a concept and its latent space representation (vector or sub-space) is called the *concept embedding.* In case a vector is associated, this vector is called the concept vector or *concept activation vector (CAV)* (Kim et al, 2018).

Concept analysis goals

As mentioned previously, the main goals of CA are to answer *whether*, *how well*, *how*, and with *which properties* information about a predefined or mined semantic concept is represented within a latent space. To just show *whether* information about a given semantic concept is available within a latent space, one can either perform a supervised training of a concept model and check whether sufficient performance can be reached (Schwalbe, 2021), as illustrated in Fuchs et al (2018); or perform an unsupervised concept mining, later comparing found concept associations with the given concept. A proxy to measure *how well* the information is encoded generally is the concept model prediction performance. The assessment of properties now requires finding an exact representation of the concept information that can be related or even compared to representations of other concepts (including the input and output). Most common are concept representations by CAVs. The general idea of CAVs is to find a vector which is close to the latent space vectors of examples of the given concept with respect to some proximity measure.

Desirable properties of concept models

The following properties of concept models were considered useful in literature:

- *Comparable* concept models allow to assess whether the mapping between semantic and latent space preserves semantic similarities. In other words, whether semantically similar concepts are assigned to similar latent space representations (Schwalbe, 2021).
- *Vector representations* allow to utilize the rich vector space structure of latent spaces, including natural comparability:
  - In Mikolov et al (2013) it was found that for continuous space language models the *cosine similarity* on latent space vectors correspond to meaningful semantic (cor-)relations of the encoded words. An example is the famous relation “$\mathsf{king} - \mathsf{man} + \mathsf{woman} = \mathsf{queen}$”. Net2Vec (Fong and Vedaldi, 2018) and TCAV (Kim et al, 2018) showed that this also holds for CAVs in computer vision networks,

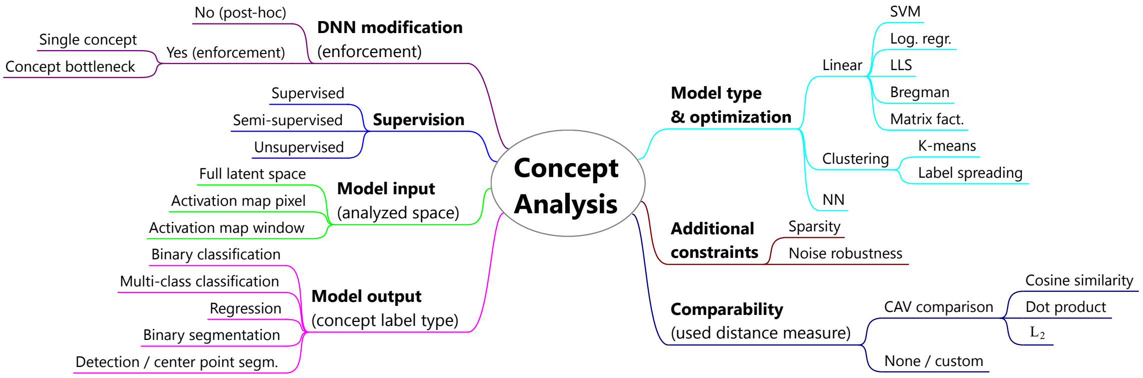


Figure 1 Overview on the discussed taxonomy aspects for concept analysis

and Zhang et al (2018b) used cosine similarity for local CAVs modeled by masked derivative vectors.

  - The local *sensitivity* of later layer representations with respect to the concept can easily be measured via the partial derivative along the CAV as proposed in Kim et al (2018).

- *Linear concept models* are considered to be natural and easy to assess for humans in Kim et al (2018).
- *Sparsity* of CAVs in case of linear encodings is postulated to increase interpretability of the concept representations (Bau et al, 2017). Bau et al (2017) even suggest using the number of concepts to which a CNN filter contributes as a comparative measure for interpretability or entanglement—the more filters, the less interpretable.
- Concept models allowing *gradient back-propagation*, like Net2Vec, can be utilized as additional outputs during training or *fine-tuning* (Schwalbe and Schels, 2020). Also, they allow to measure *attribution* of earlier layer concepts or inputs to a given concept output via relevance back-propagation or sensitivity (Lucieri et al, 2020). Otherwise, one has to turn to perturbation-based attribution methods (Wang et al, 2020).

3.2 Taxonomy

In the following we collect key aspects to differentiate approaches for concept embedding analysis, summarized in Figure 1. An overview of how the methods discussed in Section 3 fit into the suggested categorization scheme can be found in Tables 1–4.

Concept model supervision

The concepts to be analyzed can either be predefined by sample data and then associated to DNN internal representations in a supervised (superv.) fashion, or human interpretable concepts can be *mined* from the latent spaces in an unsupervised way. One example of the latter is Automated Concept-based

Explanations (ACE) by Ghorbani et al (2019). They search for clusters in the latent space vectors of super-pixels (the concept candidates), defining a concept by a cluster and the concept vectors as the cluster center. Another example of unsupervised concept analysis is feature visualization (Olah et al, 2017).

Concept model output: Concept label type

First concept analysis approaches considered binary concept model outputs, either binary classification (cls) labels as in TCAV (Kim et al, 2018) and ACE, binary segmentation (seg) masks as in Net2Vec (Fong and Vedaldi, 2018), or binary concept center point prediction (det) masks as in Schwalbe and Schels (2020); Rabold et al (2020). This binary setting can be extended to multi-class (mcls) concepts like `shape` as suggested in Kazhdan et al (2020), or a multi-dimensional concept like in the IIN (Esser et al, 2020) method, where concepts are represented by sub-spaces of the transformed latent space. Regression Concept Vectors (RCV) by Graziani et al (2018), and its successor methods consider continuous-valued instead of discrete-valued concepts.

Concept model input

TCAV considers the complete latent space of a layer for its binary classification. The same holds for successor methods that try to mitigate the resulting high memory demand by average pooling of the latent space, like Graziani et al (2020); Chyung et al (2019). However, it is hard to localize concepts in the input only from image level concept information, even using attention methods (Lucieri et al, 2020). To enable localization of concepts, Net2Vec does not consider the complete activation space of a CNN, but instead operates on single pixels of an activation map, utilizing the spatial correspondence between activation map pixels and input regions. In this case of a linear concept model, this can efficiently be implemented by a 1×1-convolution. The convolutional Net2Vec-approach is adapted in Schwalbe and Schels (2020), only using windows larger than 1×1 pixels to provide enough context information as input to the concept model. The prior work of Net2Vec on network dissection (Bau et al, 2017) was restricted not only to 1×1-windows, but also to unit vectors (i.e. single filters) per activation map pixel, similar to standard feature visualization (Olah et al, 2017).

Concept model type and optimization

The optimal retrieval of concept information from latent space input can be solved via different models, optimization methods, and regularization constraints. Examples for model types are: small neural networks (e.g., Neural Stethoscopes by Fuchs et al (2018), IIN); k-means clustering (e.g., SeVec by Gu and Tresp (2019), ACE); or more general matrix factorization (e.g., NCAV by Zhang et al (2021); Saini and Papalexakis (2020)); and linear models (e.g., TCAV, Net2Vec). The latter features, e.g., usage of support vector machines (e.g., TCAV), logistic regression (log. regr.) (e.g., Net2Vec, Schwalbe and Schels (2020)), and Bregman iteration (e.g., CHAIN by Wang et al (2020)) for optimization. It also makes a difference whether a bias is allowed in the linear model or not. Examples of regularization constraints are: Sparsity, as aimed for in NetDissect (Bau et al, 2017), and in CHAIN via L_1 loss; and robustness against noise as implemented in Net2Vec and NetDissect by thresholding activation maps.

Enforcement or post-hoc

As for other XAI methods, CA approaches can also be classified into inherent interpretability methods enforcing a level of transparency, and post-hoc methods not changing the analyzed DNN like Fong and Vedaldi (2018); Kim et al (2018). Supervised post-hoc CA is called *concept localization*, and unsupervised post-hoc CA is referred to as *concept mining*. Typical inherently interpretable architectures utilizing CA are so-called *concept-bottleneck models (CBM)*: These DNNs feature a bottleneck layer in which all or a sub-set of the nodes are trained to be directly associated with a semantic concept. Example architectures and training schemes can be found in Koh et al (2020); Losch et al (2019) (with concept supervision), and ProtoPNet (Chen et al, 2019a) (without concept model output supervision). Different architectures to integrate and connect the concept bottleneck into the DNN are compared in Kronenberger and Haselhoff (2019). Interestingly, good embeddings of task relevant concepts seem to support or at least not decrease the final performance of the network (Losch et al, 2019), and allow for better plausibility checks during operation (Kronenberger and Haselhoff, 2019).

Comparability: CAV distance measure

In case concept activation vectors are used to represent concepts in the latent space, both finding and interpreting the vectors often relies on a vector distance measure: For a concept, a CAV is selected which is “close” to latent space vector representations of samples; and to compare concept models based on CAVs, one can also utilize this proximity measure. To our knowledge, three measures have been used in literature:

- *L_2 distance* is used for concept mining via clustering in ACE and its successor Yeh et al (2020). It is also used for training a concept localization linear model in the method CHAIN, and the matrix factorization approach NCAV.
- *Cosine similarity* between two vectors v_1, v_2 encodes the angle between the vectors and is defined as the normalized dot product

  $$\mathrm{CosSim}(v_1, v_2) := \frac{v_1 \circ v_2}{\|v_1\| \cdot \|v_2\|} \in [0, 1]$$

  which is 1 for parallel, 0 for orthogonal, and -1 for anti-parallel vectors. It was used in SeVec (Gu and Tresp, 2019) for concept localization via clustering.
- *Dot product* is the standard distance measure used to define linear models: The model defines a hyperplane $H = \{d_H(v) = v \mid v \circ v_H + b_H = 0\}$ (biased

by b_H) where v_H is the normal vector (the concept vector) and d_H defines the signed shortest distance to H. This is utilized in the concept localization approaches of TCAV, Net2Vec, and their successors Schwalbe and Schels (2020); Schwalbe (2021), and Kronenberger and Haselhoff (2019).

Other known model types have to use learned distance metrics for comparability.

4 Concept analysis methods

As becomes apparent from the count of taxonomy aspects, there is an abundance of methods to obtain the desired additional concept outputs and concept models inherent to concept analysis. This section deals with prominent example methods to achieve this, and categorizes them according to main aspects of the proposed taxonomy. The first part in Subsection 4.1 discusses methods that work towards inherent transparency and change the underlying model architecture. And the second part in Subsection 4.2 presents methods working on pre-trained models without modifying them, i.e. post-hoc explainability methods. An overview over discussed methods is given in Tables 1–4. Applications for the concept models and outputs will be discussed later in Section 5.

4.1 Inherent concept models

In many applications, inherent interpretability is desirable and may even improve model performance. Examples are safety critical domains like automated driving or medicine, or applications with ethical implications like job application preprocessing. Structuring and training a DNN to use concept embeddings leads to more interpretable intermediate representations respectively models. In the following, it is differentiated between two types of enforcing rich concept outputs: (independent) enforcement of single concepts (Subsubsection 4.1.1), and so-called concept bottleneck models (CBM) enforcing a complete interpretable layer (Subsubsection 4.1.2).

4.1.1 Single concept enforcement

One way to ensure good embeddings of concepts is to add additional outputs for the respective concepts to the DNN and include their predictions into the loss. Such context aware models can be structured like regular DNNs but they use additional regularization to force the network to focus on specific concept embeddings. For example, Schwalbe and Schels (2020) suggested to attach a single neuron per concept (respectively per concept location in the image) and simply add a negative log-likelihood for the prediction performance as weighted summand to the loss.

Neural Stethoscopes. This was implemented in Fuchs et al (2018). They attach 1-hidden-layer DNNs, called Neural Stethoscopes, to the main model which are trained to predict an image-level concept alongside the main output. Their focus was on comparability of the layers, and the Neural Stethoscopes can both be used for concept enforcement and post-hoc concept analysis.

Table 1 Properties of concept analysis methods discussed in Subsubsection 4.1.1 according to the taxonomy defined in Subsection 3.2.

Method	Enforced	Superv.	Model	Distance measure	Optim.	Label type	Input
Single concept enforcement (Subsubsection 4.1.1)							
Neural Stethoscopes (Fuchs et al, 2018)	✓	✓	DNN	learned	log. regr.	cls	full
EDD (Kronenberger and Haselhoff, 2019)	(✓)	✓	linear	dot	log. regr.	cls	full
UPerNet (Xiao et al, 2018)	✓	✓	DNN	learned	log. regr.	seg	window
Interpretable CNNs (Zhang et al, 2018c)	✓	-	linear	dot	–	cls	window

EDD. Similarly, Explanation Dependency Decomposition (EDD) by Kronenberger and Haselhoff (2019) uses additional classifiers to extract the presence of visual embeddings from the convolutional layer outputs. They assessed diverse information flow settings: in-between concept models, and between concept and main model outputs (for details cf. (Kronenberger and Haselhoff, 2019, Fig. 2)). On a simple traffic sign classification setup, adding concept outputs did not infringe performance, and top performance could be achieved by an architecture merging latent space and concept model outputs before predicting the final output.

UPerNet. This is extended to a new discipline, unified perceptual parsing, in Xiao et al (2018): The output of a DNN is not restricted to a main task but extended by a large set of related concepts, as demonstrated in their UPerNet architecture.

Interpretable CNNs. While the other approaches are supervised and rely on concept labels, Zhang et al (2018c) enforces in an unsupervised fashion the alignment of convolutional filters with concepts, which are in this case object parts. This is done by using filter-specific losses that encourage the filter to only activate locally around its maximum activation in the image. This assumes that each object part type occurs once in an input image.

4.1.2 Concept bottleneck models

General concept enforcement will ensure a set of interpretable outputs. So-called concept bottleneck models (CBM) additionally require that the output of a complete layer should be interpretable, i.e. all units should correspond to human interpretable concepts. CBMs primarily can be differentiated by the applied training scheme (important ones depicted in Figure 2).

Concept whitening. An example of a fine-tuning approach that re-trains a pretrained network is concept whitening (Chen et al, 2020). Concept whitening

Table 2 Properties of concept analysis methods discussed in Subsubsection 4.1.2 according to the taxonomy defined in Subsection 3.2.

Method	***Enforced***	***Superv.***	***Model***	***Distance measure***	***Optim.***	***Label type***	***Input***
Concept bottleneck models (Subsubsection 4.1.2)							
Concept Whitening (Chen et al, 2020)	✓	✓	linear	dot	LPQC	any	full
ProtoPNet (Chen et al, 2019a), CSPP (Feifel et al, 2021)	✓	-	cluster	L_2	custom	det	window
Semantic bottlenecks (Losch et al, 2019)	✓	✓	linear	dot	log. regr.	seg	pixel
Concept bottlenecks (Koh et al, 2020)	✓	✓	linear	dot	log. regr.	cls	full
Weakly supervised CBMs (Belém et al, 2021)	✓	✓	linear	dot	log. regr.	cls	full
Concept Groups (Marcos et al, 2020)	✓	✓	custom	dot	custom	cls	full

suggests a replacement layer for batch normalization that is trained in a multi-task setting to apply batch whitening and align the dimensions of the latent space to given concepts. This is formulated as a linear programming problem with quadratic constraints (LPQC) and requires few epochs to disentangle the dimensions after replacing the batch normalization.

ProtoPNet. A clustering-based and unsupervised concept model approach is considered by Chen et al (2019a) in form of the ProtoPNet architecture. Here, the bottleneck layer consists of a prototype predictor: Windows in the CNN layer activation output are compared to unsupervised learned concept prototype CAVs, and the final output is derived from the resulting prototype scores. The CAV cluster centers and the rest of the model are trained jointly.

CSPP. An extension to object detection tasks was proposed with the Center Scale and Prototype Prediction (CSPP) architecture by Feifel et al (2021).

Semantic bottlenecks. The semantic bottlenecks architecture (Losch et al, 2019) also considers a fine-tuning training scheme. Here, the output of a complete layer of a pretrained DNN is linearly transformed to a representation where every filter represents one predefined concept, while the later layers are retrained.

CBM. Different schemes for directly training the concept bottleneck model with both the concept and the main objective were compared in Koh et al (2020). Settings were considered in which the part up-to the bottleneck and the part top-of the bottleneck were trained independently, jointly, or sequentially, as illustrated in Figure 2. Results revealed joint training as the tight

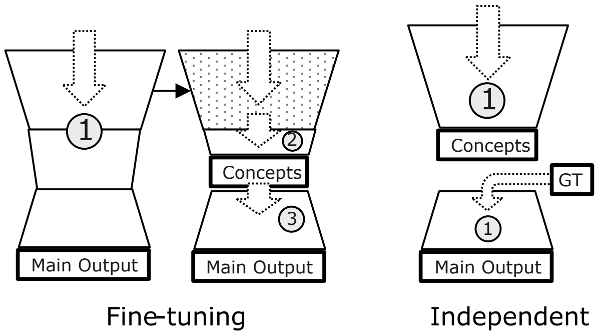


Figure 2 Different schemes for training concept bottleneck models both with the concepts and the main output objective, as suggested in literature (Losch et al, 2019; Koh et al, 2020)

winner in performance. As in Losch et al (2019), performance was shown to be competitive to non-interpretable standard models without concept bottleneck, and also substantially more robust to spurious correlations with background features.

Weakly supervised CBM. Though Koh et al (2020) found that CBMs can mediocrely cope with small data tasks, especially the joint training requires a considerable amount of possibly costly concept labels. Therefore, Belém et al (2021) suggested a weak supervision of concept models by generating noisy labels from approximate rules when such are available from domain knowledge. In a fraud detection example setting, they found sequential training to work best, where first the noisy then the accurate labels are used.

Concept Groups. Another shortcoming of vanilla CBMs is that it may not be clear how a predefined concept from the bottleneck is used for the final output, respectively what its task-specific meaning is. To tackle this, Marcos et al (2020) suggests Concept Groups, a second bottleneck layer directly attached to the first one where nodes represent task-specific groups of concepts. These are obtained unsupervised, and the complete model is trained in a three-step sequential manner.

Lastly, it must be noted that the constraint imposed by a CBM, namely that single units in a layer correspond to interpretable concepts, does in general not guarantee that only information about those concepts is passed through the layer. Due to correlation of the concepts, further information can be encoded which is possibly non-semantic, as was shown and criticized in Mahinpei et al (2021).

4.2 Post-hoc concept models

Many applications require working with a pretrained DNN without changing its architecture. In such cases, post-hoc concept analysis methods are required that train concept models on given latent space outputs. The two main classes of methods to associate single concepts with latent space representations are supervised (concept localization) and unsupervised (concept mining) post-hoc

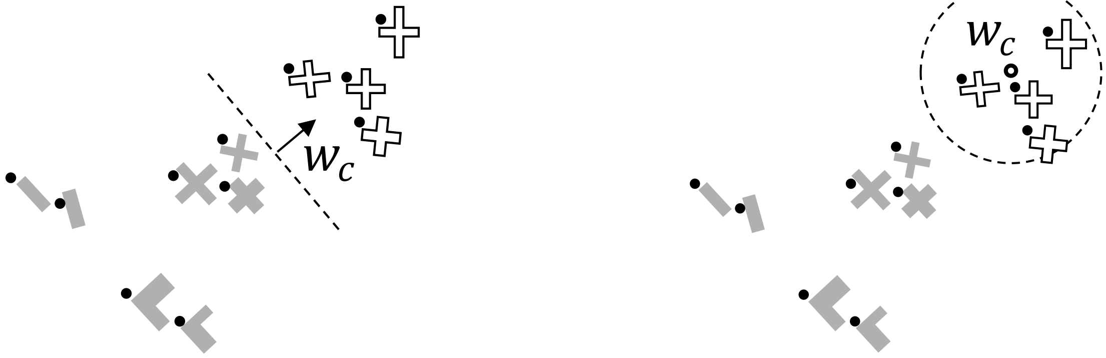


Figure 3 Comparison of concept vectors w_c for two standard concept analysis model types, and the binary visual shape concept plus (unfilled). Dots (•) represent points in the simplified latent space, accompanied by visualizations of the shape occurring in the image (patch) that produced this point. *Left:* Linear concept model with a hyperplane as decision boundary and the hyperplane normal vector as CAV w_c; *right:* L_2-clustering-based concept model with a spherical decision boundary and cluster center (unfilled) as CAV w_c.

CA. Furthermore, we will discuss examples of the special case when a complete latent space is post-hoc disentangled on the basis of semantic concepts.

4.2.1 Concept localization

By now there is a formidable selection of supervised post-hoc concept analysis methods described in literature. For an overview, we cluster them by the model type into linear and non-linear models.

Linear concept localization

NetDissect. An early work on supervised post-hoc concept analysis, and basis for many further methods, was Network Dissection (NetDissect) by Bau et al (2017). They associate single filters of a convolutional DNN with one or several predefined concepts. To do this, upscaled and denoised activation maps of each filter were compared with ground truth concept segmentation masks, and the concepts with best agreement are picked. They also introduced the BRODEN dataset combination later often used as a baseline in other methods.

Net2Vec. A direct successor of NetDissect was Net2Vec (Fong and Vedaldi, 2018) where not a filter is associated to several concepts, but a concept is associated to a linear combination of filters. This is done by training a linear model to classify denoised activation map pixels into "belongs to concept" or "does not belong to concept", resulting in binary segmentation masks. For the implementation (cf. Figure 4), a 1 × 1-convolution is used, the kernel weights of which are the CAV for the concept. For linear models like this the CAV points in the latent space direction of the concept as illustrated in Figure 3. Investigation of those CAVs for the first time revealed that such linearly obtained concept vectors behave like word vectors in word vector spaces, with cosine similarity encoding some semantic similarity.

Table 3 Properties of concept analysis methods discussed in Subsubsection 4.2.1 according to the taxonomy defined in Subsection 3.2.

Method	**Enforced**	**Superv.**	**Model**	**Distance measure**	**Optim.**	**Label type**	**Input**
Post-hoc concept localization (Subsubsection 4.2.1)							
NetDissect (Bau et al, 2017)	-	✓	linear	dot	picking	seg	pixel
Net2Vec (Fong and Vedaldi, 2018), Schwalbe (2021)	-	✓	linear	dot	log. regr.	seg	pixel
Schwalbe and Schels (2020), Rabold et al (2020)	-	✓	linear	dot	log. regr.	det	window
TCAV (Kim et al, 2018), Chyung et al (2019)	-	✓	linear	dot	SVM	cls	full
CLM (Lucieri et al, 2020)	-	✓	linear	dot	SVM	seg	full
RCV (Graziani et al, 2018, 2020)	-	✓	linear	dot	LLS	reg	full
CHAIN (Wang et al, 2020)	-	✓	linear	L_2	Bregman	seg	pixel
Local CAVs (Zhang et al, 2018b)	-	✓	loc. lin.	cosine	Lasso	seg	window
AfI (Wu et al, 2020)	-	✓	loc. lin.	dot	custom	cls	full
SeVec (Gu and Tresp, 2019)	-	✓	cluster	cosine	?	cls	full
CME (Kazhdan et al, 2020)	-	semi	cluster	learned	label spreading	mcls	full
IIN (Esser et al, 2020)	-	(✓)	DNN	L_2	custom	any	full
CBM student (Haselhoff et al, 2021)	-	✓	linear	cosine	custom	cls	full

Net2Vec extensions. There are several extensions of Net2Vec to concept center point prediction (Schwalbe and Schels, 2020; Rabold et al, 2020) instead of binary segmentation, and to object detection DNNs (Schwalbe, 2021). In both Schwalbe and Schels (2020) and Rabold et al (2020) a new ground truth encoding is used that highlights center points of object part concepts. This allows to learn center point prediction even if only binary segmentation masks are available, which is the case for most standard CA datasets. Schwalbe and Schels (2020) further suggests and compares different optimization settings for the concept model and suggests to increase the convolution kernel size from one pixel to a window, in order to provide more context for each single prediction.
Net2Vec for OD. This is used and further evaluated in Schwalbe (2021), where efficiency is improved to apply a modified Net2Vec method to large object detector CNNs.

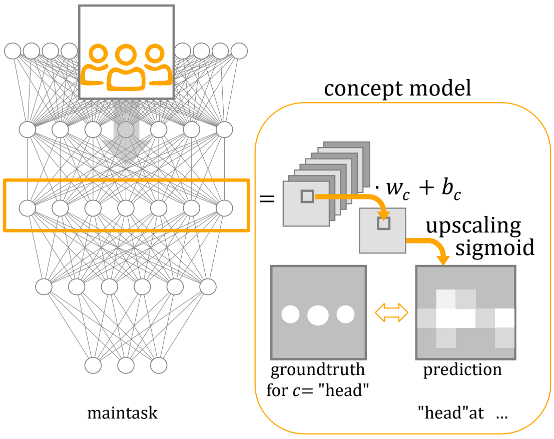


Figure 4 Exemplary linear concept localization model from Schwalbe (2021). The simple Net2Vec (Fong and Vedaldi, 2018) approach learns to predict segmentation masks for head using a 1 × 1-convolution followed by upscaling and normalization. The kernel vector w_c then is the CAV. *(Figure taken from (Schwalbe, 2021, Fig. 1))*

TCAV. In parallel with Net2Vec, the similar method TCAV (Kim et al, 2018) for linear CAV retrieval was developed. Instead of single pixels they consider the complete activation map, and their linear support vector machine (SVM) concept model does image-level concept classification. They also suggest relative CAVs by training a concept model to differentiate just two concepts. Furthermore, in this work some concrete application of CAVs for fairness analysis are suggested and demonstrated which will be discussed later.

CLM. TCAV has also inspired several extensions. One of them is CLM by Lucieri et al (2020), where the concept classification is refined to a concept segmentation via the input attention to a concept output. The intriguingly simple approach achieved mediocre performance both for perturbation- and gradient-based heatmapping on a generated dataset and the CelebA dataset (Liu et al, 2015).

RCV. Another extension of TCAV concept models are Regression Concept Vectors (RCV) by Graziani et al (2018). Here, instead of binary classification, the linear model is trained to regress a continuous-valued concept using linear least squares (LLS) optimization. They also suggest an improved global concept-to-output attribution score (cf. Subsection 5.3) and demonstrate their approach on medical data in Graziani et al (2018) and Graziani et al (2020). The later work Graziani et al (2020) on RCVs, just as Kazhdan et al (2020), proposes to apply global average pooling to the full activation maps before training the linear classifier. This reduces the latent space and CAV size.

CHAIN. Not based on TCAV or Net2Vec, but also linear in nature, are the binary segmentation concept models proposed in the work on Concept-harmonized HierArchical INference (CHAIN) (Wang et al, 2020). Using Bregman iteration these are trained for a least squares distance to concept samples. The core idea of CHAIN is to assess the dependencies of CAVs in later layers with respect to CAVs in earlier layers. This turns a DNN into a concept graph of hierarchical inference.

Locally linear concept localization

Some CA methods do not rely on linear concept models, but still locally give rise to a linear interpretation and CAVs (loc. lin.).

Local CAVs. One such approach is pursued by Zhang et al (2018b) in order to locally represent outputs of a DNN by activation map vectors. They consider the derivative vector of the output with respect to the masked activation map. The mask for this purpose is learned globally for the respective output, using a Lasso-like optimization. Though the global concept model—containing the derivative function of the DNN—is highly non-linear, per-sample this still gives rise to comparable local CAVs.

AfI. Another approach, which is also locally producing CAVs, is Attacking for Interpretability (AfI) (Wu et al, 2020). Here, concepts represented in the final output classes of a classifier are located within activation maps. To obtain a CAV for an input instance and a selected output concept, they consider the local neuron-to-output attribution of neurons in a layer to the output of interest. The local CAV is the result of a global average pooling on the neuron attribution maps. In this case, the attribution is not defined via sensitivity, but as the difference between the vanilla and a perturbed version of the input, with perturbation done by a trained global attacker model.

Non-linear concept localization

SeVec. A simple clustering-based approach to obtain CAVs is followed by SeVec in Gu and Tresp (2019). Here, the (binarized) activation vectors for positive concept examples are clustered by cosine distance. The center of the most prominent cluster is taken as a CAV for the concept. From their diverse experiments, they report that there always was a dominant cluster containing most of the positive samples.

CME. Another example of CAVs from clustering under supervision was used for Concept-based Model Extraction (CME) by Kazhdan et al (2020). Their two specialties are that they also consider multi-class concepts like shape (with values, e.g., rectangle, triangle, circle), and that they assume that also unlabeled inputs are available for concept training. Thus, they use a semi-supervised multi-task learning approach, namely label spreading for spectral clustering[3], to train their concept models. Similar performance results to Net2Vec are reported on the dSprites dataset (Matthey et al, 2017) and the Caltech-USCD birds dataset (Wah et al, 2011).

IIN. Other than the clustering-based approaches, Esser et al (2020) train a bijection between a layer's output space to a vector space of the same size. The idea is that clusters of dimensions in that vector space represent given concepts. The bijection is realized by invertible DNNs, so-called normalizing flows, and termed Invertible Interpretation Network (IIN). An interesting label format is used: The image-level concept vectors are trained to be the difference vector between pairs of samples representing an increase of the concept (e.g., non-smiling → smiling). They also suggest to use their approach for concept mining by only enforcing independence of the dimension clusters.

[3] See, e.g., the used implementation at https://scikit-learn.org/stable/modules/semi_supervised.html#label-propagation

Table 4 Properties of concept analysis methods discussed in Subsubsection 4.2.2 according to the taxonomy defined in Subsection 3.2.

Method	**Enforced**	**Superv.**	**Model**	**Distance measure**	**Optim.**	**Label type**	**Input**
Post-hoc concept mining (Subsubsection 4.2.2)							
Feature visualization (Olah et al, 2017; Nguyen et al, 2019)	-	-	linear	L_2	–	seg	channel
ACE (Ghorbani et al, 2019), VRX (Ge et al, 2021)	-	-	cluster	L_2	k-means	cls	full
Yeh et al (2020)	-	-	cluster	L_2	custom	cls	full
NBDT (Wan et al, 2020)	(✓)	-	cluster	L_2	HAC	cls	full
Explanatory Graph (Zhang et al, 2018a)	-	-	(cluster)	-	custom	det	window
NCAV (Zhang et al, 2021), Saini and Papalexakis (2020)	-	-	linear	L_2	matrix fact.	seg	pixel

CBM student. While all previous supervised methods assume full white-box access to the DNN of interest, Haselhoff et al (2021) break with this assumption and investigates how to obtain concept attribution from a black-box function. The idea pursued in the teacher-student approach in Haselhoff et al (2021) is to distill a concept bottleneck model from a trained function. Their student model is a concept bottleneck with a ResNet50 (He et al, 2016) back-end and a Gaussian discriminant analysis after the concept bottleneck. This allows a supervised concept analysis on black-box functions.

4.2.2 Concept mining

In literature, two method classes for finding concepts in latent space representations are prominent: Clustering-based approaches, and matrix factorization (matrix fact.) approaches. Examples of both are discussed in the following.

Concept mining using clustering

Feature Visualization. As for concept localization, the simplest approaches of unsupervised post-hoc concept analysis, or concept mining, are to investigate what concepts are encoded by single units of a DNN, like filters of a convolutional DNN. This is done in the broad spectrum of methods for feature visualization (Olah et al, 2017; Nguyen et al, 2019) where exemplary inputs or generated prototypes help to explain the meaning of such single units.
ACE. An approach respecting distributed representations of information is followed by Automatic Concept-based Explanations (ACE) in Ghorbani et al (2019) for visual concepts. They assume that meaningful visual concepts are represented by superpixels, i.e. connected patches, in images. Thus, they

cluster the latent space representation of size-normalized super-pixels using k-means clustering, and associate a concept with each cluster while the centers of the clusters serve as CAVs.

(Yeh et al, 2020). The same approach is utilized by the direct successor by Yeh et al (2020), but with additional custom optimization criteria for the clustering. These shall ensure natural properties of concepts like spatial proximity of similar concepts. Also, they aim to capture a selection of concepts that covers the complete amount of information encoded in one latent space.

VRX. Another extension of ACE is the Visual Reasoning eXplanation framework (VRX) from Ge et al (2021). Before applying the super-pixel segmentation to an image, they mask out regions in an image that had little attribution to the output. For the attention scoring they use the standard heatmapping method Grad-CAM (Selvaraju et al, 2017).

NBDT. Other than previous k-means based methods, Neural-Backed Decision Trees (NBDT) (Wan et al, 2020) uses hierarchical agglomerative clustering (HAC) to find a hierarchy of CAVs in the last DNN hidden layer. The leaves of the hierarchical tree are concepts represented by the DNN outputs, with their CAVs being the respective weights from the last hidden layer. Using HAC with respect to L_2 distance on those leaves, a hierarchy of clusters is established. As before, the cluster center points are interpreted as CAV. For example, this hierarchy of CAVs can be used to define a decision tree surrogate model.

Explanatory Graph. A totally different approach that is loosely related to hierarchical and spectral clustering is pursued by the Explanatory Graph (Zhang et al, 2018a) extraction method. Their goal is to represent the inference of a CNN purely via a hierarchical graph of part objects and their part-relation: From small and simple concepts in earlier layers, to large and complex concepts in later layers. As constraints to define proper concepts, they assert that a concept prediction must (1) coincide with a high activation of a filter map; and (2) be spatially consistent with predictions of its parent concept in the successive layer. Due to this hierarchical dependency of concepts, they always produce a family of concept models that can only be inferred following the hierarchical structure. These concept models are each anchored and restricted to an image region.

Concept mining using matrix factorization

NCAV. A generalization of the previous clustering methods was proposed with Non-negative CAVs (NCAV) in Zhang et al (2021). They view k-means clustering as a special case of matrix factorization, and use this to find pixel-wise linear concept models for CNNs, i.e. linear weights for the convolutional filters. The idea of matrix factorization here is to find a global matrix P (of size #channels × #concepts) for a layer, where the columns of the matrix encode CAVs. For each input a matrix S is obtained via optimization in a way that $S \cdot P$ approximately is the original activation map. S then consists of the pixel-wise scores for the concepts, i.e. the per-pixel outputs of the concept models. The work compares three matrix factorization approaches, namely k-means clustering, principal component analysis, and non-negative matrix factorization for ReLU nets, with most interpretable concepts found by the last one.

(Saini and Papalexakis, 2020). A similar approach to NCAV, also using non-negative matrix factorization, is pursued by Saini and Papalexakis (2020). The important difference is that they do not only associate *concepts* with filters (cf. dimensionality of P), but also input pixels and single neurons at once, leading to much more memory intensive optimization, but a little more direct association of concepts to input parts.

4.2.3 Complete latent space disentanglement

Some of the discussed methods allow not only to replace parts of the model but provide an (invertible) interpretable representation for a complete layer output. This is relevant if it is the aim to capture as much of the encoded information as possible, or to turn a model post-hoc into a concept bottleneck model. Some of the discussed methods are recapitulated here under this aspect.

Concretely, Yeh et al (2020) suggest a measure for evaluating completeness of a set of concepts: They intercept the connection between two layers by an interpretable intermediate output where each single dimension corresponds to one concept. While concept models are used to connect the earlier layer with the interpretable units, linear weights are learned to connect the interpretable representation with the next layer. They take the performance drop due to this interpretable interception as a measure for the *completeness* of the set of chosen concepts. Such an interpretable interception is also achieved by the matrix factorization approach of NCAV. Their idea to approximately represent the layer output as a product $S \cdot P$ of interpretable matrices allows to directly replace the layer output with this concept-based approximation. And the NBDT approach inherently ensures to capture all relevant information of the last hidden layer they consider: They take into account all weights connecting the last hidden with the output layer. Note that the supervised IIN method also achieves concept-based latent space disentanglement, but adding residual “non-semantic” dimensions as a fallback for encoded but not predefined concepts, to ensure invertibility.

5 Applications

This chapter provides an overview of core applications of concept analysis, and discusses interesting examples. The first sub-section 5.1 considers applications that utilize the additional concept outputs produced by concept analysis. The interactive model correction method of concept intervention is discussed in Subsection 5.2, and Subsection 5.3 gives an overview of useful metrics and approaches for qualitative verification based on CA.

Table 5 Overview of the presented model distillation approaches based on concept analysis outputs

Distillation Method	**Surrogate Model Type**
CA & ILP (Rabold et al, 2020)	Expressive rules
CME (Kazhdan et al, 2020)	Logistic regression, decision tree
Additive explainer (Chen et al, 2019b)	General additive model
Interpretable Basis Decomposition (Zhou et al, 2018)	Local linear model
NBDT (Wan et al, 2020)	Decision tree (semantic hierarchies)
Explanatory Graph (Zhang et al, 2018a)	Hierarchical graph (only part-relations)
CHAIN (Wang et al, 2020)	Hierarchical graph (inference dependencies)
VRX (Ge et al, 2021)	Graph DNN
CBM student (Haselhoff et al, 2021)	Concept bottleneck model

5.1 Using enriched DNN output

Several applications require access to a rich set of features in the DNN (intermediate) output. Concept analysis provides methods to either enrich the output after training the model with few data samples, or directly and easily include rich semantic outputs into the architecture. We first discuss examples of model distillation utilizing concept analysis, and then verification methods that rely on these rich outputs.

5.1.1 Model distillation

By *model distillation* we understand the approximation of the functionality of the complete or parts of a DNN by a more interpretable model, which can be, e.g., a linear one or a decision tree (Arrieta et al, 2020). Model distillation into an interpretable surrogate model requires interpretable symbolic inputs for that surrogate model, such as color or age. However, many input domains for DNNs are non-symbolic, like images. Post-hoc attached concept model outputs can be used to provide the necessary symbolic input to the surrogate. This allows to replace bottom parts of the DNN similar to the fine-tuning approach used for the Semantic Bottleneck Models in Losch et al (2019) (cf. Figure 2). In the following, examples for different types of surrogate models are summarized. An overview is given in Table 5.

Rules and additive models

CA and ILP. For example, in Rabold et al (2020), the later layers are replaced by a set of rules learned using inductive logic programming. Also, the concepts are not located in one layer, but for each concept the layer with the best embedding is chosen.

CME. CME (Kazhdan et al, 2020) also considers all layers to find the best embedding of the predefined set of concepts. They, however, rely on simpler surrogate models, namely a linear one trained using logistic regression, and decision trees.

Additive explainer. Another example of an additive surrogate model for a classifier is the additive explainer model presented in Chen et al (2019b). As concept models they use the approach by NetDissect (Bau et al, 2017), in order to associate a predefined concept with the filter of a layer that mainly activates for this concepts. Chen et al (2019b) now trains a general additive model to predict the final classification output from those concept model outputs. For each instance, the general additive model predicts weights to linearly combine the concept model outputs.

Interpretable Basis Decomposition. A simple but local linear surrogate is considered in the Interpretable Basis Decomposition method (Zhou et al, 2018). They decompose a latent space vector representation of an input into a linear combination of CAVs. This is done by means of greedy least squares basis decomposition. Concretely, those local surrogates are then used to provide per-concept attribution heatmaps.

Hierarchical models

NBDT. The method NBDT (Wan et al, 2020) mines a hierarchy of CAVs, each of which represents the center point of a (sub-)cluster. They suggest to use this concept hierarchy to define a decision tree: Top-down, at each node one chooses the sub-cluster for which the representing CAV and the current activation vector have the smallest dot-product (note: not L_2 distance as used for the clustering). To ensure good fidelity of the decision tree, a fine-tuning loss is tested that encourages the model to better sort activations into the correct clusters.

Explanatory Graph. Other than NBDT, which is restricted to the last layer, the Explanatory Graph method (Zhang et al, 2018a) discussed in Subsubsection 4.2.2 represents the complete model by a hierarchy of concepts. For this, they mine concepts from different layers. Here, the type of concepts and hierarchical relations are restricted to object parts and part relations.

CHAIN. Similarly, but without that restriction, the super-vised method CHAIN (Wang et al, 2020) produces a hierarchical inference graph. CHAIN linearly models how concepts in later layers depend on concepts in earlier layers. For this, pixel-wise linear CA models are used that provide CAVs. The main idea for obtaining the weights for linear dependency in CHAIN is to lift CAVs of later concepts to earlier layers. Weights are then trained to represent a lifted CAV as linear combination of CAVs from that layer.

Complex models

VRX. To capture and model both spatial and semantic relations between mined visual concepts, VRX (Ge et al, 2021) distills a graph neural network. As inference input, concepts are detected in the image (associated with image patches), and a fully-connected graph of the matched patches is created that encodes spatial relationships and (learnable) semantic relationships of concept instances.

CBM student. An approach to distill an interpretable model from a classifier applicable without white-box access is chosen in Haselhoff et al (2021). As a surrogate, they train a concept bottleneck model with the part after the bottleneck being a decoder based on Gaussian discriminant analysis. This setup quite easily allows to determine the attribution of a concept to an output as the ratio of concept log likelihood and output class log likelihood.

5.1.2 Verification of symbolic properties

Besides model distillation, another usage of rich DNN output is for the verification of logical properties. Logical properties must be defined on symbolic features in the DNN (intermediate) output, which can be accessed using concept analysis. An exemplary verification of hierarchical relations like "dogs are mammals" is conducted in Roychowdhury et al (2018). They translate first-order logic rules into continuous-valued fuzzy logic rules that accept output scores of the DNN. The truth values of the rules can then be tested on a test set.

5.2 Concept intervention

For interactive human-machine-systems, like experts interacting with an assistant system, it may be helpful to allow an expert to correct intermediate steps in the model decision process. This can be done by changing the intermediate output of the model. However, to allow helpful intervention, an expert both needs to (1) understand the meaning of the intermediate output, as well as (2) be able to modify it according to symbolic knowledge. Part one is solved by using concept analysis methods. Part two can be tackled using so-called *concept intervention*. There are two approaches that rely on different prerequisites: (a) Either a vector representation of the concept must be available, or (b) the concept model output must directly influence later reasoning steps of the considered model.

Concept model output intervention

If the concept model output is part of the model inference process, the expert can manually change the output of the concept model and re-do the inference from the concept model output onwards. This is suggested and done for CBMs in Koh et al (2020). CME (Kazhdan et al, 2020) does this post-hoc for standard DNNs. They attach concept outputs post-hoc and then train a surrogate upon them. The surrogate decisions can now be intervened by changing concept outputs.

Concept representation intervention

The other setting for concept intervention is that a latent space vector is given that represents the direction towards a concept in the latent space. Then, the intermediate output vector in that latent space can be modified to point more or less into the direction of the concept. For example, projecting it to the plane orthogonal to the CAV eliminates the concept from the intermediate output. The modified vector can then be fed to the next layer. This approach is, e.g., used for the Counterfactual Explanation Score (CES) (Abid and Zou, 2021), where the difference between adding the concept and removing the concept is measured. In GAN dissection (Bau et al, 2018), the concept localization method NetDissect (Bau et al, 2017) is used to associate single units in a generative adversarial network for image generation with semantic concepts. Intervention—i.e. manipulation—of those unit values results in spatially local semantic manipulation of the created image.

