Otto-Friedrich-Universität Bamberg

Doctoral Thesis

Explainable and Interactive Link Prediction in Knowledge Graphs

Author:
Christoph Wehner

Committee:
Prof. Dr. Daniela Nicklas (Head of Committee)
Prof. Dr. Ute Schmid (First Examiner and Supervisor)
Prof. Dr. Stefan Ultes (Second Examiner)

A thesis submitted in partial fulfillment of the requirements for the degree of
Doctor of Natural Sciences (Dr. rer. nat.)
at the

Fakultät Wirtschaftsinformatik und Angewandte Informatik
Otto-Friedrich-Universität Bamberg

June 12, 2026

Bamberg 2026

Diese Arbeit hat der Fakultät Wirtschaftsinformatik und Angewandte Informatik der Otto-Friedrich-Universität Bamberg als Dissertation vorgelegen.
Gutachterin: Prof. Dr. Ute SCHMID
Gutachter: Prof. Dr. Stefan ULTES
Vorsitzende der Promotionskommission: Prof. Dr. Daniela NICKLAS
Tag der mündlichen Prüfung: 09.06.2026

URN: urn:nbn:de:bvb:473-irb-115564x

DOI: https://doi.org/10.20378/irb-115564

"No man is an island entire of itself; every man is a piece of the continent, a part of the main; ..."

MEDITATION XVII
Devotions upon Emergent Occasions
John Donne

Zusammenfassung

"Erklärbare und Interaktive
Kantenvorhersage in Wissensgraphen"

Maschinelle Lernmodelle verlassen zunehmend die sicheren Mauern der Forschungslabore und werden in risikoreichen Anwendungen eingesetzt. In diesen Umgebungen müssen Nutzerinnen und Nutzer verstehen, warum sich ein Modell so verhält, wie es sich verhält, und es bei Bedarf korrigieren können. Eines dieser Anwendungsgebiet ist die Root Cause Analysis (RCA) in der Fertigung, bei der datengetriebene Modelle die Ursachen von Fehlern in zeitkritischen Umgebungen identifizieren sollen. Link-Prediction-Modelle in Knowledge Graphs (KGs) können für diese Aufgabe eingesetzt werden. Solche Modelle, insbesondere embedding-basierte Link-Prediction-Modelle, sind jedoch oft Black-Boxes. Dies schränkt die menschliche Kontrolle ein, erschwert die Validierung und mindert das Vertrauen in ihre Entscheidungen.

Diese Dissertation befasst sich damit, die internen Schlussfolgerungsprozesse von Link-Prediction-Modellen mit menschlichem Feedback in Einklang zu bringen. Sie schlägt ein technisches und konzeptionelles Vorgehen vor, bei dem Modelleklärungen als Schnittstellen zur Erfassung und Integration von Nutzerfeedback dienen. Dadurch wird das Modellverhalten schrittweise entsprechend menschlichen Feedbacks angepasst.

Drei Forschungsfragen gliedern die Arbeit:

(1) Wie kann der Entscheidungsprozess eines Link-Prediction-Modells so erklärt werden, dass die Erklärung eine nachgelagerten Modellanpassungen unterstützt?

(2) Welche Arten von Nutzerfeedback zu Erklärungen sind wirksam, um Modellanpassungen zu steuern?

(3) Wie kann Feedback zu Modellerklärungen in den Trainings- und Inferenzprozesse eines Link-Prediction-Modell integriert werden?

Zur Beantwortung dieser Fragen werden in der Arbeit drei zentrale Beiträge geleistet:

Erstens wird KGEPrisma vorgestellt, eine post-hoc Erklärungsmethode für embedding-basierte Link-Prediction-Modelle. KGEPrisma übersetzt latente Embeddings in instanzbasierte, analogiebasierte und regelbasierte Erklärungen. Diese Methode ermöglicht eine umfassende Sicht auf Modellentscheidungen.

Zweitens präsentiert die Arbeit RootFinder, ein System, das RCA als Link-Prediction-Aufgabe in einem Fertigungs-KG interpretiert. RootFinder ermöglicht es Domänenexpertinnen und -experten, Modelleklärungen zu bewerten und einfaches, auf den Anwendungskontext zugeschnittenes Feedback zu geben. Diese Anwendung motiviert eine vertiefte Diskussion über Feedbackmodalitäten, die sich etwa für den Einsatz in der Praxis besser skalieren lassen.

Drittens wird LiEr eingeführt, ein Reinforcement-Learning-Ansatz, der iterativ, mithilfe von präferenzbasiertem Feedback zu Modellerklärungen, lernt fehlende Kanten im KG vorherzusagen. LiEr formuliert Knowledge Graph Completion (KGC) als Markov Decision Process (MDP) und integriert präferenzbasiertes Nutzerfeedback in die Belohnungsfunktion. Dadurch kann das Modell Strategien erlernen, die sich menschlich nachvollziehbaren Schlussfolgerungen annähern. LiEr wird auf RCA in der Elektrofahrzeugfertigung angewendet. Die Fallstudie zeigt, dass präferenzbasiertes Feedback über Erklärungen eine Angleichung an Domänenwissen ermöglicht.

Insgesamt bietet diese Arbeit eine methodische und empirische Grundlage für den Einsatz von Erklärungen, um menschliches Feedback zu erfassen und das Verhalten von Link-Prediction-Modellen an menschliches Feedback anzupassen. Erklärungen können einen interaktiven Lernprozess ermöglichen. Diese Dissertation stellt die erste Arbeit zu interaktiver und erklärbarer KGC dar.

Abstract

Machine learning models move from the safe walls of research labs to high-risk real-world applications. In these environments, users need to understand why a machine learning model behaves as it does and correct it if necessary. One critical application domain is RCA in manufacturing, where data-driven models identify fault causes in time-critical environments. Link prediction models in KGs can be used to tackle this task. However, KGC models, particularly embedding-based link predictors, often operate as black-boxes. This limits human oversight, complicates validation, and undermines trust.

This thesis addresses the challenge of making link prediction models correct for the correct reasons with the help of human feedback. Specifically, it proposes a technical and conceptual pipeline in which model explanations serve as interfaces for collecting and incorporating user feedback to iteratively align model behaviour with human intent.

Three research questions are addressed in this work:

(1) How can the decision-making process of a link prediction model be explained in ways that support downstream alignment?

(2) What types of user feedback on explanations are effective for guiding model updates?

(3) How can feedback on model explanations be integrated into the training and inference processes of link prediction models?

In response, three major contributions are made in this thesis:

First, KGEPrisma is introduced, a post-hoc explanation method for embedding-based link prediction models. KGEPrisma translates latent embeddings into instance-based, analogy-based, and rule-based explanations. This method provides a multi-perspective view on model decisions and establishes a foundation for the subsequent exploration of feedback modalities.

Second, RootFinder is presented, a system that treats RCA as a link prediction task in a manufacturing KG. RootFinder enables domain experts to assess model explanations and to provide simple, customised feedback for real-world settings. This application motivates an in-depth discussion of feedback modalities.

Finally, LiEr is introduced in the thesis. LiEr is a reinforcement learning approach that iteratively aligns link prediction with preference-based feedback over model explanations. LiEr reformulates KGC as a MDP and incorporates preference-based user feedback into the reward function. This enables the model to learn policies that converge toward human-plausible reasoning. LiEr is applied to RCA in electric vehicle manufacturing. The case study confirms that preference-based feedback over explanations enables root cause prediction alignment with domain knowledge.

Together, these contributions provide a methodological and empirical foundation for using explanations to collect human feedback and align the behaviour of link prediction models with it. Explanations enable an interactive learning loop. This thesis presents the first work on interactive and explainable KGC.

Acknowledgments

My biggest gratitude goes to my parents. You unconditionally believed in me and supported me along this long path. Without you, I would not have had the opportunity to try out so many new things and discover where my true passion lies. I love you.

I also want to sincerely thank Prof. Dr. Ute Schmid. You gave me freedom in my research and opened so many doors for me. I will always be thankful for your trust and support. After warmly welcoming me to your research group, the second door you opened was introducing me to the IBM Watson Centre Munich team. There, I meet amazing people like Francis Powlesland.

Francis, thank you for always taking the time for me. Your patience meant a lot to me. I learned so much from our discussions about coding, software and how to deliver projects. It was truly a pleasure to work with you.

Through Ute, I also had the opportunity to be part of the research project KIProQua with the BMW Group. This brought me into contact with so many great people who shaped my research, and also me as a person. Simon Schramm, thank you for the exchange on knowledge graphs and for taking me to many events and meetings at BMW. Those experiences left a mark and shaped my research. Lukas Bahr, thank you for being around whenever it mattered. You were there when things got tough, professionally and personally. Judith Wewerka, thank you for your constant support and for your feedback on so many of the things I worked on. Maximilian Kertel, thank you for the cooperation on RootFinder. You helped bring the idea to life and gave it direction.

Finally, Ute also introduced me to Tarek R. Besold, and for that I am incredibly grateful. Tarek, thank you for the opportunity to join your team at SonyAI. Working on biomedical hypothesis generation and knowledge graph completion in an international environment, based in Barcelona and visiting Tokyo, was an unforgettable once-in-a-lifetime experience. You made it possible for me to see more of the world and to grow as a researcher and as a person. There, I met Chrysa Iliopoulou. Thank you for teaching me what it means to write clean, robust software, for your support on the Sony project, and for being a friend. Also, at SonyAI, I met Hatem Elshazly. Thank you for your emotional support and your calm and steady presence during times of pressure.

There were also many other people whose presence meant a great deal to me during this time. Pedro Giesteira Cotovio, thank you for your emotional support, the exchange, and for inviting me to spend time with your group at the University of Lisbon. That experience broadened my horizon in many ways. Jonas Troles, thank you for being there and for the conversations that helped me stay grounded. This journey would have been much harder without all of you. Thank you.

Contents

Zusammenfassung iii

Abstract v

Acknowledgments vii

Contents xi

List of Figures xiii

List of Tables xv

List of Algorithms xvii

1 Introduction 1
- 1.1 The Research Questions, Objectives and Contributions . . . 3
  - 1.1.1 The Research Questions . . . 4
  - 1.1.2 The Objectives . . . 5
  - 1.1.3 The Contributions . . . 6
- 1.2 Related Work . . . 7
  - 1.2.1 Root Cause Analysis in Manufacturing . . . 7
  - 1.2.2 Link Prediction in Knowledge Graphs . . . 8
  - 1.2.3 Explainability . . . 10
  - 1.2.4 Artificial Intelligence Alignment . . . 12
- 1.3 Synopsis . . . 13