5.3 Analysis of concept model properties

Concept analysis gives rise to some interesting qualitative and quantitative analysis options. These can be used to globally or locally verify a model by uncovering failure modes (Zhang et al, 2018b), or allowing for manual inspection of the rich semantics assessable via concept analysis. In the following, we first discuss concept-driven qualitative inspection methods enabled by CA (Subsubsection 5.3.1), and then quantitative measures related to CA results (Subsubsection 5.3.2).

5.3.1 Qualitative analysis methods

Some qualitative analysis methods are enabled by CA which are discussed in the following.

Local input-to-concept attribution. In the Interpretable Basis Decomposition approach (Zhou et al, 2018) the attribution of input pixels to a concept is visualized. For this, they use a standard and interchangeable heatmapping method (Grad CAM by Selvaraju et al (2017)), and treat the post-hoc attached concept model outputs as regular DNN outputs. They consider concepts that define a local basis for the output in order to semantically dissect the attribution to the final output. In CLM (Lucieri et al, 2020) also a perturbation based attribution method is applied and the heatmaps are used to allocate concept classification results to image regions.

Concept prototypes. For manual inspection, it may be interesting to get an impression of how a model "perceives" a localized or mined concept. For this, one can either provide examples (Ghorbani et al, 2019), patch examples to get a prototype (Ghorbani et al, 2019), or optimize a candidate to obtain a prototype for the concept. The latter is done in Kim et al (2018) by applying the DeepDream (Mordvintsev et al, 2015) optimization with respect to the concept outputs.

5.3.2 Quantitative properties

The following semantically grounded metrics directly arise from concept analysis.

Notation: We consider an exemplary DNN f and a concept c in layer l with CAV w_c. T denotes a test set of samples, and p a single input sample. The DNN sub-function up to layer l is denoted $f_{\to l}$, $f_{l\to}$ denotes the one from layer l onwards, and—in case f has multiple outputs—the function for the kth output of f is written f^k.

Concept embedding quality. The following types of concept embedding quality measures are proposed in literature:

- As suggested in Schwalbe and Schels (2020), the best achievable performance of a concept model for a concept can be interpreted as the *concept embedding strength* or quality, i.e. how well information about the concept is embedded in the DNN. Standard performance metrics are, e.g., accuracy or R^2 for binary classification (Kim et al, 2018), and set intersection over union for binary segmentation (Fong and Vedaldi, 2018; Schwalbe, 2021).
- *Misprediction overlap (MPO)*: In case of a family of concept models, Kazhdan et al (2020) argues that it is important to assess the distribution of errors over concepts. They suggest to measure the misprediction overlap as the proportion of test samples for which at least n relevant concepts are mispredicted.

Concept similarity. As discussed in Subsection 3.2, there are several approaches to measure the similarity of concept embeddings in the case CAVs are available. These usually rely on CAVs residing in the same layer, even though CAV-lifting can be performed to also compare concepts embedded in different layers as done in Wang et al (2020). The embedding similarity can be compared with prior knowledge on the ground truth semantic similarity for verification purposes as done in Schwalbe (2021). Here, one has to differentiate between globally and locally available CAVs:

- As discussed in Subsection 3.2, global CAVs are often compared using one of *cosine similarity* (e.g., Fong and Vedaldi (2018); Schwalbe (2021); Kim et al (2018)) or L_2 *distance* (e.g., Ghorbani et al (2019); Yeh et al (2020); Wang et al (2020)).
- For the local CAVs used in Zhang et al (2018b), the authors suggest to assess the *distribution of concept relation values* given a test set T. As measure of concept similarity for CAVs in the same layer they consider cosine distance. They assume a prior distribution of values and apply Kullback-Leibler divergence to compare this with the measured distribution.

Local concept-to-output attribution. Given a sample, there are several suggestions on how to quantify the attribution of a concept embedding to an output (or another concept model output).

- *Local TCAV score*: Kim et al (2018) measures local sensitivity (Baehrens et al, 2010) of an output with respect to a CAV by measuring the partial derivative along the CAV:

$$\mathrm{TCAV}(f;\ c \to k)(p) := \nabla f^k_{l\to}(f_{\to l}(p)) \cdot w_c \tag{1}$$

- *Counterfactual explanation score (CES)*: To robustify the TCAV sensitivity score, Abid and Zou (2021) suggests to use an approximation of the direct derivative via a step-perturbation. Using concept intervention (cf. Subsection 5.2), they add the concept of interest by moving the latent space vector one step into the direction of the CAV. For a step size δ (originally 10,000) the approximation is then

$$\mathrm{CES}(f;\ c \to k)(p) := f^k_{l\to}(f_{\to l}(p) + \delta w_c) - f^k(p)\ .$$

- The CHAIN method by Wang et al (2020) assesses the *local hierarchical dependency* of pixel-wise CAVs in later layers to CAVs in earlier layers. They suggest to use the partial derivative of a concept output along the CAV of an earlier-layer concept as local concept-to-concept attribution.
- An approach not relying on CAVs is to find a *linear surrogate model* based on concept outputs that directly provides concept-to-output attribution weights, as is done in a local fashion in Chen et al (2019b).

Global concept-to-output attribution. Local concept-to-output attribution values can be aggregated on a test set T to an approximate global attribution value. While this was first suggested in Kim et al (2018), the aggregation method was refined in later work:

- The original way to aggregate local TCAV scores as defined in Equation 1 to a *global TCAV score* was proposed in Kim et al (2018). The metric is specific to classifiers since it relies on a partition of the test set T into subsets T_k of the respective output class k. What is measured is the proportion of samples of a class k with positive local TCAV sensitivity score:

$$\mathrm{TCAV}_{\mathrm{global}}(f;\ c \to k) := \frac{1}{\#T_k} \sum_{p \in T_k} \mathbb{1}_{\mathrm{TCAV}(f;\ c\to k)(p)>0} \tag{2}$$

- The *normalized bidirectional relevance* (NBR) is an improved version of Equation 2 suggested in Graziani et al (2018). They aggregate local sensitivities as: (1) the mean sensitivity $\mu_{c\to k} = \mathrm{mean}_{p\in T_k}\, \mathrm{TCAV}(f;\ c \to k)(p)$ for an output class k over samples T_k of that class, (2) weighted by the concept model's R-squared value R^2_c, and finally (3) normalized by the standard deviation $\sigma_{c\to k}$ of the sensitivity:

$$NBR(f;\ c) := \frac{\mathrm{R}^2_c \cdot \mu_{c\to k}}{\sigma_{c\to k}} \tag{3}$$

This yields stronger sensitivity for accurate concept models and such with less output variance (Graziani et al, 2020).

- The CHAIN method lifts CAVs from later layers to an earlier layer in order to represent the lifted CAV as linear combination of CAVs in this earlier layer. These *global concept hierarchical inference weights* are further used to define a surrogate model (cf. Subsubsection 5.1.1).
- Sometimes, one is interested in the influence of concept c_A on concept c_B, however, has no concept model for c_A, but only concept samples. In this case, both the AfI framework (Wu et al, 2020) and Schwalbe (2021) suggest to observe the discrepancy in c_B's output between a set of samples with c_A, and a set without c_A. In Schwalbe (2021) the discrepancy is simply measured as the *discrepancy in performance* of c_B. AfI measures the *maximum mean discrepancy (MMD)* (Gretton et al, 2012), a standard measure to compare two distributions, between the output distributions of c_B on the different input datasets.

Local concept interactions. One might be interested in the *local interaction of concepts c_A and c_B*, i.e. how the local attribution of c_A to an output will change with changes to c_B. In case of CBMs, or if c_A and c_B have a CAV in the same layer, standard interaction analysis methods can be applied, such as integrated Hessians (Janizek et al, 2020).

Concept embedding validity. As was already found in Kim et al (2018) and confirmed in Rabold et al (2020), especially linear (supervised) concept models are non-robust with respect to outliers, and the training may not be stable. Thus, measures are necessary that confirm whether a found CAV is a stable and meaningful solution or not:

- In Kim et al (2018) a *t-test* is used to check whether several CAVs trained for the same concept behave consistently.
- In Pfau et al (2021) a *hypothesis test* is used that checks whether the concept sensitivity is higher than that of a reference noise vector. This is also applicable to check validity of mined CAVs.

In Rabold et al (2020) an ensembling approach is suggested to stabilize linear models. An invalid or instable concept embedding may unmask an inconsistently defined concept, or an inappropriate CA method.

Concept intervention scores. The concept intervention used in Koh et al (2020) (manually changing the bottleneck outputs of CBMs) may serve to measure the influence of wrongly predicted concepts onto the final decision. In that work they compare the performance of the original CBM with CBMs with an increasing rate of concepts being intervened, i.e. their concept bottleneck outputs being replaced with ground truth labels. They found that for most concepts the correct prediction is substantial for correct output of the final model (cf. (Koh et al, 2020, Fig. 4)). Concept intervention scores can also be seen as a local concept-to-output attribution measure that is perturbation-based and specific to CBMs.

Concept completeness. The notion of concept completeness was introduced in Yeh et al (2020). It measures how much of the task-relevant information encoded in a DNN is covered by a set of concept embeddings.

They suggest to measure this as the decrease in model performance when turning it post-hoc into a concept bottleneck. The bottleneck is inserted between layers l and $l+1$ as follows: First, the bottleneck layer after l is inserted, with the connections from l to the bottleneck concept outputs being the concept models. Then, weights are learned to connect the bottleneck to the next layer $l+1$. This differs from Semantic Bottlenecks (Losch et al, 2019), where the complete part $f_{l\rightarrow}$ is replaced and retrained. While the original method from Yeh et al (2020) assumes the existence of linear models defined by CAVs, the performance discrepancy can be measured for any other method that post-hoc inserts a bottleneck (cf. Subsubsection 4.2.3).

Concept interpretability. Due to the distributed representations occurring in a DNN, one unit, respectively one filter in case of CNNs, may be involved in representing several semantic concepts. In the work on NetDissect, Bau et al (2017) suggest to take the amount of these concepts as a measure for the interpretability of the filter. Fong and Vedaldi (2018) inverted this and took sparsity of a CAV (i.e. how many filters are needed to encode the concept) as a measure for the interpretability, or *disentanglement*, of the concept representation (cf. Subsection 3.2).

6 Image datasets for supervised CA

In this chapter we collect and compare different datasets that so far have been used for supervised concept analysis methods presented in Section 4. This is meant as a reference for researchers in search of suitable datasets for evaluating CA approaches, and to demonstrate the breadth of application fields. A tabular overview of the datasets and their key properties is compiled in Table 6.

Datasets are classified as follows: We start in Subsection 6.1 with datasets of diverse real world images that are suitable for practical baseline evaluation of DNNs for complex computer vision tasks. Next, in Subsection 6.2, a collection of domain specific image concept datasets are presented, that can serve for evaluation of domain specific applications, or of CA methods on more complex tasks. Lastly, an overview of "toy" and simple image datasets is given in Subsection 6.3 that contain few simple concept classes like shapes, textures, or colors. These may serve for initial experimentation or comparison.

6.1 Large and diverse real-world image datasets

The following large real-world image datasets with concept annotations are used throughout several papers to establish a performance baseline for CA approaches (Fong and Vedaldi, 2018; Kim et al, 2018; Schwalbe, 2021). They are suitable for practical evaluation of large DNNs in complex computer vision applications like object detection (Schwalbe, 2021).

ImageNet. ImageNet (Deng et al, 2009) is a popular and large image dataset that aims to provide sample images for the abundance of 80 000 hierarchical lexical synonym sets defined in the WordNet lexical database (Fellbaum, 1998). ImageNet features more than 3.2 million samples with binary classification

Table 6: Overview of image datasets used for supervised concept (C) analysis. Besides concept type and label type, we collected the number of samples, concept classes, and the number of samples per least dominant class value of the concept. Abbreviations: attr.=attributes, avg.=average, equ. dist.=equally distributed.

Name		**C Label Type**	**C Types**	**# Samples**	**# C Classes**	**# Samples / C Class**
Large diverse datasets (Subsection 6.1)						
ImageNet	(Deng et al, 2009)	cls,det	objects, parts	3 000 000<	5247	600 avg.
BRODEN	(Bau et al, 2017)	seg,cls	all	63 305	1197	10 ≤
BRODEN+	(Xiao et al, 2018)	seg,cls	all	57 095	1002	20 ≤
MS COCO	(Lin et al, 2014)	seg	body parts	118 287	9	40000 ≤
Domain specific datasets (Subsection 6.2)						
LFW	(Huang et al, 2008)	cls	gender	13 233	1	-
LFWA+	(Liu et al, 2015)	cls	facial attr.	13 233	40	-
CelebA	(Liu et al, 2015)	cls	facial attr.	202 599	40	-
FASSEG	(Khan et al, 2015)	seg	facial parts	270	3	-
Picasso	(Rabold et al, 2020)	seg	facial parts	452	3	452
Knee X-Rays	(Koh et al, 2020)	cls	clinical attr.	36 369	10	5% ≤
CUB	(Wah et al, 2011)	cls,det	bird attr./parts	11 788	327	100%
Simple datasets (Subsection 6.3)						
GTSRB	(Stallkamp et al, 2011; Schwalbe and Schels, 2020)	seg	color, shape, part	50 000<	10	211 ≤
A-GTSRB	(Stallkamp et al, 2011; Kronenberger and Haselhoff, 2019)	cls	color, shape, part	70 000<	32	211 ≤
FMD	(Sharan et al, 2014)	cls	material	1000	10	100
Google-512	(Schauerte, 2010)	cls	color	5632	11	512
dSprites	(Matthey et al, 2017)	cls,det	shape, spatial	737 280	4	equ. dist.
3dshapes	(DeepMind, 2021)	cls	shape, spatial	480 000	6	equ. dist.
SCDB	(Lucieri et al, 2020)	seg	shape	6000	10	equ. dist.

labels for 5247 hierarchically related synonym sets. The number of samples per synonym set varies a lot, with an average of 600 samples per set, and 50% of the sets containing more than 500 samples. Selected concept classes served, e.g., for validation purposes in the TCAV paper by Kim et al (2018).
Source: https://image-net.org/

BRODEN. The BRoadly and DEnsely labeled dataset (BRODEN) was first introduced in Bau et al (2017). It unifies diverse other image datasets (ADE20k by Zhou et al (2017), OpenSurfaces by Bell et al (2014), Pascal-Context by Mottaghi et al (2014), Pascal-Part by Chen et al (2014), and the Describable Textures Dataset by Cimpoi et al (2014)). In total, Broden features 63 305 different images resized to unified resolution 224 × 224, 227 × 227, or 384 × 384. With 1197 classes in total, the dataset labels encompass pixel-wise concept segmentations of types color (11), texture (47), material (32), object (584), part object (234), and classification labels for 468 scene type concepts. The amount of labels varies greatly, going down to as few as 10 samples per class.
Source: https://github.com/CSAILVision/NetDissect

BRODEN+. In Xiao et al (2018) the original BRODEN dataset was reworked to make classes more distinct, and feature a sufficient number of samples per class to serve for DNN training. Mainly, they merged highly similar concept classes, removed color classes and classes with too few samples, and renamed labels to match the naming in the Places dataset (Zhou et al, 2014). This results in 57 095 image samples and 1002 classes in total.
Source: https://github.com/CSAILVision/unifiedparsing

MS COCO Bodyparts. The Microsoft Common Objects in COntext object detection dataset (MS COCO) (Lin et al, 2014) features over 1.7 million keypoint labels for over 150 000 person instances in 118 287 training and 5000 validation images with comparatively high resolution. Schwalbe (2021) suggests and demonstrates a general method to turn person keypoint labels into CA-ready segmentations of body parts described by keypoints (here: eye, nose, ear, arm, leg, wrist, ankle, shoulder, hip).
Source: https://cocodataset.org/

6.2 Domain specific image datasets

This section collects several domain specific real-world image datasets that have so far been used for use-case specific evaluation of CA methods. Since the collection is dominated by concept datasets for facial features annotated in portraits, these are gathered into their own subsection (Subsubsection 6.2.1) and discussed first. Then, two further examples of detailed labeled domain specific datasets are presented in Subsubsection 6.2.2.

6.2.1 Facial features concept datasets

The following facial features datasets have served for CA evaluations:
LFW. The Labeled Faces in the Wild (LFW) dataset (Huang et al, 2008) features 13 233 portraits of 5749 people with labels of name and gender. It was

used by Kim et al (2018) in the fairness evaluations of the TCAV method.
Source: http://vis-www.cs.umass.edu/lfw/

LFWA+. The LFWA+ dataset provides the CelebA additional label types of 5 landmark locations and 40 binary attributes for the original LFW image samples. It was introduced in Liu et al (2015) alongside the CelebA dataset.
Source: https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

CelebA. The Large-scale CelebFaces Attributes (CelebA) dataset (Liu et al, 2015) contains 202 599 portraits of 10 177 celebrities annotated with 5 landmark locations and 40 different binary attributes, like bald, mustache, and makeup. It was used, e.g., in Lucieri et al (2020) for evaluation of the concept localization method CLM on a model trained on CelebA for gender classification. The same label classes are used as in the LFWA+ dataset.
Source: https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

FASSEG. The FAce Semantic SEGmentation dataset (Khan et al, 2015) contains 70 frontal face and 200 multi-pose face images with semantic segmentation of the object part concepts eye, nose, mouth. The portraits are sections from images taken from the MIT-CBCL (for Biological and Computational Learning , CBCL), FEI (Thomaz and Giraldi, 2010), and Pointing04 (Gourier et al, 2004) datasets. FASSEG served as a basis for the generated dataset used in Rabold et al (2020) for CA-based surrogate model creation.
Source: http://massimomauro.github.io/FASSEG-repository/

Picasso. In Rabold et al (2020) the generated Picasso dataset is presented for binary classification of images into "valid" and "invalid" face. In each image, clips of facial features (eye, nose, mouth) from the FASSEG dataset (Khan et al, 2015) are pasted upon portrait images in which facial features were erased. The arrangement of the features is considered an "invalid" (scrambled) face if the feature arrangement is unnatural, e.g., eye and nose swapped. For concept segmentation labels, the original segmentation information was retained. For each concept, 452 training samples and 48 test generated samples were used.
Source: https://github.com/mc-lovin-mlem/concept-embeddings-and-ilp/tree/ki2020

6.2.2 Other domain specific image datasets

In the following, two other examples of domain specific image datasets are described that feature interesting concept labels.

Knee X-Rays. For CBM training, in Koh et al (2020) a clinical dataset of diagnosis from knee X-ray images was used. The dataset was compiled by the Osteoarthritis Initiative (Initiative, 2021). The dataset prepared by Koh et al (2020) consisted of 36 369 X-ray observations of 4172 patients, at a high uniform resolution of 512 × 512 pixels. Koh et al (2020) filtered highly unbalanced concepts. Only those were included where more than 5% of the images feature the non-dominant class. In addition to the multi-value diagnosis classification, binary labels for further 18 other clinical concepts are included.
Source: https://nda.nih.gov/oai

CUB. The Caltech-UCSD Birds-200-2011 (CUB) dataset (Wah et al, 2011) consists of 11 788 bird images with classification annotations for the 200 contained bird species, and additional annotations of 15 part locations, 312 binary attributes (e.g., wing color, beak shape) with visibility information, and the bounding box of the depicted bird. The dataset was used, e.g., for CBM training in Koh et al (2020), and for concept localization in CME (Kazhdan et al, 2020).
Source: http://www.vision.caltech.edu/visipedia/CUB-200-2011.html

In TCAV, the medical use-case of predicting diabetic retinopathy from retinal fundus images was analyzed with respect to diagnostic concepts like microaneurysms or pan-retinal laser scars. However, the concept data is not freely available.

6.3 Simple evaluation datasets

While large real-world datasets may be interesting for a practical evaluation of DNNs or CA methods, simpler settings are preferred for initial method evaluation and comparison. Thus, the following subsection collects several small-resolution datasets that concentrate on annotations for simple concepts like shape, color, and texture. The examples are classified into datasets based on real-world images (Subsubsection 6.3.1) and artificial ones using generated samples (Subsubsection 6.3.2).

6.3.1 Simple real-world concept datasets

The following examples of real-world datasets provide annotations for simple concepts and can be used to validate small computer vision DNNs using CA.

GTSRB. The German Traffic Signs Recognition Benchmark (GTSRB) dataset (Stallkamp et al, 2011) is a classification dataset that features 43 classes of traffic signs, distributed onto more than 50 000 small (15 × 15 pixels) to middle (250 × 250) sized real world images. All images are close-up and fairly frontal, and the labels provide the exact bounding box of the contained sign. The bounding box labels allow to transform images to centered and uniformly sized sign photo sections. This was utilized in Schwalbe and Schels (2020) to automatically annotate bounding boxes for contained letters, by using static positions. This served as simplistic setting for evaluating different CA methods.
Source: https://benchmark.ini.rub.de/gtsrb_dataset.html

A-GTSRB. For the experiments in Kronenberger and Haselhoff (2019), an augmented version of the GTSRB dataset was created, enlarging it by about 60% and adding classification labels for diverse concept types: main color and border color (5 values each), shape (4), and contained letters (10) and symbols (13). The new data samples were created by domain randomization (new combinations of shape and color) and background randomization (adding patches to the background).
Source: https://benchmark.ini.rub.de/gtsrb_dataset.html

FMD. The Flickr Material Database (FMD) (Sharan et al, 2014) provides each 100 images for ten common material classes. It was used alongside the Describable Textures Dataset (Cimpoi et al, 2014) contained in BRODEN (Bau et al, 2017), and the Google-512 dataset (Schauerte, 2010) for evaluation in the SeVec concept localization method paper Gu and Tresp (2019).
Source: https://people.csail.mit.edu/lavanya/fmd.html

Google-512. The Google-512 dataset (Schauerte, 2010) consists of 512 sample object images for each of 11 basic color terms, collected using the Google search engine. It was used in Gu and Tresp (2019) for color concept localization.
Source: https://cvhci.anthropomatik.kit.edu/~bschauer/datasets/google-512/

6.3.2 Simple artificial concept datasets

Finally, we collect some simple artificial image datasets that allow for first evaluations and comparisons of concept models.

dSprites. The dSprites dataset (Matthey et al, 2017) is a popular simple artificial dataset for evaluation of latent space disentanglement methods. It consists of images in which a geometric shape is pasted onto a black canvas, with the following varying (discretized) latent factors: shape (3), scale (6), orientation (40), and position (each 32 values for x- and y-position). The dataset contains images of all 737 280 possible combinations of the latent factors, in low resolution of 64×64 pixels. In Kazhdan et al (2020) the dataset was considered for evaluation of concept localization and CBM methods.
Source: https://github.com/deepmind/dsprites-dataset/

3dshapes. Similar to dSprites, 3dshapes (DeepMind, 2021) is an artificial dataset generated via varying a fixed set of latent factors, originally used and intended for unsupervised disentanglement research (Kim and Mnih, 2018). The 480 000 samples of size 64×64 pixels each depict a 3D geometric shape centered within a rectangular room, with varying floor, wall, and object color (each 10 hue values), as well as object scale (8), shape (4), and camera orientation (15 values). The dataset was used in Kazhdan et al (2021) for comparison of CA methods with disentanglement methods.
Source: https://github.com/deepmind/3d-shapes

SCDB. The synthetic Simple Concept DataBase (SCDB) for binary image classification was presented in Lucieri et al (2020) in order to mimic challenges in skin lesion classification using dermatoscopic images. It contains 6000 samples for concept training with circular binary segmentation masks of 10 shape concepts. Each image contains one large geometric shape placed on black background, which contains and is surrounded by smaller geometric shapes which represent the labeled concepts. Class labels of the two classes are determined by simple disjunctive predicate rules on co-appearance of small shapes, e.g., $(\mathsf{hexagon} \land \mathsf{star}) \lor (\mathsf{ellipse} \land \mathsf{star}) \lor (\mathsf{triangle} \land \mathsf{ellipse} \land \mathsf{star})$. The position, rotation, color, count, as well as appearance of two task unrelated small shapes

are randomized.
Source: https://github.com/adriano-lucieri/SCDB

7 Challenges and research directions

The previous chapters provided a broad overview of investigated methods, applications, and datasets. On this basis, this section gives some possible further research directions to enrich the field of concept analysis, and foster its practical application.

Method combinations

As can be seen from Tables 1–4, not all combinations of properties for concept localization are fully leveraged so far. Further non-linear models like clustering may be promising for supervised concept analysis, and unsupervised concept analysis methods are still scarce, despite their value for qualitative explanations. Also, further investigation of mining segmentation or detection concept models may be valuable. And finally, further investigation in detection approaches may be promising to enable less texture-focused localization: Other than segmentation, they use more concept information (more than one activation map pixel).

Linear models and clustering

Concept localization using linear models suffers from inherent *instability*: The hyperplane defining the concept model strongly depends on the support points. A solution may be ensembling of models (Rabold et al, 2020). Furthermore, Goyal et al (2019) showed that linear models like TCAV (Kim et al, 2018) are mostly *overconfident* and may require additional confidence calibration. For concept mining approaches, a big challenge is to find *suitable concept candidates*. The mined concepts should both be relevant to the model functioning, and meaningful to humans. The current superpixeling approaches used for clustering approaches may not capture all types of concepts, and in general need not to be well align with human intuition on a concept. Moreover, other methods like matrix factorization cannot guarantee meaningfulness of the obtained concepts (Zhang et al, 2021). Hence, for concept mining, further priors and constraints could help to improve meaningfulness for humans. To give an example, one could use tracking information from temporal sequences to find independently moving sub-parts of objects.

Completeness and minimality

When using concepts to explain the inner workings of a model as in Rabold et al (2020), it is desirable to have a set of concepts that is *both minimal and complete*. It is unclear how to achieve completeness in supervised setting (what concepts are still missing?). And minimality is likewise hard to achieve because solutions for picking a generating set of concepts are not unique with both concepts and their CAVs not being independent.

Other domains and architectures

Finally, it would be an interesting challenge to apply concept analysis to other domains and to other DNN architectures than standard single-frame visual tasks. Examples may be video processing, audio or text processing with transformers. Concepts for text and audio might be emotions, single words or typical sentence constructions. Also, the networks analyzed are usually small and far from sizes needed for some important applications like automated driving perception. Only two of the presented methods consider complex tasks like object detection: CSPP (Feifel et al, 2021) and Schwalbe (2021), the latter uncovering severe performance issues for extension to larger models in the baseline method Net2Vec (Fong and Vedaldi, 2018).

Assessment applications

As shown in Schwalbe (2021), concept analysis is suitable for diverse verification methods applicable in domains where responsible AI, and thus thorough functional assessment, is relevant. It promises to both provide semantic alignment of DNN intermediate outputs and quantitative measures that allow to define local and global performance indicators, easing automatic assessments. More practical evaluations could be a great guideline to improve upon the suggested metrics, find meaningful reference values, and finally fill gaps in current DNN assessment recommendations.

Availability of data

A challenge common to all types of supervised concept analysis: Richly and densely annotated data is required like in the combined Broden dataset from Bau et al (2017), which means high labeling effort.

8 Conclusion

After a rapid development over the last years, concept embedding analysis has matured to an interesting sub-field of explainable artificial intelligence. This survey has established a common definition of efforts and related terms. Specifically, CA is roughly summarized as associating semantic, human interpretable concepts to intermediate output representations of deep neural networks (DNN), by means of simple helper concept models. To allow comparison of related approaches, a taxonomy is suggested that, amongst others, differentiates the types, inputs, outputs, and supervision of concept models.

Using this scheme, more than 30 divers CA methods are reviewed, categorized, and compared, providing a broad overview of approaches for CA. This should provide a good starting point for researchers seeking to position their CA related methods, or looking for ones that suit their use-case. Applications and use-cases are also gathered and discussed in detail. The most often encountered one is model distillation using concept outputs. Nevertheless, other interesting applications are found and reviewed, like intervention of concept outputs for interactive human-machine-systems, and verification relying either on the additional concept outputs, qualitative assessments, or metrics arising from CA results. An extensive list of metrics occurring in literature is compiled that allows to easily find appropriate measures for DNN assessment using CA. For the practical entrance to the topic, a broad collection of image datasets with concept labels useful for evaluation of CA methods is provided. Datasets are classified according to their use-cases in CA research. Relevant statistics and traits are summarized, showing that many helpful resources are currently available.

Altogether, this review should give researchers interested in applications or methods for semantic DNN latent space assessment a good starting point, and clearly establish the XAI sub-field of concept embedding analysis. We look forward to more interesting results in the field, especially with respect to the identified open research challenges: more datasets; improvement of the linear and unsupervised clustering approaches; selecting a good set of concepts for explaining a DNN (especially one that is both minimal and complete); and leveraging CA in further practical and complex use-cases and applications.

Declarations

Funding

Funding. The research leading to these results was partly funded by by the German Federal Ministry for Economic Affairs and Energy within the projects “KI Wissen – Automotive AI powered by Knowledge” and “KI Absicherung – Safe AI for automated driving”. Thanks to the consortia for the successful cooperation.

Employment. This work was authored in the course of doctoral research in an employment at Continental Automotive GmbH, Germany.

Competing interests

Non-financial interests. This doctoral research was supervised by Prof. Dr. Ute Schmid at the University of Bamberg.

Financial interests. The author declares that they have no financial interests.

Ethics approval

Not applicable.

Consent to participate

Not applicable.

Availability of data and materials

Not applicable.

Code availability

Not applicable.

Authors' Contribution Statement

No contributors were involved other than the single author.

References

Abid A, Zou J (2021) Meaningfully explaining a model's mistakes. ArXiv210612723 Cs URL http://arxiv.org/abs/2106.12723

Adadi A, Berrada M (2018) Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). In: IEEE Access, pp 52,138–52,160, https://doi.org/10.1109/ACCESS.2018.2870052

Arrieta AB, Rodríguez ND, Ser JD, et al (2020) Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Inf Fusion 58:82–115. https://doi.org/10.1016/j.inffus.2019.12.012

Bach S, Binder A, Montavon G, et al (2015) On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLOS ONE 10(7):e0130,140. https://doi.org/10.1371/journal.pone.0130140

Baehrens D, Schroeter T, Harmeling S, et al (2010) How to explain individual classification decisions. J Mach Learn Res 11:1803–1831. URL http://portal.acm.org/citation.cfm?id=1859912

Bau D, Zhou B, Khosla A, et al (2017) Network dissection: Quantifying interpretability of deep visual representations. In: Proc. 2017 IEEE Conf. Comput. Vision and Pattern Recognition. IEEE Computer Society, pp 3319–3327, https://doi.org/10.1109/CVPR.2017.354

Bau D, Zhu JY, Strobelt H, et al (2018) GAN dissection: Visualizing and understanding generative adversarial networks. In: Posters 2021 Int. Conf. Learning Representations, URL https://openreview.net/forum?id=Hyg_X2C5FX

Belém C, Balayan V, Saleiro P, et al (2021) Weakly supervised multi-task learning for concept-based explainability. In: Proc. ICLR 2021 Workshop Weakly Supervised Learning, vol abs/2104.12459. CoRR, URL https://arxiv.org/abs/2104.12459

Bell S, Bala K, Snavely N (2014) Intrinsic images in the wild. ACM Trans Graph 33(4):159:1–159:12. https://doi.org/10.1145/2601097.2601206 for Biological and Computational Learning (CBCL) MC (2021) MIT-CBCL database. URL http://cbcl.mit.edu/software-datasets/FaceData2.html

Carvalho DV, Pereira EM, Cardoso JS (2019) Machine learning interpretability: A survey on methods and metrics. Electronics 8(8):832. https://doi.org/10.3390/electronics8080832, URL https://www.mdpi.com/2079-9292/8/8/832

Chen C, Li O, Tao D, et al (2019a) This looks like that: Deep learning for interpretable image recognition. In: Advances in Neural Information Processing Systems 32, pp 8928–8939, URL https://proceedings.neurips.cc/paper/2019/hash/adf7ee2dcf142b0e11888e72b43fcb75-Abstract.html

Chen R, Chen H, Huang G, et al (2019b) Explaining neural networks semantically and quantitatively. In: Proc. 2019 IEEE/CVF Int. Conf. Computer Vision. IEEE, pp 9186–9195, https://doi.org/10.1109/ICCV.2019.00928

Chen X, Mottaghi R, Liu X, et al (2014) Detect what you can: Detecting and representing objects using holistic models and body parts. In: Proc. IEEE Conf. Comput. Vision and Pattern Recognition. IEEE Computer Society, pp 1979–1986, https://doi.org/10.1109/CVPR.2014.254

Chen Z, Bei Y, Rudin C (2020) Concept whitening for interpretable image recognition. CoRR abs/2002.01650. URL https://arxiv.org/abs/2002.01650

Chyung C, Tsang M, Liu Y (2019) Extracting interpretable concept-based decision trees from CNNs. In: Proc. 2019 ICML Workshop Human in the Loop Learning, vol 1906.04664. CoRR, URL http://arxiv.org/abs/1906.04664

Cimpoi M, Maji S, Kokkinos I, et al (2014) Describing textures in the wild. In: Proc. 2014 IEEE Conf. Comput. Vision and Pattern Recognition, pp 3606–3613, https://doi.org/10.1109/CVPR.2014.461

DARPA (2016) Explainable artificial intelligence. URL https://www.darpa.mil/program/explainable-artificial-intelligence

DeepMind (2021) 3dshapes - 3d shapes dataset. URL https://github.com/deepmind/3d-shapes

Deng J, Dong W, Socher R, et al (2009) ImageNet: A large-scale hierarchical image database. In: Proc. 2009 IEEE Conf. Comput. Vision and Pattern Recognition. IEEE, pp 248–255, https://doi.org/10.1109/CVPR.2009.5206848

Esser P, Rombach R, Ommer B (2020) A disentangling invertible interpretation network for explaining latent representations. In: Proc. 2020 IEEE Conf. Comput. Vision and Pattern Recognition. IEEE, pp 9220–9229, https:

//doi.org/10.1109/CVPR42600.2020.00924

Feifel P, Bonarens F, Koster F (2021) Reevaluating the safety impact of inherent interpretability on deep neural networks for pedestrian detection. In: Proc. IEEE/CVF Conf. Comput. Vision and Pattern Recognition, pp 29–37, https://doi.org/10.1109/CVPRW53098.2021.00012

Fellbaum C (ed) (1998) WordNet: An Electronic Lexical Database. Language, Speech, and Communication, A Bradford Book

Fong R, Vedaldi A (2018) Net2Vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. In: Proc. 2018 IEEE Conf. Comput. Vision and Pattern Recognition. IEEE Computer Society, pp 8730–8738, https://doi.org/10.1109/CVPR.2018.00910

Fong RC, Vedaldi A (2017) Interpretable explanations of black boxes by meaningful perturbation. In: Proc. 2017 IEEE Int. Conf. Comput. Vision. IEEE Computer Society, pp 3449–3457, https://doi.org/10.1109/ICCV.2017.371

Fuchs FB, Groth O, Kosiorek AR, et al (2018) Neural Stethoscopes: Unifying analytic, auxiliary and adversarial network probing. CoRR abs/1806.05502. URL http://arxiv.org/abs/1806.05502

Ge Y, Xiao Y, Xu Z, et al (2021) A peek into the reasoning of neural networks: Interpreting with structural visual concepts. In: Proc. 2021 IEEE/CVF Conf. Comput. Vision and Pattern Recognition, pp 2195–2204

Ghorbani A, Wexler J, Zou JY, et al (2019) Towards automatic concept-based explanations. In: Advances in Neural Information Processing Systems 32, pp 9273–9282, URL http://papers.nips.cc/paper/9126-towards-automatic-concept-based-explanations

Gilpin LH, Bau D, Yuan BZ, et al (2018) Explaining explanations: An overview of interpretability of machine learning. In: Proc. 5th IEEE Int. Conf. Data Science and Advanced Analytics. IEEE, pp 80–89, https://doi.org/10.1109/DSAA.2018.00018

Goodman B, Flaxman S (2017) European union regulations on algorithmic decision-making and a "right to explanation". AIMag 38(3):50–57. https://doi.org/10.1609/aimag.v38i3.2741

Gourier N, Hall D, Crowley JL (2004) Estimating face orientation from robust detection of salient facial structures. In: Proc. FG Net Workshop Visual Observation of Deictic Gestures, URL http://crowley-coutaz.fr/FGnet/reports/Pointing04-Proceedings.pdf

Goyal Y, Shalit U, Kim B (2019) Explaining classifiers with causal concept effect (CaCE). CoRR abs/1907.07165. URL http://arxiv.org/abs/1907.07165

Graziani M, Andrearczyk V, Müller H (2018) Regression concept vectors for bidirectional explanations in histopathology. In: Understanding and Interpreting Machine Learning in Medical Image Computing Applications. Springer International Publishing, Lecture Notes in Computer Science, pp 124–132, https://doi.org/10.1007/978-3-030-02628-8_14

Graziani M, Andrearczyk V, Marchand-Maillet S, et al (2020) Concept attribution: Explaining CNN decisions to physicians. Computers in Biology and Medicine 123:103,865. https://doi.org/10.1016/j.compbiomed.2020.103865

Gretton A, Borgwardt KM, Rasch MJ, et al (2012) A kernel two-sample test. J Mach Learn Res 13:723–773

Gu J, Tresp V (2019) Semantics for global and local interpretation of deep neural networks. CoRR abs/1910.09085. URL http://arxiv.org/abs/1910.09085

Hailesilassie T (2016) Rule extraction algorithm for deep neural networks: A review. CoRR abs/1610.05267. URL http://arxiv.org/abs/1610.05267

Haselhoff A, Kronenberger J, Kuppers F, et al (2021) Towards black-box explainability with Gaussian discriminant knowledge distillation. In: Proc. IEEE/CVF Conf. Comput. Vision and Pattern Recognition. Computer Vision Foundation / IEEE, pp 21–28, https://doi.org/10.1109/CVPRW53098.2021.00011

He K, Zhang X, Ren S, et al (2016) Deep residual learning for image recognition. In: Proc. 2016 IEEE Conf. Comput. Vision and Pattern Recognition, pp 770–778, https://doi.org/10.1109/CVPR.2016.90

Heuillet A, Couthouis F, Díaz-Rodríguez N (2021) Explainability in deep reinforcement learning. Knowledge-Based Systems 214:106,685. https://doi.org/10.1016/j.knosys.2020.106685

Higgins I, Matthey L, Pal A, et al (2016) Beta-VAE: Learning basic visual concepts with a constrained variational framework. In: Posters 5th Int. Conf. Learning Representations, URL https://openreview.net/forum?id=Sy2fzU9gl

Huang GB, Mattar M, Berg T, et al (2008) Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In: Proc. Workshop Faces in 'Real-Life' Images: Detection, Alignment, and Recognition, URL https://hal.inria.fr/inria-00321923

Initiative TO (2021) The osteoarthritis initiative. URL https://nda.nih.gov/oai

ISO/TC 22/SC 32 (2018) ISO 26262-1:2018(En): Road Vehicles — Functional Safety — Part 1: Vocabulary, ISO 26262:2018(En), vol 1, 2nd edn. International Organization for Standardization, URL https://www.iso.org/standard/68383.html

Janizek JD, Sturmfels P, Lee SI (2020) Explaining explanations: Axiomatic feature interactions for deep networks. CoRR abs/2002.04138. URL https://arxiv.org/abs/2002.04138

Kazhdan D, Dimanov B, Jamnik M, et al (2020) Now you see me (CME): Concept-based model extraction. In: Proc. 29th ACM Int. Conf. Information and Knowledge Management Workshops, CEUR Workshop Proceedings, vol 2699. CEUR-WS.org, URL http://ceur-ws.org/Vol-2699/paper02.pdf

Kazhdan D, Dimanov B, Terre HA, et al (2021) Is disentanglement all you need? comparing concept-based & disentanglement approaches. ArXiv210406917 Cs URL http://arxiv.org/abs/2104.06917

Khan K, Mauro M, Leonardi R (2015) Multi-class semantic segmentation of faces. In: Proc. 2015 IEEE Int. Conf. Image Processing. IEEE, pp 827–831, https://doi.org/10.1109/ICIP.2015.7350915

Kim B, Wattenberg M, Gilmer J, et al (2018) Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In: Proc. 35th Int. Conf. Machine Learning, Proc. Machine Learning Research, vol 80. PMLR, pp 2668–2677, URL http://proceedings.mlr.press/v80/kim18d.html

Kim H, Mnih A (2018) Disentangling by factorising. In: Proc. 2018 Int. Conf. Machine Learning. PMLR, pp 2649–2658, URL https://proceedings.mlr.press/v80/kim18b.html

Kingma DP, Welling M (2014) Auto-encoding variational Bayes. In: Proc. 2nd Int. Conf. Learning Representations, URL http://arxiv.org/abs/1312.6114

Koh PW, Nguyen T, Tang YS, et al (2020) Concept bottleneck models. In: Proc. 2020 Int. Conf. Machine Learning. PMLR, pp 5338–5348, URL http://proceedings.mlr.press/v119/koh20a.html

Kronenberger J, Haselhoff A (2019) Dependency decomposition and a reject option for explainable models. In: Proc. 2019 IEEE Conf. Comput. Vision and Pattern Recognition Workshops, URL http://arxiv.org/abs/2012.06523

Lin TY, Maire M, Belongie SJ, et al (2014) Microsoft COCO: Common objects in context. In: Proc. 13th European Conf. Computer Vision - Part V, Lecture Notes in Computer Science, vol 8693. Springer International Publishing, pp 740–755, https://doi.org/10.1007/978-3-319-10602-1_48

Linardatos P, Papastefanopoulos V, Kotsiantis S (2021) Explainable AI: A review of machine learning interpretability methods. Entropy 23(1):18. https://doi.org/10.3390/e23010018

Liu Z, Luo P, Wang X, et al (2015) Deep learning face attributes in the wild. In: Proc. 2015 IEEE Int. Conf. Computer Vision, pp 3730–3738, https://doi.org/10.1109/ICCV.2015.425

Losch M, Fritz M, Schiele B (2019) Interpretability beyond classification output: Semantic bottleneck networks. In: Proc. 3rd ACM Computer Science in Cars Symp. Extended Abstracts, URL https://arxiv.org/pdf/1907.10882.pdf

Lucieri A, Bajwa MN, Dengel A, et al (2020) Explaining AI-based decision support systems using concept localization maps. In: Neural Information Processing. Springer International Publishing, Communications in Comput. and Information Science, pp 185–193, https://doi.org/10.1007/978-3-030-63820-7_21

Mahinpei A, Clark J, Lage I, et al (2021) Promises and pitfalls of black-box concept learning models. ArXiv210613314 Cs URL http://arxiv.org/abs/2106.13314

Marcos D, Fong R, Lobry S, et al (2020) Contextual semantic interpretability. In: Proc. 15th Asian Conf. Comput. Vision Revised Selected Papers, Part IV, Lecture Notes in Computer Science, vol 12625. Springer, pp 351–368, https://doi.org/10.1007/978-3-030-69538-5_22