2 Foundations 17
- 2.1 Knowledge Graphs . . . 17
- 2.2 Link Prediction in Knowledge Graphs . . . 18
- 2.3 Explainability . . . 20
  - 2.3.1 Metrics for Explanation Quality . . . 21
- 2.4 Artificial Intelligence Alignment . . . 22

3 Explaining Link Prediction in Knowledge Graphs 24
- 3.1 Explainability for Knowledge Graph Completion . . . 24
  - 3.1.1 Instance-Based Explanations . . . 25
  - 3.1.2 Saliency-Based Explanations . . . 27
  - 3.1.3 Counterfactual Explanations . . . 29
  - 3.1.4 Analogy-Based Explanations . . . 31
  - 3.1.5 Rule-Based Explanations . . . 33
  - 3.1.6 Combining Explanation Modalities to Exploit Strengths and Mitigate Drawbacks . . . 34
- 3.2 Generating Explanations for Knowledge Graph Embedding Models with KGEPrisma . . . 34
  - 3.2.1 From Latent Representations to Rules, Analogies and Instance-Based Explanations . . . 36
  - 3.2.2 Evaluation . . . 44
- 3.3 Summary and Answer to Research Question 1 . . . 54

4 On Feedback Modalities for Aligning Link Prediction in Knowledge Graphs 56
4.1 Interactive Root Cause Analysis in Electric Vehicle Manufacturing with Causal Bayesian Networks and Knowledge Graphs . . . 57
4.1.1 System Overview . . . 59
4.1.2 The Manufacturing Knowledge Graph . . . 60
4.1.3 Temporal-Spatial Relations and Variable Subclass Properties . . . 62
4.1.4 Causal Bayesian Network . . . 64
4.1.5 Interactive User Interface . . . 65
4.1.6 Evaluation . . . 67
4.2 Discussion on Feedback Modalities for Downstream Model Alignment . . . 69
4.2.1 Positive Example as Feedback Modality . . . 70
4.2.2 Negative Examples as Feedback Modality . . . 71
4.2.3 Rules as Feedback Modality . . . 73
4.2.4 Ratings as Feedback Modality . . . 74
4.2.5 Preferences as Feedback Modality . . . 76
4.3 Summary and Answer to Research Question 2 . . . 78

5 Iteratively Aligning Link Prediction with Human Intent 80
5.1 Aligning Path-based Link Prediction with Human Understanding of Valid Reasoning with LiEr . . . 81
5.1.1 Definitions . . . 83
5.1.2 From Knowledge Graph to Markov Decision Process . . . 85
5.1.3 Solving the Markov Decision Process . . . 86
5.1.4 Aligning the Policy Network with Human Understanding of Valid Reasoning via the Reward . . . 87
5.2 Evaluating LiEr‘s Predictive Performance and Reasoning Alignment . . . 90
5.2.1 Experimental Setup . . . 91
5.2.2 Evaluation Metrics . . . 93
5.2.3 Benchmark Comparison . . . 94
5.2.4 Evaluating LiEr's Alignment with Valid Reasoning . . . 100
5.2.5 Remarks on the Collection of Preference-based Feedback . 102
5.2.6 Discussion and Limitations . . . 103
5.2.7 LiEr and Research Question 3 . . . 104
5.3 Applying LiEr to Root Cause Analysis in Electric Vehicle Manufacturing . . . 105
5.3.1 Manufacturing Data as a Knowledge Graph . . . 106
5.3.2 Synthetic Root Cause Data Generation . . . 107
5.3.3 Measurement-Based Feature Construction . . . 109
5.3.4 Evaluation . . . 110
5.3.5 Limitations and Lessons Learned . . . 111
5.3.6 Discussion of Applying LiEr to Root Cause Analysis . . . 113
5.4 Summary and Answer to Research Question 3 . . . 113

6 Conclusion **115**
6.1 Summary 115
6.2 Discussion 116
6.3 Open Potentials 117

References **119**

Appendix **136**

A Peer-Reviewed Publications **136**
A.1 Comprehensible Artificial Intelligence on Knowledge Graphs: A survey 136
A.2 Interactive and Intelligent Root Cause Analysis in Manufacturing with Causal Bayesian Networks and Knowledge Graphs 136
A.3 ManuKnowVis: How to Support Different User Groups in Contextualizing and Leveraging Knowledge Repositories 137
A.4 Literature-based Hypothesis Generation: Predicting the evolution of scientific literature to support scientists 137
A.5 Knowledge graph enhanced retrieval-augmented generation for failure mode and effects analysis 138
A.6 Anwendung von Causal-Discovery-Algorithmen zur Root-Cause-Analyse in der Fahrzeugmontage 138

B Scientific activities **140**
B.1 Non-Peer-Reviewed Publications 140
B.1.1 From Latent to Lucid: Transforming Knowledge Graph Embeddings into Interpretable Structures with KGEPrisma 140
B.1.2 Aligning Path-based Link Prediction with Human Understanding of Valid Reasoning 140
B.2 Reviews 141
B.3 Advised Thesis 141
B.4 Other Activities 141

List of Figures

1 Illustration of root cause analysis setting in manufacturing 1
2 Reframing root cause analysis as link prediction in a knowledge graph 2
3 Overview of the thesis' approach to explainable and interactive link prediction in knowledge graphs 3
4 Root cause analysis workflow from problem identification to corrective action 8
5 Example of two entities and a relation represented in a 2D space 9
6 Explanation of a convolutional network prediction for an image of a cat 11
7 Example of the PrimeKG knowledge graph 18
8 Path and subgraph theme used to explain a predicted link 25
9 Saliency-based explanations 27
10 Visualization of a counterfactual explanation 30
11 An analogy-based explanation 32
12 Overview KGEPrisma 35
13 The perturbation-based faithfulness evaluation protocol 49
14 Comparison of two different explanations for one predicted triple 50
15 Instance-based and analogy-based explanation of a predicted triple 51
16 Runtime evaluation of KGEPrisma 53
17 The system architecture of RootFinder 58
18 Schema of the RootFinder manufacturing KG 60
19 Example of the RootFinder manufacturing KG 61
20 Possible cause effect relations among sensor variables in the manufacturing knowledge graph 61
21 Possible cause effect relations among sensor variables with the process structure taken into account 61
22 The true root cause graph among the sensor variables, ... 61
23 The user interface of RootFinder 66
24 Evaluation results of RootFinder 67
25 The positive example feedback modality 70
26 The negative example feedback modality 72
27 The rating feedback modality 75
28 The preference feedback modality 77
29 Minimal example of a knowledge graph where a path-based predictor infers a missing relation 82
30 Schematic overview of LiEr 82
31 The architecture of LiEr‘s policy network 87
32 Screenshot of LiEr's CLI 102
33 Performance of LiEr with increasing levels of erroneous preferences 103
34 An excerpt of MP07 represented as a KG 107
35 The schema of the production line represented as a KG. 108
36 Rewards of LiEr on the MP07 KG 112

List of Tables

1 Summary of included publications in this thesis 15
2 Overview of the three link prediction tasks in knowledge graphs . 19
3 Pros and cons of instance-based explanations for the downstream task of model alignment. 26
4 Pros and cons of saliency-based explanations for the downstream-task of model alignment 29
5 Pros and cons of counterfactual explanations for the downstream-task of model alignment. 30
6 Pros and cons of analogy-based explanations for the downstream-task of model alignment 32
7 Pros and cons of rule-based explanations for the downstream-task of model alignment 33
8 Examples for the explanation modalities generated by KGEPrisma 37
9 Comparison of explanation modalities generated by KGEPrisma 42
10 Results for KGEPrisma on FB15k-237 48
11 Results for KGEPrisma on WN18RR 50
12 Results for KGEPrisma on Kinship 52
13 Alignment risk assessment of the positive example feedback modality. 71
14 Alignment risk assessment of the negative example feedback modality. 72
15 Alignment risk assessment of the rule-based feedback modality. . 74
16 Alignment risk assessment of the rating feedback modality. . . . 76
17 Alignment risk assessment of the preference feedback modality. . 78
18 Three example reasoning paths for one query 89
19 Reasoning patterns required to solve the Countries tasks 94
20 MRR for tasks S1, S2, and S3 in the Countries knowledge graph 96
21 MRLR for tasks S1, S2, and S3 in the Countries knowledge graph 96
22 MRR for the Family, Sports, and Locations knowledge graphs . . 98
23 MRLR for the Family, Sports, and Locations knowledge graphs . 98
24 MRR for the Clever Hans Countries knowledge graph 100
25 MRLR for the Clever Hans Countries knowledge graph 100
26 Top three reasoning paths and their probabilities for MINERVA, NeuralLP, and LiEr 101
27 Entity types in the MP07 KG. 106
28 Relation types in the MP07 KG. 106
29 Types of synthetic root causes and their number of occurrences. . 109
30 Final hyperparameter configuration used for evaluation of LiEr on the MPO7 KG 111
31 Evaluation results of LiEr for RCA 111

List of Algorithms

1 Clause Mining and Frequency Calculation 39

List of Abbreviations

AI Artificial Intelligence. 12, 13, 15, 22, 23, 116

aSHD adapted Structural Hamming Distance. 68

CAM Causal Additive Model. 65

CBN Causal Bayesian Network. 7, 8, 14, 15, 57–68

CER Cause-Effect Relationship. 7, 8, 57, 59–68

FMEA Failure Mode and Effect Analysis. 7

GNN Graph Neural Network. 11

HSIC-Lasso Hilbert-Schmidt Independence Criterion Lasso. 40–42, 46

ILP Inductive Logic Programming. 9, 17

IML Interpretable Machine Learning. 20

KG Knowledge Graph. iii, v, xiii, xv, 1–4, 6, 8–20, 24–29, 31–36, 43–46, 50–54, 56–68, 70–78, 80–88, 90, 92, 94, 96–100, 102, 104–114, 116, 117, 137, 138

KGC Knowledge Graph Completion. iii–v, 2, 3, 5–11, 13–15, 18–20, 23, 24, 28–36, 43, 44, 56, 58–60, 69, 70, 72–78, 80, 81, 83, 93–95, 100, 103, 105, 115–118

KGE Knowledge Graph Embedding. 15, 16, 34–50, 52–55, 93, 115

LRP Layer-wise Relevance Propagation. 11, 27, 35

LSTM Long Short-Term Memory. 86, 87, 111

MDI Mean Decrease in Impurity. 40, 41, 46

MDP Markov Decision Process. iii, v, 10, 14, 80–82, 85, 86, 110, 114, 116

ML Machine Learning. 1, 2, 5, 7, 8, 10–12, 15, 16, 18, 20–23, 33, 35, 37, 59

MRLR Mean Reciprocal Localisation Rank. 6, 7, 14, 16, 93–101, 114, 116

MRR Mean Reciprocal Rank. 93–101, 114

RCA Root Cause Analysis. iii, v, xv, 1, 2, 5–8, 13–16, 18–20, 57–61, 63, 66–69, 80, 81, 105–108, 110–116, 139