Matthey L, Higgins I, Hassabis D, et al (2017) dSprites: Disentanglement testing Sprites dataset. URL https://github.com/deepmind/dsprites-dataset/

Mikolov T, Yih Wt, Zweig G (2013) Linguistic regularities in continuous space word representations. In: Proc. 2013 Conf. North American Chapter Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, pp 746–751, URL https://www.aclweb.org/anthology/N13-1090

Molnar C (2020) Interpretable Machine Learning. Lulu.com, URL https://christophm.github.io/interpretable-ml-book/

Mordvintsev A, Olah C, Tyka M (2015) Inceptionism: Going deeper into neural networks. URL http://ai.googleblog.com/2015/06/

inceptionism-going-deeper-into-neural.html

Mottaghi R, Chen X, Liu X, et al (2014) The role of context for object detection and semantic segmentation in the wild. In: Proc. 2014 IEEE Conf. Comput. Vision and Pattern Recognition, pp 891–898, https://doi.org/10.1109/CVPR.2014.119

Nguyen A, Yosinski J, Clune J (2019) Understanding neural networks via feature visualization: A survey. In: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. Lecture Notes in Computer Science, Springer International Publishing, p 55–76, https://doi.org/10.1007/978-3-030-28954-6_4

Olah C, Mordvintsev A, Schubert L (2017) Feature visualization. Distill 2(11):e7. https://doi.org/10.23915/distill.00007

Pfau J, Young AT, Wei J, et al (2021) Robust semantic interpretability: Revisiting concept activation vectors. In: Proc. 2021 ICML Workshop Human Interpretability in Machine Learning. CoRR, URL http://arxiv.org/abs/2104.02768

Rabold J, Schwalbe G, Schmid U (2020) Expressive explanations of DNNs by combining concept analysis with ILP. In: KI 2020: Advances in Artificial Intelligence. Springer International Publishing, Lecture Notes in Computer Science, pp 148–162, https://doi.org/10.1007/978-3-030-58285-2_11

Ribeiro MT, Singh S, Guestrin C (2016) ”Why should I trust you?”: Explaining the predictions of any classifier. In: Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining. ACM, KDD ’16, pp 1135–1144

Roychowdhury S, Diligenti M, Gori M (2018) Image classification using deep learning and prior knowledge. In: Workshops of the 32nd AAAI Conf. Artificial Intelligence, AAAI Workshops, vol WS-18. AAAI Press, pp 336–343, URL https://aaai.org/ocs/index.php/WS/AAAIW18/paper/view/16575

Saini US, Papalexakis EE (2020) Analyzing representations inside convolutional neural networks. ArXiv201212516 Cs URL http://arxiv.org/abs/2012.12516

Schauerte B (2010) Google-512, color term learning data set. URL https://cvhci.anthropomatik.kit.edu/~bschauer/datasets/google-512/

Schwalbe G (2021) Verification of size invariance in DNN activations using concept embeddings. In: Artificial Intelligence Applications and Innovations. Springer International Publishing, IFIP Advances in Information and Communication Technology, pp 374–386, https://doi.org/10.1007/978-3-030-79150-6_30

Schwalbe G, Finzel B (2021) XAI Method Properties: A (Meta-)study. CoRR abs/2105.07190. URL http://arxiv.org/abs/2105.07190

Schwalbe G, Schels M (2020) Concept enforcement and modularization as methods for the ISO 26262 safety argumentation of neural networks. In: Proc. 10th European Congress Embedded Real Time Software and Systems, URL https://hal.archives-ouvertes.fr/hal-02442796

Selvaraju RR, Cogswell M, Das A, et al (2017) Grad-CAM: Visual explanations from deep networks via gradient-based localization. In: Proc. 2017 IEEE Int. Conf. Computer Vision. IEEE, pp 618–626, https://doi.org/10.1109/ICCV.2017.74

Sharan L, Rosenholtz R, Adelson EH (2014) Accuracy and speed of material categorization in real-world images. Journal of Vision 14(9):12. https://doi.org/10.1167/14.9.12

Stallkamp J, Schlipsing M, Salmen J, et al (2011) The german traffic sign recognition benchmark: A multi-class classification competition. In: Proc. 2011 Int. Joint Conf. Neural Networks. IEEE, Y, pp 1453–1460, https://doi.org/10.1109/IJCNN.2011.6033395

Sundararajan M, Taly A, Yan Q (2017) Axiomatic attribution for deep networks. In: Proc. 34th Int. Conf. Machine Learning, Proc. Machine Learning Research, vol 70. PMLR, pp 3319–3328, URL http://proceedings.mlr.press/v70/sundararajan17a.html

Thomaz CE, Giraldi GA (2010) A new ranking method for principal components analysis and its application to face image analysis. Image and Vision Computing 28(6):902–913. https://doi.org/10.1016/j.imavis.2009.11.005

Tjoa E, Guan C (2020) A survey on explainable artificial intelligence (XAI): Toward medical XAI. IEEE Trans Neural Netw Learn Syst pp 1–21. https://doi.org/10.1109/TNNLS.2020.3027314

Vilone G, Longo L (2020) Explainable artificial intelligence: A systematic review. CoRR URL http://arxiv.org/abs/2006.00093

Wah C, Branson S, Welinder P, et al (2011) The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, URL http://www.vision.caltech.edu/visipedia/papers/CUB_200_2011.pdf

Wan A, Dunlap L, Ho D, et al (2020) NBDT: Neural-backed decision tree. In: Posters 2021 Int. Conf. Learning Representations, URL https://openreview.net/forum?id=mCLVeEpplNE

Wang D, Cui X, Wang ZJ (2020) CHAIN: Concept-harmonized hierarchical inference interpretation of deep convolutional neural networks. CoRR abs/2002.01660. URL https://arxiv.org/abs/2002.01660

Wu W, Su Y, Chen X, et al (2020) Towards global explanations of convolutional neural networks with concept attribution. In: Proc. 2020 IEEE/CVF Conf. Comput. Vision and Pattern Recognition, pp 8649–8658, https://doi.org/10.1109/CVPR42600.2020.00868

Xiao T, Liu Y, Zhou B, et al (2018) Unified perceptual parsing for scene understanding. In: Proc. 15th European Conf. Comput. Vision, Part V, Lecture Notes in Computer Science, vol 11209. Springer International Publishing, pp 432–448, https://doi.org/10.1007/978-3-030-01228-1_26

Yeh CK, Kim B, Arik S, et al (2020) On completeness-aware concept-based explanations in deep neural networks. In: Advances in Neural Information Processing Systems 33, pp 20,554–20,565, URL https://proceedings.neurips.cc/paper/2020/hash/ecb287ff763c169694f682af52c1f309-Abstract.html

Zhang Q, Zhu SC (2018) Visual interpretability for deep learning: A survey. Front IT EE 19(1):27–39. https://doi.org/10.1631/FITEE.1700808

Zhang Q, Cao R, Shi F, et al (2018a) Interpreting CNN knowledge via an explanatory graph. In: Proc. 32nd AAAI Conf. Artificial Intelligence. AAAI Press, pp 4454–4463, URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17354

Zhang Q, Wang W, Zhu SC (2018b) Examining CNN representations with respect to dataset bias. In: Proc. 32nd AAAI Conf. Artificial Intelligence. AAAI Press, pp 4464–4473, URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17429

Zhang Q, Wu YN, Zhu S (2018c) Interpretable convolutional neural networks. In: Proc. 2018 IEEE/CVF Conf. Comput. Vision and Pattern Recognition. IEEE, pp 8827–8836, https://doi.org/10.1109/CVPR.2018.00920

Zhang R, Madumal P, Miller T, et al (2021) Invertible concept-based explanations for CNN models with non-negative concept activation vectors. In: Proc. 35th AAAI Conf. Artificial Intelligence, vol 35. AAAI Press, pp 11,682–11,690, URL https://ojs.aaai.org/index.php/AAAI/article/view/17389

Zhou B, Lapedriza A, Xiao J, et al (2014) Learning deep features for scene recognition using places database. In: Advances in Neural Information Processing Systems, vol 27. Curran Associates, Inc., URL https://papers.nips.cc/paper/2014/hash/3fe94a002317b5f9259f82690aeea4cd-Abstract.html

Zhou B, Khosla A, Lapedriza À, et al (2016) Learning deep features for discriminative localization. In: Proc. 2016 IEEE Conf. Comput. Vision and Pattern Recognition. IEEE Computer Society, pp 2921–2929, https://doi.org/10.1109/CVPR.2016.319

Zhou B, Zhao H, Puig X, et al (2017) Scene parsing through ADE20K dataset. In: Proc. 2017 IEEE Conf. Comput. Vision and Pattern Recognition, pp 5122–5130, https://doi.org/10.1109/CVPR.2017.544

Zhou B, Sun Y, Bau D, et al (2018) Interpretable basis decomposition for visual explanation. In: Computer Vision – ECCV 2018. Springer International Publishing, Lecture Notes in Computer Science, pp 122–138, https://doi.org/10.1007/978-3-030-01237-3_8

Zhou J, Gandomi AH, Chen F, et al (2021) Evaluating the quality of machine learning explanations: A survey on methods and metrics. Electronics 10(5):593. https://doi.org/10.3390/electronics10050593

Verification of Size Invariance in DNN Activations Using Concept Embeddings

Gesina Schwalbe[1,2](✉)

[1] Continental AG, Regensburg, Germany
gesina.schwalbe@continental-corporation.com
[2] Cognitive Systems, University of Bamberg, Bamberg, Germany

Abstract. The benefits of deep neural networks (DNNs) have become of interest for safety critical applications like medical ones or automated driving. Here, however, quantitative insights into the DNN inner representations are mandatory [10]. One approach to this is *concept analysis*, which aims to establish a mapping between the internal representation of a DNN and intuitive semantic concepts. Such can be sub-objects like human body parts that are valuable for validation of pedestrian detection. To our knowledge, concept analysis has not yet been applied to large object detectors, specifically not for sub-parts. Therefore, this work first suggests a substantially improved version of the Net2Vec approach [5] for post-hoc segmentation of sub-objects. Its practical applicability is then demonstrated on a new concept dataset by two exemplary assessments of three standard networks, including the larger Mask R-CNN model [9]: (1) the consistency of body part similarity, and (2) the invariance of internal representations of body parts with respect to the size in pixels of the depicted person. The findings show that the representation of body parts is mostly size invariant, which may suggest an early intelligent fusion of information in different size categories.

Keywords: Concept embedding analysis · MS COCO · Explainable AI

1 Introduction

The high performance and flexibility of deep neural networks (DNNs) makes them interesting for many complex computer vision applications. This includes ethically involved or safety critical fields where silent biases can become a matter of unfairness, or even life-threatening decisions. Hence, responsible artificial intelligence is required, allowing for thorough assessments of trained DNNs [1]. For example in the area of automated driving, quantitative insights into the inner representation of trained DNNs are mandatory [10].

The research leading to these results was partly funded by the German Federal Ministry for Economic Affairs and Energy within the project "KI-Absicherung".

Published by Springer Nature Switzerland AG 2021
I. Maglogiannis et al. (Eds.): AIAI 2021, IFIP AICT 627, pp. 374–386, 2021.
https://doi.org/10.1007/978-3-030-79150-6_30

One step towards this direction is the research area of *concept (embedding) analysis* [5,11], that started with the work in [2] in 2017. Supervised concept analysis aims to answer the question how (well) information on a visual semantic concept is embedded in the intermediate output of one layer of a trained DNN. A concept can be, e.g., a texture, material, scene, object, or object part [2], and should be given as samples with binary annotations. The question is answered by training a simple model to predict the concept from the layer output, with simplicity ensuring interpretability [2,11]. Once these *concept models* are obtained, several means for verification open up: The embedding quality can be measured as the prediction quality of the concept model; if a similarity measure for concept models is available, semantic relations between concepts can be validated (e.g., legs are similar to arms) [5]; and lastly, if the information embedding can be represented as a vector in the latent space, one can use sensitivity analysis to investigate dependencies of outputs on certain concepts [11]. The vector can be the normal vector of a linear model, or the center of a cluster, and is called the embedding of the concept [5].

While concept analysis opens up a wide variety of verification options, existing methods suffer from some limitations. For one, the methods are either restricted to image-level concepts, or, like Net2Vec [5], report poor results for part objects. But part objects, such as human body parts, are substantial to verify logical plausibility of pedestrian detectors needed for automated driving. This brings up the second issue: To our knowledge, existing proposals have not been evaluated on large networks like standard object detectors. Concretely, we uncovered some severe performance limitations of Net2Vec that impede application to state-of-the-art sized DNNs or larger concept datasets than the original Broden dataset proposed in [2].

In order to overcome the practical issues, this paper suggests some substantial improvements to the Net2Vec method, demonstrated on a new large MS COCO [15] based concept dataset for human body parts. The applicability of concept analysis for simple verification is demonstrated by exemplary assessment of three networks with different sizes, tasks, and training data (AlexNet, VGG16, and Mask R-CNN [9]). Concretely, the semantic similarity of different concepts is validated, and it is checked whether the internal representation of the networks is biased towards one size category (i.e., distance to the camera). Interestingly, our findings suggest that convolutional DNNs are not biased towards one size. Instead, from early layers onwards, they use an efficient common representation for instances of the same part object but different sizes. The main contributions of this paper are:

- A Net2Vec-based concept analysis approach that can deal with large models and large concept datasets is presented and demonstrated. It is found to considerably surpass the state-of-the-art for object part concepts.
- The first evaluation of size invariance of the internal representations of convolutional DNNs is conducted.
- For this, an approach is presented to estimate the pixel size of persons in 2D images from skeletal annotations, which can be used for concept mask generation.

After a revision of concept analysis methods in Sect. 2, our approach is introduced in Sect. 3. Finally, in Sect. 4 it is validated and used for the exemplary assessment.

2 Related Work

An early approach to supervised concept analysis was NetDissect by Bau et al. [2]. They associated single convolutional filters with semantic concepts, and provided the Broden dataset, a combined dataset featuring a wide variety of concepts. Building upon NetDissect, Net2Vec [5] expanded from single filters to combinations of filters and trains a 1 × 1 convolution to predict concept masks from latent space outputs. Intuitively, the weight vector of the convolution, the concept embedding vector, encodes the direction within the layer output space that adds the concept. The authors investigate the similarity of different concepts via cosine similarity of their concept vectors. Similarly, TCAV by Kim et al. [11] trains linear models but using a support vector machine on the complete latent space output for binary classification instead of segmentation. The authors suggest to use partial directional derivatives along the concept vectors to assess how outputs depend on concepts in intermediate layers. Future work may investigate how this can be applied to concept segmentation instead of concept classification. Other than the previous linear model approaches, SeVec [8] uses a k-means clustering approach, demonstrated on a dataset other than Broden, but again only on image-level concepts and with a manual step in the process. The Net2Vec-based proof of concept in [19] uses a small concept dataset for traffic sign letters. This work also details the value of concept analysis for safety assessment, and suggests that a mismatch of concept size and receptive field of concept models can cause a texture bias, that is, predictions purely based on texture. This is investigated for larger nets in this paper.

More relevant for inspection than requirements verification purposes, unsupervised concept analysis approaches aim to find and visualize repetitive concepts in the intermediate output of a DNN. A simple example is feature visualization [16], where noise is optimized to maximally activate a convolutional filter. ACE [7] instead uses super-pixels as concept candidates, which are then clustered by their latent space proximity. In the direction of representation disentanglement, [21] introduced a measure for the completeness of a set of concept vectors with respect a given task. Their idea is that post-hoc adding a semantic bottleneck with unit vectors aligned to the concepts should not decrease the model performance. Similarly, but with invertible DNNs instead of linear maps, the recent work in [4] tries to find a bijection of a latent space to a product of semantically aligned sub-spaces (and a residual space).

Experiments so far have been conducted on small resolution image datasets like the 224 × 224 pixels of the Broden [2,5,11] and ImageNet [8] data or less [19], and have not yet been applied to large models or object detectors.

3 Approach

This section introduces our concept analysis approach with its modifications to the Net2Vec concept analysis method, and an approach to estimate the body size in pixels for persons in 2D images from skeletal annotations. Size estimation is later used to generate ground truth segmentation masks for body part concepts, and to categorize them by size.

3.1 Concept Embedding Analysis

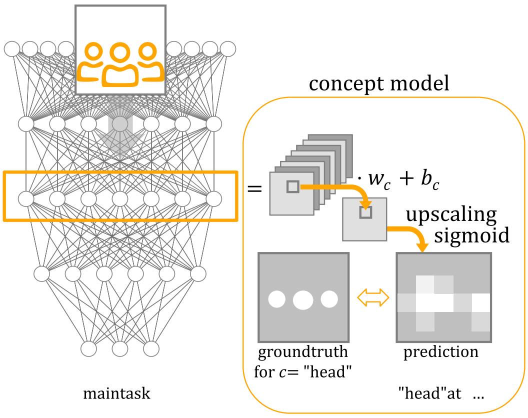


Fig. 1. Illustration of the used concept analysis approach. For a concept c, here "head", it predicts binary segmentation masks from convolutional activation maps via (1) a 1-filter convolution with kernel w_c and bias b_c, (2) bilinear upscaling (here depicted using nearest pixel upscaling), and finally (3) normalization.

Our method is illustrated in Fig. 1. As a foundation, the Net2Vec [5] approach was chosen. It allows to localize non-image-level concepts like object parts in convolutional activation maps by predicting binary segmentation masks. For this the spatial resolution of the convolutional layers is used: The mask predictor is a 1 × 1 convolution followed by bilinear upscaling, then sigmoid normalization (and, for binarizing, thresholding at 0.5). For comparability, the quality measure set intersection over union (set IoU) is adopted which divides the total area of intersections by the total area of unions between the binary ground truth masks M_i and the predicted segmentation masks M_i^{pr} (binarized at 0.5):

$$\text{set IoU}\left((M_i)_i, (M_i^{\mathrm{pr}})_i\right) = \frac{\sum_i \sum M_i \cap (M_i^{\mathrm{pr}} > 0.5)}{\sum_i \sum M_i \cup (M_i^{\mathrm{pr}} > 0.5)} .$$

We here took the mean of batch-wise set IoU values. It must be noted that the set IoU measure penalizes errors on small objects more than on large ones, and it suffers under low resolution of activation maps. Below, four further limitations of Net2Vec are collected and our countermeasures explained.

Most notably, Net2Vec uses denoising of activation maps with a ReLU suggested in [2]. The threshold is calculated to keep the 0.5% highest activations. This requires to load activations of the complete dataset into memory at once, which is infeasible both for large models and large datasets. Experiments showed that denoising makes no noticeable difference, hence it is skipped.

Next, to push the decision boundary of the model towards a value of 0.5, Net2Vec uses a binary cross-entropy (BCE) loss with per-class-weights where each weight is the mean proportion of pixels of the opposite class. Obtaining the weight is an expensive pre-processing step. Here we found that global weighting could be replaced either by calculating weights batch-wise, or using an unweighted Tversky loss [18]—also called Dice or F1 loss—that directly optimizes the F1 score of the segmentation per $h \times w$ image ("$\cdot$" pixel-wise):

$$\text{Dice loss}\,(M, M^{\text{pr}}) = 1 - \frac{2\sum M \cdot M^{\text{pr}}}{\sum M + \sum M^{\text{pr}}} \quad \text{for } M, M^{\text{pr}} \in [0,1]^{h \times w}.$$

Further, the original method reports average set IoU values lower than 0.05 for part objects, which we could confirm. This can be achieved by constantly predicting white masks (1 for all pixels). In a series of experiments we found settings fixing this: We use a Tversky loss, Adam optimizer [12] instead of stochastic gradient descent (SGD), with a batch size of 8 instead of 64, and learning rate of 10^{-3} rather than 10^{-4}.

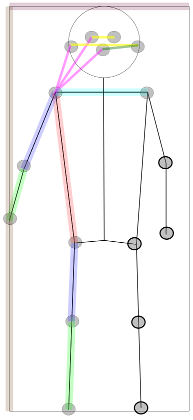
Another modification used in some experiments is *adaptive kernel size*. The original Net2Vec method uses a 1×1 kernel, so each prediction only has a spatial context of one pixel. As found in [19], this may be too small for larger concepts and layers with higher resolution, and possibly leads to a texture bias [6]. Following [19], this can be mitigated using a kernel size that covers an area of the expected size of the concept, assuming low size variance. Note that this may improve performance of a concept model, but destroys comparability amongst models of differently sized concepts due to incompatible kernel shapes.

3.2 Distance Category Estimation

Assume an image dataset is given that provides 2D skeletal information of persons (cf. Fig. 2), possibly bounding boxes, but no depth information. The goal of this approach is to estimate and categorize the body size of a person depicted in a 2D image in pixels, given only lengths between keypoints. The challenges are that keypoint information may be incomplete due to occlusion or cropping, and that 2D projected lengths may be inaccurately short. Thus, redundant calculation of body size from as many link types as possible is needed. One major drawback of this method is that assumptions on "standard" proportions must be made. This cannot be overcome without additional information. Despite that, it was found that the approach yields surprisingly good and intuitive results, with errors not infringing the quality of body size categorization.

The following sources were used to relate the true length l of (combinations of) keypoint links to the true body height h: arts for facial and head-to-body

Formulas for total height
$1 \cdot \texttt{bbox\_width/\_height}$
$1.1 \cdot \texttt{body\_height}$
$2.4 \cdot \texttt{hip\_to\_shoulder}$
$7 \cdot \texttt{head\_height}$
$1.485 \cdot \texttt{leg} \cdot \frac{h}{h-0.433}$
$2.77 \cdot \texttt{upper\_leg} \cdot \frac{h}{h-0.405}$
$3.075 \cdot \texttt{lower\_leg} \cdot \frac{h}{h-0.501}$
$3.72 \cdot \texttt{upper\_arm} \cdot \frac{h}{h-0.449}$
$4.46 \cdot \texttt{lower\_arm} \cdot \frac{h}{h-0.569}$



Formulas for derived lengths	
`body_height`	$\approx$ `leg`* + `hip_to_shoulder`* + `shoulder_to_eye`*/`_ear`*/`_nose` $\approx$ `arm + shoulder_width`
`leg`	$\approx$ `lower_leg + upper_leg`
`arm`	$\approx$ `lower_arm + upper_arm`
`head_height`	$\approx 1.1 \cdot \texttt{head\_width}$ $\approx \frac{8}{7}\,\texttt{head\_depth}$
`head_width`	$\approx$ `ear_to_opposite_ear` $\approx 2.5 \cdot \texttt{eye\_to\_eye}$
`head_depth`	$\approx 2 \cdot \texttt{ear\_to\_eye}$* $\approx \frac{7}{4}\,\texttt{ear\_to\_nose}$

* involved keypoints must be on the same side

Fig. 2. Used keypoints and links with formulas to estimate the total body height of a person from link lengths and assumed "standard" height h in meters.

proportions [14]; standard educational material relating the wrist-to-wrist span [3]; and linear models used in archeology to estimate the body size from single long bones [14]. For long bone relations, the mean model parameters were taken between genders. Relations involving approximate `body_height` and bounding box (`bbox`) dimensions were estimated manually. The resulting relation models are all of the linear nature $h = s \cdot l + c$ for some slope s, and offset c in meters. Given the downscaled length $l' = f \cdot l$ in pixels, the formula for the downscaled body size is $f \cdot h = l' \cdot \frac{h}{h-c}$. If c is non-zero, a "standard" real person size h must be assumed, which was set to 1.7 m following [14]. The final formulas are shown in Fig. 2. Whenever one value has several estimation results, the maximum was used to cope with possible length shortenings due to 2D projection.

The formulas were now leveraged to sort annotations into four size categories. These were chosen such that in each the minimum and maximum estimated size do not differ by more than a factor of two. Very small and very large persons were discarded. The resulting size categories with their range of relative sizes are: *far* in $[0.2, 0.38]$, *middle* in $[0.38, 0.71]$, *close* in $[0.71, 1.33]$, and *very close* in $[1.33, 2.5]$. Categories are depicted in Fig. 3.

4 Experiments

After detailing our experiment settings, this section will validate the used approach including hyperparameters (Sect. 4.1), then conduct exemplary assessments of embedding quality and semantics (Sect. 4.2), as well as size bias (Sect. 4.3).

For the experiments, we chose three networks of different size, training objective, and training dataset, with pretrained weights from the torchvision model zoo[1]: ImageNet-trained classifiers VGG16 [20] and AlexNet [13], and the object detector Mask R-CNN [9] trained on MS COCO.

[1] https://pytorch.org/vision/stable/models.html.

As layers we chose the activated convolutional output before each downsampling step except the first. In case of Mask R-CNN we considered the output of each residual grouping in the backbone and the feature pyramid. For concepts we selected a set of five large and small concepts that are common to the Broden and MS COCO datasets: leg, arm, foot and hand (for COCO approximated via ankle and wrist keypoints), and eye. Images were zero-padded to square size then resized to 400 × 400 respectively 224 × 224 (VGG16, AlexNet) pixels to avoid feature distortion. Ground truth masks on concepts for MS COCO images were generated by drawing links (leg, arm) respectively points (foot, hand, eye) of width 0.025 times the image height or times the body size if this can be estimated (see Fig. 5). Intermediate outputs of the DNNs were cached with bi-float 16 precision to speed up experiments, except for the very large Mask R-CNN feature pyramid blocks 0 and 1, and backbone layer 1. On cached layers, concept model training was done with 5-fold cross-validation. The original train-test splits were used.

4.1 Validation of the Proposed Methods

Size Distribution in MS COCO To validate the size estimation approach, an analysis of the size distribution on the popular MS COCO dataset was conducted. The size of an annotation cannot be estimated if it provides no keypoints (very small persons and crowds) or only disconnected keypoints. Nevertheless, size estimations could be made for a decent proportion of the annotations in both training and test set (ca. 46%), see Fig. 3. Also, for each chosen body part, a sufficient amount of annotations is available in size categories *far*, *middle*, and *close*: more than 10,000 in the train, and 300 in the test set. These categories were thus used for experiments. Training and test set were found to be similarly distributed, with significantly more keypoint annotations estimated far than close. This suggests, that the total amount of positive pixels for masks of small far body parts is similar to that of large close ones.

Improvement of Net2Vec. Settings for the concept analysis approach were needed that (1) work for large models and datasets (no expensive thresholds and weights), and (2) work for part objects. First, the necessity to threshold the activation maps was disproved: Skipping it had no effect on the test performance. Then, without thresholding, a series of experiments was conducted comparing different optimizers (SGD, Adam), batch sizes (bs), learning rates (lr), and losses (globally weighted BCE, batch-wise weighted BCE, and Tversky). Each setting was evaluated on AlexNet and VGG16, on all selected layers and concepts, using the original Broden dataset [2]. Firstly, the better optimizer was chosen, then the better batch-size and learning rate combination, and lastly losses were compared. The consistently best and stable setting proved to be Adam optimizer with lr 10^{-3}, bs 8, with maximum five epochs due to fast convergence. Find a comparison to the baseline for AlexNet in Fig. 3 (VGG16 similar).

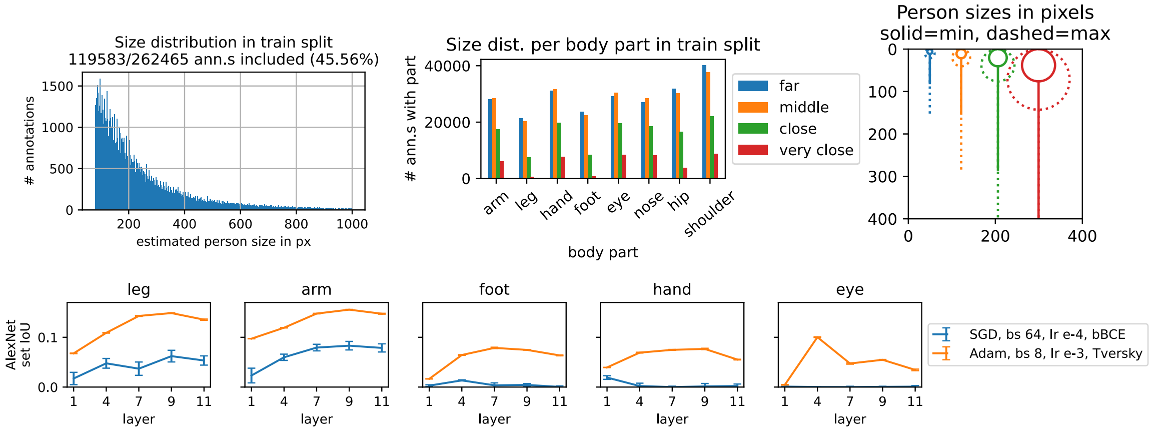


Fig. 3. *Top:* Distribution by estimated person size of all annotations (*left*) and per body part (*center*) for COCO train images (padded to square size and resized to 400 × 400 px); used person size categories are illustrated on the *right*. *Bottom:* Performance comparison of used settings with Net2Vec baseline (cf. [5, Fig. 2]). Best viewed in color. (Color figure online)

4.2 Embedding Quality and Similarity Validation

Two applications of concept analysis are to verify sufficient embedding qualities, and to use the similarity measure on the concept models for validation of semantic relations. For this, concept models for all chosen concepts, networks, and layers were trained on the complete MS COCO concept dataset. Performance results are depicted in Fig. 6 (size category *all*). As before, the variance was very low, evidencing good convergence of the linear concept models. Convincing embedding qualities were found except for the small concepts hand and foot in small networks. This may origin from the approximate and noisy ground truth, and the higher set IoU penalties for low resolution. In general, later layers with low resolution showed the best embedding results, indicating that body parts are relatively complex concepts. These may require larger network structures, since Mask R-CNN significantly surpassed AlexNet and VGG16. Some exemplary output for Mask R-CNN is shown in Fig. 6.

Next, the concept model similarities were measured as the mean cosine similarity between the normal vectors of the models, shown in Fig. 4. Cosine similarity $\frac{a \cdot b}{\|a\| \cdot \|b\|}$ of vectors $a, b \in \mathbb{R}^n$ is the cosine of the angle between the vectors, with 1 meaning 0° angle, and -1 meaning 180°. The results show intuitive relations between the concepts: No similarity values were below zero, so no body part concepts are mutually exclusive. And eye is most different from the other parts, while hand and arm are more closely related, similarly leg and foot. Interestingly, in the later low-resolution layers concepts were more dissimilar, so better distinguishable, explaining the better performance here.

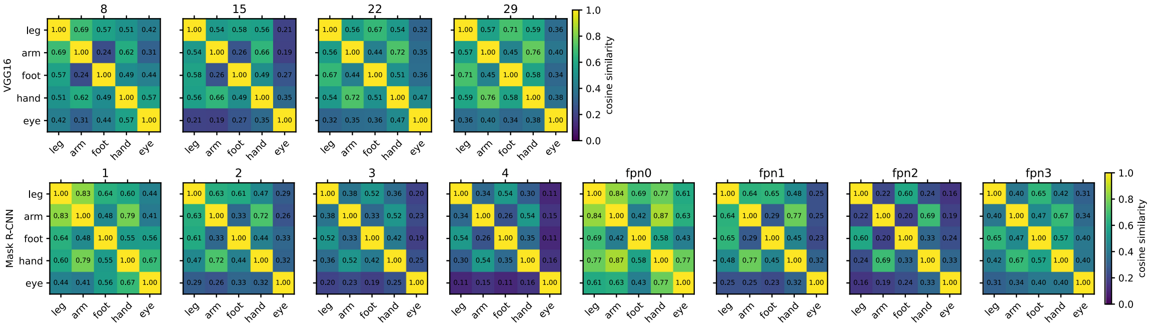


Fig. 4. Mean cosine similarities between normal vectors of concept models for different body parts, by layer. AlexNet results were similar to VGG16 (*top*).

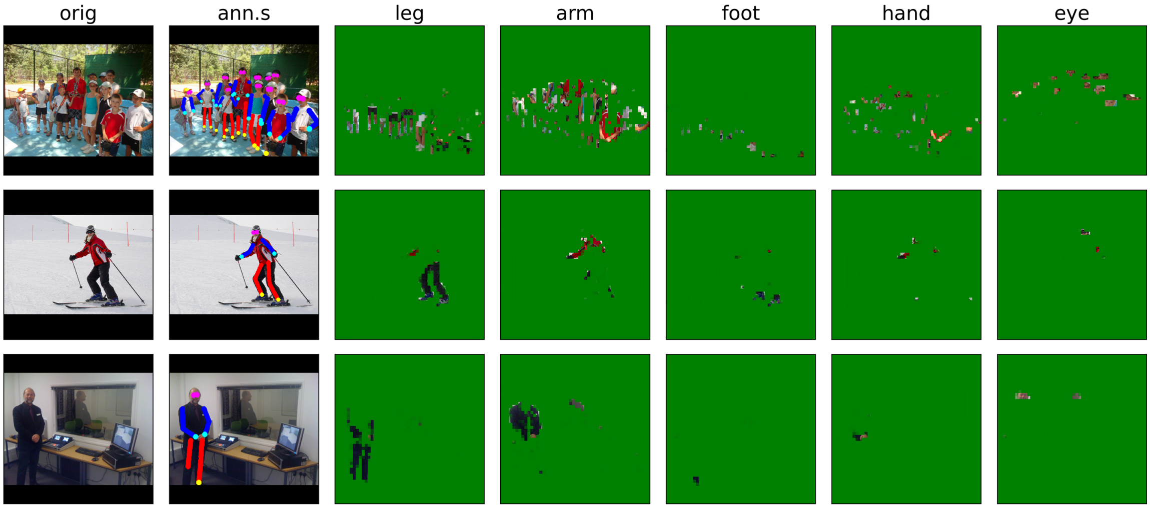


Fig. 5. Overlay (green) of segmentation masks predicted by the mean concept model of Mask R-CNN backbone layer block 3. Original images are shown in first column, generated ground truth annotations in second column in different colors. Mean is taken over normalized concept vectors, cf. [17]. Masks are upsampled using nearest rule to demonstrate resolution. Note how predicted masks clearly correlate with the ground truth instead of being purely white. Images are taken from test set (MS COCO `val2017`). For the used images thanks to: top: Oleg Klementiev http://farm5.staticflickr.com/4115/4906536419_6113bd7de4_z.jpg © CC BY 2.0; middle: Nick Webb http://farm8.staticflickr.com/7015/6795644157_f019453ae7_z.jpg © CC BY 2.0; bottom: Yandle http://farm4.staticflickr.com/3179/2986591710_d76622fdf0_z.jpg © CC BY 2.0 (Color figure online)

4.3 Correlations of Person Size and Segmentation Quality

The last series of experiments means to investigate the following questions: 1) Does the internal representation of a concept differ for different sizes of the concept? And is there a safety critical bias towards one size (i.e., distance from the camera)? 2) Is adaptive kernel size needed to avoid texture bias? A yes to 1) would suggest that different and better embeddings can be found if concept information is assessed separately for each size category. Then, especially for larger size categories, adaptive kernel sizes might be necessary. To answer the

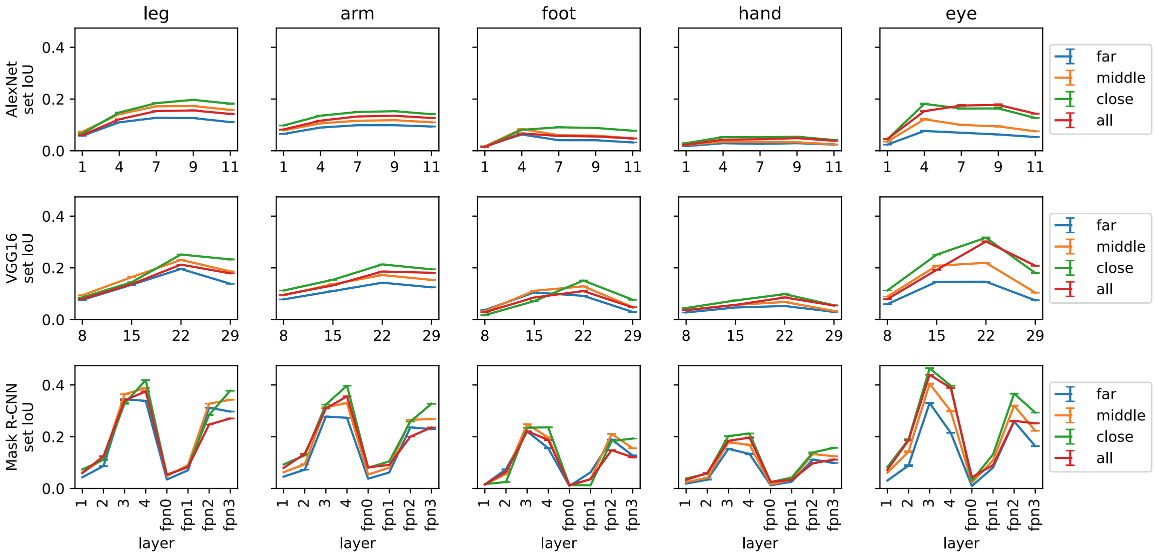


Fig. 6. Test set IoU results of concept models trained on the complete dataset, by layer, annotation size category, model (top to bottom), and concept (left to right). Note that standard deviations are marked but negligible. Best viewed in color. (Color figure online)

questions, results on a test set restricted to one size category were collected on the MS COCO concept data. This was done and compared for concept models trained on the complete dataset (all nets), and models trained on the respective size category either with 1 × 1 or adaptive kernel size (AlexNet, VGG16). Adaptive kernel sizes were chosen to cover the following areas relative to the mean person size of the size category (in height × width): 0.3 × 0.1 for leg, 0.2 × 0.15 for arm, 0.1 × 0.1 for foot and hand, and 0.04 × 0.04 for eye. The two main results were: The internal representation of body parts seems to be mostly size invariant, and adaptive kernel sizes are not necessary.

Results. In Fig. 6 the performance of concept models trained on *all* data was compared for different test subsets. The differences between size categories did not exceed what was expected from the different set IoU penalties, so no bias could be found here. Since a concept model trained on all data may not be the optimal one for a size, we also compared the *all* models to models solely trained on the respective size category, see Fig. 7. This showed that a restricted training set may even decrease test results, indicating that at least parts of the internal representation used for the different sizes must be shared. To substantiate that, the normal vectors of the linear models for the different size categories were compared. In the best performing layers the restricted size categories had high cosine similarity (over 0.7) to *all*, with *far* deviating the most. However, *all* could not be represented as a linear combination of the sub-sizes, and cosine similarities of least squares solutions did not top 0.95. So, some information even seems to get lost when training only on single size categories. Lastly, results on single size categories were also compared to adaptive kernel size results in Fig. 7. Adaptive kernel size would show some improvements but the best layer rarely changed.

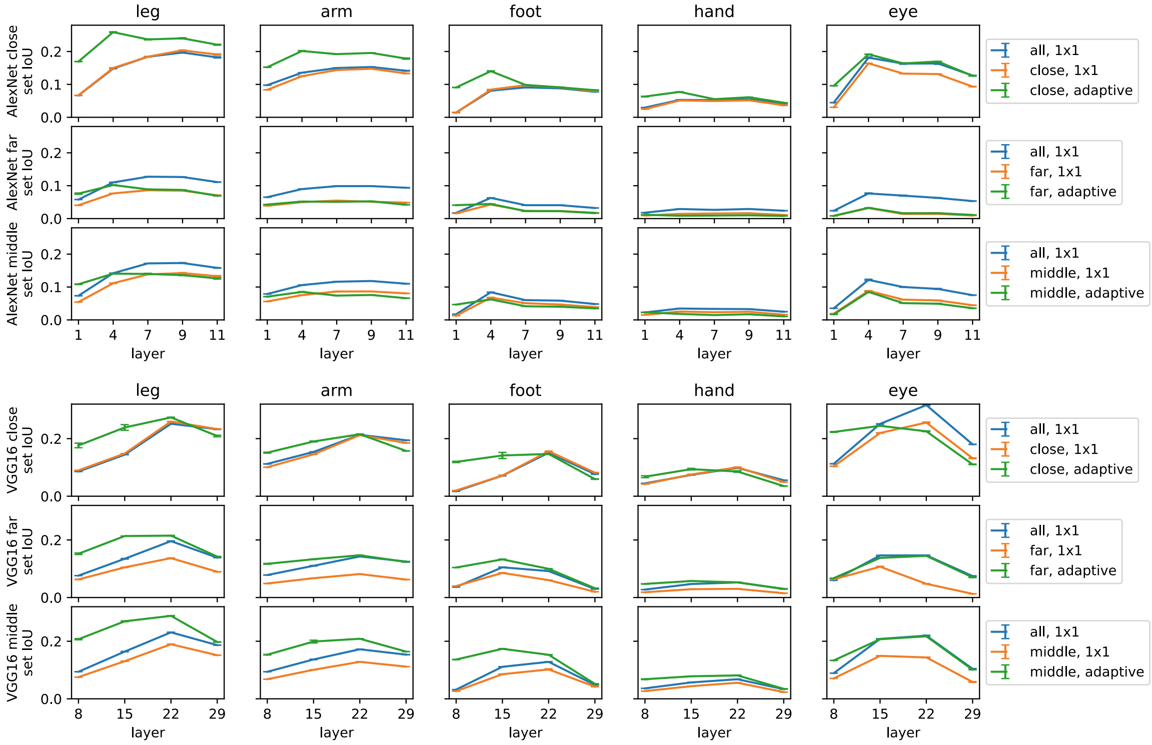


Fig. 7. Test set IoU results of concept models trained on a size subset by layer and kernel setting. Results were obtained on the test set restricted to the respective training size category. Results for the models trained on all data are added for comparison, cf. Fig. 6. Best viewed in color. (Color figure online)

5 Conclusion

This paper proposed an efficient concept analysis approach that is fit for practical application to large networks and datasets, and to object part concepts. It was demonstrated how this method can be used to assess consistency and size bias in the internal representations of a DNN. For this, a size estimation method from 2D skeletal data is proposed together with a new concept dataset based on MS COCO keypoint annotations. Our assessment results suggest that the representations within standard models AlexNet, VGG16, and Mask R-CNN are mostly invariant to different sizes respectively camera distances. Despite drawbacks of the used set IoU metric, concept analysis is shown to be a promising approach for verification and validation of deep neural networks. Future work will investigate further assessment possibilities arising from post-hoc explainable access to DNN intermediate output. This can, e.g., be used to test or formally verify logical properties that are formulated on the enriched DNN output.

References

1. Arrieta, A.B., et al.: Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. Inf. Fusion **58**, 82–115 (2020)

2. Bau, D., Zhou, B., Khosla, A., Oliva, A., Torralba, A.: Network dissection: quantifying interpretability of deep visual representations. In: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3319–3327. IEEE Computer Society (2017). https://doi.org/10.1109/CVPR.2017.354
3. De Brabandere, S.: Human Body Ratios. Scientific American (Bring Science Home) (March 2017). https://www.scientificamerican.com/article/human-body-ratios/
4. Esser, P., Rombach, R., Ommer, B.: A disentangling invertible interpretation network for explaining latent representations. In: Proceedings of the 2020 IEEE Conference on Computer Vision and Pattern Recognition, pp. 9220–9229 (June 2020)
5. Fong, R., Vedaldi, A.: Net2Vec: quantifying and explaining how concepts are encoded by filters in deep neural networks. In: Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, pp. 8730–8738. IEEE Computer Society (2018)
6. Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F.A., Brendel, W.: ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In: Proceedings of the 7th International Conference on Learning Representations. OpenReview.net (2019)
7. Ghorbani, A., Wexler, J., Zou, J.Y., Kim, B.: Towards automatic concept-based explanations. Adv. Neural. Inf. Process. Syst. **32**, 9273–9282 (2019)
8. Gu, J., Tresp, V.: Semantics for global and local interpretation of deep neural networks. CoRR abs/1910.09085 (October 2019). http://arxiv.org/abs/1910.09085
9. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2980–2988 (October 2017)
10. ISO/TC 22/SC 32: ISO 26262–6:2018(En): Road Vehicles—Functional Safety—Part 6: Product Development at the Software Level, ISO 26262:2018(En), 2nd edn., vol. 6. International Organization for Standardization (December 2018)
11. Kim, B., et al.: Interpretability beyond feature attribution: quantitative testing with concept activation vectors (TCAV). In: Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 2668–2677. PMLR (July 2018)
12. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Proceedings of the 3rd International Conference on Learning Representations (2015)
13. Krizhevsky, A.: One weird trick for parallelizing convolutional neural networks. CoRR abs/1404.5997 (2014). http://arxiv.org/abs/1404.5997
14. Larson, D.: Standard proportions of the human body (January 2014). https://www.makingcomics.com/2014/01/19/standard-proportions-human-body/
15. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
16. Olah, C., Mordvintsev, A., Schubert, L.: Feature visualization. Distill, vol. 2, no. 11 (November 2017). https://doi.org/10.23915/distill.00007
17. Rabold, J., Schwalbe, G., Schmid, U.: Expressive explanations of DNNs by combining concept analysis with ILP. In: Schmid, U., Klügl, F., Wolter, D. (eds.) KI 2020. LNCS (LNAI), vol. 12325, pp. 148–162. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58285-2_11
18. Salehi, S.S.M., Erdogmus, D., Gholipour, A.: Tversky loss function for image segmentation using 3D fully convolutional deep networks. In: Wang, Q., Shi, Y., Suk, H.-I., Suzuki, K. (eds.) MLMI 2017. LNCS, vol. 10541, pp. 379–387. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67389-9_44

19. Schwalbe, G., Schels, M.: Concept enforcement and modularization as methods for the ISO 26262 safety argumentation of neural networks. In: Proceedings of the 10th European Congress Embedded Real Time Software and Systems (January 2020). https://hal.archives-ouvertes.fr/hal-02442796
20. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: Proceedings of the 3rd International Conference on Learning Representations (2015)
21. Yeh, C.K., Kim, B., Arik, S., Li, C.L., Pfister, T., Ravikumar, P.: On completeness-aware concept-based explanations in deep neural networks. In: Advances in Neural Information Processing Systems, vol. 33, pp. 20554–20565 (2020)

Expressive Explanations of DNNs by Combining Concept Analysis with ILP

Johannes Rabold[1](✉), Gesina Schwalbe[1,2], and Ute Schmid[1]

[1] Cognitive Systems, University of Bamberg, Bamberg, Germany
{johannes.rabold,gesina.schwalbe,ute.schmid}@uni-bamberg.de

[2] Holistic Engineering and Technologies, Artificial Intelligence,
Continental AG, Regensburg, Germany
gesina.schwalbe@continental-corporation.com

Abstract. Explainable AI has emerged to be a key component for black-box machine learning approaches in domains with a high demand for reliability or transparency. Examples are medical assistant systems, and applications concerned with the General Data Protection Regulation of the European Union, which features transparency as a cornerstone. Such demands require the ability to audit the rationale behind a classifier's decision. While visualizations are the de facto standard of explanations, they come short in terms of expressiveness in many ways: They cannot distinguish between different attribute manifestations of visual features (e.g. eye open vs. closed), and they cannot accurately describe the influence of *absence* of, and *relations* between features. An alternative would be more expressive symbolic surrogate models. However, these require symbolic inputs, which are not readily available in most computer vision tasks. In this paper we investigate how to overcome this: We use inherent features learned by the network to build a global, expressive, verbal explanation of the rationale of a feed-forward convolutional deep neural network (DNN). The semantics of the features are mined by a concept analysis approach trained on a set of human understandable visual concepts. The explanation is found by an Inductive Logic Programming (ILP) method and presented as first-order rules. We show that our explanation is faithful to the original black-box model (The code for our experiments is available at https://github.com/mc-lovin-mlem/concept-embeddings-and-ilp/tree/ki2020).

Keywords: Explainable AI · Concept analysis · Concept embeddings · Inductive Logic Programming

1 Introduction

Machine learning went through several changes of research perspective since its beginnings more than fifty years ago. Initially, machine learning algorithms were inspired by human learning [14]. Inductive Logic Programming (ILP) [17] and

U. Schmid et al. (Eds.): KI 2020, LNAI 12325, pp. 148–162, 2020.
https://doi.org/10.1007/978-3-030-58285-2_11

explanation-based generalization [16] were introduced as integrated approaches which combine reasoning in first-order logic and inductive learning.

With the rise of statistical approaches to machine learning, focus shifted from human-like learning to optimizing learning for high predictive accuracy. Deep learning architectures [7] resulted in data-intensive, black-box approaches with impressive performances in domains such as object recognition, machine translation, and game playing. However, since machine learning more and more is moving from the lab to the real world, researchers and practitioners alike realize that interpretable, human-like approaches to machine learning are necessary to allow developers as well as end-users to evaluate and understand classifier decisions or possibly also the learned models themselves.

Consequently there is a growing number of approaches to support explainability of black-box machine learning [1]. Explainable AI (XAI) approaches are proposed to support developers to recognize oversampling and problems with data quality such as number of available data, class imbalance, expensive labeling, and sampling biases [2,13]. For many application domains, it is a legal as well as an ethical obligation to make classifier decisions transparent and comprehensible to end-users who need to make sense of complex information, for instance in medical diagnosis, automotive safety, or quality control.

A main focus of research on explanations for image classifications is on visual explanations, that is, highlighting of relevant pixels such as LRP [23] or showing relevant areas in the image such as LIME [21]. However, visual explanations can only show which conjunction of information in an image is relevant. In many domains, more sophisticated information needs to be taken into account [24]:

- **Feature values:** highlighting the area of the eye in an image is not helpful to understand that it is important for the class decision that the lids are tightened (indicating pain) in contrast to eyes which are wide open (indicating startle, [29]);
- **Quantification:** highlighting all blowholes on the supporting parts of a rim does not make clear that the rim is not a reject because *all* blowholes are smaller than 0,5 mm;
- **Negation:** highlighting the flower in the hand of a person does not transport the information that this person is *not* a terrorist because he or she does *not* hold a weapon;
- **Relations:** highlighting all windows in a building cannot help to discriminate between a tower, where windows are *above* each other and a bungalow, where windows are *beside* each other [19];
- **Recursion:** highlighting all stones within a circle of stones cannot transport the information that there must be a sequence of an arbitrary number of stones with increasing size [20].

Such information can only be expressed in an expressive language, for instance some subset of first-order logic [18]. In previous work, it has been shown how ILP can be applied to replace the simple linear model agnostic explanations of LIME [3,19,20,25]. Alternatively, it is investigated how knowledge can be incorporated into deep networks. For example, capsule networks [22] are proposed to model hierarchical relationships and embeddings of knowledge graphs allow to grasp relationships between entities [8].

In this paper, we investigate how symbolic knowledge can be extracted from the inner layers of a deep convolutional neural network to uncover and extract relational information to build an expressive global explanation for the network. In the following, we first introduce concept embedding analysis (to extract visual concepts) and ILP (to build the explanation). In Sect. 3, the proposed approach to model-inherent generation of symbolic relational explanations is presented. We present a variety of experiments on a new "Picasso" data set of faces with permuted positions of sub-parts such as eyes, mouth, and nose. We conclude with an outlook to extend this first, preliminary investigation in the future.

2 Theoretical Background

2.1 Concept Embedding Analysis

To understand the process flow of an algorithm, it is of great value to have access to interpretable intermediate outputs. The goal of concept embedding analysis is to answer *whether*, *how well*, *how*, and with what *contribution to the reasoning* information about semantic concepts is embedded into the latent spaces (intermediate outputs) of DNNs, and to provide the result in an explainable way. Focus currently lies on finding embeddings in either the complete output of a layer (image-level concepts), or single pixels of an activation map of a convolutional DNN (concept segmentation). To answer the *whether*, one can try to find a decoder for the information about the concept of interest, the *concept embedding*. This means, one is looking for a classifier on the latent space that can predict the presence of the concept. The performance of the classifier provides a measure of *how well* the concept is embedded. For an explainable answer of *how* a concept is embedded, the decoder should be easily interpretable. One constraint to this is introduced by the rich vector space structure of the space of semantic concepts respectively word vector spaces [15]: The decoder map from latent to semantic space should preserve at least a similarity measure. For example, the encodings of "cat" and "dog" should be quite similar, whereas that of a "car" should be relatively distant from the two. The methods in literature can essentially be grouped by their choice of distance measure $\langle -, - \rangle$ used on the latent vector space. A concept embedding classifier E_c predicting the presence of concept c in the latent space L then is of the form $E_c(v) = \langle v_c, v \rangle > t_c$ for $v \in L$, where $v_c \in L$ is the concept vector of the embedding, and $t_c \in \mathbb{R}$.

Automated concept explanations [6] uses L_2 distance as similarity measure. They discover concepts in an unsupervised fashion by k-means clustering of the latent space representations of input samples. The concept vectors of the discovered concepts are the cluster centers. In TCAV [11] it is claimed that the mapping from semantic to latent space should be linear for best interpretability. To achieve this, they suggest to use linear classifiers as concept embeddings. This means they try to find a separation hyperplane between the latent space representations of positive and negative samples of the concept. A normal vector of the hyperplane then is their concept vector, and the distance to the hyperplane is used as distance measure. As method to obtain the embedding they use support vector machines (SVMs). TCAV further investigated the contribution of concepts to given output classes by sensitivity analysis. A very similar approach to TCAV, only instead relying on logistic regression, is followed by Net2Vec [5]. As a regularization, they add a filter-specific cut-off before the concept embedding analysis to remove noisy small activations. The advantage of Net2Vec over the SVMs in TCAV is that they can more easily be used in a convolutional setting: They used a 1×1-convolution to do a prediction of the concept for each activation map pixel, providing a segmentation of the concept. This was extended by [26], who suggested to allow larger convolution windows to ensure that the receptive field of the window can cover the complete concept. This avoids a focus on local patterns. A measure that can be applied to concept vectors of the same layer regardless of the analysis method, is that of *completeness* suggested in [31]. They try to measure, how much of the information relevant to the final output of the DNN is covered by a chosen set of concepts vectors. They also suggested a metric to compare the attribution of each concept to the completeness score of a set of concepts.

2.2 Inductive Logic Programming

Inductive Logic Programming (ILP) [17] is a machine learning technique that builds a logic theory over positive and negative examples (E^+, E^-). The examples consist of symbolic background knowledge (BK) in the form of first-order logic predicates, e.g. `contains(Example, Part), isa(Part, nose)`. Here the upper case symbols are variables and the lower case symbol is a constant. The given BK describes that example `Example` contains a part `Part` which is a nose. Based on the examples, a logic theory can be learned. The hypothesis language of this theory consists of logic Horn clauses that contain predicates from the BK. We write the Horn clauses as implication rules, e.g.

```
face(Example):- contains(Example, Part), isa(Part, nose).
```

For this work we obey the syntactic rules of the Prolog programming language. The `:-` denotes the logic implication (←). We call the part before the implication the *head* of a rule and the part after it the *body* or *preconditions* of a rule.

We use the framework Aleph [28] for this work since it is a flexible and adaptive general purpose ILP toolbox. Aleph's built in algorithm attempts to induce a logic theory from the given BK to cover as many positive examples E^+ as possible while avoiding covering the negative examples E^-. The general algorithm of Aleph can be summarized as follows [28]:

1. As long as positive examples exist, select one. Otherwise halt.
2. Construct the most-specific clause that entails the selected example and is within the language constraints.
3. Find a more general clause which is a subset of the current literals in the clause.

4. Remove examples covered by the current clause.
5. Repeat from step 1.

3 Explaining a DNN with Concept Localization and ILP

When building explanations for a DNN via approximate rule sets, the underlying logic and predicates of the rules should reflect the capabilities of the model. For example, spatial relations like `top_of` or `right_of` should be covered, as e.g. dense layers of a DNN are capable of encoding these. Spatial relations cannot be represented by current visualization methods for explainable AI, which only feature predicates of the form `contains(Example, Part)` and `at_position(Part, xy)`. Rule-based methods like ILP are able to incorporate richer predicates into the output. However, their input must be symbolic background knowledge about the training and inference samples which is formulated using these predicates *explicitly*. For computer vision tasks with pixel-level input, this encoding of the background knowledge about samples is not available. To remedy this, we propose to use existing concept mining techniques for extraction of the required background knowledge:

1. Associate pre-defined visual semantic concepts with intermediate output of the DNN. Concepts can be local, like parts and textures, or image-level.
2. Automatically infer the background knowledge about a sample `Ex` given the additional concept output, which defines predicates `isa(C, concept)`, and `isa(Ex, C)` (image-level) or `contains(Ex, C)` with `at_position(C, xy)`. From this, spatial relations and negations can be extracted.
3. Given background knowledge for a set of training samples, apply an inductive logic programming approach to learn an expressive set of rules for the DNN.

The approach presented in this paper differs from the previous work outlined in [19] by the following main aspects:

- We will find a global verbal explanation for a black-box decision in contrast to a local explanation.
- We directly make use of information stored in the building blocks of the DNN instead of relying on the linear surrogate model generated by LIME.

3.1 Enrich DNN Output via Concept Embedding Analysis

We directly built upon the concept detection approach from [5,26], suggesting some further improvements. Net2Vec bilinearly upscaled the predicted masks before applying the sigmoid for logistic regression. This overrates the contribution to the loss by pixels at the edges from positive to negative predicted pixels. We instead apply upscaling after applying the sigmoid. Instead of the suggested IoU penalty from [26], we propose a more stable Dice loss to fit the overlap objective, supported by a small summand of the balanced binary cross-entropy (bBCE) suggested in Net2Vec to ensure pixel-wise accuracy.

One major disadvantage of the linear model approaches over the clustering ones is their instability, i.e. several runs for the same concept yield different concept vectors. Reasons may be dependence on the outliers of the concept cluster (SVM); non-unique solutions due to a margin between the clusters; and inherent variance of the used optimization methods. To decrease dependence on the training set selection and ordering, and the initialization values, we for now simply use ensembling. For this we define a hyperplane H as the zero set of the distance function $d_H(v) = (v - b_H \cdot v_H) \circ v_H$ for the normal vector v_H and the support vector $b_H v_H$, $b_H \in \mathbb{R}$. Then, the zero set of the mean $\frac{1}{N}\sum_{i=1}^{N} d_{H_i}$ of the distance functions of hyperplanes H_i again defines a hyperplane with

$$v_H = \frac{1}{N}\sum_{i=1}^{N} v_{H_i} \qquad \text{and} \qquad b_H = \frac{1}{\|v_H\|^2}\frac{1}{N}\sum_{i=1}^{N}(b_{H_i}\|v_{H_i}\|^2) \ .$$

Note, that hyperplanes with longer normal vectors (i.e. higher confidence values) are overrated in this calculation. To remedy this, concept vectors are normalized before ensembling, using the property $(w - b \cdot v) \circ v = \|v\| \cdot (w - (b\|v\|)\frac{v}{\|v\|}) \cdot \frac{v}{\|v\|}$ of the distance function for scalar b and vectors v, w.

3.2 Automatic Generation of Symbolic Background Knowledge

The output of the concept analysis step (binary masks indicating the spatial location of semantic concepts) can be used to build a symbolic global explanation for the behavior of the original black-box model. We obtain the explanation by finding a first-order logic theory with the ILP approach Aleph (see Sect. 2.2). Since Aleph needs a set of positive and negative examples (E^+, E^-), the first step is to obtain these examples along with their corresponding symbolic background knowledge (BK). In order to obtain a good approximation of the behavior of the model, we sample N^+ binary masks from positively predicted images and N^- binary masks from negatively predicted images that lie close to the decision boundary of the original black-box model using the concept analysis model described above. Let M^+, M^- be the set of positive and negative binary masks. Let $m^+ \in M^+$, $m^- \in M^-$ be single masks. Each mask (e.g. m^+) consists of multiple mask layers (e.g. $l_c \in m^+$, $c \in C$) for the different human understandable concepts from the pool of concepts C. These mask layers are sparse matrices upsampled to the same size as the original images they are masking. The matrices have the value 1 at all the positions where the concept analysis model detected the respective concept and 0 at all other positions.

The symbolic explanation of the original model should consist of logic rules that establish the prototypical constellation of visual parts of an image that resembles the positive class as seen by the DNN. We therefore need not only the information about occurrence of certain visual parts in the sampled examples but also the different relations that hold between the parts. In the next sections we adhere to the following general workflow:

1. Find positions of visual parts in the examples and name them.
2. Find relations between parts.
3. Build BK with the information from step 1 and 2.
4. Induce a logic theory with Aleph.

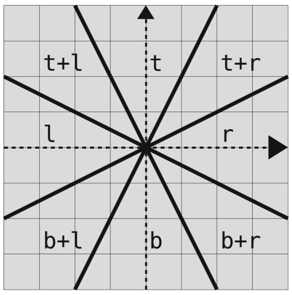


Fig. 1. Potential positions of an object to be only **t**op of, **r**ight of, **b**ottom of, or **l**eft of a reference object located in the origin. Also the four overlapping regions are indicated.

Find Visual Parts. One mask layer l_c contains possibly multiple contiguous clusters of concept propositions. Therefore, as a denoising step, we only take the cluster with the largest area into account. As a proposition for the position of the concept c in the picture, in this cluster we take the mean point of the area A_{max} of 1's in l_c. We therefore find the position $p_c = (x, y)$ with $x = (\min_x(A_{max}) + \max_x(A_{max}))/2$ and $y = (\min_y(A_{max}) + \max_y(A_{max}))/2$. This procedure can be followed for all masks l_c that are contained in all $m^+ \in M^+$ and $m^- \in M^-$.

Find Relations Between Parts. By taking relationships between the parts into account, we strive for more expressive explanations. For this work we limit ourselves to spatial relationships that hold between the parts that were found in the previous steps. We assume that pairs of two parts can be in the following four relationships to each other: `left_of`, `right_of`, `top_of`, `bottom_of`. We declare part `A` to be `top_of` part `B` if the vertical component y_A of the position p_A is above y_B and the value for the horizontal offset $\Delta x = x_A - x_B$ does not diverge from the value for the vertical offset $\Delta y = y_A - y_B$ by more than double. The other spatial relations can be formalized in an analogous manner. Thus, the relations that can hold between two parts `A` and `B` can be visualized as in Fig. 1.

Inferring Global Symbolic Explanations. After the inference of visual parts and the relationships that hold between them, we can build the BK needed for Aleph. Part affiliation to an example can be declared by the `contains` predicate. We give all parts a unique name over all examples. Suppose part `A` is part of a particular example `E` and describes the human understandable concept `c` $\in C$. Then the example affiliation can be stated by the predicates `contains(E, A)`, `isa(A, c)`. Likewise for the relations we can use 2-ary predicates that state the constellation that holds between the parts. When part `A` is left of part `B` we incorporate the predicate `left_of(A, B)` in the BK and likewise for the other relations. When the BK for the positive and negative examples E^+ and E^- is found, we can use Aleph's induction mechanism to find the set of rules that best fit the examples. The complete algorithm for the process is stated in Algorithm 1.

Algorithm 1. Verbal Explanation Generation for DNNs

```
**Require:** Positive and negative binary masks $M^+$, $M^-$
**Require:** Pool of human understandable concepts $C$
$E^+ \leftarrow \{\}$
$E^- \leftarrow \{\}$
**for each** $\odot \in \{+, -\}$ **do**
    **for each** $m^\odot \in M^\odot$ **do**
        $P \leftarrow \{\}$
        **for each** $l_c \in m^\odot$ where $c \in C$ **do**
            $p_c \leftarrow calculatePartPosition(l_c)$
            $P \leftarrow P \cup \{\langle c, p_c \rangle\}$
        $R \leftarrow calculateRelations(P)$
        $E^\odot \leftarrow E^\odot \cup \langle P, R \rangle$
$T \leftarrow \text{Aleph}(E^+, E^-)$
**return** $T$
```

4 Experiments and Results

We conducted a variety of experiments to audit our previously described approach. As a running example we used a DNN which we trained on images from a generated dataset we dubbed *Picasso Dataset*. The foundation are images of human faces we took from the FASSEG dataset [9,10]. The Picasso Dataset contains collage images of faces with the facial features (eyes, mouth, nose) either in the correct constellation (positive class) or in a mixed-up constellation (negative class). See Fig. 2 for examples. No distinction is made between originally left and right eyes.

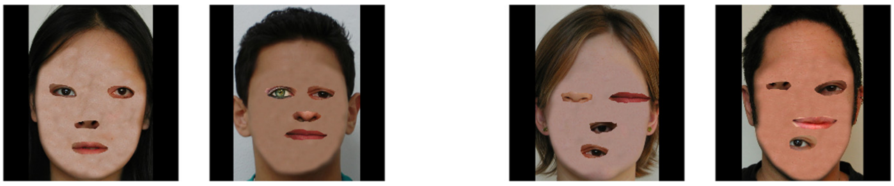


Fig. 2. Examples from the Picasso Dataset (*left:* positive class, *right:* negative).

In order to not establish a divergence in the image space of the two classes, the positive and negative classes contain facial features that were cut out of a set of images from the original FASSEG dataset. As a canvas to include the features, we took a set of original faces and got rid of the facial features by giving the complete facial area a similar skin-like texture. Then we included the cut out facial features onto the original positions of the original features in the faces.

The face images in Fig. 2 show that the resulting dataset is rather constructed. This however will suffice for a proof of concept to show that our approach in fact exploits object parts and their relations. In the future we plan on moving towards more natural datasets.

4.1 Analyzed DNNs

We evaluated our method on three different architectures from the pytorch model-zoo[1]: AlexNet [12], VGG16 [27], and ResNeXt-50 [30]. The convolutional parts of the networks were initialized with weights pre-trained on the ImageNet dataset. For fine-tuning the DNNs for the Picasso Dataset task, the output dimension was reduced to one and the in- and output dimension of the second to last hidden dense layer was reduced to 512 for AlexNet and VGG16. Then the dense layers and the last two convolutional layers (AlexNet, VGG16) respectively bottleneck blocks (ResNeXt) were fine-tuned. The fine-tuning was conducted in one epoch on a training set of 18,002 generated, 224 × 224-sized picasso samples with equal distribution of positive and negative class. All models achieved accuracy greater than 99% on a test set of 999 positive and 999 negative samples.

4.2 Training the Concept Models

In our example we determined the best ensembled detection concept vectors for the concepts EYES, MOUTH and NOSE amongst the considered layers. We excluded layers with low receptive field, as they are assumed to hold only very local features (for the layers used see Fig. 3). Convolutional output was only considered after the activation. For each concept, 452 training/validation, and 48 test picasso samples with segmentation masks were used. The training objective was: Predict at each activation map pixel whether the kernel window centered there lies "over" an instance of the concept. Over meant that the fuzzy intersection of the concept segmentation and the kernel window area exceeds a threshold (*intersection encoding*). This fuzzy definition of a box center tackles the problem of sub-optimal intersections in later layers due to low resolution. Too high values may lead to elimination of an instance, and thresholds were chosen to avoid such issues with values 0.5/0.8/0.7 for nose/mouth/eye. We implemented the encoding via a convolution. As evaluation metric we use set IoU (sIoU) between the detection masks and the intersection encoded masks as in Net2Vec. On each dataset and each layer, 15 concept models were trained in three 5-fold-cross-validation runs with the following settings: Adam optimization with mean best learning rate of 0.001, a weighting of 5:1 of Dice to bBCE loss, batch size of 8, and two epochs (all layers showed quick convergence).

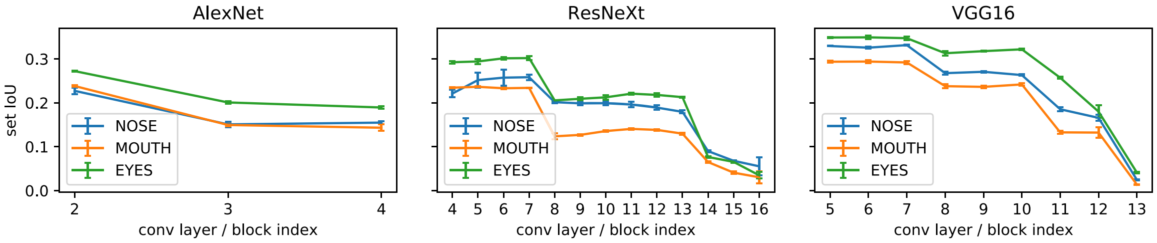


Fig. 3. The layer-wise mean set IoU results of the concept analysis runs.

[1] https://pytorch.org/docs/stable/torchvision/models.html.

Results. Our normalized ensembling approach proved valuable as it yielded mean or slightly better performance compared to the single runs. For the considered models, meaningful embeddings of all concepts could be found (see Table 1): The layers all reached sIoU values greater than 0.22 despite of the still seemingly high influence of sub-optimal resolutions of the activation maps. Figure 4 shows some exemplary outputs. The concepts were best embedded in earlier layers, while different concepts did not necessarily share the same layer.

Table 1. Results for ensemble embeddings with set IoU (sIoU), mean cosine distance to the runs (Cos.d.), and index of conv layer or block (L) (cf. Fig. 3).

AlexNet		L	sIoU	Cos.d.
	NOSE	2	0.228	0.040
	MOUTH	2	0.239	0.040
	EYES	2	0.272	0.058

VGG16		L	sIoU	Cos.d.
	NOSE	7	0.332	0.104
	MOUTH	6	0.296	0.154
	EYES	6	0.350	0.197

ResNeXt		L	sIoU	Cos.d.
	NOSE	6	0.264	0.017
	MOUTH	5	0.237	0.020
	EYES	7	0.302	0.020

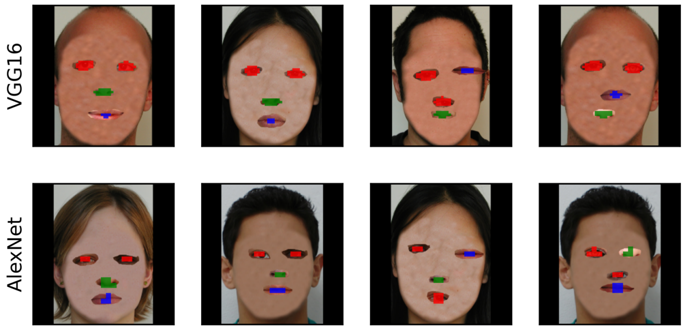


Fig. 4. Ensemble embedding outputs of NOSE (green), MOUTH (blue), EYES (red). (Color figure online)

4.3 Example Selection for ILP Training

The goal of the ILP model is to approximate the behavior of the main DNN, i.e. its decision boundary. For this, few but meaningful training samples and their DNN output are needed: class-prototypes as well as ones that tightly frame the DNN decision boundary. From the 1,998 samples in the picasso test set, in total 100 samples were chosen from the DNN test set to train the ILP model. The DNN confidence score here was used to estimate the proximity of a data point to the decision boundary. For each class, we selected the 50 samples predicted to be in this class and with confidence closest to the class boundary of 0.5. In our setup this provided a wide range of confidence values (including 0 and 1).

4.4 Finding the Symbolic Explanation

In order to find the background knowledge needed for Aleph to generate the explanation, we need to extract the information about the facial features and their constellations from the masks of the samples drawn in the previous step. Abiding the procedure described in Sect. 3.2, we first find contiguous clusters in the mask layers to then infer the positional information for them. This is straight-forward for the nose and the mouth but imposes a problem for the eyes, since we do not want to have a single position proposal for them in the eye that produces the biggest cluster in the mask layer. Thus, we allow for the top two biggest clusters to infer a position. Although we give them unique constants in the BK, we both give them the type `eye` $\in C$.

The next step consists of the extraction of the spatial features between the found parts. Since the relation pair `left_of`/`right_of` as well as `top_of`/`bottom_of` can be seen as the inverses of the respective other relation, we omit the relations `right_of` and `bottom_of` in the BK. This is possible, because the *Closed World Assumption* holds (Everything that is not stated explicitly is false).

Once the BK is found for all examples, we can let Aleph induce a theory of logic rules. Consider the induced theory for the trained VGG16 network:

```
face(F):- contains(F, A), isa(A, nose), contains(F, B), isa(B, mouth),
          top_of(A, B), contains(F, C), top_of(C, A).
```

The rule explicitly names the required facial concepts `nose` and `mouth` and the fact that the nose has to be above the mouth in order for an image to be a face. Further there is another unnamed component `C` required which has to be placed above the nose. By construction this has to be one of the eyes. The rule makes sense intuitively as it describes a subset of correct constellations of the features of a human face.

To further test the fidelity of the generated explanations to the original black-box network, we calculated several performance metrics for a test set of 1998 test images (999 positive and 999 negative examples). We handled the learned explanation rules as binary classification model for the test images in BK representation. If an image representation is covered by the explanation rules, it is predicted to be positive, otherwise negative. We now can handle the binary output of the black-box model as ground truth to our explanation predictions. The performance metrics together with the induced explanation rules for several DNN architectures are listed in Table 2. It can be seen that the explanations stay true to the original black-box model.

Table 2. Learned rules for different architectures and their fidelity scores (accuracy and F1 score wrt. to the original model predictions). Learned rules are of common form `face(F):- contains(F, A), isa(A, nose), contains(F, B), isa(B, mouth), distinctPart`

Arch.	Accuracy	F1	Distinct rule part
VGG16	99.60%	99.60%	`top_of(A, B), contains(F, C), top_of(C, A)`
AlexNet	99.05%	99.04%	`contains(F, C), left_of(C, A), top_of(C, B), top_of(C, A)`
ResNext	99.75%	99.75%	`top_of(A, B), contains(F, C), top_of(C, A)`

5 Conclusion and Future Work

Within the described simple experiment we showed that expressive, verbal surrogate models with high fidelity can be found for DNNs using the developed methodology. We suggest that the approach is promising and worth future research and optimization.

The proposed concept detection approach requires a concept to have little variance in its size. It should easily extend to a concept with several size categories (e.g. close by and far away faces) by merging the result for each category. A next step for the background knowledge extraction would be to extend it to an arbitrary number of concept occurrences per image, where currently the algorithm assumes a fixed amount (exactly one `mouth`, one `nose`, two `eyes`). This could e.g. be achieved by allowing a maximum number per sliding window rather than an exact amount per image. In cases, where the predicates cannot be pre-defined, one can learn the relations as functions on the DNN output from examples as demonstrated in [4].

We further did not consider completeness (cf. Sect. 2.1) of the chosen concepts: They may not be well aligned with the decision relevant features used by the DNN, infringing fidelity of the surrogate model. We suggest two ways to remedy this: One could rely on (possibly less interpretable) concepts found via concept mining [6]. Or, since ILP is good at rejecting irrelevant information, one can start with a much larger set of pre-defined, domain related concepts. We further assume that best fidelity can only be achieved with the *minimal* complete sub-set of most decision-relevant concepts, which fosters uniqueness of the solution. For a decision relevance measure see e.g. [6].

It may be noted that the presented concept analysis approach is not tied to image classification: As long as the ground truth for concepts in the form of masks or classification values is available, the method can be applied to any DNN latent space (imagine e.g. audio, text, or video classification). However, spatial or temporal positions and relations are currently inferred using the receptive field information of convolutional DNNs. This restriction may again be resolved by learning of relations.

Lastly, in order to examine the understandability of the induced explanation in a real world scenario, we need to let explanations be evaluated in a human user study. For this matter, subjective evaluation measures have to be specifically designed for verbal explanations.

References

1. Adadi, A., Berrada, M.: Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). IEEE Access **6**, 52138–52160 (2018)
2. Arya, V., et al.: One explanation does not fit all: a toolkit and taxonomy of AI explainability techniques. CoRR (2019). http://arxiv.org/abs/1909.03012
3. Dai, W.Z., Xu, Q., Yu, Y., Zhou, Z.H.: Bridging machine learning and logical reasoning by abductive learning. In: Advances in Neural Information Processing Systems, pp. 2811–2822 (2019)
4. Donadello, I., Serafini, L., d'Avila Garcez, A.S.: Logic tensor networks for semantic image interpretation. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence, pp. 1596–1602. ijcai.org (2017). https://doi.org/10.24963/ijcai.2017/221
5. Fong, R., Vedaldi, A.: Net2Vec: quantifying and explaining how concepts are encoded by filters in deep neural networks. In: Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, pp. 8730–8738. IEEE (2018). https://doi.org/10.1109/CVPR.2018.00910
6. Ghorbani, A., Wexler, J., Zou, J.Y., Kim, B.: Towards automatic concept-based explanations. In: Advances in Neural Information Processing Systems 32, pp. 9273–9282 (2019). http://papers.nips.cc/paper/9126-towards-automatic-concept-based-explanations
7. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
8. Ji, G., He, S., Xu, L., Liu, K., Zhao, J.: Knowledge graph embedding via dynamic mapping matrix. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Vol. 1: Long Papers), pp. 687–696 (2015)
9. Khan, K., Mauro, M., Leonardi, R.: Multi-class semantic segmentation of faces. In: Proceedings of the 2015 IEEE International Conference on Image Processing (ICIP), pp. 827–831. IEEE (2015)
10. Khan, K., Mauro, M., Migliorati, P., Leonardi, R.: Head pose estimation through multi-class face segmentation. In: Proceedings of the 2017 IEEE International Conference on Multimedia and Expo (ICME). pp. 175–180. IEEE (2017)
11. Kim, B., et al.: Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In: Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 2668–2677. PMLR (2018). http://proceedings.mlr.press/v80/kim18d.html
12. Krizhevsky, A.: One weird trick for parallelizing convolutional neural networks. CoRR (2014). http://arxiv.org/abs/1404.5997
13. Lapuschkin, S., Wäldchen, S., Binder, A., Montavon, G., Samek, W., Müller, K.R.: Unmasking clever hans predictors and assessing what machines really learn. Nat. Commun. **10**(1), 1–8 (2019)

14. Michalski, R.S., Carbonell, J.G., Mitchell, T.M. (eds.): Machine Learning - An Artificial Intelligence Approach. Tioga, Palo Alto (1983)
15. Mikolov, T., Yih, W.T., Zweig, G.: Linguistic regularities in continuous space word representations. In: Proceedings of the 2013 Conference on North American Chapter Association for Computational Linguistics: Human Language Technologies, pp. 746–751. Association for Computational Linguistics (2013). https://www.aclweb.org/anthology/N13-1090
16. Mitchell, T.M., Keller, R.M., Kedar-Cabelli, S.T.: Explanation-based generalization: a unifying view. Mach. Learn. **1**(1), 47–80 (1986). https://doi.org/10.1023/A:1022691120807
17. Muggleton, S.: Inductive logic programming. New Gener. Comput. **8**(4), 295–318 (1991)
18. Muggleton, S., Schmid, U., Zeller, C., Tamaddoni-Nezhad, A., Besold, T.: Ultra-strong machine learning: comprehensibility of programs learned with ILP. Mach. Learn. **107**(7), 1119–1140 (2018). https://doi.org/10.1007/s10994-018-5707-3
19. Rabold, J., Deininger, H., Siebers, M., Schmid, U.: Enriching visual with verbal explanations for relational concepts-combining lime with Aleph. arXiv preprint arXiv:1910.01837 (2019)
20. Rabold, J., Siebers, M., Schmid, U.: Explaining black-box classifiers with ILP – empowering LIME with Aleph to approximate non-linear decisions with relational rules. In: Riguzzi, F., Bellodi, E., Zese, R. (eds.) ILP 2018. LNCS (LNAI), vol. 11105, pp. 105–117. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99960-9_7
21. Ribeiro, M.T., Singh, S., Guestrin, C.: Why should i trust you?: Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1135–1144. ACM (2016)
22. Sabour, S., Frosst, N., Hinton, G.E.: Dynamic routing between capsules. In: Advances in Neural Information Processing Systems, pp. 3856–3866 (2017)
23. Samek, W., Wiegand, T., Müller, K.R.: Explainable artificial intelligence: understanding, visualizing and interpreting deep learning models. CoRR (2017). http://arxiv.org/abs/1708.08296
24. Schmid, U.: Inductive programming as approach to comprehensible machine learning. In: Proceedings of the 6th Workshop KI & Kognition, KIK-2018. Co-located with KI 2018 (2018). http://ceur-ws.org/Vol-2194/schmid.pdf
25. Schmid, U., Finzel, B.: Mutual explanations for cooperative decision making in medicine. KI - Künstliche Intelligenz, Special Issue Challenges in Interactive Machine Learning 34 (2020)
26. Schwalbe, G., Schels, M.: Concept enforcement and modularization as methods for the ISO 26262 safety argumentation of neural networks. In: Proceedings of the 10th European Congress Embedded Real Time Software and Systems (2020). https://hal.archives-ouvertes.fr/hal-02442796
27. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: Proceedings of the 3rd International Conference on Learning Representations (2015). http://arxiv.org/abs/1409.1556
28. Srinivasan, A.: The Aleph Manual (2004). https://www.cs.ox.ac.uk/activities/programinduction/Aleph
29. Weitz, K., Hassan, T., Schmid, U., Garbas, J.U.: Deep-learned faces of pain and emotions: elucidating the differences of facial expressions with the help of explainable AI methods. tm-Technisches Messen **86**(7–8), 404–412 (2019)

30. Xie, S., Girshick, R.B., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, pp. 5987–5995. IEEE (2017). https://doi.org/10.1109/CVPR.2017.634
31. Yeh, C.K., Kim, B., Arik, S.O., Li, C.L., Pfister, T., Ravikumar, P.: On completeness-aware concept-based explanations in deep neural networks. CoRR (2020). http://arxiv.org/abs/1910.07969

Enabling Verification of Deep Neural Networks in Perception Tasks Using Fuzzy Logic and Concept Embeddings

Gesina Schwalbe$^{1,2[0000-0003-2690-2478]}$, Christian Wirth^{1}, and Ute Schmid$^{2[0000-0002-1301-0326]}$

[1] Continental AG, Regensburg, Germany
gesina.schwalbe@conti.de
christian.2.wirth@continental-corporation.com
[2] Cognitive Systems, University of Bamberg, Bamberg, Germany
ute.schmid@uni-bamberg.de

Abstract. One major drawback of deep convolutional neural networks (CNNs) for use in safety critical applications is their black-box nature. This makes it hard to verify or monitor complex, symbolic requirements on already trained computer vision CNNs. In this work, we present a simple, yet effective, approach to verify that a CNN complies with symbolic predicate logic rules which relate visual concepts. It is the first that (1) does not modify the CNN, (2) may use visual concepts that are no CNN in- or output feature, and (3) can leverage continuous CNN confidence outputs. To achieve this, we newly combine methods from explainable artificial intelligence and logic: First, using supervised concept embedding analysis, the output of a CNN is post-hoc enriched by concept outputs. Second, rules from prior knowledge are modelled as truth functions that accept the CNN outputs, and can be evaluated with little computational overhead. We here investigate the use of fuzzy logic, i.e., continuous truth values, and of proper output calibration, which both theoretically and practically show slight benefits. Applicability is demonstrated on state-of-the-art object detectors for three verification use-cases, where monitoring of rule breaches can reveal detection errors.

Keywords: Concept Activation Vectors, Fuzzy Logic, Calibration, Verification, Runtime Monitoring, CNN, XAI

1 Introduction

Deep neural network (DNN) debugging and safety critical use-cases like automated driving perception require verification of semantic prior domain knowledge [32]. Such knowledge relates concepts that are semantically meaningful to humans, e.g., objects, sub-objects, and object properties, and is potentially complex and fuzzy [25] Examples are physical laws or typical anatomical structures.

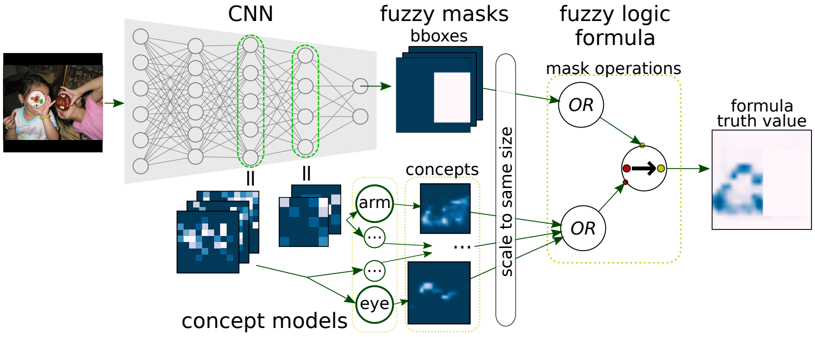


Fig. 1: Visualization of our approach from Sec. 3 for the pixel-wise formula $F(p) = (\mathtt{Is}_{\mathsf{eye}}(p) \lor \mathtt{Is}_{\mathsf{arm}}(p) \lor \dots) \to \mathtt{Is}_{\mathsf{Person}}(p)$. (Image attribution in Fig. 2).

While it would be desirable to formulate the prior knowledge as verifiable constraints, this fails for standard black-box object detection CNNs for computer vision tasks [21]. In particular, they lack an association of semantic concepts with inputs or (intermediate) outputs: (1) inputs are non-symbolic, (2) final outputs only describe few concepts to reduce labeling costs (e.g., few object classes, no sub-object classes), and (3) intermediate outputs are non-interpretable. Hence, standard work on improving [6] or formally verifying [20] rule compliance are not applicable without custom-built DNN architectures. One way to overcome this and post-hoc extract learned concept information from a trained CNN is concept (embedding) analysis (CA) [3,7,13]. The idea is to attach cheap and small models to the CNN intermediate outputs. These are trained in a supervised manner to predict the existence of a concept. This also works for large object detectors as shown in [31], requires only few additional labels [7,12], and leaves the original DNN unchanged.