RL Reinforcement Learning. 10, 13

ROI Region of Interest. 92, 94–99, 103, 112, 117

SEM Structural Equation Model. 64, 65

XAI Explainable Artificial Intelligence. 10–12, 15, 16, 20–22, 24, 27, 35, 46, 48, 49, 52, 116

List of Symbols

C A set of classes. 17, 38, 39

D A dictionary to map each entity pair to its clauses and frequencies. 39–41

E A set of entities. 8, 9, 17–19, 38, 39, 83–85

G A knowledge graph. 8, 17–20, 37–39, 46–48, 83

N A set of triples with similar embeddings. 37

P A set of filtered entity pairs. 38–40, 43

R A set of relations. 8, 9, 17–19, 83–85

T A set of triples. 46–48

V A set of embeddings. 37

Λ A human oracle. 88–90

Σ An alphabet. 17, 18, 83, 84

α An action embedding. 86, 88

δ A transition function. 85

ℓ A language mapping usually from an edge or relation to a symbol of an alphabet. 17, 18, 83, 84

ϵ A normally distributed noise term. 64, 65

η A hidden state of a recurrent neural network. 86–88, 110

κ A reasoning step, consisting of a state and action. 86, 88

$\mathcal{A}$ An action space. 85–87

$\mathcal{G}$ A knowledge graph as a quiver. 17

$\mathcal{R}$ A reward function. 85–87

$\mathcal{S}$ A state space. 85, 87

ϕ An interaction function of a knowledge graph embedding model. 9, 27, 45–47

π A policy. 86

ρ A tail prediction mapping. 84

σ A symbol of an edge or relation. 85, 88, 90, 110

τ A scalar reward. 88–90

$\varkappa$ A reasoning step embedding. 88, 90

ς A state embedding. 86, 88

ζ A semantic schema of a knowledge graph. 17, 38, 39

a An action. 85, 86, 88

c A conjunctive clause. 39, 40

d A policy step. 86

e An entity. 8, 17–19, 24–26, 29, 31, 33, 36–39, 41, 45–47, 84–86, 110

h A head mapping for relations. 17, 18, 83–85

r A relation. 8, 17–19, 24–26, 29, 31, 33, 36–38, 45–47, 84, 85, 88, 90, 110

s A state. 85, 86, 88, 110

t A tail mapping for relations. 17, 18, 83–85

v An embedding. 31, 37, 40, 41, 45, 46

w A tuple/sequence of entities and relations. 38, 39, 86, 88, 90

1 Introduction

Machine Learning (ML) models gradually move outside the safe walls of research labs and invade our daily lives. With that, the predictions and decisions made by ML models have impactful consequences (Schramm et al., 2023). Consider the real-world application of RCA in the manufacturing domain (Oliveira et al., 2022). RCA is about identifying the manufacturing step where the error occurred after a faulty part has been detected during the quality control (ISO/IEC, 2019) (see Figure 1). Traditionally, RCA is a manual, labor-intense, and thus expensive process (Serrat, 2017; Tay and Lim, 2008; Ruijters and Stoelinga, 2015). It involves the design of experiments to study manipulations and their effects. RCA requires high amounts of process knowledge. As a result, only process experts can conduct a RCA in a reasonable timeframe. With the emergence of the Industrial Internet of Things and Big Data, ML approaches were employed to make RCA data-driven, for example decision trees, regression models, or Bayesian networks (Oliveira et al., 2022; Wehner et al., 2023).

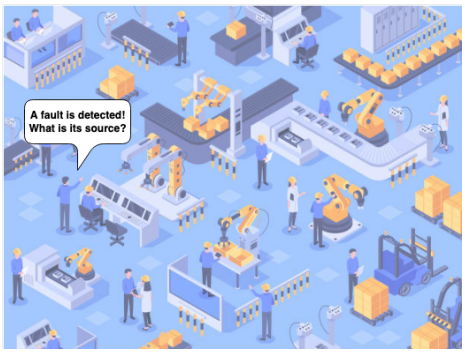


Figure 1: A defective part is detected at a test step of a manufacturing process. The challenge of RCA is to determine the origin of the fault within the manufacturing process.

The ML models identify potential root causes of faulty parts by analyzing manufacturing data. This technology aims to point engineers to the sources of a faulty part directly, without lengthy manual analysis. It enables the engineers to target the error source directly, which eventually results in a reduced production downtime. However, the models rely purely on data and do not utilize any existing expert knowledge. Nevertheless, this unused knowledge holds large potential.

KGs are an expressive tool to formalize and enrich knowledge for downstream ML tasks. KGs store symbolic facts by representing data as nodes and relations in a directed, labeled multigraph (Hogan et al., 2022; Nickel et al., 2011). The symbolic topology of the KG contextualizes the data. Contextualization allows reasoners to derive known and novel insights from the data (Hogan et al., 2022; Schramm et al., 2023). In particular, the ML task of link prediction involves identifying links that were previously unknown between different nodes within the KG (Zhang et al., 2021). The modelling of the manufacturing process, including expert knowledge, as a KG allows framing the RCA as a link prediction task. A missing link is to be predicted from a node representing a faulty variable to another node representing the process step where the fault was induced. This

can be used to combine expert knowledge about the manufacturing process and root causes with the pattern learning capabilities of ML models. Thus, to reframe the RCA to a link prediction task unlocks the pattern-matching capabilities of data-driven ML approaches for RCA and the benefits of the knowledge corpus of process experts formalized in a KG.

Figure 2 illustrates how RCA can be reframed as a link prediction task. The manufacturing process is represented as a KG where nodes encode stations, parts, process steps, or measurements, and edges represent functional dependencies or interactions. Once a defective part is identified, the link prediction model is tasked with predicting a plausible missing root cause link pointing to the source of the fault. This reframing leverages the symbolic structure of KGs and the pattern-matching ability of machine learning models and allows for incoperation of expert knowledge.

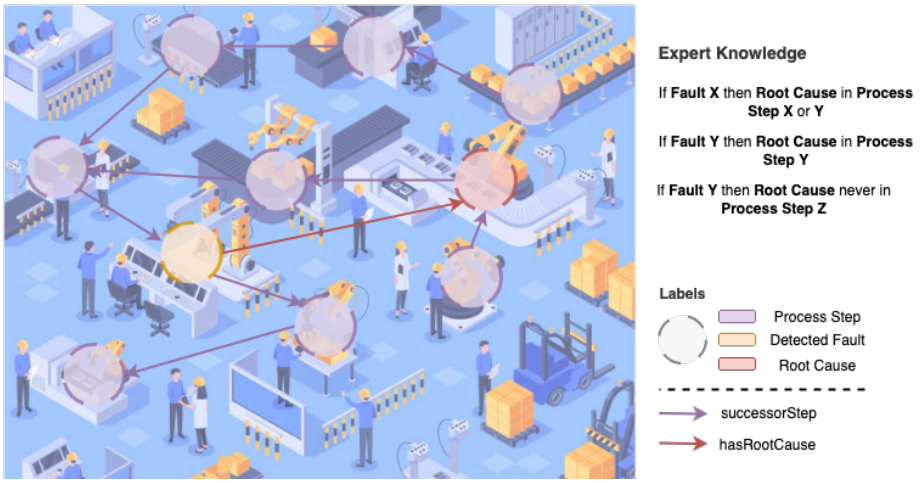


Figure 2: An illustration of reframing RCA as a link prediction task. The manufacturing process is formalized as a KG, where nodes represent process steps and edges represent relations between them. Once a defect is detected, the goal is to predict a missing link from the defective part to the most likely root cause. The nodes in the graph represent process step entities. The purple arrows represent the relation `successorStep`. The red arrow visualizes the predicted link from the faulty part to its potential root cause, which was previously missing from the KG and is inferred by the KGC model. Expert knowledge can be expressed as rules that describe plausible causal chains and used to support link prediction.

Occasionally, a link prediction model, like any other ML model, is wrong. This results in engineers searching for the error source in the wrong place and in increased downtime for the manufacturing process. Engineers therefore need to be able to sanity check the link prediction model's decision-making process. This allows them to assess if the predicted root cause is trustworthy. The need for explainability of the link prediction model emerges.

Additionally, more than plain explainability is needed for a domain expert to gain justified trust in a link prediction model. After the process expert observed an invalid, and thus unintended, decision-making process, trust in the model's predictions can only be restored if the model's decision-making process is corrected and once again aligns with the model's intended behavior (Turek,

2018). Thus, the domain expert has to be empowered to interact with the model iteratively and to align it whenever it is wrong. Suppose a root cause prediction is false, or the decision-making process is invalid and thus untrustworthy. In that case, the domain expert wants to inject their knowledge into the link prediction model to improve its decision-making process such that future predictions are correct. Feedback on the KGC model's decision-making process to improve it is thus the next step following explainability. This motivates the thesis and its research question to explore the underlying methodological aspects of using explanations to iteratively align KGC models.

1.1 The Research Questions, Objectives and Contributions

This section outlines the research questions, objectives, and contributions of the thesis. The research is motivated by the need for explainable link prediction models that interactively align their decision-making process with human expectations. To this end, the thesis investigates how explanations can support users in understanding model decisions, how different forms of user knowledge can be captured as feedback on explanations, and how such feedback can be incorporated to iteratively refine the KGC model's behavior. The resulting approaches enable human-in-the-loop KGC that is interpretable and iteratively adaptable to domain expertise.

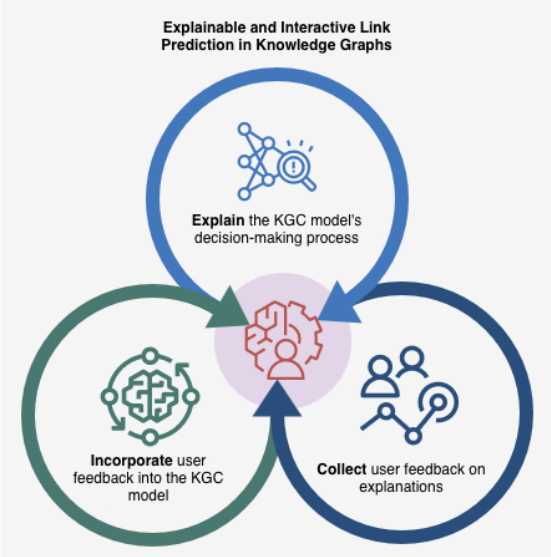


Figure 3: Overview of the thesis exploration of explainable and interactive link prediction in KGs. First, explanations provide insights into the KGC model's decision-making process. Next, these explanations serve as the interface for collecting user feedback, which captures domain expertise. The gathered feedback is then incorporated into the model to refine its behaviour. This process is iterative. It enables continuous alignment of the model behaviour with human expectations.