We here combine CA with methods of (fuzzy) formal rule formulation. This for the first time allows to *post-hoc* score the compliance of CNN outputs *and intermediate outputs* with *predicate logic rules*, such as formalizations of "Limbs usually belong to a person". For this, the outputs of a CNN are interpreted as family of logical predicates that produce (fuzzy) truth values from image inputs. Rules are modelled to be truth functions accepting these truth values and yielding pixel- or region-wise *logical consistency scores*. Once the rule model is established, the evaluation works in a *self-supervised* manner, suitable for several debugging and verification use-cases: (1) identification of corner cases (examples with low score), (2) comparison of the logical consistency of different CNNs, and (3) monitoring of logical consistency as error indicator during runtime.

Besides the mentioned use-cases, this work investigates the benefits of using t-norm *fuzzy logic* [25] for modelling, and of *calibrating* the newly attached CNN outputs. Fuzzy logic promises to better leverage the continuity of CNN outputs [11]. And, for a mathematically sound interpretation as truth values, CNN outputs should be calibrated (cf. Sec. 3.3). Our contributions are:

(i) A novel, simple, extensible and scalable framework to construct a *truth value monitor* for logical constraints on semantic concepts (Sec. 3, cf. Fig. 1),
(ii) Development and comparison of *modeling options for formulating the monitor*, including different fuzzification methods (Secs. 3.1 and 4.2),
(iii) Analysis of benefits from *model calibration* and *fuzziness*,
(iv) Demonstration on two *state-of-the-art object detectors* and two safety relevant occlusion robustness rules for pedestrian detection (Sec. 4.2);
(v) We show that the framework can *uncover a substantial proportion of detection errors* with a single rule.

2 Related Work

Fuzzy Logic Integration into DNNs: Diligenti et al. [5] suggested to use fuzzy logic rules as loss functions for DNN training, relying on continuous t-norm fuzzy logic [25]. This inclusion of semantic prior knowledge to the training was shown to bring significant performance improvements for different types of rules [30,24,2], and was extended to several frameworks [23,2]. While we build upon this idea, the prior work requires all referenced concepts to be features in the DNN output, and does not guarantee that classifier outputs are calibrated.

Concept Embedding Analysis: The basic works from CA associate semantic concepts with linear combinations (*concept activation vectors*) of neurons [13] or CNN filters [7]. For this, small linear *concept models* are attached to CNN intermediate outputs. We use the successor [31] of [7] as part of our framework (cf. Sec. 3.2), but apply calibration and optimize the choice of loss towards this. Prior work proposes two semantic consistency metrics based on CA: correct similarity between concept encodings [7,31], and attribution of early-layer concepts to later-layer ones [13,34]. Both are restricted in the complexity of verifiable relations, whereas this work can deal with general predicate logic rules.

Calibration: For obtaining calibrated confidence estimates, we use model calibration. Common post-calibration methods [9] are computationally cheap, but approximate Bayesian learning is usually outperforming these methods [16]. However, common variational inference methods [15] are mostly restricted to a mean field approximation, disregarding the covariance, due to computational cost. Hence, we use a full covariance Laplace approximation [17], as this is computationally viable with our approach. Find a comprehensive overview in [1].

Other Monitoring Approaches: Self-supervised runtime monitoring of DNNs usually relies on uncertainty estimation, like [29], or consistency checks based on pixel-attribution or additional inputs, e.g. LiDAR. Uncertainty can detect outliers or proximity to decision boundaries [26], but no logical inconsistencies. Pixel-attribution can be used to check for spurious input attribution patterns of the model [8,18], or for attention not local to detected objects [4]. While this is useful for manual inspection, attention methods are restricted in the rules that can be checked, and inference of our method requires only a single forward pass.

3 Approach

Assume one wants to verify that an object detector respects the rule “Heads and limbs belong to a person” formalized to the predicate logic formula F

$$
\begin{aligned}
&& F(p) &:= \mathtt{IsBodyPart}(p) \to \mathtt{IsPartOfAPerson}(p) && (1)\\
\text{with} && \mathtt{IsPartOfAPerson}(p) &:= (\exists q \in P\colon \mathtt{Is}_{\mathsf{person}}(q) \wedge \mathtt{CloseBy}(p,q)) && (2)\\
\text{and} && \mathtt{IsBodyPart}(p) &:= \textstyle\bigvee_{\mathtt{b}\in\mathtt{BodyParts}} \mathtt{Is}_{\mathsf{b}}(p) && (3)
\end{aligned}
$$

with the free variable $p \in \text{Images} \times \text{PixelPositions}$ a pixel position in an image, and some pre-defined $\mathtt{BodyParts}$ predicates $\mathtt{Is}_{\mathsf{b}}$ for $b \in \mathtt{BodyParts}$. Such a formula with free variables can be viewed as a function on the variable values that outputs a single truth value for “Is the rule fulfilled”. It can be modeled as computational tree with the nodes being functions and logical operations, i.e. logical conjunctions $\vee, \wedge, \neg$; quantifiers $\forall, \exists$; and predicates like $\mathtt{Is}_{\mathsf{person}}$. The main idea for verifying a rule on CNN outputs is to interpret the CNN as family of predicates, i.e., functions that output truth values. We dissect the formula evaluation into the following steps (cf. Fig. 1):

1. **Obtain outputs of predicates** (calibrated, Sec. 3.3) and functions:
   - Predicates describing a CNN output, like $\mathtt{Is}_{\mathsf{person}}$: derived from CNN output.
   - Predicates describing internal knowledge of the CNN, like $\mathtt{Is}_{\mathsf{arm}}$: extracted from the activation maps via concept analysis (Sec. 3.2).
2. **Evaluate the residual computational tree of the formula** to obtain a final (fuzzy) truth value.

To leverage the knowledge encoded in the confidences of the DNN outputs, we also consider to *fuzzify* all logical operations (Sec. 3.1). It should be noted that all components (additional concept outputs, Sec. 3.2; calibration, Sec. 3.3; rule evaluation) only require few standard tensor operations, and thus produce negligible computational overhead.

3.1 Fuzzy Logic on DNN Outputs

For fuzzification of logical operations, we rely on the theory of t-norm fuzzy logic [25]. We first recapitulate needed basics thereof. Then, further modeling aspects for fuzzy rules are detailed, and the considered applications.

Basics of Fuzzy Logic Fuzzy logic generalizes standard predicate logic by allowing more than two truth values. This concerns the logical operations, which are the following: *Logical connectives NOT* ($\neg$), *AND* ($\wedge$), *OR* ($\vee$), and *implication* ($\to$) reduce one or several truth values to a single truth value. *Quantifiers all* ($\forall$), and *exists* ($\exists$) reduce a given domain of values (e.g. pixel positions) to a single truth value using a body formula. *Predicates* take symbol values (constants or instantiated variables), and return a single truth value. For example, predicates arising from DNN confidence outputs are inherently fuzzy. Intuitive

fuzzy logical connectives may be defined using a t-norm fuzzy logic [25]. The t-norm fuzzy logics from Tab. 1 represent a generating basis of all t-norm logics with continuous *AND* [25, sec. 2.3.1, p. 48]. Quantifiers can either be naturally expressed via the chosen logical connectives by $(\forall x \in X\colon F(x)) \coloneqq \big(\bigwedge_{x\in X} F(x)\big)$ and $(\exists x \in X\colon F(x)) \coloneqq \big(\bigvee_{x\in X} F(x)\big)$, or be implemented by a mean operation, as suggested in [6].
Notation: Predicates with binary output are <u>underlined</u>.

Table 1: The fuzzy connectives *AND* (*t-norm*), *OR* (*t-conorm*), *residuated implication* (R-implication) and *strong implication* (S-implication) defined by Boolean logic (Bool) and standard t-norm fuzzy logics Łukasiewicz (Ł), Goedel/Minimum (G), and Product/Goguen (P). Bool applies the standard Boolean operations to values binarized at a threshold $\mathtt{t} = \mathtt{t}_{\mathrm{Bool}}$ (default: 0.5).

Logic	Negation $\neg a$	Conjunction $a \wedge b$	Disjunction $a \vee b$	R-Implication $a \rightarrow_R b$	S-Implication $(\neg a) \vee b$
Ł	$1 - a$	$\max(0, a + b - 1)$	$\min(1, a + b)$	$\min(1, 1 - a + b)$	$\min(1, 1 - a + b)$
G	$1 - a$	$\min(a, b)$	$\max(a, b)$	1 if $a \leq b$ else b	$\max(1 - a, b)$
P	$1 - a$	$a \cdot b$	$a + b - a \cdot b$	$\min(1, \frac{b}{a})$	$1 - a + a \cdot b$
Bool	$\neg(a \geq \mathtt{t})$	$(a \geq \mathtt{t}) \wedge (b \geq \mathtt{t})$	$(a \geq \mathtt{t}) \vee (b \geq \mathtt{t})$	$(a < \mathtt{t}) \vee (b \geq \mathtt{t})$	$(a < \mathtt{t}) \vee (b \geq \mathtt{t})$

Formulation of Rules for Object Detection Object detection is subject to several fuzzy logic rules that are based, e.g., on laws, physical constraints, or statistical prior knowledge like typical anatomy. Simple examples are spatial rules, e.g., “Road users usually do not fly.”, or “A person located on top of a bike is usually a cyclist.”; and hierarchical rules, e.g., “Cars usually have wheels.”, or the example rule from Eq. (1). Note that the last one describes a safety relevant occlusion robustness: Infringements of the rule may indicate that the detector can fail in cases of high occlusion when only few body parts are visible. To leverage pixel-wise information obtained from the DNN, we propose a pixel-wise formulation of rules, as done in the example rule from Eq. (1). This allows to model the logical operations via few standard tensor operations on the mask tensors. Further details of our modeling approach are given in the following.

For **fuzzy connectives** we consider the three t-norm fuzzy logics detailed in Tab. 1. Note that the non-continuous R-implication $a \rightarrow_R b$ of product and Goedel logic is instable if truth values of both a and b are small [2]. As baseline we consider non-fuzzy Boolean logic, where $\wedge$ and $\vee$ translate to standard mask intersection and union. For this, all inputs to logical connectives must be binarized at a threshold $\mathtt{t}_{\mathrm{Bool}}$ (default 0.5). This may reduce memory consumption, but discards potentially valuable confidence information.

In [6], the **fuzzy quantifier** definition $\forall$ = mean was suggested as an intuitive and differentiable implementation. Note that, other than $\forall$ quantifiers derived from a fuzzy *OR*, mean gives rise to the following unequality for a subset $Q \subset P$ and a t-norm fuzzy implication:

$$(\forall p \in Q\colon F(p)) \neq (\forall p \in P\colon (p \in Q) \to F(p)) \tag{4}$$

The right hand formulation produces generally large values for small Q[3]. Thus, to keep intuition intact, we recommend the left hand formulation from (4) where applicable. When choosing the $\exists$ quantifier, it must be noted that for $\forall = \text{mean}$ the standard definition $(\exists x\colon F(x)) = (\neg\forall x\colon \neg F(x))$ for a formula body F would produce $\exists = \text{mean}$. This ignores high truth values when their proportion in the domain is too low. Thus, a $\exists$ defined from a fuzzy logic $\bigwedge$ is preferrable.

Typical **predicates** occurring in rules for object detection are unary concept predicates ("Does the concept apply to the location/region?") and spatial relations. Using examples from Eq. (1), we demonstrate different ways to model predicates: using predicted or ground truth (GT) bounding boxes (e.g., $\texttt{Is}_{\text{person}}, \texttt{Is}_{\text{GTPerson}}$), using concept segmentation masks (e.g., `IsBodyPart`), and manually (e.g., `CloseBy`).

Box information. Instance-wise bounding box information for an object class can be turned into a fuzzy semantic segmentation mask, e.g., with the following simplistic transformation: (1) Each bounding box is turned into a mask by setting pixels inside the bounding box to their bounding box objectness score value, and (2) if needed, the box masks are combined using pixel-wise logical *OR* of the respective fuzzy logic. For GT bounding boxes, the objectness score may be set constant to 1.

Segmentation masks. Segmentation outputs, e.g. from CA (here called *concept masks*, cf. Sec. 3.2), can serve as unary concept predicates, each pixel value being a truth value for a pixel position. The masks of different concepts may differ in resolution, e.g. those from CA have the resolution of CNN activation maps. To ensure compatible sizes, one can: (a) upscale masks to the common largest resolution, e.g. bilinearly [31], or (b) downscale masks to the common smallest resolution, e.g. using maxpooling[4].

`CloseBy`. We suggest two formulations of a spatial close-by relations on pixel coordinates a, b, a Gaussian radial distance function (with a low cut for memory efficiency), and a simpler binary one that checks L_1 proximity:

$$\texttt{CloseBy}_{\sigma,r}(a,b) := \exp(-d(a,b)^2/(2\sigma^2)) \tag{5}$$

$$\texttt{CloseBy}(p,q) := \left(\|p-q\|_1 \leq \lfloor\tfrac{1}{2}(\texttt{ksize}-1)\rfloor\right)\ , \tag{6}$$

with $d(a,b) := \|a-b\|_2$ if $\|a-b\|_1 \leq r$ for a window half size r, else 0. $\texttt{CloseBy}_\sigma$ becomes trivial if $\sigma = 0$, i.e., $\texttt{CloseBy}_0(a,b) = 1$ if $a \equiv b$ else 0. Note that `CloseBy` can be defined from a neighborhood definition $\texttt{Nbh}(\cdot)$ via set membership: $\texttt{CloseBy}(p,q) = (q \in \texttt{Nbh}(p))$ (cf. Eq. (4)). For Eq. (6) $\texttt{Nbh}(p)$ is a

[3] $(\forall p \in P\colon (p \in Q) \to F(p)) = \frac{\#P - \#Q}{\#P} + \frac{\#Q}{\#P}\,\text{mean}_{p\in Q} F(p)$

[4] Formally, maxpooling on a unary predicate $\texttt{Is}_\texttt{X}$ can be written as $\exists p' \in P'\colon \texttt{CloseBy}(p,p') \wedge \texttt{Is}_\texttt{X}(p')$ for a pixel $p \in P$ on some other resolution P', and binary $\texttt{CloseBy}(p,p') = (\|p-p'\|_1 \leq \lfloor\frac{1}{2}(\texttt{ksize}-1\rfloor))$ (cf. Eq. (8)).

square window of size `ksize` around p. A non-trivial spatial relation may help to mitigate segmentation mask resolution problems: Upscaling low resolution masks may cause positive regions to become too large. For example, in the rule from Eq. (1) body part masks may "shoot over" the person areas, leading to rule infringements at person area boundaries (cf. Fig. 2).

One **pattern** for rule formulation we consider here is that of a neighborhood prior for pixel-wise formulas F. The neighbor condition $\texttt{NbCond}_F(p, P)$ should down-weight $F(p)$ if it is spatially isolated (noise) within the region P. We suggest to score p isolated if none (Eq. (8)) or, alternatively, not all (Eq. (7)) of p's neighbors have a high $F(p)$ value, formally:

$$\texttt{NbCond}_F(p, P) = \forall q \in P \colon \texttt{CloseBy}(p, q) \to F(q) \underset{\texttt{CloseBy}}{\overset{\text{binary}}{=}} \forall q \in \texttt{Nbh}(p) \colon M(q) \tag{7}$$

$$\texttt{NbCond}_F(p, P) = F(p) \wedge (\exists q \in P \setminus \{p\} \colon \texttt{CloseBy}(p, q) \wedge F(q)) \tag{8}$$

If `CloseBy` is radial symmetric, Eq. (7) selects center points of radial peaks. For `CloseBy` defined via a square `Nbh` as in Eq. (6), and using Eq. (4) and $\forall = \text{mean}$, the neighborhood condition Eq. (7) can be efficiently implemented as a stride 1 average pooling. In this case, to not loose any peak, the kernel size `ksize` is best chosen to cover at least the minimum expected area of an interesting peak.

Fuzzy Rules for Verification Assume one is given a logical rule of the form $R(P) = \forall p \in P \colon F(p)$ where the variable P is an image or image region, and $p \in P$ means a pixel position in that region, and F a truth function on such valid pixels $p \in \text{ImageRegions} \times \text{PixelPositions}$, like in Eq. (1). For a concrete image, the pixel outputs of F together form a mask with values in $[0, 1]$ that are high where the formula is fulfilled, and low otherwise. R yields for an image or image region P a single truth value that can be interpreted as a logical consistency score for P. We suggest the following use-cases to utilize R and F for verification, given a test set T.

Corner Case Search: R can be evaluated for each image in T. Outliers with respect to the consistency score distribution may serve as pre-selection for manual analysis in order to identify root causes of illogical behavior.

Global Logical Consistency Score: The aggregation $\forall P \in T \colon R(P)$ of image-wise scores over T yields a global consistency score over T. The score allows, e.g., comparison of different networks trained on the same dataset, or comparison of different training datasets.

Monitoring: The pixel-wise truth scores given by F give rise to several runtime monitoring scenarios. For our example formula Eq. (1), a monitor would have the task to reveal cases of detector false negatives where some body part was found but no person predicted in the end. A *pixel-wise monitor* that observes whether pixels p have a suspiciously low truth value can be defined as

$$M(p) \coloneqq \neg F(p) \in [0, 1]. \tag{9}$$

Alarm is raised if an inconsistency bound is breached, i.e., $M(p) \geq \texttt{t}_{\text{px}}$. The benefit of the pixel-wise approach is that errors can be localized. Hence, only suspicious parts of the prediction can be discarded; or warnings can ignored for image regions that are not of interest, such as regions far off the vehicle trajectory in case of a pedestrian detector. A *region-wise monitor* for image regions P marks complete regions or images as spurious based on the pixel truth values. We suggest two definitions derived from M, a simple one and a smoothed one:

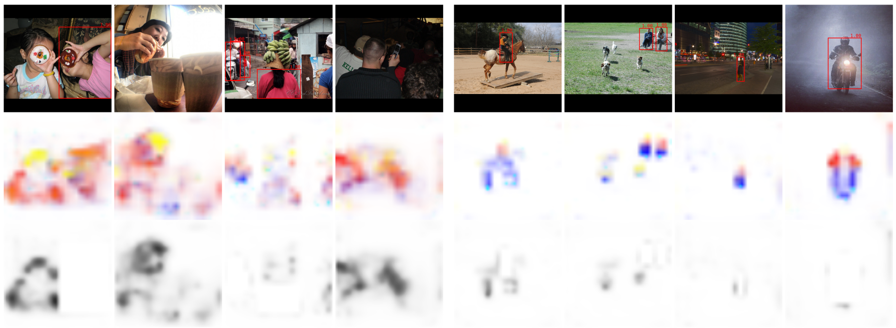
$$M_{\mathrm{reg}}^{\mathrm{simple}}(P) \coloneqq \exists p \in P \colon M(p) \overset{(*)}{=} \neg R(P) \tag{10}$$

$$M_{\mathrm{reg}}^{\mathrm{peaks}}(P) \coloneqq \exists p \in P \colon \mathtt{NbCond}_M(p, P) \overset{(**)}{=} \max_{p \in P} \mathrm{AvgPool2D}\left(\left(M(p)\right)_{p \in P}\right) \tag{11}$$

where equality $(*)$ holds if $(\exists x \colon f(x)) \equiv (\neg\forall x \colon \neg f(x))$, and equality $(**)$ for $\exists =$ max and the average pool formulation of `NbCond` from Eq. (7). The smoothing in Eq. (11) essentially adds a prior on the shape of peaks of monitor pixel values.In general, the choice of quantifier is important: Some are sensitive to outliers, i.e. isolated pixel alarms, and need the smoothing in Eq. (11) (e.g. $\exists = \max$ from Goedel logic), others are sensitive to the ratio between alarm and non-alarm pixels (e.g. mean), or are computationally expensive (e.g. $\exists = \bigvee$ for the probabilistic sum $\bigvee$ from Product logic).

3.2 Concept Analysis for Additional DNN Outputs

Standard object detection outputs usually include only a small set of object classes, e.g. person, and no sub-objects or other object attributes allowing for rich semantics. Our goal is to post-hoc enrich the output of a DNN by those concepts that are necessary to formulate given prior domain knowledge like Eq. (1) on DNN inputs and outputs. This should be done (1) without changing the trained DNN, (2) with little additional training and labeling effort, and (3) extensible, i.e. allowing to later add further concepts in case more domain knowledge is collected. For this, supervised post-hoc concept analysis constitutes a suitable candidate. *Concept (embedding) analysis* (CA) in general aims to associate in a simple way semantic concepts with elements in the intermediate output of a DNN, usually that of a layer [31]. Standard types of visual semantic concepts are texture, material, objects, object parts, or image-level concepts like the scene [3]. To achieve the association, supervised CA methods learn to predict information about the concept of interest, e.g. a binary segmentation mask, from the intermediate outputs. We here use the CA method suggested in [31], which was shown to work on CNN object detection. The concept models are attached to the output of one CNN layer. Each consists of a 1×1-convolution, followed for inference by a sigmoid normalization (and potentially upscaling). This method only introduces minor computational overhead. The output of the concept models are pixel-masks with confidence values in $[0, 1]$, which are trained to match ground truth binary segmentations of the concept. Due to our additional need for well-calibrated outputs (Sec. 3.3), we suggest to exchange the original self-balancing losses from [31] for standard binary cross-entropy loss (BCE) which has less impact on calibration.



Left to right thanks to David Quitoriano, j bizzie, CCFoodTravel.com, k.steudel, Fort Rucker, Donelle, Jacob Bøtter, Vir Nakai; © CC BY 2.0

Fig. 2: EffDet predicted boxes (*top*), concepts (leg, ankle, arm, wrist, eye; *middle row*), and logical consistency pixel values (*bottom*) for Eq. (11) (before max) with P logic cal setting from Tab. 4. *Left*: Monitor true positives; *right*: false positives.

3.3 Concept Model Calibration

DNN confidence outputs, including our post-hoc attached concept models, may be badly calibrated. This means that the confidence values do not well match the actual probability with which the prediction is correct [9]. When interpreting confidences as fuzzy truth values, bad calibration violates the fundamental assumption that the truth *meaning* monotonously increases with the truth *value*.

Therefore, we suggest to employ approximate Bayesian learning for calibrating our concept models (cf. Sec. 2). We compute a full covariance approximation of the posterior via Laplace approximate [17]. Despite the required Hessian approximation, we still maintain limited computational overhead, as we only apply it to a single layer (cf. Sec. 3.2) with few parameters. This method can be applied without changing or retraining the base model, making it applicable to any pretrained network.

4 Experiments

This section discusses evaluation results for concept model creation (Sec. 4.1) and the three verification use-cases (Sec. 4.2), as well as limitations of our approach (Sec. 4.3). The use-cases are tested on the example occlusion robustness rule from Eq. (1), and for body parts eye, arm, wrist, leg, ankle. The $\mathtt{Is}_{\text{person}}$ predicate is derived from person bounding box outputs of the two networks Mask R-CNN [10] (MR) with ResNet50 backbone, and EfficientDet D1 [33] (EffDet). The weights are trained on the MS COCO train2017 dataset [19], and are taken from PyTorch [27] respectively TensorFlow modelzoo [35] (cf. Tab. 2a for statistics). The ground truth for the body part concepts was derived from the MS COCO 2017 [19] keypoint annotations as done in [31]. Concept models were trained and calibrated on the MS COCO train2017 dataset, with a train/validation split of 4:1. Evaluation results both for the concept model performance and the fuzzy

Table 2: Statistics of used models (2a, 2c) and global consistency scores (2b)

(a) Statistics of used $\mathtt{Is}_{\mathsf{person}}$ predicates

Model	ECE	Pixel-Acc.	sIoU
MR [10]	0.044	0.956	0.810
EffDet [33]	0.071	0.929	0.684

(b) Global logical consistency scores for rule from Eq. (1)

	G	G cal	Ł	Ł cal	P	P cal
MR	0.994	0.994	0.992	0.991	0.992	0.991
EffDet	0.988	0.985	0.981	0.975	0.982	0.977

(c) Layer (L) and performance of the used BCE-trained concept models.

	MR			EffDet		
C	L	sIoU_b	sIoU	L	sIoU_b	sIoU
ankle	3	**0.210**	0.132	4	0.142	0.059
arm	4	**0.330**	0.254	6	0.305	0.242
eye	3	**0.436**	0.424	5	0.340	0.296
leg	3	**0.326**	0.264	5	0.310	0.246
wrist	4	**0.184**	0.119	6	0.150	0.069

Table 3: Calibration and performance of best concept models for MR (statistics averaged over concepts)

	ECE	MCE	sIoU_b	$\mathtt{t}_{\mathrm{sIoU}_\mathrm{b}}$	sIoU
BCE	0.001 ± 0.000	0.146 ± 0.079	0.297 ± 0.102	0.23 ± 0.06	0.218 ± 0.126
cal	**0.001 ± 0.000**	**0.140 ± 0.082**	**0.297 ± 0.102**	0.24 ± 0.06	0.218 ± 0.126
Dice	0.010 ± 0.008	0.621 ± 0.082	0.265 ± 0.079	0.59 ± 0.20	0.263 ± 0.081
cal	0.001 ± 0.001	0.451 ± 0.121	0.264 ± 0.080	0.52 ± 0.07	**0.263 ± 0.080**
bBCE	0.083 ± 0.047	0.879 ± 0.037	0.115 ± 0.052	0.95 ± 0.00	0.047 ± 0.031
cal	0.022 ± 0.012	0.772 ± 0.071	0.228 ± 0.054	0.91 ± 0.05	0.056 ± 0.032

logic are collected on the MS COCO val2017 dataset. The performance of $\mathtt{Is}_{\mathsf{person}}$ and the concept model segmentations is evaluated as the set intersection over union (sIoU) [7] between the binary ground truth concept masks $(m_i)_i$ and the predicted and upscaled concept masks $(m_i^{\mathrm{pr}})_i$:

$$\mathrm{sIoU}\left((m_i)_i, (m_i^{\mathrm{pr}})_i\right) = \frac{\sum_i \sum m_i \cap (m_i^{\mathrm{pr}} > \mathtt{t}_{\mathrm{sIoU}})}{\sum_i \sum m_i \cup (m_i^{\mathrm{pr}} > 0.5)} \;. \tag{12}$$

Notation: sIoU_b is measured at optimal $\mathtt{t}_{\mathrm{sIoU}}$ (determined on a the validation set), sIoU at 0.5.

4.1 Concept Analysis and Calibration

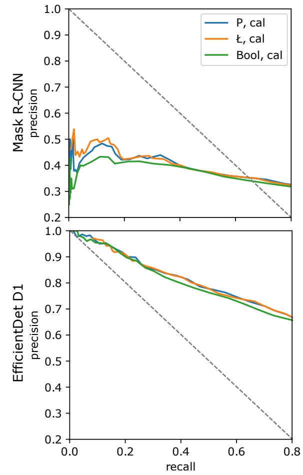
Literature suggests several losses for concept analysis, including balanced Dice loss [31,28], and a class-balanced binary cross-entropy (bBCE) [7]. We here also consider standard BCE. Tab. 3 compares calibration and performance of these losses in terms of sIoU, expected calibration error (ECE) and maximum calibration error (MCE) [9]. Results show that Laplace-calibrated BCE outperforms all

Table 4: Performance comparison of different formula formulations for image-level monitoring of rule Eq. (1). Metrics: precision-recall (PR) curve, area under ROC curve (AUC), F1 at $\mathtt{t}_{\text{reg}}^{\text{peaks}} = 0.5$, and each $\overline{\text{F1}}$, $\overline{\text{F0.1}}$, and $\overline{\text{F10}}$ for the $\mathtt{t}_{\text{reg}}^{\text{peaks}}$ yielding the theoretically best corresponding score value.

(a) Image-level results for formula Eq. (1)

		AUC	$\text{F1}_{0.5}$	$\overline{\text{F1}}$	$\overline{\text{F0.1}}$	$\overline{\text{F10}}$
Mask R-CNN	Bool	0.619	0.146	0.455	0.421	0.975
	Bool cal	0.620	0.144	0.458	0.423	**0.975**
	Ł	0.632	0.282	0.462	0.484	0.975
	Ł cal	0.632	**0.295**	0.461	**0.492**	0.975
	P	0.632	0.244	**0.466**	0.467	0.975
	P cal	**0.632**	0.243	0.465	0.470	0.975
EfficientDet D1	Bool	0.671	0.421	0.749	0.899	0.993
	Bool cal	0.676	0.411	0.749	0.895	0.993
	Ł	0.690	0.557	0.752	0.895	0.993
	Ł cal	0.695	**0.592**	**0.755**	**0.899**	0.993
	P	0.689	0.500	0.751	0.897	0.993
	P cal	**0.696**	0.523	0.753	0.896	0.993

(b) Upper left part of PR curves



other variants, *given a tuned sIoU threshold.* Interestingly, balanced losses (Dice, bBCE) suffer from substantial miscalibration. This can be partially countered with the Laplace method, but results are still worse. Calibrating the BCE result only shows minor advantages, therefore we re-evaluated the calibration effects for the full approach (cf. Tab. 4). Improvements are still small, but consistent (cf. Sec. 4.2). The sIoU performance of the BCE-trained concept models used in later experiments is given in Tab. 2c. Representations of the larger Mask R-CNN model consistently outperform those of EfficientDet D1 (cf. Sec. 4.2).

4.2 Logical Consistency Monitor Applications

If not stated otherwise (cf. Sec. 4.2), experiments were conducted for the S-implication formulation of the formula from Eq. (1), with trivial $\mathtt{CloseBy}_{\sigma}$ from Eq. (5), and the arithmetic mean for ∀, and max for ∃ (this is $\bigvee_x F(x)$ in Goedel logic). The Goedel exists quantifier was chosen as it is the most conservative one of those defined by fuzzy logics, i.e. yields the smallest values [25, Lemma 2.19]. This is a desirable property for safety evaluation with the example rule. Concept masks are bilinearly upscaled to ensure comparability amongst CNNs. For non-fuzzy logic (Tab. 1), the concept masks are binarized at the same thresholds as the monitor outputs, i.e. $\mathtt{t}_{\text{Bool}} = \mathtt{t}_{\text{px}}, \mathtt{t}^{\cdot}_{\text{reg}}$ for $\mathtt{t}_{\text{px}}, \mathtt{t}^{\cdot}_{\text{reg}}$ defined below. A pixel p is a *false negative pixel* ($\underline{\mathtt{Is}_{\mathsf{FN}}}(p)$), if $\mathtt{Is}_{\mathsf{FN}}(p) \geq \mathtt{t}_{\mathsf{ped}} = 0.5$ for $\mathtt{Is}_{\mathsf{FN}}(p) := \neg\mathtt{Is}_{\mathsf{person}}(p) \wedge \mathtt{Is}_{\mathsf{GTPerson}}(p)$. Since the logical consistency monitor Eq. (1) shall highlight false negatives of the detector, $\underline{\mathtt{Is}_{\mathsf{FN}}}$ is taken as pixel-wise ground

Table 5: Performance of monitors by binarization threshold $\mathtt{t}_{\text{px}}$, resp. $\mathtt{t}_{\text{reg}}$.

(a) Pixel-level ROC AUC for rule from Eq. (1)

	MR	EffDet
Bool	0.829	0.840
Bool cal	0.830	0.843
Ł	0.833	0.843
Ł cal	0.833	**0.847**
P	0.833	0.843
P cal	**0.834**	0.847

(b) Prediction-level performance for rule Eq. (13)

		AUC	$\mathrm{F1}_{0.5}$	$\overline{\mathrm{F1}}$	$\overline{\mathrm{F0.1}}$	$\overline{\mathrm{F10}}$
MR	Bool cal	0.929	0.123	0.230	0.145	0.854
	Ł cal	0.927	**0.136**	**0.251**	**0.172**	0.844
	P cal	**0.930**	0.132	0.250	0.172	**0.847**
EffDet	Bool cal	**0.986**	0.024	**0.118**	**0.115**	**0.480**
	Ł cal	0.889	**0.028**	0.082	0.050	0.464
	P cal	0.985	0.028	0.080	0.048	0.452

truth for the monitor. For evaluation of the pixel-level monitor we binarize the outputs of M (Eq. (9)) at threshold $\mathtt{t}_{\text{px}}$. Note that our formula will only highlight those parts of false negative areas for which the selected body parts were predicted (e.g. not the torso). Hence, pixel-level recall is naturally comparatively low. To investigate suitability for finding images with *some* logical inconsistency, we have a look at the region monitor formulations $M_{\text{reg}}^{\text{simple}}$ (Eq. (10)) and $M_{\text{reg}}^{\text{peaks}}$ (Eq. (11)) for complete images as regions. The ground truth $\mathtt{GT}_{\text{reg}}^{\cdot}(P)$ for an image P is defined from the pixel-wise $\underline{\mathtt{Is}_{\mathtt{FN}}}$ values using the same formula as for deriving $M_{\text{reg}}^{\cdot}$ from M. For evaluation, the predicted and ground truth image scores are binarized using thresholds $\mathtt{t}_{\text{reg}}^{\cdot}$ respectively $\mathtt{t}_{\text{GTreg}}^{\cdot} = 0.5$. The average pooling kernel size $\mathtt{ksize}$ of both $\mathtt{GT}_{\text{reg}}^{\text{peaks}}$ and $M_{\text{reg}}^{\text{peaks}}$ is set to 33 pixels[5], which approximately coincides with the typical height of a head in the test data (cf. size statistics in [31]) at acceptable memory consumption. This setting means, if half of the pixels within any 33×33 pixels window are false negatives, the image is marked ground truth faulty. In our setting, 27.7 % of the test images are marked faulty. For the Bool baseline, $\mathtt{t}_{\text{Bool}}$ was set to $\mathtt{t}_{\text{px}}$.

Comparing Monitor Formulations Compared were F1 score, precision, recall and true negative rate of different pixel- (Eq. (9)) and simple image-level (Eq. (10)) monitor formulations at $\mathtt{t}_{\text{px}} = \mathtt{t}_{\text{reg}}^{\text{simple}} = 0.5$. Aspects of variation were: fuzzy logics from Tab. 1; standard R-implication versus strong implication; adding calibration; adding denoising of masks, i.e. setting concept mask values < 0.005 to 0; bilinear upscaling versus maxpool downscaling of concept masks; and the trivial $\mathtt{CloseBy}_{\sigma,r}$ from Eq. (5) versus (only for upscaling) the non-trivial one with $\sigma \approx 2.77$ (truth value of 0.8 at distance of 4px) at $r = 12$px (window of 25×25px, cutting at truth values below 0.1). Preliminary experiments showed that R-implication produces many false positives at small input truth values, even with denoising. Slight performance improvements were achieved by: fuzziness; calibration for S-implication; the memory-intense $\mathtt{CloseBy}_{\sigma\neq 0}$ on

[5] Preliminary experiments showed: For $\mathtt{ksize} = 2^i + 1, i \leq 5$, differences between $\mathtt{ksize}$ for $\underline{\mathtt{GT}}_{\text{reg}}$ and M_{reg} only had little influence on the results.

pixel-level; and downscaling for Ł and P logic, at the cost of truth mask resolution. Therefore, we used the initially described setup for all later experiments.

Self-supervised Error Monitoring Results Compared were Bool, Ł, and P logic (cf. Tab. 1), each with and without calibration (cal), both for pixel-level (Eq. (9), Tab. 5a) and image-level (Eq. (11), Tab. 4) monitoring. Results show that *(1)* good error identification performance can be achieved, uncovering a substantial amount of detector errors (cf. contribution (v)); *(2)* fuzzy evaluations slightly but consistently outperform the non-fuzzy ones and substantially outperform them for precision-biased metrics like $\overline{\text{F0.1}}$ score (cf. contribution (iii)),; and *(3)* calibrated slightly outperform non-calibrated versions (cf. contribution (iii)). The image-level results (Tab. 4) further reveal that *(4)* acceptable precision-recall balances can be achieved for EffDet (e.g. recall of ≥ 0.98 at precision ≥ 0.60, or precision of ≥ 0.95 at recall ≥ 0.1). This can be seen on the precision-recall curves, as well as from results for F0.1 score (favoring precision) and F10 score (favoring recall) in Tab. 4. Another finding was that *(5)* the optimal $\mathtt{t}_{\text{reg}}^{\text{peaks}}$ alarm threshold consistently diverged from the default value 0.5. This suggests that tuning of the threshold on a validation set, similar to $\mathtt{t}_{\text{sIoU}}$, may boost performance values. Without tuning, fuzziness brings a considerable benefit (cf. $\text{F1}_{0.5}$ in Tab. 4). This is especially of interest for cases when threshold tuning is not viable, e.g., because of tuning data availability or bad generalization of the tuned threshold.

We also considered monitoring of the prediction-level formula for “A person should have a body part” which should reveal false positives of the object detector. With $\mathtt{Is}_{\text{person,i}}$ refering to the ith prediction for P:

$$F(P) = (\exists p \in P\colon \mathtt{Is}_{\text{person,i}}(p)) \to \big(\exists p \in P\colon \mathtt{Is}_{\text{person,i}}(p) \land \mathtt{IsBodyPart}(p)\big)\ . \qquad (13)$$

A predicted box was defined as false positive (i.e., monitor GT positive) if its person class score was greater 0.5 and it was covered by less than 20 % by the union of GT boxes, producing positive rates of 0.03 % (EffDet) and 1.60 % (MR). Evaluation was conducted using the P logic cal setting from Tab. 4. Here, also, fuzzy approaches outperformed the Boolean one without threshold tuning (Tab. 5b). While performance was low for very small predicted boxes due to the static `ksize` setting and strong class imbalance, it achieved formidable F-scores on larger sized boxes, especially for recall balanced settings which are relevant, e.g. , for sample selection for manual analysis.

Comparing Logical Consistency of Models The better performing and calibrated (cf. Tab. 2a), and ca. three times larger MR model also gives rise to better concept models (Tab. 2c), and achieves higher global logical consistency scores (Tab. 2b). with respect to the the rule from Eq. (1). This aligns with the findings in Tabs. 4 and 5a, where fewer—but still a considerable amount—of the MR false negatives could be recovered using a logical consistency monitor. On the other hand, most errors of the smaller EffDet were recovered using our simple example rule, attesting the model some simple logical gaps. This promises high potential for improvement of EffDet performance on person detection by fine-tuning towards better logical consistency, or by post-processing based on the monitor outputs.

Corner Case Analysis Finally, we manually inspected the samples with smallest mean pixel-wise logical consistency score for rule Eq. (1)[6]. Here, we exclude pixels p regarded trivial (e.g. background), i.e. such with low alarm score $M(p) \leq 10^{-3}$. See Fig. 2 for example outputs. The analysis revealed for Mask R-CNN: *(1)* persons segmented by (self-)occlusions may get too small bounding boxes, and *(2)* person features seem to be confused with that of animals and puppets. And *(3)* EffDet produces false negatives if the face is only slightly occluded, e.g. by perspective, objects or the image boundary. The found symbolic error modes directly allow to define data augmentation strategies, e.g. adding persons segmented by occlusion. To automate corner case selection from new samples, a threshold $\mathtt{t}^{\cdot}_{\mathrm{reg}}$ for the logical inconsistency score could be fine-tuned on a validation set. Altogether, the fuzzy rule evaluation helps in finding semantic error modes.

4.3 Limitations

Result quality depends on that of the framework ingredients: the concept models and the rules. Bad convergence, a bias in the concept samples, or simply insufficient CNN representations may lead to erroneous concept masks. And the prior knowledge rules are naturally susceptible to human bias (e.g. anatomical assumptions), potentially leading to unfair safety guarantees. Besides that, inconsistencies with those pre-defined rules are not guaranteed to accurately correspond to errors. Our approach potentially considerably reduces the amount of test data needed to uncover issues and it enables self-supervision, but, thus, naturally inherits all limitations of data based verification methods, like test data representativity. And lastly, parts of our simple initial modeling approach may be improved. This concerns the simplistic CNN output transformation, GT definition, and the neighborhood condition from Eq. (7) which currently does not take into account object sizes, so still cannot prevent all occurrences of "over-shooting".

5 Conclusion and Outlook

This work presents a simple, yet flexible, post-hoc, self-supervised method to verify and monitor outputs of trained CNNs regarding compliance with symbolic domain knowledge rules. It is the first that requires no architectural constraints or changes to already trained CNNs, and permits later extension of the rule base.

[6] Precisely, images P_{img} with highest $M_{\mathrm{reg}}^{\mathrm{simple}}(P)$ value from Eq. (10) for the region $P = \{p \in P_{\mathrm{img}} \mid M(p) \geq 10^{-3}\} \subset P_{\mathrm{img}}$.

The method comes with little computational overhead, and allows to leverage fuzziness and calibration for slight performance benefits. We showed how to use it to identify a considerable proportion of detection errors. Furthermore, the used logical consistency scores can help in finding error modes, and to directly compare CNNs. For example, in comparison to EfficientDet D1, Mask R-CNN shows considerably better representations and conformity with a simple occlusion robustness rule for person detection.

We are looking forward to see results on further models, tasks, and example rules, including investigation of how to model temporal aspects, and how to advance our initial modeling suggestions. Also, it seems promising to study how much a trained model could be improved by self-supervised fine-tuning via the logical consistency scores.

Acknowledgments

The research leading to these results is partly funded by the German Federal Ministry for Economic Affairs and Energy within the project “KI Absicherung – Safe AI for Automated Driving”. The authors would like to thank the consortium for the successful cooperation.