1.1.1 The Research Questions

In this doctoral thesis, we investigate how to align link prediction models with human intent using human feedback. The primary objective is to develop a method that integrates user knowledge iteratively and interactively into the decision-making process of link prediction models. This addresses a gap in current research, as, to the best of the author's knowledge, no existing method targets the alignment of link prediction models with human intent. We aim to bridge this gap by providing a methodology which enables model alignment with user feedback on model explanations.

The following research questions are addressed and answered in this thesis:

1. **How can we explain the decision-making process of a link prediction model, and how can this explanation assist a user in the downstream task of aligning the model with human intent?**

   In this research question, we explore first how to explain link prediction models in KGs. This includes the discussion of how explanation modalities, such as instance-based, saliency-based, counterfactual-based, analogy-based, and rule-based approaches, enable domain experts to understand and critique the model's decision-making process. It shall be discussed which explanation modality can provide actionable insights that facilitate user intervention to improve model alignment with human expectations and values. Based on the findings of the discussion, a novel post-hoc explainability method for embedding-based link prediction models shall be proposed.

2. **How effective are different types of feedback modalities in collecting user knowledge for the downstream task of aligning a link prediction model with human intent?**

   Here, the focus is on identifying, formalizing and discussing the diverse types of feedback modalities users can provide on model explanations, including rules and preferences, for the downstream task of aligning a link prediction model. This research question investigates how effective different feedback modalities impact the model's training or inference processes. Thus, with Research Question 2, we will answer how different feedback modalities can be used to align with human intent from a user's perspective.

3. **How can user knowledge be incorporated into a link prediction model for iterative alignment with human intent?**

   With this research question, we investigate the feasibility and effectiveness of a link prediction model that incorporates user feedback on the model's behaviour to refine the model iteratively. We aim to determine whether such a model can align with human feedback while maintaining or improving its predictive performance.

To answer these questions requires developing and evaluating a method for generating actionable explanations of the link prediction model's behavior. Additionally, we will explore how these explanations can support users in iteratively providing feedback and continuously aligning the model with human intent. The

research fosters justified trust in link prediction models by placing humans in the loop. Thereby, it enhances their adoption and effectiveness in real-world applications, such as in RCA.

The findings and methodologies presented in this thesis provide a technical and methodological perspective on integrating human feedback into link prediction models.

1.1.2 The Objectives

There are three challenges in using user feedback on the decision-making process of a link prediction model to align the model interactively.

First, how can explanations of a link prediction model's decision-making process support a user in the downstream task of aligning the model with human intent?
Explanations of an ML model's decision-making process come in many shapes. Some explanations are symbolic structures, like extracted rules (Rabold et al., 2018), which approximate the model's decision-making process. Others are saliency maps (Bach et al., 2015) indicating which input feature is and which is not relevant for classification. Yet other explanations provide counterfactual examples (Guidotti, 2022). Some explanations explain one prediction at a time only. They locally (Dwivedi et al., 2023) explain the model's behavior. Other explanations give insights into the general decision-making process of a model. They globally (Dwivedi et al., 2023) explain the model's behavior. With this thesis, we explore the benefits and downsides of different types of explanations when interactively and iteratively aligning the decision-making process of a link prediction model with human intent via user feedback. Focusing on the usability aspect of explanations is not novel. However, state-of-the-art research mainly focuses on understandability (e.g., relevancy, meaningfulness) (Phillips et al., 2021; Murdoch et al., 2019), meaning how effective an explanation is in conveying information to a user, given the environment in which a system operates (e.g., while driving a car, or diagnosing a patient). We aim with this doctoral thesis to broaden the scientific landscape by exploring what effective explanations are for the downstream task of human-in-the-loop KGC.

Second, how effective are different types of user feedback on KGC model explanations to align the model with human intent?
The Human-in-the-loop paradigm incorporate strategies such as demonstrating positive behaviors (Teso and Kersting, 2019), collecting ratings (Knox and Stone, 2011), and gathering preferences (Christiano et al., 2017) as feedback signals for model updates (Mosqueira-Rey et al., 2023; Najar and Chetouani, 2021). However, these approaches have been applied predominantly in contexts like image classification, natural language, or simulated agent environments. In contrast, the link prediction tasks involve structured knowledge representations that require domain-specific constraints or rule-based formulations (Schramm et al., 2023; Rabold et al., 2018). Thus, we will examine in this thesis the user perspective on different feedback modalities, ranging from preference annotations to formal logic rules, to align the model's decision-making process. The focus is to understand how these feedback modalities enable a user to interact with the link prediction model.

Finally, how can user feedback be incorporated into a link prediction model for iterative alignment with human intent?

The link prediction landscape includes a wide range of methods (Zhang et al., 2021), such as translational (Bordes et al., 2013), rule-based (Galárraga et al., 2015), and path-based models (Das et al., 2018). Yet, to the best of the author's knowledge, none have been explicitly tailored for continuous alignment with user feedback. Aligning a link prediction model requires adapting its objective function to consider human feedback and knowledge during training and inference. The thesis objective is to explore a way to align the decision-making process of a link prediction model with human feedback. We address this by proposing a novel link prediction model with a learning objective that iteratively integrates user feedback. This ensures that the KGC model progressively converges towards human-specified behaviour.

1.1.3 The Contributions

The thesis makes six contributions to the field of explainable and interactive link prediction in KGs. Each contribution builds upon the previous one, moving from explanation to interaction, measurement, and finally, iterative model alignment with human feedback.