References

1. Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., Fieguth, P., Cao, X., Khosravi, A., Acharya, U.R., Makarenkov, V., Nahavandi, S.: A review of uncertainty quantification in deep learning: Techniques, applications and challenges. Information Fusion **76**, 243–297 (2021). https://doi.org/https://doi.org/10.1016/j.inffus.2021.05.008 3
2. Badreddine, S., d’Avila Garcez, A., Serafini, L., Spranger, M.: Logic tensor networks. CoRR **abs/2012.13635** (Jan 2021), https://arxiv.org/abs/2012.13635 3, 5
3. Bau, D., Zhou, B., Khosla, A., Oliva, A., Torralba, A.: Network dissection: Quantifying interpretability of deep visual representations. In: Proc. 2017 IEEE Conf. Comput. Vision and Pattern Recognition. pp. 3319–3327. IEEE Computer Society (2017). https://doi.org/10.1109/CVPR.2017.354 2, 8
4. Cheng, C.H., Nührenberg, G., Huang, C.H., Ruess, H., Yasuoka, H.: Towards dependability metrics for neural networks. In: 16th ACM/IEEE Int. Conf. Formal Methods and Models for System Design. pp. 43–46. IEEE (2018). https://doi.org/10.1109/MEMCOD.2018.8556962 3
5. Diligenti, M., Gori, M., Saccà, C.: Semantic-based regularization for learning and inference. Artificial Intelligence **244**, 143–165 (Mar 2017). https://doi.org/10.1016/j.artint.2015.08.011 3
6. Donadello, I., Serafini, L., d’Avila Garcez, A.S.: Logic tensor networks for semantic image interpretation. In: Proc. 26th Int. Joint Conf. Artificial Intelligence. pp. 1596–1602. ijcai.org (2017). https://doi.org/10.24963/ijcai.2017/221 2, 5, 22, 23
7. Fong, R., Vedaldi, A.: Net2Vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. In: Proc. 2018 IEEE Conf. Comput. Vision and Pattern Recognition. pp. 8730–8738. IEEE Computer Society (2018). https://doi.org/10.1109/CVPR.2018.00910 2, 3, 10

8. Fong, R.C., Vedaldi, A.: Interpretable explanations of black boxes by meaningful perturbation. In: Proc. 2017 IEEE Int. Conf. on Comput. Vision. pp. 3449–3457. IEEE Computer Society (2017). https://doi.org/10.1109/ICCV.2017.371 3
9. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017. Proceedings of Machine Learning Research, vol. 70, pp. 1321–1330. PMLR (2017), http://proceedings.mlr.press/v70/guo17a.html 3, 9, 10
10. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: 2017 IEEE International Conference on Computer Vision (ICCV). pp. 2980–2988 (Oct 2017). https://doi.org/10.1109/ICCV.2017.322 9, 10
11. Hüllermeier, E.: Does machine learning need fuzzy logic? Fuzzy Sets and Systems **281**, 292–299 (Dec 2015). https://doi.org/10.1016/j.fss.2015.09.001 2
12. Kazhdan, D., Dimanov, B., Terre, H.A., Jamnik, M., Liò, P., Weller, A.: Is disentanglement all you need? comparing concept-based & disentanglement approaches. arXiv:2104.06917 [cs] (Apr 2021), http://arxiv.org/abs/2104.06917 2
13. Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., Sayres, R.: Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In: Proc. 35th Int. Conf. Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 2668–2677. PMLR (Jul 2018), http://proceedings.mlr.press/v80/kim18d.html 2, 3
14. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Proc. 3rd Int. Conf. Learning Representations (2015), http://arxiv.org/abs/1412.6980 20
15. Kingma, D.P., Salimans, T., Welling, M.: Variational dropout and the local reparameterization trick. In: Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 28. Curran Associates, Inc. (2015), https://proceedings.neurips.cc/paper/2015/file/bc7316929fe1545bf0b98d114ee3ecb8-Paper.pdf 3
16. Krishnan, R., Tickoo, O.: Improving model calibration with accuracy versus uncertainty optimization. In: Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual (2020), https://proceedings.neurips.cc/paper/2020/hash/d3d9446802a44259755d38e6d163e820-Abstract.html 3
17. Kristiadi, A., Hein, M., Hennig, P.: Being bayesian, even just a bit, fixes overconfidence in relu networks. In: Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event. Proceedings of Machine Learning Research, vol. 119, pp. 5436–5446. PMLR (2020), http://proceedings.mlr.press/v119/kristiadi20a.html 3, 9
18. Lapuschkin, S., Wäldchen, S., Binder, A., Montavon, G., Samek, W., Müller, K.R.: Unmasking Clever Hans predictors and assessing what machines really learn. Nature Communications **10**(1), 1096 (Mar 2019). https://doi.org/10.1038/s41467-019-08987-4 3
19. Lin, T.Y., Maire, M., Belongie, S.J., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: Proc. 13th European Conf. Computer Vision - Part V. Lecture Notes in Computer Science, vol. 8693, pp. 740–755. Springer International Publishing (2014). https://doi.org/10.1007/978-3-319-10602-1_48 9, 20
20. Liu, C., Arnon, T., Lazarus, C., Strong, C., Barrett, C., Kochenderfer, M.J.: Algorithms for verifying deep neural networks. Foundations and Trends® in Optimization **4**(3-4), 244–404 (Feb 2021). https://doi.org/10.1561/2400000035 2

21. Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., Pietikäinen, M.: Deep learning for generic object detection: A survey. International Journal of Computer Vision **128**(2), 261–318 (Feb 2020). https://doi.org/10.1007/s11263-019-01247-4 2
22. MacKay, D.J.C.: The evidence framework applied to classification networks. Neural Comput. **4**(5), 720–736 (1992). https://doi.org/10.1162/neco.1992.4.5.720 20, 21
23. Marra, G., Giannini, F., Diligenti, M., Gori, M.: LYRICS: A general interface layer to integrate logic inference and deep learning. In: Proc. European Conf. Machine Learning and Principles of Knowledge Discovery 2019 (Sep 2019), http://arxiv.org/abs/1903.07534 3
24. Nandwani, Y., Pathak, A., Mausam, Singla, P.: A primal dual formulation for deep learning with constraints. In: Advances in Neural Information Processing Systems 32. pp. 12157–12168. Curran Associates, Inc. (2019), http://papers.nips.cc/paper/9385-a-primal-dual-formulation-for-deep-learning-with-constraints.pdf 3
25. Novák, V., Perfilieva, I., Mockor, J.: Mathematical Principles of Fuzzy Logic. The Springer International Series in Engineering and Computer Science, Springer US (Jan 1999). https://doi.org/10.1007/978-1-4615-5217-8 1, 2, 3, 4, 5, 11
26. Perello-Nieto, M., Filho, T.D.M.E.S., Kull, M., Flach, P.: Background check: A general technique to build more reliable and versatile classifiers. In: 2016 IEEE 16th International Conference on Data Mining (ICDM). pp. 1143–1148. IEEE (Dec 2016). https://doi.org/10.1109/ICDM.2016.0150 3
27. pytorch: Torchvision (Oct 2021), https://github.com/pytorch/vision/v0.10.0 9
28. Rabold, J., Schwalbe, G., Schmid, U.: Expressive explanations of DNNs by combining concept analysis with ILP. In: KI 2020: Advances in Artificial Intelligence. pp. 148–162. Lecture Notes in Computer Science, Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-58285-2_11 10
29. Rottmann, M., Colling, P., Paul Hack, T., Chan, R., Hüger, F., Schlicht, P., Gottschalk, H.: Prediction error meta classification in semantic segmentation: Detection via aggregated dispersion measures of softmax probabilities. In: Proc. 2020 Int. Joint Conf. Neural Networks. pp. 1–9 (Jul 2020). https://doi.org/10.1109/IJCNN48605.2020.9206659 3
30. Roychowdhury, S., Diligenti, M., Gori, M.: Image classification using deep learning and prior knowledge. In: Workshops of the 32nd AAAI Conf. Artificial Intelligence. AAAI Workshops, vol. WS-18, pp. 336–343. AAAI Press (Jun 2018), https://aaai.org/ocs/index.php/WS/AAAIW18/paper/view/16575 3
31. Schwalbe, G.: Verification of size invariance in DNN activations using concept embeddings. In: Artificial Intelligence Applications and Innovations. pp. 374–386. IFIP Advances in Information and Communication Technology, Springer International Publishing (2021). https://doi.org/10.1007/978-3-030-79150-6_30 2, 3, 6, 8, 9, 10, 12, 20
32. Schwalbe, G., Schels, M.: A survey on methods for the safety assurance of machine learning based systems. In: Proc. 10th European Congress Embedded Real Time Software and Systems (Jan 2020), https://hal.archives-ouvertes.fr/hal-02442819 1
33. Tan, M., Pang, R., Le, Q.V.: EfficientDet: Scalable and efficient object detection. In: Proc. 2020 IEEE/CVF Conf. Comput. Vision and Pattern Recognition. pp. 10781–10790 (2020) 9, 10

34. Wang, D., Cui, X., Wang, Z.J.: CHAIN: Concept-harmonized hierarchical inference interpretation of deep convolutional neural networks. CoRR **abs/2002.01660** (2020), https://arxiv.org/abs/2002.01660 3
35. Wightman, R.: EfficientDet (PyTorch) (Oct 2021), https://github.com/rwightman/efficientdet-pytorch/tree/75e16c2f 9

Appendix Overview

This supplemental material collects additional results and details for the experiments described in the main paper. Specifically, it gives further evidence for the following claims made in the main paper:

- Appendix B: *The main formula for false negative detection used in the main paper can be implemented as parallelizable windowed operation* under some modeling constraints.
- Appendix C: This section gives detailed results of the preliminary experiments on the comparison of monitor formulations referenced in the main paper. These show that *the simple rule formulation used in the main paper is a valid choice* for the later experiments in the main paper.
- Appendix D: *Fuzziness improves the error-detection performance of a logical consistency monitor when the hyperparameter for the binarization threshold cannot be tuned* on a validation set.
- Appendix E: For the peak-concentrated image-level monitor formulation, *the kernel size hyperparameter of the monitor can be chosen independently of that of the ground truth.*

Appendix A gives details on the chosen hyperparameter values for the experiments to ensure reducibility.

Table 6: Overview of hyperparameters and their default values used in the formula calculation (here and in the main paper).

Name	Default	Function
$\mathtt{t}_{\mathtt{sIoU}}$	0.5	Threshold used to determine the set intersection over union performance
$\mathtt{t}_{\mathtt{denoise}}$	0.005	Denoise low values of $\mathtt{IsBodyPart}(\bullet)$ masks
$\mathtt{t}_{\mathtt{ped}}$	0.5	Binarizing threshold for $\mathtt{Is}_{\mathsf{person}}(\bullet)$ masks
$\mathtt{t}_{\mathtt{px}}$	0.5	Binarizing threshold for pixel-level monitor output masks
$\dot{\mathtt{t}}_{\mathtt{reg}}$	0.5	Binarizing threshold for image-level monitor outputs
$\mathtt{t}_{\mathtt{sIoU}}$	0.5	Binarizing threshold for pixel-level monitor outputs for measuring sIoU
$\mathtt{t}_{\mathtt{Bool}}$	0.5	Binarizing threshold for all mask in non-fuzzy logic
ksize	33	Width of the quadratic kernel for defining a (binary) pixel neighborhood

A Implementation Notes: Hyperparameter Choices

In the following, hyperparameter choices for our experiments are detailed. An overview on further hyperparameters introduced in this supplemental material and the main paper can be found in Tab. 6.

Settings applicable to all experiments are:

- An image size of 400 × 400 pixels was used for Mask R-CNN, 640 × 640 pixels for EfficientDet D1.
- Images were first zero-padded to square size, then resized to the desired image size (keeping the aspect ratio).
- For metrics working on binary pixel values, output masks (after transformation of bounding boxes to masks) are thresholded at a value of 0.5.

Concept Data The body part segmentation masks were generated from keypoint annotations as proposed in [31]. Point concepts (eye, wrist, ankle) are added as filled circle of white pixels. Limb concepts (arm, leg) get filled circles at edges and joints, and lines connecting them where skeletal connections exist. The concepts consist of the following keypoints:

- arm: left/right shoulder, ellbow, wrist
- leg: left/right hip, knee, ankle
- eye, wrist, ankle: corresponding left/right keypoints

Circle diameter and line width were set to 5 % of the estimated person body height in pixels (using estimation algorithm from [31]). The body height defaults to the bounding box height if it cannot be estimated from keypoint links of the annotation. Keypoints annotated as occluded are treated as not present.

Calibration and Formula Evaluations The measurement of calibration and performance metrics on the main CNN, as well as the evaluations of the fuzzy logic formulas, used the following settings:

- Evaluation batch size: 64;
- Data split: All images from the MS COCO dataset [19] val2017 split are used for evaluation.

Concept Model Training The training and evaluation of the concept models used the following settings in accordance with [31]:

- Metric and loss calculation: Concept model outputs (after convolution) are first bilinearly upscaled to match the ground truth resolution, then normalized. For BCE-losses, normalization is skipped and the loss is calculated in logit space.
- Loss for second stage training of the Laplace approximation parameters [22]: binary cross-entropy (in logit space);
- Data split: The training and validation data sets are taken from the MS COCO dataset [19] train2017 split, the test data from the val2017 split. For each concept, only those samples are included that contain any positive concept mask pixel. The train2017 samples for a concept are randomly split into training and validation set, at a ratio of 4:1.
- Optimizer and learning rate: Adam [14] from PyTorch version 1.9.0 implementation, with a learning rate of 0.001, and default beta values of 0.9 and 0.999; no weight decay;

– Batch sizes: 8 for training, 64 for validation, and 6 for the second stage training of the Laplace approximation parameters [22];
– Early stopping: Concept models are trained for at most 7 epochs. Training is stopped early after an epoch if the validation loss decreases less than 0.001 for each in three successive epochs.

B Implementation Notes: Parallelization of Example Formula

Consider the mask operation $\texttt{CloseToA}_{\texttt{b}}$ that accepts a mask Q and returns a mask P with each pixel p holding the truth value for *"p is close to a pixel q in Q at which concept* `b` *is present"*. This is used in the example formula from the main paper as will become clear from Def. 2. This section shows that under some conditions this **$\texttt{CloseToA}_{\texttt{b}}$ can be implemented as a windowed operation** on the mask Q, similar to a convolution or pooling operation. Furthermore, an upper bound is given for the minimal window size (cf. Prop. 1). The **windowing allows parallelization, and thus a potentially massive speedup**. Note, however, that this may be quite memory intensive, as it is not guaranteed that the operation can be represented as sparse matrix operations.

Before formally defining $\texttt{CloseToA}_{\texttt{b}}$ in Def. 2, some notation for pixel indices is introduced that allows to interpret them as coordinates (with standard distance measures like L_1).

Definition 1. Let h, h', w, w' be natural numbers (mask heights and widths), and let $Q \in [0,1]^{h' \times w'}, P \in [0,1]^{h \times w}$ be 2D masks. A mask coordinate system here denotes an injective mapping of dimensions (i_h, i_w) in a mask P to pixel positions in 2D space $\mathbb{R}^2$. By default map the dimension (i_h, i_w) to the point $(i_h + 0.5, i_w + 0.5)$, i.e. a pixel has a width of 1×1 and pixel coordinates refer to pixel centers. Let $p \in \mathbb{R}^2$ be a pixel position.

(a) *Pixel positions in a mask:* Denote by $p \in P$ that p is a pixel in the mask P.
(b) *Pixel values of a mask:* Given that $p \in P$, denote by $P(p)$ the value at the pixel position p in the mask P.
(c) *Sub-masks of a mask:* A mask $W \in [0,1]^{h \times w}$ is a window or sub-mask in $P \in [0,1]^{h' \times w'}$, denoted $W \subset P$, if $h \leq h', w \leq w'$, and there is a mask coordinate system for W and P such that for all pixel positions $p \in W$ holds $p \in P$ and $W(p) = P(p)$.

Definition 2. Let h, h', w, w' be integers and $Q \in [0,1]^{h' \times w'}$ be a mask. Assume a logic is given with predicates $\texttt{Is}_{\texttt{b}}$ and $\texttt{CloseBy}$ of arity one respectively two, which both operate on pixel positions. We define the following mask operation:

$$\texttt{CloseToA}_{\texttt{b}} \colon [0,1]^{h' \times w'} \to [0,1]^{h \times w} \tag{14}$$

$$P(p) \coloneqq (\exists q \in Q \colon \texttt{Is}_{\texttt{b}}(q) \wedge \texttt{CloseBy}(p, q)) \tag{15}$$

for $p \in P \coloneqq \texttt{CloseToA}_{\texttt{b}}(Q)$.

This is used in the example formula from the main paper. (Note that P and Q need not have the same resolution.)

The following proposition states that, under some conditions, $\mathtt{CloseToA_b}$ can be implemented as a windowed operation. And, if the $\mathtt{CloseBy}$ predicate becomes zero at some L_1 distance r, the window size of the windowed operation may be chosen $2r + 1$ (or smaller). This means, $2r + 1$ is the maximum window size necessary to ensure an exact implementation.

Proposition 1. *Consider a (fuzzy) logic, and* $\mathit{Is_b}$, $\mathit{CloseBy}$, *and the masks* Q *and* $P = \mathit{CloseToA_b}(Q)$ *as in Def. 2. Assume the following:*

(a) $\exists$ *is either defined as an arithmetic* mean [6], *or defined using the logical* OR, *i.e. for a domain* X *and a formula* f *holds* $\exists x \in X \colon f(x) := \bigvee_{x\in X} f(x)$*.*
(b) There is a square integer window size $2r + 1$ *such that* $\mathit{CloseBy}(p, q) = 0$ *if* $\|p-q\|_1 > r$ *for any pixel positions* p, q*. This is e.g. the case for the suggested maxpooling or Gaussian* $\mathit{CloseBy}$ *from the main paper.*

Then, $P(p)$ *for a pixel position* $p \in P$ *at most depends on the pixel values in the window* $W_p \subset Q$ *of size* $(2r + 1) \times (2r + 1)$ *centered at* p*, i.e.*

$$\mathit{CloseToA_b}(Q)(p) = (\exists q \in W_p \colon \mathit{Is_b}(q) \wedge \mathit{CloseBy}(p, q)) \tag{16}$$

Proof. Let $f(p, q) := \mathit{Is_b}(q) \wedge \mathit{CloseBy}(p, q)$*. Note that*
(i) $f(p, q) = 0$ *if* $\mathit{CloseBy}(p, q) = 0$*, and*
(ii) $q \in W_p$ *if* $f(p, q) \neq 0$*, because for* $q \in Q \setminus W_p$ *holds:*

- $\|p - q\|_1 > r$ *by definition of the window* W_p*, thus*
- $\mathit{CloseBy}(p, q) = 0$ *by (b), and so*
- $f(p, q) = 0$ *by (*).*

For $\exists$ *defined via logical* OR *according to assumption (a), the formula becomes*

$$P(p) = \exists q \in Q \colon f(p, q) \overset{(a)}{=} \bigvee_{q\in Q} f(p, q) \tag{17}$$

$$\overset{(i)}{=} 0 \vee \bigvee_{\substack{q\in Q\\ f(p,q)\neq 0}} f(p, q) \overset{(ii)}{=} 0 \vee \bigvee_{\substack{q\in W_p\\ f(p,q)\neq 0}} f(p, q) \tag{18}$$

$$\overset{(a)}{=} \exists q \in W_p \colon f(p, q) \tag{19}$$

For $\exists$ *defined via arithmetic mean, note that*

$$\exists q \in Q \colon f(p, q) = \frac{1}{\#Q} \sum_{q\in Q} f(p, q) \; . \tag{20}$$

Replacing all occurrences of $\bigvee_{q\in Q}$ *with* $\frac{1}{\#Q} \sum_{q\in Q}$ *in the above calculation finalizes the proof.*

Remark 1. Note that the notation, Def. 2, and Prop. 1 all can easily be generalized to masks $Q \in [0,1]^{n_1 \times \dots \times n_d}$ of arbitrary dimension $d \geq 1$, with pixel positions in $\mathbb{R}^d$. So, the rule formulation and efficient implementation can be applied not only to 2D masks, but also e.g. 3D masks consisting of several channels or frames.

Remark 2. The following special cases further simplify the implementation:

(a) *Trivial case:* In case $r = 1$, as in the experiments in the main paper, $\texttt{CloseToA}_\texttt{b}(Q)(p) = \texttt{Is}_\texttt{b}(p) \wedge \texttt{CloseBy}(p,p)$.

(b) *Maxpooling case:* For $\exists = \max$ and binary

$$\texttt{CloseBy}(p,q) \coloneqq (\|p-q\|_1 \leq r) \tag{21}$$

($\texttt{CloseBy}(p,q)$ is one if q lies in a square window around p, else 0) $\texttt{CloseToA}_\texttt{b}$ becomes a simple maxpooling operation on Q with kernel size $2r+1$. The stride of the pooling operation is such that the size $h' \times w'$ of Q is reduced to the size $h \times w$ of P. This is used for the downscaling operation which was suggested in the main paper and is further investigated in Appendix C.

(c) *Convolutional case:* For $\exists = $ mean [6] and Product logic, $\texttt{CloseToA}_\texttt{b}$ becomes a simple linear convolution followed by applying a pixel-wise factor $\frac{1}{(h' \cdot w')^2}$. Kernel weights for the kernel window with center p are defined by $\texttt{CloseBy}(p,\cdot)$.

C Comparison of Monitor Formulations

This section collects further considerations and the results for directly comparing different monitor formulations. The following variations are considered (cf. Tab. 7):

Fuzzy logics from the main paper: Łukasiewicz, Product, and Goedel fuzzy logic; and non-fuzzy predicate logic on concept masks binarized at $\texttt{t}_{\text{Bool}}$.

Implication style: Sstrong implication (S-implies) or residuated implication (R-implies).

Denoising: With or without setting concept mask values lower than $\texttt{t}_{\text{denoise}} = 0.005$ to 0. This can be modeled as pixel-wise operation $(\texttt{Is}_\texttt{b}(p) \geq \texttt{t}_{\text{denoise}}) \wedge \texttt{Is}_\texttt{b}(p)$ for pixels p and concept b with concept mask $\texttt{Is}_\texttt{b}(\cdot)$.

Choice of `CloseBy`*:* Considered are the Gaussian $\texttt{CloseBy}_{\sigma,r}$ from the main paper, different mask scaling options, and trivial $\texttt{CloseBy}_{\sigma=0,r=1}$. The following three settings are compared here:

- bilinear upscaling with trivial `CloseBy`
- bilinear upscaling with non-trivial $\texttt{CloseBy}_{\sigma,r}$, $\sigma \approx 2.77$ and $r = 12$px (*in visualizations shorted to* `CoveredBy`)
- downscaling using maxpooling with trivial `CloseBy`

Calibration: With or without post-hoc Laplace calibration of concept model outputs.

On the Denoising The *denoising* intends to reduce the instable cases of pixels for R-implies. These are the cases where both values of $\texttt{IsBodyPart}$ and $\texttt{Is}_{\textsf{person}}$ are very low, but $\texttt{IsBodyPart}$ is larger than $\texttt{Is}_{\textsf{person}}$. For Product and Goedel logic this causes $\texttt{IsBodyPart}(p) \to_R \texttt{Is}_{\textsf{person}}(p)$ to jump from 1 to a smaller value (cf. R-implies definitions from the main paper).

On the $\texttt{CloseBy}$ *Choice* Either approach for *implementing* $\texttt{CloseBy}$—the fuzzy but memory-intense Gaussian $\texttt{CloseBy}_{\sigma,r}$, and the non-fuzzy maxpooling based one—has the effect of enlarging the area of positive truth values. This is useful if concept masks have different resolutions and might not match precisely at the boundaries. This leads to low truth values in boundary regions if a low-resolution mask and a higher-resolution mask are scaled and combined in a *AND* or *implies* operation.

Results Results for the following monitor formulations can be found below (both using binarization thresholds $\texttt{t}_{\text{px}} = 0.5 = \texttt{t}_{\text{reg}}^{\text{simple}}$):

- Pixel-wise monitor M: Fig. 3 for Mask R-CNN, Fig. 4 for EfficientDet D1.
- Image-level monitor $M_{\text{reg}}^{\text{simple}}$: Fig. 5 for Mask R-CNN, Fig. 6 for EfficientDet D1.

Furthermore, global scores for different monitor formulations are compared in Fig. 7. It must be noted that the pixel-level monitor is expected to have low recall: Only such detector false negative pixels should be highlighted that are inconsistent with the considered rule, which can only be those that are part of one of the considered body parts (no torso, bounding box corners, etc.). The following can be seen from the visualizations:

(1) *Fuzziness*: For S-implication, **non-fuzzy logic performs consistently slightly worse than fuzzy logic.**

(2) *Implication style* and *denoising*: Despite denoising, **R-implication is still much too sensitive** and causes many false positives (cf. low true negative rates, and suspiciously high recall on pixel-level).

(3) $\texttt{CloseBy}$: **Using a non-trivial $\texttt{CloseBy}$ gives slight (downscaling) up to significant (non-trivial $\texttt{CloseBy}_{\sigma,r}$) performance benefits**, as can be seen e.g. from the F1 scores. However, $\texttt{CloseBy}_{\sigma,r}$ either cannot be parallelized or requires lots of memory for the windowed operation, and downscaling significantly reduces (localization) information. Thus, in the main paper only the trivial $\texttt{CloseBy}$ is used, and noise problems smoothed using the neighborhood condition for the image-level monitor (cf. Appendix E).

(4) *Calibration* on pixel-level seems to have a **negative effect on precision and a positive one on recall, resulting in comparable F1 scores**. On image-level, calibration seems to have the least effect of all aspects of variation on the monitor performance. **For the global scores calibration has the visible effect of decreasing logical consistency for any rule formulation**.

(5) The comparison of the lack of *global logical consistency* in Fig. 7 shows that **the choice of rule formulation has no influence on the trend of the global consistency**. Any choice of variation attests EfficientDet D1 the clearly worse score.

Table 7: Overview on the different aspects of variation tested for the formulation of the formula from the main paper (cf. Appendix C)

Aspect	Description
Implication	Whether S- or R-implication is used
`CloseBy`, Scaling	Formulation of the `CloseBy`: upscaling of small masks with either non- or trivial $\mathtt{CloseBy}_{\sigma,r}$; or max-downscaling of large masks
Calibration	Whether Laplace calibration is applied to the concept models during inference
Denoising	Whether truth values in concept masks $\mathtt{Is}_{\mathrm{b}}(\cdot)$ that are below $\mathtt{t}_{\mathrm{denoise}}$ are set to 0

D Benefits of Fuzziness without Monitor Threshold Tuning

Fig. 8 shows the F1 scores of the pixel- and the image-level monitors considered in the main paper, plotted by the binarization threshold $\mathtt{t}_{\mathrm{px}}$ respectively $\mathtt{t}_{\mathrm{reg}}^{\mathrm{peaks}}$. Concretely compared are the non-fuzzy baseline with Łukasiewicz and Product t-norm fuzzy logic, with calibrated and non-calibrated concept models. The following can be seen:

(1) **When deviating from the optimal threshold, fuzzy formulations clearly outperform the non-fuzzy alternative.**

(2) The optimal thresholds, and corresponding optimal $\overline{\mathrm{F1}}$ scores on the test set nearly coincide for all formulations.

(3) In all cases, F1-by-threshold curve has a unique maximum, and the threshold for this optimal F1 score is far from the default of 0.5. This **highlights the potential performance increase by tuning the hyperparameter $\mathtt{t}_{\mathbf{px}}$ respectively $\mathtt{t}_{\mathbf{reg}}^{\cdot}$.**

(4) It can be seen a consistent slight performance benefit of calibration.

E Choice of `ksize` for Image-level Monitor

The peak-concentrated image-level monitor $M_{\mathrm{reg}}^{\mathrm{peaks}}$ used in experiments in the main paper has the additional hyperparameter of the average pooling kernel size `ksize` for the image-level monitor neighborhood condition. As both the monitor $M_{\mathrm{reg}}^{\mathrm{peaks}}$ and $\mathtt{GT}_{\mathrm{reg}}^{\mathrm{peaks}}$ were defined using the neighborhood condition, each has a tunable `ksize` parameter, $\mathtt{ksize}_{M}$ and $\mathtt{ksize}_{\mathrm{GT}}$.

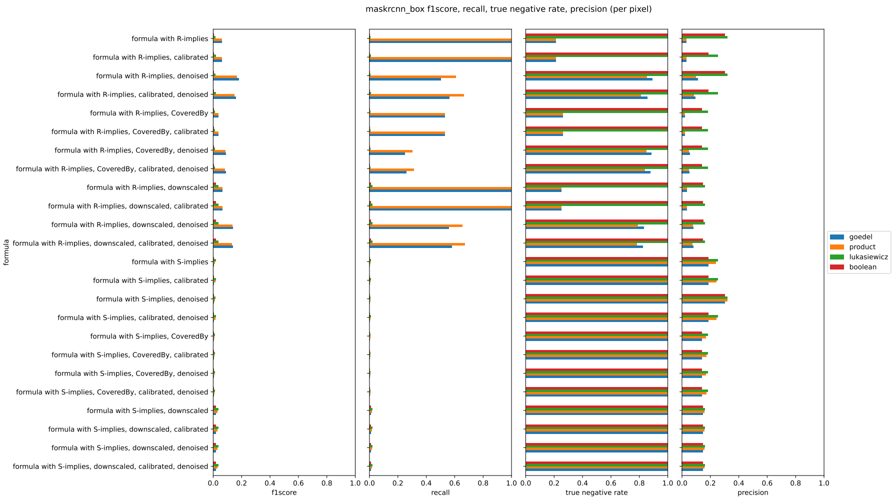


Fig. 3: **Mask R-CNN**, **pixel-level** monitor: Visual comparison of performance metrics for different monitor formulations (cf. Tab. 7). *Left to right:* F1 score, recall, precision, and true negative rate.

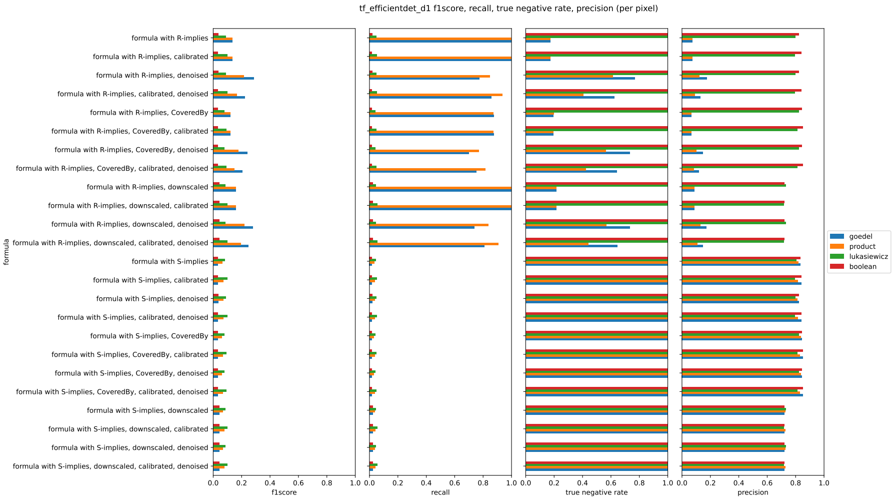


Fig. 4: **EfficientDet D1**, **pixel-level** monitor: Visual comparison of performance metrics for different monitor formulations (cf. Tab. 7). *Left to right:* F1 score, recall, precision, and true negative rate.

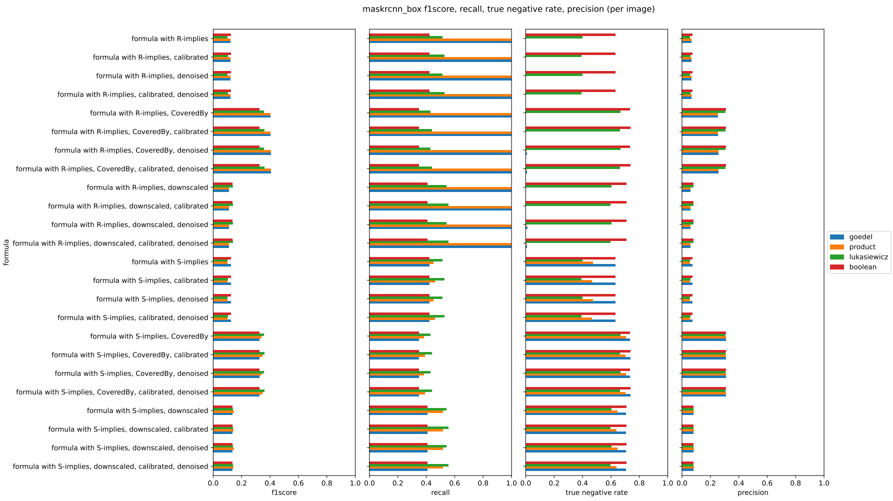


Fig. 5: **Mask R-CNN**, **image-level** monitor: Visual comparison of performance metrics for different monitor formulations (cf. Tab. 7). *Left to right:* F1 score, recall, precision, and true negative rate.

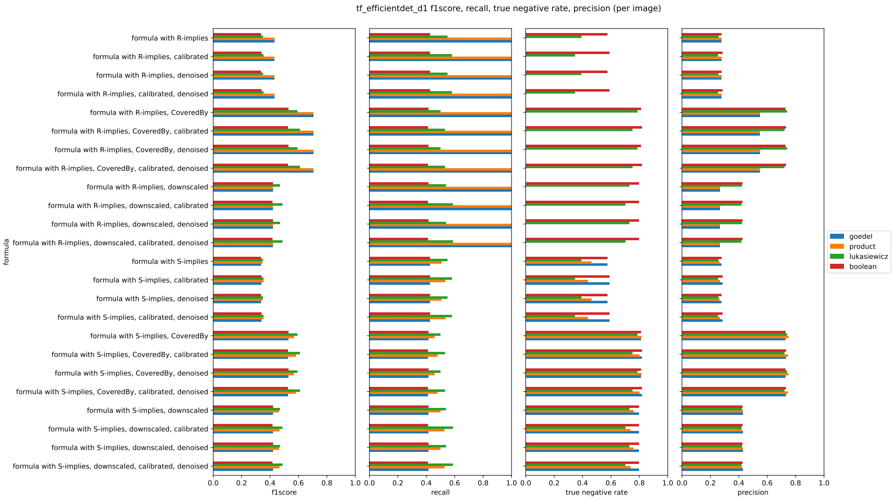


Fig. 6: **EfficientDet D1**, **image-level** monitor: Visual comparison of performance metrics for different monitor formulations (cf. Tab. 7). *Left to right:* F1 score, recall, precision, and true negative rate.

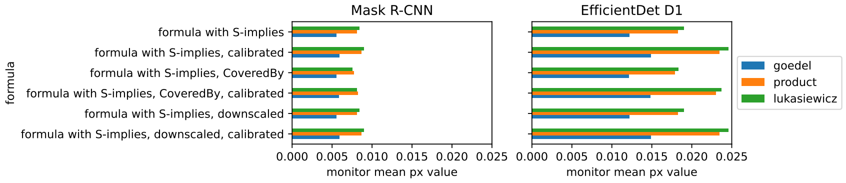
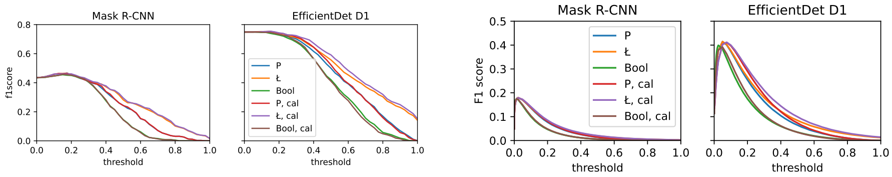


Fig. 7: Comparison of values of (1 − global score) for different monitor formulations. This calculates as the mean of the pixel-wise monitor values over the complete dataset.



(a) Per-image F1 score by threshold

(b) Per-pixel F1 score by threshold

Fig. 8: F1 score by $\texttt{t}_{\text{reg}}^{\text{peaks}}$ (image-level, Fig. 8a) and $\texttt{t}_{\text{px}}$ (pixel-level, Fig. 8b) for the versions of the simple rule formulations compared in the main paper.

Ground Truth Positive Rates $\texttt{ksize}_{\text{GT}}$ influences the ground truth positive rate of images in the dataset, as smaller kernel window size values are more sensible to smaller (possibly noisy) peaks. Tab. 8 shows the positive rates for exponentially increasing $\texttt{ksize}_{\text{GT}}$ and two $\texttt{t}_{\text{GTreg}}^{\text{peaks}}$ settings. What positive rate is obtained for which $\texttt{ksize}$ setting depends on the distribution of alarm area sizes in the test dataset. For example, for $\texttt{t}_{\text{GTreg}}^{\text{peaks}}$ the positive rate approximately linearly increases with the kernel size. Note that for $\texttt{ksize}_{\text{GT}}$, this is identical with the ground truth for the simple monitor formulation $M_{\text{reg}}^{\text{simple}}$.

Dependence of Monitor Results on $\texttt{ksize}_{GT}$ The following results show that larger kernel sizes of the monitor (prediction) bring some benefit, regardless of the ground truth kernel size. Investigated were odd kernel sizes $2^i + 1, 2 \leq i \leq 5$ and kernel size of 1 as baseline. Here evaluated were the neighborhood (nb) image-level formulation of the monitor from the main paper, for the rule formulation with S-implication and calibration. Compared were different Łukasiewicz and non-fuzzy logic, different kernel sizes $\texttt{ksize}_M, \texttt{ksize}_{\text{GT}}$, and binarization thresholds $\texttt{t}_{\text{reg}}^{\text{peaks}}, \texttt{t}_{\text{GTreg}}^{\text{peaks}}$. Results are shown

- for fixed middle sized $\texttt{ksize}_{\text{GT}} = 9$ in Fig. 9,
- for $\texttt{ksize}_M$ fixed in Fig. 11,
- for $\texttt{ksize}_M = \texttt{ksize}_{\text{GT}}$ in Fig. 10.

The sampling rate of the plotted curves increases towards the value boundaries of $\texttt{t}_{\text{reg}}^{\text{peaks}}$. The results show:

(1) **Varying $\texttt{ksize}_{\mathrm{GT}}$ for a fixed $\texttt{ksize}_M$, and the other way round, both have little influence on the performance.**

(2) **Larger $\texttt{ksize}_M$ have slight performance benefits** for common fixed $\texttt{ksize}_{\mathrm{GT}}$. This may indicate a better noise robustness of larger kernel sizes. Larger kernel sizes can differentiate the quality of truth peaks in a larger range of peak sizes, boosting the influence of large interesting areas while decreasing that of small ones. Therefore, in the main paper we used the largest considered kernel size of 33, which also nicely fits typical body part proportions in the test dataset.

(3) Fuzziness shows a consistent performance benefit (cf. precision-recall curves).

(4) Increasing $\texttt{ksize}_{\mathrm{GT}}$ decreases the monitor precision. This is expected, as a higher $\texttt{ksize}_{\mathrm{GT}}$ also decreases the positive rate (cf. Tab. 8).

Table 8: Percentage of the 2693 MS COCO val2017 images with positive image-level ground truth annotations $\underline{\texttt{GT}}^{\mathrm{peaks}}_{\mathrm{reg}}$ for different choices of $\texttt{ksize}_{\mathrm{GT}}$ and $\texttt{t}^{\mathrm{peaks}}_{\mathrm{GTreg}}$.

$\texttt{ksize}_{\mathrm{GT}}$	$\texttt{t}^{\mathrm{peaks}}_{\mathrm{GTreg}}$	GT Pos. [%]
1	0.500	0.950
5	0.500	0.844
9	0.500	0.703
9	0.700	0.599
17	0.500	0.491
33	0.500	0.277

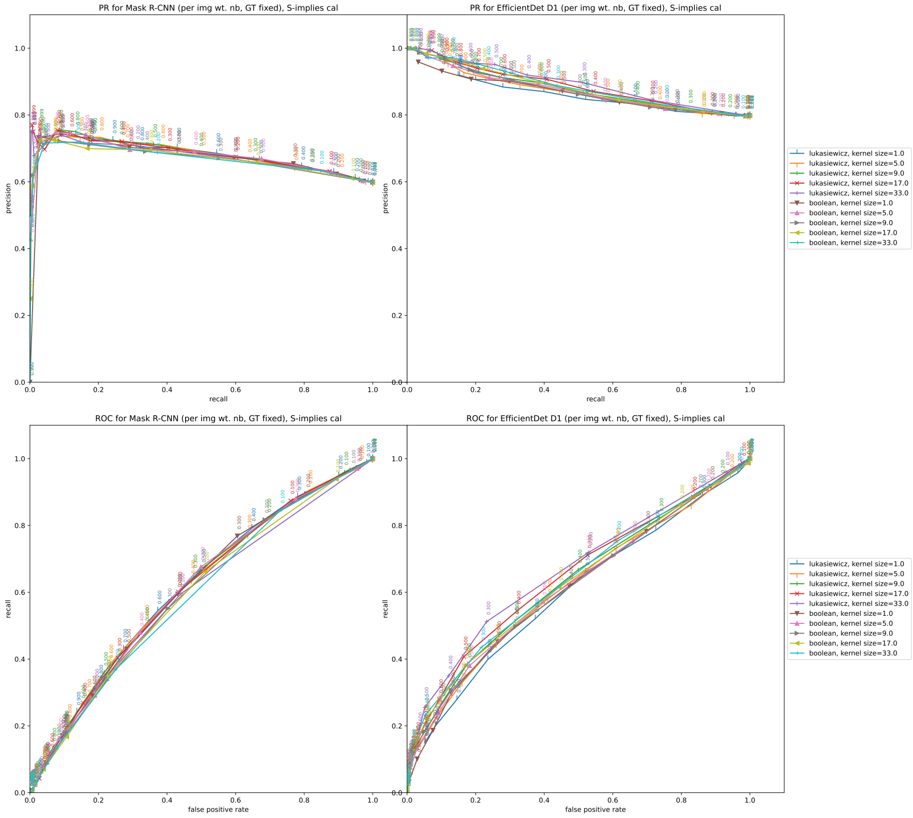


Fig. 9: Results for fixed $\texttt{ksize}_{\mathrm{GT}} = 9$ and $\mathtt{t}^{\mathrm{peaks}}_{\mathrm{GTreg}} = 0.7$. *Top:* Precision-recall curves, *bottom:* ROC curves over $\mathtt{t}^{\mathrm{peaks}}_{\mathrm{reg}}$, *left:* Mask R-CNN, *right:* EfficientDet D1.

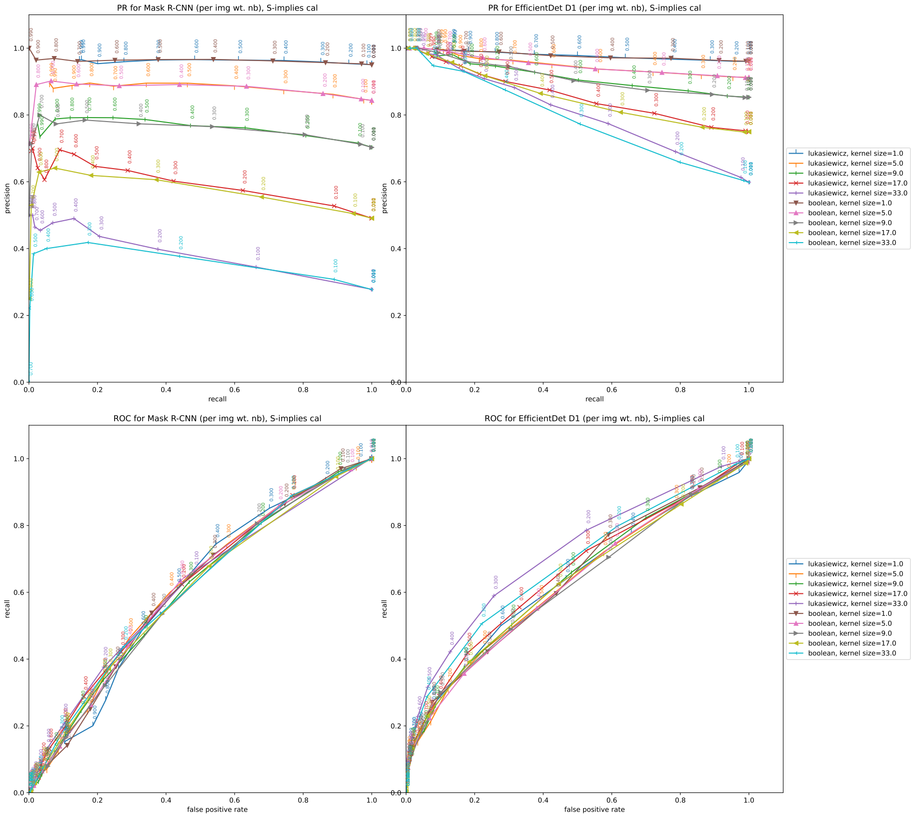


Fig. 10: Results for $\texttt{ksize}_M = \texttt{ksize}_{\mathrm{GT}}$ and $\texttt{t}^{\mathrm{peaks}}_{\mathrm{GTreg}} = 0.5$. *Top:* Precision-recall curves, *bottom:* ROC curves over $\texttt{t}^{\mathrm{peaks}}_{\mathrm{reg}}$, *left:* Mask R-CNN, *right:* EfficientDet D1.

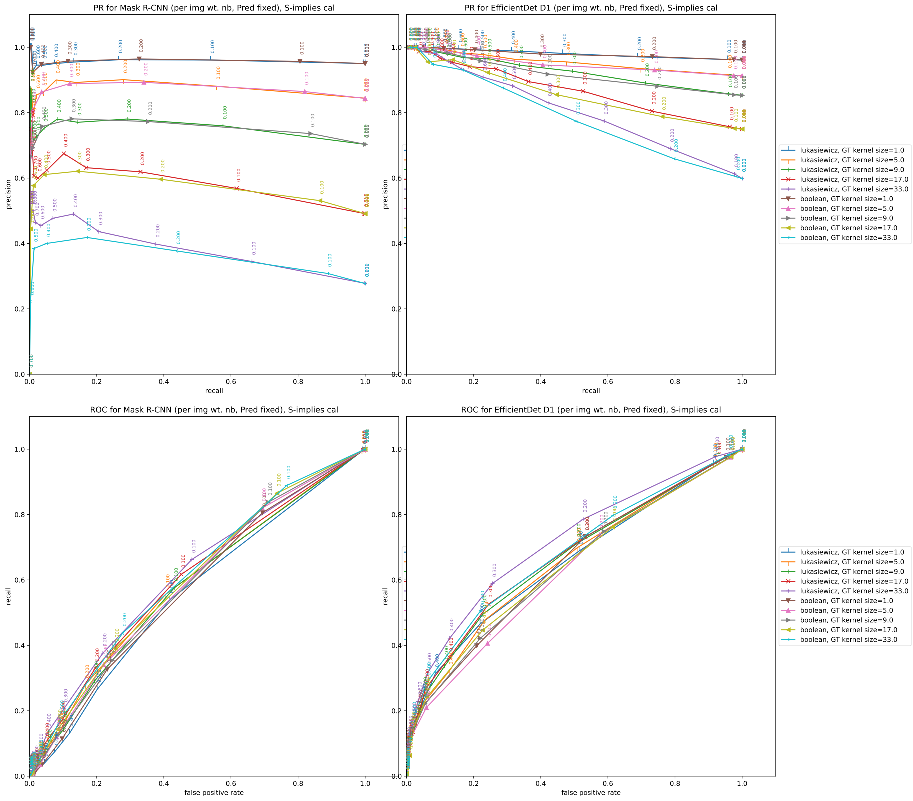


Fig. 11: Results for fixed $\mathtt{ksize}_M = 33$ and $\mathtt{t}^{\text{peaks}}_{\text{GTreg}} = 0.5$. *Top:* Precision-recall curves, *bottom:* ROC curves over $\mathtt{t}^{\text{peaks}}_{\text{reg}}$, *left:* Mask R-CNN, *right:* EfficientDet D1.

A Survey on Methods for the Safety Assurance of Machine Learning Based Systems

Gesina Schwalbe and Martin Schels

Artificial Intelligence and Robotics Laboratory,
Continental AG, Regensburg, Germany
{forename.lastname}@continental-corporation.com

Abstract. Methods for safety assurance suggested by the ISO 26262 automotive functional safety standard are not sufficient for applications based on machine learning (ML). We provide a structured, certification oriented overview on available methods supporting the safety argumentation of a ML based system. It is sorted into life-cycle phases, and maturity of the approach as well as applicability to different ML types are collected. From this we deduce current open challenges: powerful solvers, inclusion of expert knowledge, validation of data representativity and model diversity, and model introspection with provable guarantees.

Keywords: functional safety, life-cycle, ISO 26262, machine learning, explainable AI

1 Introduction

Machine learning is currently rapidly finding its way into the domain of safety critical applications in the scope of the ISO 26262 automotive functional safety standard. Examples are computer vision systems or trajectory planning for autonomous driving (AD) [25, 39]. The reason for this advance of machine learned models is its applicability to inherently complex, little understood problems merely via sample data. Unfortunately, this benefit is often accompanied by a mostly black-box character and high complexity of the final model in use, rendering conventional methods for safety assurance insufficient or inapplicable. For example, inductive/deductive inspection and extensive interface/unit testing are inapplicable due to model complexity. The same holds for the "proven in use" argument due to the problem complexity. The latter is reasoned in [21], which historically derives problems for risk assessment of AI. A mathematical deduction can be found in [39, sec. 2.2].

In [37], a complete applicability assessment of ISO 26262 methods to deep neural networks (DNNs) was conducted. They also found that design guidelines, model inspection techniques, formal verification and training data assessment techniques are required. Such new methods for verification and validation were loosely collected e.g. in the surveys [44, p. 36–37] and [19], to which we would like to refer the reader for a more extensive method catalogue. In this paper we proceed with the next step: Sort available methods (respectively method categories) into the main stages of the standard development life-cycle, and from this systematically identify open challenges. This paper provides:

- a *short overview* of ML specific approaches and methods supporting a safety case, *structured* by the phases of the life-cycle (see Tab. 1 for a summary). The focus will be on DNNs as in [37] in the context of AD.
- evaluation of their applicability to different ML types.
- identification of *open challenges* for safety argumentation.

We claim that this novel view on ML development methodologies from a certification perspective is a key to enable ML in safety critical applications. The view should be seen as a top-down approach towards a safety argumentation strategy.

2 ML Specific Safety Argumentation

There are important differences between the traditional approach to the development and certification of safety critical software and the data driven paradigm of machine learning. This is addressed in the following, aligned to a typical life-cycle. The details of the methods, their applicability and open challenges are summarized in Table 1.

Our life-cycle skeleton is based on the ISO 26262 part 6 process [1, Figure 1] for the development of automotive software-enabled components. The main idea of the V-model suggested there is a feedback loop consisting of the phases requirements gathering, (model) development, and finally verification and validation (and integration, which is not considered here). During *requirements engineering* the system architecture is constructed up to smallest functional units, and for each the use-case is specified, including intended operational environment and behavior. Both are guided by verifiable safety goals identified in an inductive hazard and risk analysis. In the succeeding *development* phase, the model design and implementation on hardware and software side is conducted. This is succeeded by intense *verification* and *validation* based on the given requirements. The results are used to determine, whether the model (verification) or the requirements (validation) need to be updated.

In ML, the model design is data driven and automatized, and the implementation consists of classically implemented interpreters for the generated model definitions. We here solely concentrate on the new aspects introduced by the design, verification and validation of (ML) models.

2.1 Requirements Engineering

Available expert knowledge and experience should be compiled into the use-case, system and function requirements formulation, and best be formulated in specialized KPIs. With data driven development of (possibly non-robust) black boxes it becomes practically hard to do this for the little available domain knowledge. As categories of safety requirements for NNs we identified: data representativity, such as

- **scenario coverage** (for further metric suggestions see collection in [8]) applied to an input space ontology such as developed in [5] for the context of autonomous driving;
- **robustness** of the algorithm, such as adversarial robustness [e.g. definition in 22, sec. 6] of NNs;
- **fault tolerance** of the system, as could be increased by runtime monitoring and traditional model redundancy [2, (E) 7.4.12] which requires a model diversity measure;
- **safety related performance** requirements for the black-box, i.e. specialized performance measures such as detection performance for occluded objects [8];
- **plausibility** of the algorithm behavior, such as sensible intermediate steps, or modeling quality of domain specific physical rules, like gravity, translation invariance, or legal ones like collected for the definition of responsibility sensitive safety for AD trajectory planning in [39]; and finally
- requirements arising from **experience** about algorithm specific faults, such as can be found in the AD specific collection of problems and faults in [28].

These cornerstones need to be respected during requirements gathering. Data representativity and model diversity measures are open challenges since methods are scarce and untested.

2.2 Development

Prevention of faults by proper design choices during development are an important building block for a safety case. Some ML specific quality criteria are collected below. Since meaningful KPIs are necessary in order to conduct quantitative analyses, suggestions on such are given as well.

Design based on experience All general design choices not specified in the requirements need to be reasoned, best by experts in the field. Choices include at least the training objective, the model type, the training method including early stopping criteria, the micro design—e.g. activation functions, topology, and loss function for NNs—and initialization values, for both hyperparameters and parameters meant for optimization. An example collection of five design problems for the machine learning area of reinforcement learning with a focus on NN solutions can be found in [4].

Incorporation of uncertainty Most machine learning models will not provide a proper uncertainty output, i.e. estimation of the prediction variance. [29] advises to propagate uncertainty of any component through the system to enable monitoring of accumulated uncertainty. Examples of ML models with uncertainty output are classical Bayesian networks, or for NNs Bayesian new deep learning methods like treating weights as distributions [14], specialized loss (e.g. [38] models classification output as Dirichlet distributions), or via learned specialized weight decay and dropout [23]. Uncertainty treatment was also shown to improve properties like adversarial robustness [38]. Even though available methods are theoretically mature, there is little practical experience and a lot of ongoing research in this direction.

Inclusion of expert knowledge Available expert knowledge should be included into the model in a measurable way for better behavior control. Ways to include prior knowledge are e.g. via the training data as shown in [9], which is framework for automated counterexample generation and training data augmentation; via the loss function as done for soft logic in a teacher-network approach in [18], for fuzzy logic in [35], and for a general objective function in [13]; via statistical model repair methods for reinforcement learning (RL) as developed in [15]; via safe state enforcement for RL as can be achieved through the adaptive control strategy developed in [3] and through the planner switching strategy suggested in [12]; or via topology as in convolutional NNs (CNNs) or in the ReNN architecture [42] which introduces an interpretable and verifiable intermediate layer. All mentioned methods report to also increase performance, but yield little guarantees and are an open field of research.

Robustness Enhancements Robustness here means the indifference of the model output for slight (with regard to a given metric) changes for the given data samples. Non-robust models exhibit chaotic behavior, and deprive testing data of their significance since one sample generalizes to a comparatively small input range. Furthermore, measures must be taken to increase and assure robustness, as e.g. regularization and training data preparation, either by adding adversarial counterexamples [9] or by removing non-robust features in the data [20]. Uncertainty treatment was shown to help as well (see above).

Proper design choices during development contribute to a convincing process argumentation. However, for data-driven automated modeling combined with high complexity, process argumentation needs to be supported by a strong product based argumentation based on verification and validation.

2.3 Verification

Verification is the formal check of the model against the defined (formal) requirements and testing data. This requires tools (e.g. solver or search algorithms) to conduct the formal model checks on the novel ML models. Means to check properties in form of rules are:

- **solvers**, e.g. based on satisfiability modulo theory (SMT) [see survey in 6], or mixed integer linear programming [10], where for both approaches the idea is to transform the condition on the network to check into a solvable equation system;
- **output bound estimation** such as ReluVal [43] optimized for networks with ReLU activations, or the more general reachability analysis tool DeepGo [36];
- **search algorithms** like [31].

The given approaches are well matured. However, they specialize on stability properties, except for the very performance limited solvers. Therefore, we see a need for rule checkers that can efficiently deal with first-order logic like the SMT based solver Reluplex [22].

2.4 Validation

Validation is the task of identifying and adding missing requirements and test cases. It takes up an essential role in a ML safety case due to the sparse specification typical for ML tasks. We identified two aspects that need to be assessed: The testing and training data, and the inner logic and representation of the ML model. Methods for the latter can be categorized in qualitative and quantitative.

Data validation Test and training cases should thoroughly cover the input space. Test cases additionally need to cover the available experience, and the model behavior. The first two were described above. For model behavior coverage, new NN specific metrics were suggested alongside a test case generation framework in [40], and mature tools for counterexample generation [9] are available such as [16] which maximizes the model output differences for minimal perturbations using efficient fuzzy testing.

Qualitative analysis Qualitative analysis may give experts valuable indications about misbehavior. One type is the assessment of "attention", i.e. the assessment which parts of the input most contributed to an output decision. In the case of computer vision, heatmapping can be used to indicate attention. There are both model specific methods available like the signal back-tracing methods suggested in [27],as well as model agnostic ones like based on local linear approximations [32, 34]or on mutual information like the attention estimator trained in [7]. Local linear approximations are created by observing the output for slight, spacially distinct, modifications of one given example. Such modifications can be blending out parts like in the original LIME method [34] or e.g. more advanced blurring techniques as in RISE [32]. Other than attention analysis, feature visualization [30] for NNs gives insight into the specialization of single units. Additional explanatory output like textual explanations as demonstrated in [26] can also be useful. Another example of additional output is hierarchical information as realized in [35] by additional output neurons together with logical constraints translated into the loss function.

Quantitative analysis Methods that yield quantitative insights into black-box NNs are scarce, especially for NNs. One is extraction of rules, either locally for a subset of examples using inductive logic programming [33] or globally. Global model-agnostic rule extraction like validity interval analysis [41], which iteratively refines valid intervals using back-propagation of outputs, is due to the complexity of concurrent networks not applicable to NNs (see the survey on rule extraction for NNs in [17]). Other methods are analysis of the performance on sub-tasks [13] and of the inherent (distributed) representation of semantic domain concepts within the network structure [11, 24]. The latter learn the representation of a concept within the NN from its intermediate output on a training set for this concept. Net2Vec [11] does this by attaching an additional output, which enables retraining for the concept, while TCAV [24] uses support vector machines to improve uniqueness. Given a representation vector of a semantic concept, its attribution to output classes can be determined via directed derivative along the found representation vector [24]. Concept analysis could enable topological modularization as in the simple example in the ReNN architecture [42].

3 Conclusion and Outlook

ML requires new approaches for a convincing safety argument, and such are available for different ML types. The identification of open challenges is eased by a structured, certification oriented overview of these methods such as summarized from the above in Table 1. This now needs to be extended and aligned with a safety argumentation strategy, to finally be established in standards and industries.

Table 1. Overview of method categories for the safety argumentation of ML models; for each category examples are given, accompanied by citations for further reading and by the applicability scope restrictions (applicable to all ML models if none are given). The methods are categorized by the life-cycle phase they are applicable in. Major open challenges are marked *italic*.

Phase & method goals	Method categories	Examples	Scope
Requirements Engineering Use available domain knowledge to formulate use-case and safety requirements	*Data representativity requirements*	- scenario coverage [8] - input space ontology [5]	 AD
	Robustness requirements	- adversarial robustness metric [22]	
	System fault tolerance requirements	- runtime monitoring - *model diversity measure* for redundancy [2, 7.4.12]	
	Safety performance measures	- occlusion sensitivity [8]	
	Plausibility requirements	- AD behavior rules [39] - sensible intermediate steps - domain specific rules (physical, legal)	TP[a]
	Experience collection	- AD pitfalls collected in [28]	AD
Development Apply reasonable design choices at all decision points	Design based on experience	- experience on model type, training method, initialization values - list of pitfalls in [4]	 RL
	Incorporation of uncertainty [29]	- [14, 23]	NNs
	Inclusion of expert knowledge	- via loss function [18, 35] - via topology [42] - model repair [15] - safe learning [3, 12]	NNs NNs RL RL
	Robustness enhancements	- regularization [see 20] - robustification of training data [20] - counterexample-guided data augmentation [9]	
Verification Check against test data and model requirements	Formal verification of rules, model, KPIs	- *solvers* [6, 10] - boundary approximation [36, 43] - search algorithms [31]	NNs NNs NNs
Validation Find missing ...			
... test cases	Input space coverage checks	- (see data representativity method)	
	Experience coverage checks	- (see experience collection method)	
	Model coverage checks	- concolic testing [40] - counterexample generation [9, 16, 40]	NNs NNs
... requirements via qualitative model analysis	Attention analysis through heatmapping [45]	- back-propagation based, e.g. [27] - model agnostic, e.g. [7, 32, 34]	NNs
	Feature visualization	- method collection in [30]	CNNs
	Explanatory output	- textual explanations [26] - hierarchical information [35]	NNs NNs
... requirements via *quantitative model analysis*	Rule extraction	- locally via ILP [33] - global model agnostic, e.g. VIA [41] - global NN specific methods [17]	 NNs
	Sub-task/concept analysis	- Neural Stethoscopes [13] - concept embedding and attribution analysis [11, 24] - ReNN modularized topology [42]	NNs NNs NNs

[a] trajectory planning

Bibliography

[1] 32, I.S.: Road Vehicles — Functional Safety — Part 1: Vocabulary, vol. 1, 2 edn. (2018)

[2] 32, I.S.: Road Vehicles — Functional Safety — Part 6: Product Development at the Software Level, vol. 6, 2 edn. (2018)

[3] Akametalu, A.K., Fisac, J.F., Gillula, J.H., Kaynama, S., Zeilinger, M.N., Tomlin, C.J.: Reachability-based safe learning with gaussian processes. In: Proc. 53rd IEEE Conf. Decision and Control, pp. 1424–1431 (2014)

[4] Amodei, D., Olah, C., Steinhardt, J., Christiano, P.F., Schulman, J., Mané, D.: Concrete problems in AI safety. CoRR **abs/1606.06565** (2016)

[5] Bagschik, G., Menzel, T., Maurer, M.: Ontology based scene creation for the development of automated vehicles. In: Proc. 2018 IEEE Intelligent Vehicles Symp., pp. 1813–1820 (2018)

[6] Bunel, R.R., Turkaslan, I., Torr, P., Kohli, P., Mudigonda, P.K.: A unified view of piecewise linear neural network verification. In: Advances in Neural Information Processing Systems 31, pp. 4790–4799 (2018)

[7] Chen, J., Song, L., Wainwright, M., Jordan, M.: Learning to explain: An information-theoretic perspective on model interpretation. In: Proc. 35th Int. Conf. Machine Learning, vol. 80, pp. 883–892 (2018)

[8] Cheng, C.H., Nührenberg, G., Huang, C.H., Ruess, H., Yasuoka, H.: Towards dependability metrics for neural networks. In: 16th ACM/IEEE Int. Conf. Formal Methods and Models for System Design, pp. 43–46 (2018)

[9] Dreossi, T., Ghosh, S., Yue, X., Keutzer, K., Sangiovanni-Vincentelli, A.L., Seshia, S.A.: Counterexample-guided data augmentation. In: Proc. 27th Int. Joint Conf. Artificial Intelligence, pp. 2071–2078 (2018)

[10] Dutta, S., Jha, S., Sankaranarayanan, S., Tiwari, A.: Output range analysis for deep feedforward neural networks. In: Proc. 10th Int. Symp. NASA Formal Methods, vol. 10811, pp. 121–138 (2018)

[11] Fong, R., Vedaldi, A.: Net2Vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. In: Proc. 2018 IEEE Conf. Comput. Vision and Pattern Recognition, pp. 8730–8738 (2018)

[12] Fridovich-Keil, D., Herbert, S.L., Fisac, J.F., Deglurkar, S., Tomlin, C.J.: Planning, fast and slow: A framework for adaptive real-time safe trajectory planning. In: Proc. 2018 IEEE Int. Conf. Robotics and Automation, pp. 387–394 (2018)

[13] Fuchs, F.B., Groth, O., Kosiorek, A.R., Bewley, A., Wulfmeier, M., Vedaldi, A., Posner, I.: Neural Stethoscopes: Unifying analytic, auxiliary and adversarial network probing. CoRR **abs/1806.05502** (2018)

[14] Gast, J., Roth, S.: Lightweight probabilistic deep networks. In: Proc. 2018 IEEE Conf. Comput. Vision and Pattern Recognition, pp. 3369–3378 (2018)

[15] Ghosh, S., Lincoln, P., Tiwari, A., Zhu, X.: Trusted machine learning: Model repair and data repair for probabilistic models. In: Workshops 31st AAAI Conf. Artificial Intelligence, vol. WS-17 (2017)

[16] Guo, J., Jiang, Y., Zhao, Y., Chen, Q., Sun, J.: DLFuzz: Differential fuzzing testing of deep learning systems. In: Proc. ACM Joint Meeting on European Software Engineering Conf. and Symp. Foundations of Software Engineering, pp. 739–743 (2018)

[17] Hailesilassie, T.: Rule extraction algorithm for deep neural networks: A review. CoRR **abs/1610.05267**, 555,555 (2016)

[18] Hu, Z., Ma, X., Liu, Z., Hovy, E.H., Xing, E.P.: Harnessing deep neural networks with logic rules. In: Proc. 54th Annu. Meeting of the Association for Computational Linguistics, vol. 1: Long Papers (2016)

[19] Huang, X., Kroening, D., Kwiatkowska, M., Ruan, W., Sun, Y., Thamo, E., Wu, M., Yi, X.: Safety and trustworthiness of deep neural networks: A survey. CoRR **abs/1812.08342** (2018)

[20] Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., Madry, A.: Adversarial Examples Are Not Bugs, They Are Features. CoRR **abs/1905.02175** (2019)

[21] Johnson, C.W.: The increasing risks of risk assessment: On the rise of artificial intelligence and non-determinism in safety-critical systems. In: Evolution of System Safety: Proc. Safety-Critical Systems Symp. (2017)

[22] Katz, G., Barrett, C., Dill, D.L., Julian, K., Kochenderfer, M.J.: Reluplex: An efficient SMT solver for verifying deep neural networks. In: Proc. 29th Int. Conf. Comput. Aided Verification, pp. 97–117 (2017)

[23] Kendall, A., Gal, Y.: What uncertainties do we need in Bayesian deep learning for computer vision? In: Advances in Neural Information Processing Systems 30, pp. 5580–5590 (2017)

[24] Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., Sayres, R.: Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In: Proc. 35th Int. Conf. Machine Learning, vol. 80, pp. 2668–2677 (2018)

[25] Kim, J., Canny, J.F.: Interpretable learning for self-driving cars by visualizing causal attention. In: Proc. 2017 IEEE Int. Conf. Comput. Vision, pp. 2961–2969 (2017)

[26] Kim, J., Rohrbach, A., Darrell, T., Canny, J.F., Akata, Z.: Textual explanations for self-driving vehicles. In: Proc. 15th European Conf. Comput. Vision, Part II, vol. 11206, pp. 577–593 (2018)

[27] Kindermans, P.J., Schütt, K.T., Alber, M., Müller, K.R., Erhan, D., Kim, B., Dähne, S.: Learning how to explain neural networks: PatternNet and PatternAttribution. In: Proc. 6th Int. Conf. on Learning Representations (2018)

[28] Koopman, P., Fratrik, F.: How many operational design domains, objects, and events? In: Workshops of the 32nd AAAI Conf. Artificial Intelligence (2019)

[29] McAllister, R., Gal, Y., Kendall, A., van der Wilk, M., Shah, A., Cipolla, R., Weller, A.: Concrete problems for autonomous vehicle safety: Advantages of Bayesian deep learning. In: Proc. 26th Int. Joint Conf. Artificial Intelligence, pp. 4745–4753 (2017)

[30] Olah, C., Mordvintsev, A., Schubert, L.: Feature visualization. Distill **2**(11), e7 (2017)

[31] Pei, K., Cao, Y., Yang, J., Jana, S.: Towards practical verification of machine learning: The case of computer vision systems. CoRR **abs/1712.01785** (2017)

[32] Petsiuk, V., Das, A., Saenko, K.: RISE: Randomized input sampling for explanation of black-box models. In: Proc. British Machine Vision Conf., p. 151 (2018)

[33] Rabold, J., Siebers, M., Schmid, U.: Explaining black-box classifiers with ILP – empowering LIME with Aleph to approximate non-linear decisions with relational rules. In: Proc. Int. Conf. Inductive Logic Programming, pp. 105–117 (2018)

[34] Ribeiro, M.T., Singh, S., Guestrin, C.: "Why should I trust you?": Explaining the predictions of any classifier. In: Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, pp. 1135–1144 (2016)

[35] Roychowdhury, S., Diligenti, M., Gori, M.: Image classification using deep learning and prior knowledge. In: Workshops of the 32nd AAAI Conf. Artificial Intelligence, vol. WS-18, pp. 336–343 (2018)

[36] Ruan, W., Huang, X., Kwiatkowska, M.: Reachability analysis of deep neural networks with provable guarantees. In: Proc. 27th Int. Joint Conf. Artificial Intelligence, pp. 2651–2659 (2018)

[37] Salay, R., Queiroz, R., Czarnecki, K.: An analysis of ISO 26262: Using machine learning safely in automotive software. CoRR **abs/1709.02435** (2017)

[38] Sensoy, M., Kaplan, L.M., Kandemir, M.: Evidential deep learning to quantify classification uncertainty. In: Advances in Neural Information Processing Systems 31, pp. 3183–3193 (2018)

[39] Shalev-Shwartz, S., Shammah, S., Shashua, A.: On a formal model of safe and scalable self-driving cars. CoRR **abs/1708.06374** (2017)

[40] Sun, Y., Wu, M., Ruan, W., Huang, X., Kwiatkowska, M., Kroening, D.: Concolic testing for deep neural networks. In: Proc. 33rd ACM/IEEE Int. Conf. Automated Software Engineering, pp. 109–119 (2018)

[41] Thrun, S.: Extracting rules from artificial neural networks with distributed representations. In: Advances in Neural Information Processing Systems 7, pp. 505–512 (1995)

[42] Wang, H.: ReNN: Rule-embedded neural networks. In: Proc. 24th Int. Conf. Pattern Recognition, pp. 824–829 (2018)

[43] Wang, S., Pei, K., Whitehouse, J., Yang, J., Jana, S.: Formal security analysis of neural networks using symbolic intervals. In: Proc. 27th USENIX Security Symp., pp. 1599–1614 (2018)

[44] Xiang, W., Musau, P., Wild, A.A., Lopez, D.M., Hamilton, N., Yang, X., Rosenfeld, J.A., Johnson, T.T.: Verification for machine learning, autonomy, and neural networks survey. CoRR **abs/1810.01989** (2018)

[45] Zhang, Q., Zhu, S.C.: Visual interpretability for deep learning: A survey. Front. IT EE **19**(1), 27–39 (2018)

Structuring the Safety Argumentation for Deep Neural Network Based Perception in Automotive Applications

Gesina Schwalbe[1(✉)], Bernhard Knie[2,7], Timo Sämann[3], Timo Dobberphul[4], Lydia Gauerhof[5], Shervin Raafatnia[6], and Vittorio Rocco[7]

[1] Continental AG, Regensburg, Germany
gesina.schwalbe@continental-corporation.com
[2] Automotive Safety Technologies GmbH, Wolfsburg, Germany
[3] Valeo Schalter und Sensoren GmbH, Kronach, Germany
[4] Volkswagen AG, Wolfsburg, Germany
[5] Corporate Research Robert Bosch GmbH, Renningen, Germany
[6] Robert Bosch GmbH, Stuttgart-Weilimdorf, Germany
[7] Università di Roma Tor Vergata, Rome, Italy

Abstract. Deep neural networks (DNNs) are widely considered as a key technology for perception in high and full driving automation. However, their safety assessment remains challenging, as they exhibit specific insufficiencies: *black-box nature*, *simple performance issues*, *incorrect internal logic*, and *instability*. These are not sufficiently considered in existing standards on safety argumentation. In this paper, we systematically establish and break down safety requirements to argue the sufficient absence of risk arising from such insufficiencies. We furthermore argue why diverse evidence is highly relevant for a safety argument involving DNNs, and classify available sources of evidence. Together, this yields a generic approach and template to thoroughly respect DNN specifics within a safety argumentation structure. Its applicability is shown by providing examples of methods and measures following an example use case based on pedestrian detection.

Keywords: Automated driving · Safety case · Deep neural networks

1 Introduction

Deep neural networks can solve tasks which cannot be easily specified, involving high dimensional input spaces. They promise to be an alternative to rule-based algorithms for environment perception in autonomous driving, like pedestrian detection. However, their safety assessment in the automotive context remains challenging: DNNs exhibit specific *insufficiencies*, that can lead to hazardous failures not covered by existing safety standards [15, B.2, p. 34] (see discussion in Sect. 2). For completeness, a safety case must cover the known, technology specific insufficiencies, which might involve specialized sources of evidence. A

A. Casimiro et al. (Eds.): SAFECOMP 2020 Workshops, LNCS 12235, pp. 383–394, 2020.
https://doi.org/10.1007/978-3-030-55583-2_29

structured, two-fold approach towards this challenge for the use of DNNs is presented in this work. Section 3 features the top-down part: *DNN insufficiencies* known from literature are structured (*black-box property*, *simple lack of generalization*, *incorrect internal logic*, and *instability*), to derive DNN specific safety requirements, which then are broken down into sub-requirements (see Fig. 1). Section 4 provides the bottom-up part: The influence of DNN insufficiencies on the different types of evidence (development, system level, and V&V) is investigated. According to this, a structure for DNN related evidence is suggested (see Fig. 2). Our concrete contributions towards safety critical DNN applications are as follows:

- A template to structure the safety argumentation part specific to DNNs is developed in the form of safety requirements and evidence categories. The structured approach facilitates coverage not only of known sources of evidence but also of known DNN specific insufficiencies.
- For the suggested evidence classes, typical examples of methods and measures are given. As a running example, the automated driving use case pedestrian detection is used.

Scope. As a running example use case, an automotive pedestrian perception and successive control function for an emergency braking automation is assumed, with perception realized by a DNN. With the architectural concept inspired by the STAMP/STPA model [18], hazards could be caused by a late, early, or misdetection of pedestrians by the DNN. We here want to focus on misdetections. The failures of hardware and software implementations, as well as non-DNN-specific SOTIF aspects are not considered (see e.g. [8] on how to structure the validity of assumptions). Concentrating on a first structure, the evaluation of the contribution of specific evidence methods is reserved for future work. This also holds for system architectural aspects like redundancy.

2 Related Work

Completeness of a safety argument requires that all insufficiencies of the used technology that might lead to malfunctioning are considered. However, established safety standards do not yet sufficiently cover DNN specific aspects. The automotive functional safety standard [13] focuses on failures emerging from hardware and traditional software, and few of the suggested methods are applicable to DNNs [21]. The standard on safety of the intended functionality (SOTIF) [15] extends this towards failures caused by foreseeable misuse, environment, and performance limitations of complex sensoric systems [15, Table 1]—as of today not considering DNN specific limitations in detail. The same holds for the available domain specific draft standards that collect best-practice methods for design as well as verification and validation (V&V). Examples for driving automation are the white paper [31], and the UL4600 [28]; for aviation the report [5]; and national activities are e.g. the German DIN SPEC 13266 [6] on computer vision systems. We would like to amend the above bottom-up, evidence-driven approaches with a top-down, insufficiency-oriented perspective. For this we unify following previous diverse work of the authors. DNN specific safety concerns and insufficiencies were collected in [22] and [30]. The product argumentation aspects handled here are built upon [24]. A collection of evidences was provided in [25]. And our evidence structuring can be considered a refinement of [3], who identified six types of evidence required for confidence in a safety requirement.

3 Respecting DNN Insufficiencies in Safety Requirements

The moment any safety load rests upon the functionality of a DNN, it is required to either function accurately or provide reliable failure indication for mitigation measures (see Sect. 4). Hazardous misbehavior could e.g. be overlooking a pedestrian (false negative) leading to a crash, or ghost detections (false positive) leading to unnecessary braking and rear crash. Usual assessment investigates DNN behavior (i.e. generalization ability) according to human "understanding" of the task, which includes expected decision boundaries, corner cases, and continuity of the solution. This introduces a human bias to the assessment: e.g. , an assessor will expect an algorithm to react similar on examples that he or she assumes similar or identical. DNNs—other than manually designed algorithms—are not tied to such assumptions, as will be discussed in the lack of explainability in Sect. 3.1. For the sake of completeness of a safety argument, such assumptions must be validated and avoided. To achieve this, we suggest to derive DNN related safety requirements from types of DNN specific generalization issues. This will include issues both on the semantic level, i.e. describable in natural language; and on the non-semantic level. In Sect. 3.1 we summarize and categorize known DNN insuffiencies from literature, from which generic safety requirements are derived in Sect. 3.2 (see Fig. 1).

3.1 DNN Insufficiencies

DNN insufficiencies are properties of trained DNN models inherent to their technology, and with negative impacts for the use in safety-critical systems [22]. Structured overviews of DNN insufficiencies can be found e.g. in [22] and [30]. For the sake of our arguments, we consider two super-categories of DNN insufficiencies: The *black-box property* of DNNs indirectly infringing the safety case, and generalization issues which can directly cause hazardous failures. Lack of operational efficiency [22] concerning hardware and implementation dependent aspects (inference time, memory consumption) are not considered.

The black-box property of DNNs refers to their *lack of explainability*. DNNs use learned features to derive their predictions. These are extracted from data representations rather than the semantic content of the input. This flexibility is one of their major benefits for hard-to-specify perception applications: For example, the representation of a pedestrian in all its varieties cannot be completely semantically specified such that it could always be recognized by a rule-based

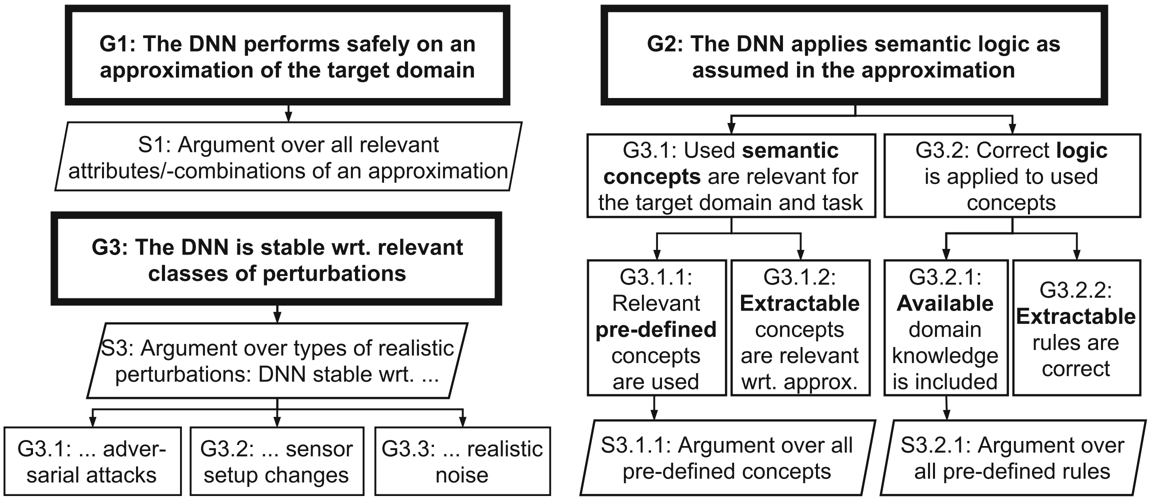

G1: The DNN performs safely on an approximation of the target domain
S1: Argument over all relevant attributes/-combinations of an approximation
G3: The DNN is stable wrt. relevant classes of perturbations
S3: Argument over types of realistic perturbations: DNN stable wrt. ...
G3.1: ... adversarial attacks
G3.2: ... sensor setup changes
G3.3: ... realistic noise
G2: The DNN applies semantic logic as assumed in the approximation
G3.1: Used semantic concepts are relevant for the target domain and task
G3.2: Correct logic is applied to used concepts
G3.1.1: Relevant pre-defined concepts are used
G3.1.2: Extractable concepts are relevant wrt. approx.
G3.2.1: Available domain knowledge is included
G3.2.2: Extractable rules are correct
S3.1.1: Argument over all pre-defined concepts
S3.2.1: Argument over all pre-defined rules

Fig. 1. Safety requirements (**goals** G) and decomposition **strategies** (S) derived in Sect. 3.2 in goal structuring notation [26].

algorithm. The flexibility comes at a cost: The learned features and their correlations are not necessarily comprehensible for humans, or even counterintuitive [9,30]. The high-dimensional internal structures of DNNs further hinder interpretable representations, which are be needed for traditional inspection and test case selection.

Lack of generalization ability summarizes *performance limitations* [15, 3.9] which can cause erroneous output. Generalization here means the performance on an unseen test set relative to that of the training set, as an estimate for the performance in the field. We suggest the following categorization: simple generalization issues, logical issues, and stability issues. The *simple generalization ability* refers to the performance on input data which is *semantically close* to the training data. This means the distributions of few semantic attributes like weather condition are changed compared to the training data. Reasons for a simple lack of generalization can be manifold, e.g. memorization of the training data, underfitting, or a underrepresentation of attributes in the training data. For example, if only few training samples feature rain, the DNN performance in rainy conditions may be very low. Another issue can be a *lack of internal logic*: The internal representation and reasoning a DNN applies originate from correlations in the training data, and may be wrong. This leads to errors that generalize according to semantic rules. For example, due to a bias in the data, a DNN may predict pedestrians not based on physiological features, but below all traffic lights. This *lack of reasoning* may already be obvious from a *lack of the internal representation* if no indicative features of pedestrians are included. Lastly, DNNs suffer from a *lack of stability* against *(slight) perturbations*—possibly imperceptible slight changes to an image that do not change the semantic information. Fairly easy-to-find perturbations may change the output of a DNN drastically and unpredictably [1]. Perturbation types and sources are manifold, e.g. : permanent or temporary sensor setup changes [3]; adversarial attacks applied in the real world or at pixel level (see taxonomy in [1]); and noise from domain (fog), sensor (dust, dirt), or intrinsic noise (faulty pixels, transmission error).

3.2 Derived Safety Requirements

We suggest to associate to each lack of generalization ability a safety requirement to mitigate it. The first is mitigation of the lack of simple generalization ability: The DNN should perform safely on a semantic approximation of the input domain, i.e. on a subset that that covers relevant semantic aspects (Fig. 1, G1). This means both a sufficient (weighted) overall performance, and sufficient performance on *each* safety critical subset (S1). Especially, one needs to argue separately over all relevant attributes and attribute combinations. In the case of pedestrian detection, such attributes could be age, occlusion, weather etc. The challenge of this semantic input space coverage is discussed later in Sect. 4.3.

When testing, the behavior of a function is interpolated from test samples. Here, implicit assumptions are made, like a certain stability or invariance of the function. For example, if the DNN detects a pedestrian, it should still do so if a non-related object like a far traffic light is changed. On a semantic level, this requires that the DNN applies semantic logic as assumed in the semantic approximation specified before (Fig. 1, G2). Sub-requirements are that relevant features or concepts are internally represented (G3.1), and that the logic/reasoning applied to them is correct (G3.2). In the case of both reasoning and concepts, one can either formally verify specified ones (G & S3.x.1), or manually inspect extracted ones (G3.x.2, see verification in Sect. 4.3). Interpolation assumptions may not only fail on a semantic level, but also due to instabilities. Therefore, the last goal is that the DNN behavior is stable in the input domain with respect to relevant slight perturbations (Fig. 1, G3). It remains a challenge to identify all safety relevant, i.e. realistic, perturbations (S3 and G3.1–3).

4 Respecting DNN Insufficiencies in Evidences

We categorize and will further structure two major types of evidence for sufficient safety: *Detection and measurement*, which is obtained via V&V; and *prevention and mitigation*, which can be subdivided into best-practice measures for item creation, and reduction of the safety load via system level mechanisms (see Fig. 2). In the following, this structure is detailed, and example methods are provided. Furthermore, we will discuss the challenges imposed by DNN insufficiencies with the result that V&V alone cannot provide sufficient confidence about safety. Best-practice cannot fully compensate this, due to the lack of field experience with DNNs in automated driving. Thus, our conclusion is that all types of evidence should be considered (compare [3]).

4.1 Mechanisms During Creation

The example use case is based on a DNN that is trained offline. For this kind of artificial intelligence component a creation process can be followed that consists

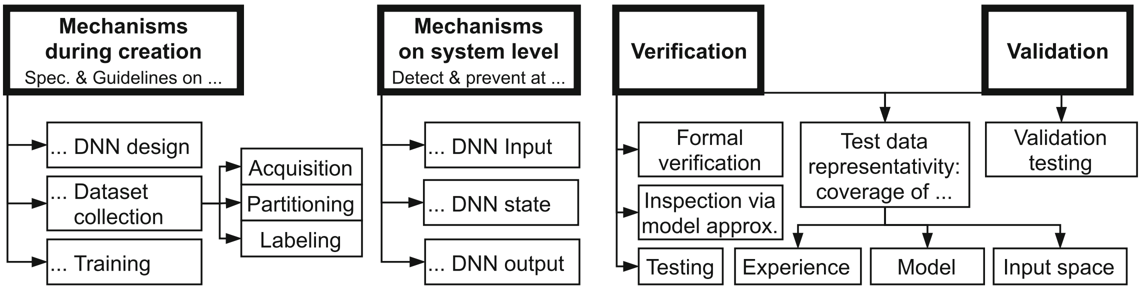


Fig. 2. Hierarchical overview of evidence categories identified in Sect. 4.

of three phases: *design of the network*, *data collection* and *training*. The corresponding intermediates are the raw network with initial values; labeled datasets for training, verification and validation; and the trained network.

Today's state of the art software engineering is requirement-based with design, implementation and verification steps following appropriate sets of rules and strategies to prove fulfillment of requirements and predefined goals (see the ISO/IEC 330xx family of standards, and [12]). Applying these general principles to the creation process of offline trained DNNs means: One has to define requirements and goals for intermediates in a design step, implement them, and prove their achievement by a verification step. Defining and complying to strategies and guidelines for certain activities supports the systematic achievement of goals and requirements where applicable.

This section structures types of requirements, goals, guidelines or strategies for the different creation phases and their intermediates, which were extracted from relevant and current references about reliable DNNs [3,12,22,29–31]. Evidencing their fulfillment helps to argue the mitigation of DNN insufficiencies during its creation process. The examples aim to show the diversity of aspects, for details consult the referenced literature.

Design of the Network. The DNN design implicitly makes prior assumptions about the application, e.g. convolutional DNNs introduce translation invariance. A DNN respecting the following aspects helps to create a reliable and robust raw network as a stable starting point for the further phases.

DNN specification [31] containing requirements for: robustness of the algorithm [31]; the architectural design (including layer parameters) [22]; the DNN class [22]; the interfaces [12] with definition of the output space and input resolution [29] to enable the output of reliable confidence information [30]; weight initialization [25].

Design guidelines defining the unified language for the specification [29] and using best practices from established design approaches.

Data Collection. High quality datasets are essential to maximize the performance of trained networks, as they are the source for the learned features.

Dataset specification [31]*:* Considering the following aspects will minimize the insufficiencies arising from incomplete datasets: dataset quality, coverage and relevance [31]; dataset representativity [3,25]; sample classification and the corresponding equivalence classes [31]; corner cases and other boundaries [31]; dataset robustification and necessary perturbations [25]; dataset augmentation including adversarial examples and attribute randomization [22].

Data acquisition strategy [30] defining an adequate choice for data sources.

Data partitioning guidelines [30] giving rules how to divide the dataset into training, verification and validation data.

Labeling specification [31] High quality labeling is essential to maximize the quality of datasets. At least the following requirements should be considered to minimize insufficiencies from weak labeling: choice and boundaries of label classes [22]; labeling accuracy and quality [31]; adversarial examples [22].

Labeling guidelines [30] to ensure unified and exact labeling and to avoid labeling errors.

Training. The training practically decides on whether the available data (and further knowledge) is processed efficiently. This includes preventing e.g. over- and underfitting. Especially, training quality directly influences the DNN quality.

Training specification: Considering the following types of goals increases the probability of getting the intended output of the training phase: hyperparameters (batch size, regularization, loss function, etc.) [31]; active learning [22]; domain randomization [22]; robustness; constraints for internal logic [25]; well-calibrated uncertainties [22]; accuracy and failure rates [3].

Training strategy defining the procedure of adjustments of training parameters and the training sequence and iterations.

Hyperparameters tuning guidelines [31] to systematically adjust the parameters to achieve the training goals.

Configuration management strategy [12] to establish training baselines [31]. This enables analyzing the training steps, setting termination points, and recovering optimal stages.

Further modification activities for deploying, optimization, compression or quantization have not the goal to increase the reliability of a DNN and therefore, no arguments for mitigation of insufficiencies are derived conducting these activities. The challenge is to prove that all assumptions and safety arguments are still valid after these further modifications.

4.2 Mechanisms on Component and System Level

Besides the mechanisms during DNN creation, other mechanisms, such as detection of causes for performance limitations, can be applied to decrease the safety load on the DNN and increase the overall safety. While detection mechanisms are the first measure to be aware of an potential error, they can be used for further handling, such as prevention, mitigation or even forecast. Examples are to filter the output, switch to alternative predictors, or increase caution, e.g. the controller initiates a gentle slow down if the prediction quality is unsure. We suggest to structure the mechanisms with respect to their intervening point: input, DNN and output of DNN. Some methods make use of their combination.

Input Before providing the input to DNN, *modification* of the input can decrease the chance for an error, e.g. by normalization, denoising, filtering, or removal of non-semantic features causing adversarial examples [25]. Furthermore, the input of DNN might be *monitored* for causes of performance limitations. For example, detection of adversarial examples [4] causing instability errors, or out-of-distribution samples, i.e. samples very different from the training data, can indicate a to-be-expected error within the pedestrian detection.

DNN Error *detection* and *mitigation* methods can be applied on the DNN itself. If the exact error is known, e.g. from input monitoring and estimation, the *DNN output might be corrected* by an additional output [23]. Uncertainty measures of the DNN, indicating the current inherent uncertainty state of the function [11,17], are treated as promising safety mechanisms. For example, the output of the DNN can be dropped in the case of high uncertainty. Otherwise, the uncertainty level should be forwarded to the next component [25].

Output The output and the behavior of the DNN can be *monitored* to detect errors. In general, anomaly detection [7] as well as plausibility checks might be applied to the DNN output to detect errors, e.g. pedestrians cannot vanish in clear sight. For doing so, the processing information of the input within the DNN is compared to the behavior for clean data. An example method is ODIN [19]: By using temperature scaling (calibration of the DNN confidence output) and exploring the behavior for small perturbations around the input, ODIN can detect samples to which the network might not be capable to generalize. Another method, GraN [20], is introduced for detecting misclassified data samples in general, and adversarial examples. It investigates the norm of the gradient of the DNN function on the current input-output combination. Other than monitoring, the DNN output might be *modified*, e.g. further processed and fused with other signals. For example, when fusing with outputs of different DNNs as part of an *ensemble*, the final output might increase the performance and reduce errors, even though ensuring sufficient model diversity for the latter may be hard [25]. A *fusion* with signals gathered from other sensors also result in redundancies that might have different limitations.

4.3 Verification and Validation

Verification activity should determine whether the specified requirements are met [13, 3.180]. and focuses on known insufficiencies [15, 3.18]. Validation tries to identify new insufficiencies affecting safety, and provides assurance, that the safety requirements are adequate [13, 3.148] in the sense of correctness and completeness. In the following, attention is concentrated on DNN-specific requirements as derived in Sect. 3.2, other than e.g. requirements on the system-level safety mechanisms.

In practice, the terms verification and validation are hard to distinguish and share the following common challenges. One is the *open-world* domain typical for DNNs, which exhibits many rare scenes and may change over time. Being high-dimensional, partly non-semantic, and complex, open world domains cannot be explored or specified thoroughly using human interpretable attributes [30].

The goal of test data is to effectively reveal inconsistencies between expected and DNN output that indicate DNN insufficiencies. Near misses are less valuable for this than implausible errors, imposing a challenge for defining *performance indicators*. For effectivity, high *test data representativity* is required, which is another V&V challenge. We propose to handle this as a coverage problem and suggest as coverage criteria both the semantic features of the input space, and the DNN-specific ones, meaning coverage of DNN state-space. The latter is hard to achieve formally due to the enormous size of the state-space and since the sub-space of valid samples is unknown. Thus, semi-formal exploration of the state-space should further aim for coverage of DNN-specific weaknesses: coverage of instability sources, and of previous counterexamples. For *semantic input space coverage*, all relevant semantic aspects of the objective need to be statistically covered, especially cases defining the true decision boundary (e.g. high occlusion). Ontologies such as [2] are a good starting point to find such aspects. Challenges remain: the real distribution of values may be hard to approximate due to the open-world context or a lack of real samples; and validating realism of synthetic samples is an unsolved problem [30]. The model decision boundaries and decision-relevant features not necessarily agree with those assumed by humans. To reveal related weaknesses, a high *coverage of the model state space* is required. This can be done globally, e.g. via coverage of neuron activation patterns [27]; or locally by estimating the expected distance to the decision boundaries (see verification). A lack of instability imposes a major challenge for the *validity range* of test samples: Due to the black-box property of correlations, we may not be able to retrace their cause, wherefore the behavior out of the validity range cannot be guaranteed or even estimated. The validity range of a sample includes how far and how stable the nearest sample with significantly different behavior is. A distance metric can be L2-distance, as used in the formal verification example in [16]. Lastly, DNNs are prone to regression as long as the effect of parameter adaptions on the global behavior cannot be controlled. Hence, *previous counterexamples* should be tested. Prominent sources are e.g. earlier test phases also from similar tasks, and operation experience like (near) failure reports and accident databases.

Verification. The types of verification activities from ISO 26262[13, 3.180] that are in principle applicable to DNNs are [21]: *walk-through or inspection* of the algorithm, *testing and semi-formal verification*, and *formal verification* requiring a formal notation that is linked to intermediate outputs of the considered model. A main verification specific challenge is to provide methods applicable to DNNs. For current formal verification methods, requirements must be formulated as range constraints on neuron outputs which is seldom possible due to the black-box property. Provers, e.g. solvers like Reluplex [16], are mostly limited in architecture types and speed due to DNN sizes. By definition, inspection and walk-through require interpretable approximations of the model. A standard but less expressive example is to indicate the attention of the DNN on selected samples via heatmapping. Alternatively, the internal logic can be assessed using local or global rule extraction, but methods that can deal with up-to-date DNN complexity are scarce [10]. For details on the example methods see [25].

Validation. Ensuring that all safety-relevant factors have been taken into account is already a non-trivial task for non-AI-based functions due to the open-world context. The state-of-the-art validation approach in automotive is to evaluate the function on a large, randomly collected amount of real-world data ([14, 8.4.3.4], [31, Sect. 3.3.2]). The goal is to test the problem space for safety-relevant factors. However, the effectivity of explorative testing decreases with increasing autonomy level, function and environment complexity: Statistical coverage is harder to achieve; given validity range problems, it is difficult to decide when enough variations of the same semantic content have been considered; and lastly, exploring features used by the DNN is infringed by the black-box problematic, which makes it impossible to specify, and thus cover, all decision-relevant features. The central activity for validating DNNs remains the collection of a (large and) representative test dataset, with emphasis on model state-space coverage.

The discussion above shows: V&V for DNNs cannot be separated clearly, and are challenged by the complexity coming along with DNNs. Especially due to the representativity and validity range problems, one cannot expect to gain sufficient confidence in safety only via V&V. One needs to address V&V at all development stages and even beyond.

5 Conclusion

Generic safety requirements and an evidence structure were suggested to include DNN specifics into a safety argument with a focus on completeness. Nevertheless, the structure revealed several open challenges and difficulties for a complete safety case for DNNs. Further research will refine the suggested argumentation on the proposed use case, fill the method gaps, and evaluate the contributions of the different methods to the overall risk reduction.

Acknowledgements. The research leading to the results presented above is funded by the German Federal Ministry for Economic Affairs and Energy within the project "KI Absicherung – Safe AI for automated driving". The authors would like to thank the consortium for the successful cooperation. Special thanks to Simon Burton, Horst Michael Groß (Ilmenau University of Technology, Neuroinformatics and Cognitive Robotics Lab), Christian Hellert, Fabian Hüger, Peter Schlicht, and Oliver Willers.

References

1. Assion, F., et al.: The attack generator: a systematic approach towards constructing adversarial attacks. In: Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition Workshops (2019)
2. Bagschik, G., Menzel, T., Maurer, M.: Ontology based scene creation for the development of automated vehicles. In: Proceedings of the 2018 IEEE Intelligent Vehicles Symposium, pp. 1813–1820. IEEE (2018). https://doi.org/10.1109/IVS.2018.8500632
3. Burton, S., Gauerhof, L., Sethy, B.B., Habli, I., Hawkins, R.: Confidence arguments for evidence of performance in machine learning for highly automated driving functions. In: Romanovsky, A., Troubitsyna, E., Gashi, I., Schoitsch, E., Bitsch, F. (eds.) SAFECOMP 2019. LNCS, vol. 11699, pp. 365–377. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26250-1_30
4. Carlini, N., Wagner, D.: Adversarial examples are not easily detected: bypassing ten detection methods. In: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec 2017, pp. 3–14. Association for Computing Machinery (2017). https://doi.org/10.1145/3128572.3140444
5. Cluzeau, J.M., Henriquel, X., Rebender, G., et al.: Concepts of design assurance for neural networks. Technical report, European Union Aviation Safety Agency (EASA) (2020)
6. Deutsches Institut für Normung e.V.: DIN SPEC 13266:2020-04: Guideline for the development of deep learning image recognition systems. Beuth Verlag, 2020-04 edn, April 2020. https://doi.org/10.31030/3134557
7. Gauerhof, L., Gu, N.: Reverse variational autoencoder for visual attribute manipulation and anomaly detection. In: Winter Application Conference on Applications of Computer Vision (2020)
8. Gauerhof, L., Munk, P., Burton, S.: Structuring validation targets of a machine learning function applied to automated driving. In: Gallina, B., Skavhaug, A., Bitsch, F. (eds.) SAFECOMP 2018. LNCS, vol. 11093, pp. 45–58. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99130-6_4
9. Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F.A., Brendel, W.: ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In: Proceedings of the 7th International Conference on Learning Representations (2018)
10. Hailesilassie, T.: Rule extraction algorithm for deep neural networks: a review. CoRR abs/1610.05267 (2016)
11. Henne, M., Schwaiger, A., Roscher, K., Weiss, G.: Benchmarking uncertainty estimation methods for deep learning with safety-related metrics. In: Proceedings of the Workshop on Artificial Intelligence Safety, vol. 2560, pp. 83–90. CEUR-WS.org (2020)
12. ISO/IEC JTC 1/SC 7: ISO/IEC/IEEE 12207:2017: Systems and Software Engineering—Software Life Cycle Processes, 1 edn. (2017)
13. ISO/TC 22/SC 32: ISO 26262–1:2018(En): Road Vehicles—Functional Safety—Part 1: Vocabulary, ISO 26262:2018(En), vol. 1. 2 edn. (2018)
14. ISO/TC 22/SC 32: ISO 26262–4:2018(En): Road Vehicles—Functional Safety—Part 4: Product Development at the System Level, ISO 26262:2018(En), vol. 4. 2 edn. (2018)
15. ISO/TC 22/SC 32: ISO/PAS 21448:2019(En): Road Vehicles—Safety of the Intended Functionality (2019)

16. Katz, G., Barrett, C., Dill, D.L., Julian, K., Kochenderfer, M.J.: Reluplex: an efficient SMT solver for verifying deep neural networks. In: Majumdar, R., Kunčak, V. (eds.) CAV 2017. LNCS, vol. 10426, pp. 97–117. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63387-9_5
17. Kendall, A., Gal, Y.: What uncertainties do we need in Bayesian deep learning for computer vision? In: Advances in Neural Information Processing Systems, vol. 30, pp. 5580–5590 (2017)
18. Leveson, N.: Engineering a Safer World: Systems Thinking Applied to Safety. Engineering Systems. MIT Press, Cambridge (2012)
19. Liang, S., Li, Y., Srikant, R.: Principled detection of out-of-distribution examples in neural networks. CoRR abs/1706.02690 (2017)
20. Lust, J., Condurache, A.: GraN: an efficient gradient-norm based detector for adversarial and misclassified examples. In: ESANN (2020). http://www.esann.org/node/8
21. Salay, R., Queiroz, R., Czarnecki, K.: An analysis of ISO 26262: using machine learning safely in automotive software. CoRR abs/1709.02435 (2017)
22. Sämann, T., Schlicht, P., Hüger, F.: Strategy to increase the safety of a dnn-based perception for HAD systems. CoRR abs/2002.08935 (2020)
23. Schorn, C., Guntoro, A., Ascheid, G.: Efficient on-line error detection and mitigation for deep neural network accelerators. In: Gallina, B., Skavhaug, A., Bitsch, F. (eds.) SAFECOMP 2018. LNCS, vol. 11093, pp. 205–219. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99130-6_14
24. Schwalbe, G., Schels, M.: Strategies for safety goal decomposition for neural networks. In: Abstracts 3rd ACM Computer Science in Cars Symposium (2019)
25. Schwalbe, G., Schels, M.: A survey on methods for the safety assurance of machine learning based systems. In: Proceedings of the 10th European Congress on Embedded Real Time Systems (2020)
26. SCSC Assurance Case Working Group: SCSC-141B: Goal Structuring Notation Community Standard (2018). https://scsc.uk/scsc-141B
27. Sun, Y., Wu, M., Ruan, W., Huang, X., Kwiatkowska, M., Kroening, D.: Concolic testing for deep neural networks. In: Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, pp. 109–119. ACM (2018). https://doi.org/10.1145/3238147.3238172
28. Underwriters Laboratories, Edge Case Research: UL4600: Standard for Safety of Autonomous Products. Edge Case Research (2019)
29. Voget, S., Rudolph, A., Mottok, J.: A consistent safety case argumentation for artificial intelligence in safety related automotive systems. In: Proceedings of the 9th European Congress Embedded Real Time Systems (2018)
30. Willers, O., Sudholt, S., Raafatnia, S., Stephanie, A.: Safety concerns and mitigation approaches regarding the use of deep learning in safety-critical perception tasks. CoRR abs/2001.08001 (2020)
31. Wood, M., Robbel, P., Wittmann, D., et al.: Safety First for Automated Driving (2019). http://www.daimler.com/documents/innovation/other/safety-first-for-automated-driving.pdf

Concept Enforcement and Modularization as Methods for the ISO 26262 Safety Argumentation of Neural Networks

Gesina Schwalbe and Martin Schels

Artificial Intelligence and Robotics Laboratory,
Continental AG, Regensburg, Germany
{forename.lastname}@continental-corporation.com

Abstract. Neural networks (NN) are prone to systematic faults which are hard to detect using the methods recommended by the ISO 26262 automotive functional safety standard. In this paper we propose a unified approach to two methods for NN safety argumentation: Assignment of human interpretable concepts to the internal representation of NNs to enable modularization and formal verification. Feasibility of the required concept embedding analysis is demonstrated in a minimal example and important aspects for generalization are investigated. The contribution of the methods is derived from a proposed generic argumentation structure for a NN model safety case.

Keywords: concept enforcement, machine learning, neural networks, functional safety, ISO 26262, goal structuring notation, explainable AI

1 Introduction

The complexity and black-box nature of NNs makes the suggested set of methods from the ISO 26262 automotive functional safety standard insufficient for plausible safety assurance. This is e.g. the case for autonomous driving perception functions, or in general computer vision applications solved with convolutional neural networks (CNN) as a safety critical function, which is the scope of this paper.

We propose a template for the part of the safety case concerning model assessment in order to identify needed methods. One is *modularization* of NNs, i.e. simplification or reduction, e.g. pruning or splitting into environment abstraction and trajectory planning instead of an end-to-end approach. For a useful split it is necessary that the interface of the sub-modules as well as their intended functionality is clear. Instead of using a fixed modular topology as in [25], we propose to utilize inherent modules of a trained NN by identifying interpretable intermediate outputs as splitting points.

One source of evidence for a safety case is formal verification, which requires a formal language on the available input and (intermediate) output of the algorithm in order to define the rules to be verified. A formal language consists of words (human interpretable concepts) and valid relations thereon, which are both seldom available for neural networks. Therefore, we suggest as a second needed method the dynamic *enforcement* of a preselected set of interpretable intermediate outputs. These can then serve as vocabulary for a formal description language and should be build up from semantic *concepts*, i.e. abstract classes of real world objects (e.g. body parts like "foot") or attributes (e.g. textures like "hairy"), which admit real world *relations* (e.g. spatial or hierarchical) between them.

The Contributions of this paper are:

- A generic model assessment part for the safety argument of NN based applications is provided.
- A theory on embeddings of visual semantic concepts into NNs is developed.
- The safety argumentation template is enriched by a unified approach for concept enforcement and modularization which based on the theory of concept embeddings. For each a workflow is suggested.
- An approach for concept embedding analysis is investigated.

2 A Safety Argumentation Structure for Neural Networks

A template for arguing the safety of the intended functionality of NN based systems can be found in [12]. Development of functional safety argumentation started with very basic considerations in [18], revised and more detailed in [24]. Contrary to the latter and inspired by the template suggested in [13, p. 14], we propose to more concretely split between product based (verification and validation) and process based argumentation. In the following a template, for product based safety argumentation of NN based algorithms is suggested with focus on the product argumentation regarding NN specific properties. As in the preliminary work, goal structuring notation [13] is used, which is regarded common practice for safety argumentation [9].

Figure 1 proposes a structure for the NN specific functional safety parts of a safety argumentation. The idea of decomposition is to split into product (**S1**) and process (**G2**) argumentation. Within the process argumentation (**S2**) we locate the need for a modularization strategy for overly complex models (**Sn1**): Due to the black-box character and complexity of NNs, the effort for the corresponding safety argumentation increases exponentially with the size of the network [14].

A NN product argumentation requires special care compared to traditional software models. Reasons are that the intended functionality might be little understood, that a high probability of remaining systematic faults requires proper safety mechanisms (**G3**), and that the absence of safety relevant systematic faults introduced by NN specific problems (**G4**) needs to be proven, but such are not considered in the ISO 26262 standard so far. Figure 1 concentrates on an argumentation strategy which can provide reasonable confidence for the last goal.

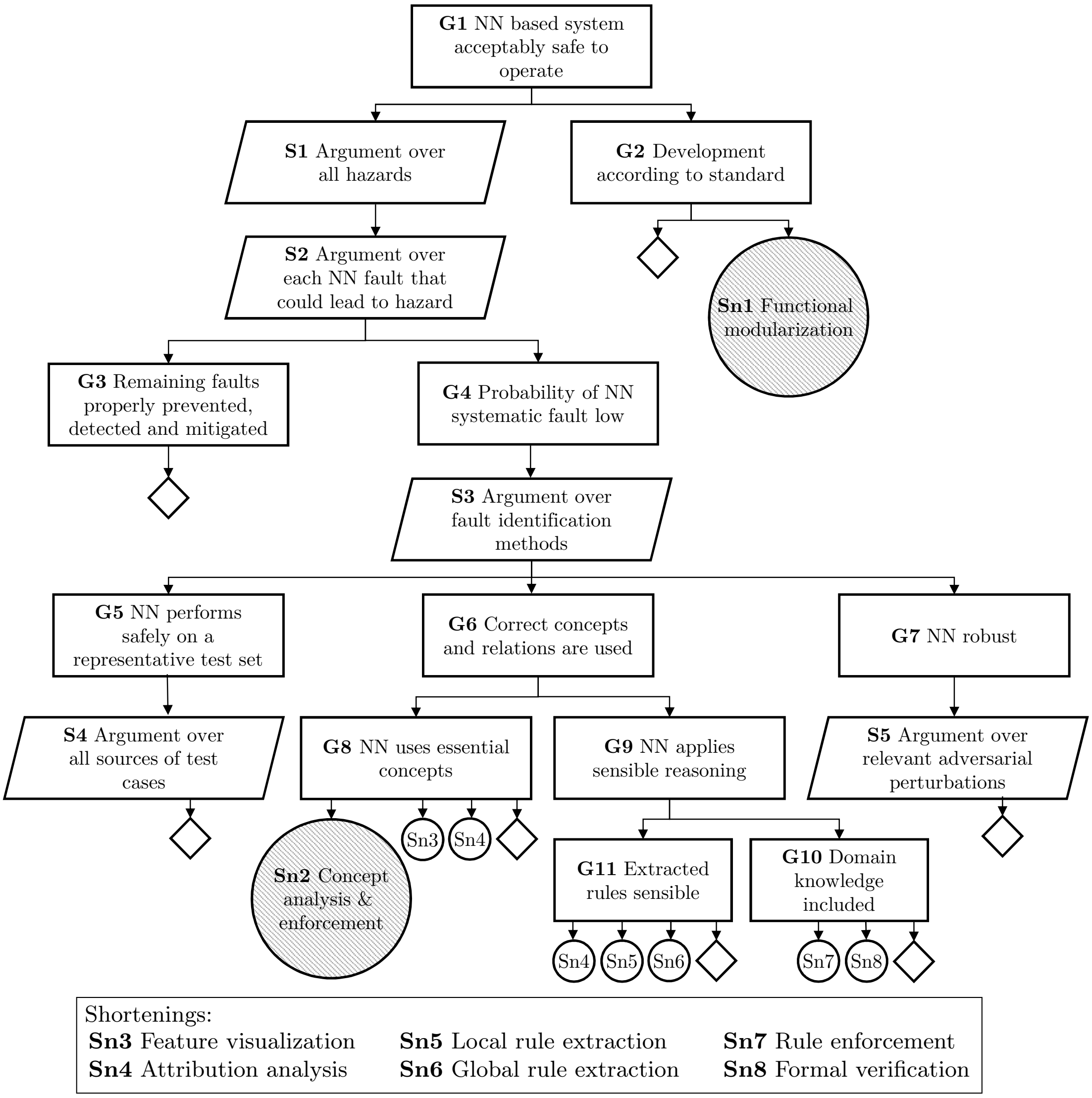

G1 NN based system acceptably safe to operate
S1 Argument over all hazards
G2 Development according to standard
S2 Argument over each NN fault that could lead to hazard
Sn1 Functional modularization
G3 Remaining faults properly prevented, detected and mitigated
G4 Probability of NN systematic fault low
S3 Argument over fault identification methods
G5 NN performs safely on a representative test set
G6 Correct concepts and relations are used
G7 NN robust
S4 Argument over all sources of test cases
G8 NN uses essential concepts
G9 NN applies sensible reasoning
S5 Argument over relevant adversarial perturbations
Sn3
Sn4
Sn2 Concept analysis & enforcement
G11 Extracted rules sensible
G10 Domain knowledge included
Sn4
Sn5
Sn6
Sn7
Sn8
Shortenings:
Sn3 Feature visualization
Sn4 Attribution analysis
Sn5 Local rule extraction
Sn6 Global rule extraction
Sn7 Rule enforcement
Sn8 Formal verification

Fig. 1: Template excerpt from a safety case structure for a NN algorithm, focusing on product argumentation regarding NN model specifics. Sub-trees where we see suggest further methods (modularization and concept enforcement) are colored. Parts: safety (sub-)goals (**G*X***), argumentation strategies (**S*X***), methods to provide evidence (**Sn*X***), and arrows depict dependencies/break downs. Skipped parts are marked with diamonds.

We categorize NNs specific failure modes into three categories: Robustness against small input changes, wrong internal representation or logic of the black-box which we are dealing with in our method proposal, and bad generalization performance due to training data representativity. We suggest that faults of NNs (**S3**) leading to one of those failure modes can be found by one of the following verification and validation approaches: One is thorough testing (**G5**) on systematically acquired test cases (**S4**). Another aspect is robustness assurance (**G7**) for all relevant input perturbations (**S5**). Lastly, due to the complexity of the considered input space, we claim that evidence for a sensible internal structure of the algorithm (**G6**) is required. Sensible here means beneficial or necessary for the intended functionality according to expert judgment. For example classifying speed signs requires the ability to detect and distinguish digits.

The internal structure includes the internal representation (**G8**) and the internal logic (**G9**) applied to it. The latter can either be directly assessed (**G11**) via local (**Sn5**, e.g. [20], [16]) or global (**Sn6**, see [4]) rule extraction, or can be verified by comparison with given rules (**G10**), which can be done by formal verification (**Sn8**, e.g. [7]), or by enforcement of rules (**Sn7**, e.g. via loss [21]). Similar to rule extraction for logic assessment, qualitative methods for inner representation assessment are available. Such are analysis of attention (**Sn4**, e.g. [17]), or analysis of inherent intermediate features (**Sn3**, e.g. [19]). However, we identified a lack of quantitative methods for the analysis and improvement of domain knowledge usage in the inner representation (**Sn2**). Method suggestions for analysis were given by [10, 15]. Our quantitative proposal of concept enforcement builds upon these and fills a gap here.

3 Unified Approach for Concept Enforcement and Modularization

As will be detailed in the following section, semantic concepts are naturally embedded into the internal representations of NNs. We suggest to utilize this property in two directions: Given desirable concepts, assess and enforce their usage. And, knowing which ones are represented internally, use these as splitting points for a modularization strategy.

3.1 Background Theory: Concept Embeddings

Consider the following definitions generalized from [15]. The intermediate output of a NN layer (or, in general, of a set of independent neurons) is a vector within the space of neuron outputs, here called the *space of abstract features* of the layer. Consider a layer, an abstract feature vector w, the projection p_w to the one dimensional sub-vector space spanned by x, and a concept c. The vector w is said to be a *concept embedding* of c if the intermediate output of the NN obtained by concatenation of the layer with p_w has a high correlation with the existence of c (in a certain spatial location). Simply speaking, w is the normal vector of the hyperplane separating the layer output into c or not-c.

It was shown that there often are concepts embedded by the unit vectors (i.e. by single neurons) [6], and that for many task related concepts embeddings can be identified as investigated in [10, 15]. They furthermore found that similar concepts are embedded by close vectors, and vector operations yield meaningful relations on the concepts, and that concept vectors can mostly be chosen sparse [10], i.e. simple basis vector combinations. Sparsity of concepts possibly gives a measure of encoding complexity and of alignment of the layer basis to the human concept base. So, a NN naturally internally represents and uses concepts, and these can be extracted as additional output. Note that previous literature was restricted to feature spaces consisting of neuron outputs of exactly one layer, not general collections of independent neurons, but the definitions and possibly the properties generalize to such.

The above approach to concept embedding analysis gives rise to valuable metrics: The correlation value of a concept embedding serves as quality metric for the embedding. The quality of the best concept embedding for a concept shows *how well the concept is embedded* into the network. Given an input, the additional output by projection to the concept embedding shows *how well the network detected the concept for the given input instance.* And directed derivative along the vector shows *the attribution of the concept towards the decision* [15].

3.2 Concept Enforcement

The inclusion of task related natural language concepts and outputs has been shown to improve the performance of NNs: Loss based examples are fuzzy logic rules on hierarchical concepts [21] and multi-task learning including sub-tasks [11]. Other ways of rule enforcement are e.g. inclusion it into the data [8], topologically [25], or in the case of reinforcement learning safe learning [2]. A possible reason for the positive impact is that natural language in general admits highly efficient abstraction levels for real world tasks [3].

Explainable intermediate output as given by concept embeddings enables one to formulate and automatically verify requirements on the NN. The above suggests that enforcement of output of preselected concepts can be realized without negative impact on the performance, thus qualifies as post-training fine-tuning method. We propose the following workflow:

1. *Identification* of task relevant concepts and relations: We propose for a start two criteria to identify relevant concepts and relations: They are used or needed for synthetic data generators, i.e. are used for the underlying ontology as in [5]. Or, they are used by NNs in similar tasks. Concepts used by NNs can be identified using concept embedding analysis on predefined concepts, or methods like explanatory graph extraction [27] that need manual assignment of natural language descriptions to the found concepts. Used relations can be found by inspecting the embedding vectors or rule extraction methods like [20] which uses inductive logic programming.
2. *Definition of a formal description language* out of these concepts and relations: The previous guidance on vocabulary selection should ensure a domain relevance.

3. *Formulation of rules* using this formal language: This is highly use-case specific and requires domain knowledge. General aspects to derive rules from can be plausibility checks, e.g. derived from physical laws (“pedestrians usually don’t fly”) or hierarchical relations (“children and adults are pedestrians”, “a head usually belongs to a pedestrian”). Or they can be derived from safety bounds, e.g. performance guarantees (“pedestrian detection performance independent of clothing color”) or safe state guarantees (“trajectories keep safe distance from obstacles”).
4. *Verification* of these rules (automatically) using available solvers, a nice overview of which can be found e.g. in [7]: Current approaches differ in performance, restrictions to the network activation functions, and the rules that can be checked; notably upper/lower bound rules verifiable via boundary approximation [26] versus general linear constraints that are solved by satisfiability modulo theory based approaches [7].
5. *Enforcement* of rules if necessary: Of the methods suggested above retraining with counterexamples and a modified loss are most promising for the chosen example use-case discussed later.

3.3 Modularization

Splitting at interpretable intermediate outputs results in several smaller and, hence, less complex sub-networks respectively functional modules, which can be iteratively optimized and verified. This is eased by reducing the number of neurons per sub-network and intersection of such using pruning of the neuron connections before splitting, i.e. deleting ones with low weight. We suggested the following recursive workflow to achieve higher modularization:

1. *Identification of learned concepts* using neuron level analysis, e.g. [6, 10].
2. *Radical pruning* of connections, e.g. weight decay and thresholding [1, 22].
3. *Identification* of the abstraction and interpretation networks using topological introspection (if possible automatically).
4. *Splitting* of the NN into the smaller NNs with the identified interfaces.
5. *Evaluation* and (partly) *retraining* of the pruned and split NN if necessary.
6. *Simplified safety assessment* on the smaller parts.

4 Experiment on Concept Embedding Analysis

The idea on how to obtain a concept embedding vector is to train it and its threshold as parameters for a linear predictor on the model intermediate output. In [15] they use a support vector machine as predictor, which improves the uniqueness of results. The approach in [10] uses a convolutional layer for the prediction introducing translation invariance, easy sparsity analysis, and the possibility of concept segmentation. We base on this setup, since it can directly be used for a multi-task training problem to enforce the desired concept. We tested whether the concept embedding analysis suggested in [10] generalizes to a minimal example and investigated several proposals for improvement.

General Setup. The setup is a simple traffic sign classifier on the subset of 15 classes of the German Traffic Sign Recognition dataset [23] with all images scaled to 48 × 48 px. The visual semantic concepts to analyze were the digits occurring in the five different speed limit sign classes. A concept training respectively testing set are all images containing the digit together with a random sampling of others at a ratio of 7:3. Due to the sign uniformity and translation invariance, the concepts were statically labeled. The pretrained classifier was realized at accuracy of 98.2 % with a feed-forward NN with four convolutional blocks, each ReLU-activated 3 × 3-convolution and 2 × 2 max pooling, then two dense blocks, and a final dense sigmoid layer. Each block is succeeded by a dropout layer. The intermediate output wherein to find a concept embedding was chosen to be a window of 3 × 4 pixels (width × height) in the activation map of the third convolutional layer. This differs from [10] where only one pixel was considered at a time because it proved to be necessary that the receptive field of the considered window (when up-scaling the window pixels to the original image) covers the complete digit, or, more generally, uniquely identifying parts of the concept. For example a "3" and "5" cannot be distinguished by their lower half. Similarly, the training objective we found to generalize best differs fundamentally from [10]: For a position, we want to predict whether it is the center of a window containing the desired concept, other than directly creating a pixel-wise mask. For translation invariance, the window setup was realized by a 3 × 4-convolution with one output filter (the concept output) and sigmoid activation on the output of the layer. The concept embedding vector to train is the weight vector of the convolution kernel. For evaluation of the embedding respectively the associated classifier we used smooth set intersection over union (siou) as in [6, 10] which is calculated as the sum of intersections of the predicted mask $M(x)$ with the ground truth mask $M_{\text{gt}}(x)$ for all samples x in the test set X

$$\text{siou}(X) = \frac{\left(\sum_{x\in X} M_{\text{gt}}(x)M(x)\right) + 1}{\left(\sum_{x\in X} M_{\text{gt}}(x) + M(x) - M_{\text{gt}}(x)M(x)\right) + 1} . \tag{1}$$

Proposals for Improvement. As losses we compared the standard losses binary cross-entropy (bc) and mean squared error (mse) to our new suggestions of negative smooth set IoU from (1) with $M(x)$ non-binary, and a variant derived from the source code of [10] (si), which optimizes for large intersections each of foreground and of background pixels:

$$\mathsf{si}(B) = -\tfrac{1}{\#B} \textstyle\sum_{x\in B}(1-b)M_{\text{gt}}(x)M(x) + b\,(1 - M_{\text{gt}}(x))\,(1 - M(x))$$

where B is a batch and b is the mean number of foreground pixels in the dataset as a weighting factor. This weighting factor (w) as well as smoothing the objective by predicting the intersection over union (IoU) value of a window (iou) were tested for performance benefits. Lastly, we investigated whether convergence could be improved by pretraining the weight vector in a digit classifier setup: A balanced set of windows with and without the digit is presented to the network. Due to the convolution size, this yields one classification output pixel per image.

Loss	**pretrained**	
	yes	**no**
si	0.313	0.016
si-w	0.308	0.138
si-iou	0.305	0.011
si-w-iou	0.386	**0.200**
siou	0.264	**0.325**
siou	0.047[a]	0.093[a]
bc	**0.473**	0.094
bc-w	–	–
bc-iou	**0.421**	0.050
bc-w-iou	–	**0.198**
mse	**0.423**	0.025
mse-w	0.223	0.099

[a] pixels binarized, not bloated

Table 1: Mean smooth set IoU results by setup (⟨loss⟩-⟨modifiers⟩) for concept “3”; gaps are numerically instable.

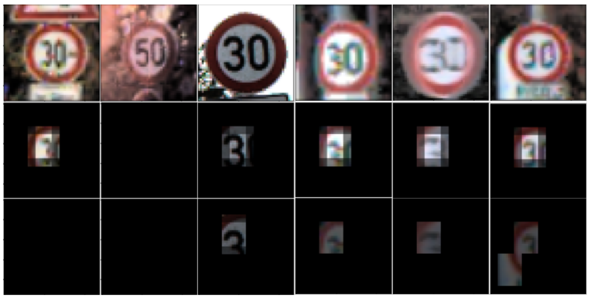


Fig. 2: Top to bottom: Originals, ground truth IoU masks, exemplary outputs of the bc model with weight pretraining.

Results. Table 1 collects the mean set IoU of 10 runs for each setup (variance each below 0.004). For evaluation, the predicted IoU values were turned into an estimated concept area by up-scaling, bloating and then adding the IoU pixel values. For weight pretraining, the mean init model accuracy was 91%. The weighting w was originally suggested to be applied to bc instead of si loss in [10], which we found to be numerically instable. In general, w and iou were shown to only optimize si loss. Pretraining the weights on a concept training set with equal class distribution generally benefits the performance, supporting the suggestions from [10, 15]. Interestingly, pretraining the weights pushes bc and mse from no convergence to best results, suggesting that the losses themselves are weaker aligned with the optimization objective. We claim that this problem is caused by giving little account to spatial distance to the ground truth center, also in the iou setting. This well fits the observation that the direct application of siou loss, where the problem is mitigated, yields best results without any pretraining. Interestingly, siou will focus on finding pixels strictly inside the concept area, which can be seen from the comparison of IoU measurement directly on the output and after bloating the pixels. Finally, we found that all converging methods produce sparse concept vectors, since more than 50 % of the weight entries can be zeroed without inflicting but even increasing the performance. Therefore, we suggest that sparsity could be improved or even enforced based on above methods.

5 Conclusion and Outlook

The two suggested methods and workflows based on concept embedding analysis promise to substantially support a safety argument (Figure 1). The experiment results give guidance on generalizing an analysis approach from [10]. Future work will focus on applying these to the more complex safety relevant use-case of pedestrian detection. The final goal is to finish the goal structuring notation tree for the safety argumentation as a template for safety critical NN applications.

Bibliography

[1] Abbasi-Asl, R., Yu, B.: Interpreting convolutional neural networks through compression. CoRR **abs/1711.02329** (2017)

[2] Akametalu, A.K., Fisac, J.F., Gillula, J.H., Kaynama, S., Zeilinger, M.N., Tomlin, C.J.: Reachability-based safe learning with gaussian processes. In: Proc. 53rd IEEE Conf. Decision and Control, pp. 1424–1431 (2014)

[3] Andreas, J., Klein, D., Levine, S.: Learning with latent language. In: Proc. Conf. North Amer. Chapter of the Assoc. for Computational Linguistics: Human Language Technologies, vol. 1, pp. 2166–2179 (2018)

[4] Augasta, M.G., Kathirvalavakumar, T.: Rule extraction from neural networks — A comparative study. In: Proc. 2012 Int. Conf. Pattern Recognition, Informatics and Medical Engineering, pp. 404–408 (2012)

[5] Bagschik, G., Menzel, T., Maurer, M.: Ontology based scene creation for the development of automated vehicles. In: Proc. 2018 IEEE Intelligent Vehicles Symp., pp. 1813–1820 (2018)

[6] Bau, D., Zhou, B., Khosla, A., Oliva, A., Torralba, A.: Network dissection: Quantifying interpretability of deep visual representations. In: Proc. 2017 IEEE Conf. Comput. Vision and Pattern Recognition, pp. 3319–3327 (2017)

[7] Bunel, R.R., Turkaslan, I., Torr, P., Kohli, P., Mudigonda, P.K.: A unified view of piecewise linear neural network verification. In: Advances in Neural Information Processing Systems 31, pp. 4790–4799 (2018)

[8] Dreossi, T., Ghosh, S., Yue, X., Keutzer, K., Sangiovanni-Vincentelli, A.L., Seshia, S.A.: Counterexample-guided data augmentation. In: Proc. 27th Int. Joint Conf. Artificial Intelligence, pp. 2071–2078 (2018)

[9] Duffau, C., Polacsek, T., Blay-Fornarino, M.: Support of justification elicitation: Two industrial reports. In: Advanced Information Systems Engineering, vol. 10816, pp. 71–86 (2018)

[10] Fong, R., Vedaldi, A.: Net2Vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. In: Proc. 2018 IEEE Conf. Comput. Vision and Pattern Recognition, pp. 8730–8738 (2018)

[11] Fuchs, F.B., Groth, O., Kosiorek, A.R., Bewley, A., Wulfmeier, M., Vedaldi, A., Posner, I.: Neural Stethoscopes: Unifying analytic, auxiliary and adversarial network probing. CoRR **abs/1806.05502** (2018)

[12] Gauerhof, L., Munk, P., Burton, S.: Structuring validation targets of a machine learning function applied to automated driving. In: Computer Safety, Reliability, and Security, pp. 45–58 (2018)

[13] Group, S.A.C.W.: Goal Structuring Notation Community Standard, jan 2018 edn. (2018)

[14] Johnson, C.W.: The increasing risks of risk assessment: On the rise of artificial intelligence and non-determinism in safety-critical systems. In: Evolution of System Safety: Proc. Safety-Critical Systems Symp. (2017)

[15] Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., Sayres, R.: Interpretability beyond feature attribution: Quantitative testing with

concept activation vectors (TCAV). In: Proc. 35th Int. Conf. Machine Learning, vol. 80, pp. 2668–2677 (2018)

[16] Kim, J., Rohrbach, A., Darrell, T., Canny, J.F., Akata, Z.: Textual explanations for self-driving vehicles. In: Proc. 15th European Conf. Comput. Vision, Part II, vol. 11206, pp. 577–593 (2018)

[17] Kindermans, P.J., Schütt, K.T., Alber, M., Müller, K.R., Erhan, D., Kim, B., Dähne, S.: Learning how to explain neural networks: PatternNet and PatternAttribution. In: Proc. 6th Int. Conf. on Learning Representations (2018)

[18] Kurd, Z., Kelly, T.: Establishing Safety Criteria for Artificial Neural Networks. In: Knowledge-Based Intelligent Information and Engineering Systems, pp. 163–169 (2003)

[19] Olah, C., Mordvintsev, A., Schubert, L.: Feature visualization. Distill **2**(11), e7 (2017)

[20] Rabold, J., Siebers, M., Schmid, U.: Explaining black-box classifiers with ILP – empowering LIME with Aleph to approximate non-linear decisions with relational rules. In: Proc. Int. Conf. Inductive Logic Programming, pp. 105–117 (2018)

[21] Roychowdhury, S., Diligenti, M., Gori, M.: Image classification using deep learning and prior knowledge. In: Workshops of the 32nd AAAI Conf. Artificial Intelligence, vol. WS-18, pp. 336–343 (2018)

[22] Setiono, R., Liu, H.: Symbolic representation of neural networks. IEEE Comput. **29**, 71–77 (1996)

[23] Stallkamp, J., Schlipsing, M., Salmen, J., Igel, C.: The German Traffic Sign Recognition benchmark: A multi-class classification competition. In: Proc. 2011 Int. Joint Conf. Neural Networks, pp. 1453–1460 (2011)

[24] Voget, S., Rudolph, A., Mottok, J.: A consistent safety case argumentation for artificial intelligence in safety related automotive systems. In: Proc. 9th European Congress on Embedded Real Time Systems (2018)

[25] Wang, H.: ReNN: Rule-embedded neural networks. In: Proc. 24th Int. Conf. Pattern Recognition, pp. 824–829 (2018)

[26] Wang, S., Pei, K., Whitehouse, J., Yang, J., Jana, S.: Formal security analysis of neural networks using symbolic intervals. In: Proc. 27th USENIX Security Symp., pp. 1599–1614 (2018)

[27] Zhang, Q., Cao, R., Shi, F., Wu, Y.N., Zhu, S.C.: Interpreting CNN knowledge via an explanatory graph. In: Proc. 32nd AAAI Conf. Artificial Intelligence, pp. 4454–4463 (2018)