First, the thesis introduces KGEPrisma, a post-hoc explanation method for embedding-based link prediction models. KGEPrisma produces local, instance-based, analogy-based, and rule-based explanations by identifying subgraph patterns that justify a KGC model's decision. It enables multi-facet perspectives on the KGC model's decision-making process and establishes a foundation for the subsequent exploration of human feedback on KGC model explanations.

Second, to investigate how feedback modalities on explanation can support users in giving feedback, RootFinder is presented. A system that reframes RCA in manufacturing as a link prediction task within a KG. RootFinder explores how explainable and interactive link prediction can serve as a decision-support tool. It connects symbolic process knowledge with data-driven Bayesian learning. Through this tool, the concept of interactive KGC model alignment is introduced and explored from a user perspective in a real-world industrial scenario.

Third, to move to a methodological exploration of model alignment, the thesis introduces the *Clever Hans Countries* dataset. It is a diagnostic benchmark designed to evaluate is link prediction models rely on spurious correlations instead of valid reasoning patterns. The dataset allows systematic detection of Clever Hans behavior (Lapuschkin et al., 2019) in link prediction and provides a basis for quantifying alignment of a KGC model's internal reasoning with human intent. Thus, it contributes to the interpretability research by offering a concrete benchmark to test KGC model alignment.

Fourth, to further build up on evaluation capabilities of KGC model alignment with human intent, the Mean Reciprocal Localisation Rank (MRLR) metric is proposed in this thesis. It serves as a quantitative measure for evaluating the validity of reasoning and explanation patterns produced by link prediction models. MRLR measures the overlap of model behavior and ground-truth regions of interest that adhere to a human understanding of valid reasoning. This provides a way to assess explanation quality.

Fifth, based on the previous contributions, the human-in-the-loop KGC model LiEr is proposed. It is a reinforcement learning model that iteratively aligns link prediction with human intent. LiEr formalizes the alignment of reasoning processes by integrating preference-based user feedback over instance-

based explanations directly into the model's reward function. This enables continuous refinement of the model's policy. LiEr represents a methodological step towards using preference-based feedback over model explanations to align KGC with human intent.

Finally, the real-world applicability of LiEr is demonstrated by applying it to RCA in electric vehicle manufacturing. This case study shows how the human-in-the-loop KGC model is used in practice to identify manufacturing faults. It confirms the transferability of LiEr to industrial applications.

Together, these contributions provide a methodological and empirical exploration of explainable and interactive link prediction. They address the open challenge of aligning KGC models´ internal reasoning with human intent via user feedback. KGEPrisma enables explanations; RootFinder explores user interaction over explanations; Clever Hans Countries and MRLR provide means to evaluate reasoning alignment; LiEr integrates the prior learned lessons into a human-in-the-loop KGC model; and the final case study validates this framework in a real-world domain. The following section introduces the related work that laid the foundations for these contributions.

1.2 Related Work

This section provides an overview of the work relevant to the thesis's objective. The use case that guides the methodological considerations of this thesis is the RCA in manufacturing.

1.2.1 Root Cause Analysis in Manufacturing

RCA in manufacturing is about finding the cause of a fault within the manufacturing process (see Figure 4). Detecting a fault in a complex manufacturing process does not necessarily uncover in which process steps the fault was induced. Finding the Cause-Effect Relationships (CERs) leading to the diagnosed fault is called RCA (Oliveira et al., 2022; Wehner et al., 2023).

RCA in practice is dominated by manual, labor-intense, and thus expensive process models defined in the ISO/IEC 31010 (ISO/IEC, 2019), including versions of the Five Why's (Serrat, 2017), the Failure Mode and Effect Analysis (FMEA) (Tay and Lim, 2008), or the Fault Tree Analysis (Ruijters and Stoelinga, 2015). The model's structure expert knowledge such that a process expert can use it as a manual for RCA. However, the decision-support tools do not take advantage of today's sensor networks and data (Wehner et al., 2023).

Thus, ML models were introduced to RCA. They enable automated and intelligent decision-support tools (Ma et al., 2021; Oliveira et al., 2022). The literature on data-driven RCA in manufacturing describes RCA with different ML models. Bayesian networks (Lokrantz et al., 2018), decision trees (Fan et al., 2016), and regression models (Ahn et al., 2019) are particularly popular due to them being per default interpretable by users. For example, Causal Bayesian Networks (CBNs) can be used to automate RCA (Pradhan et al., 2007) by learning CERs between sensor variables of the manufacturing process. The CBN constructs a root cause graph over the manufacturing process. This uncovers potential root causes (Kirchhof et al., 2020). However, learning CBNs for large manufacturing processes becomes prohibitively expensive as the search space for potential cause-effect relations explodes. This problem can be mitigated by

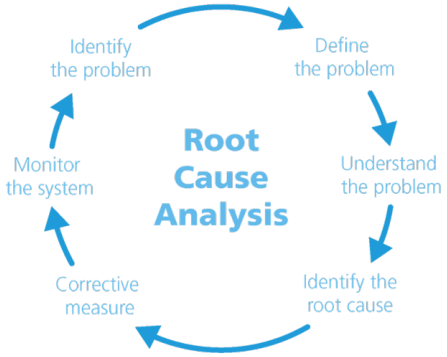


Figure 4: Typical workflow of RCA. The process starts with identifying and defining the problem. Next, a team of process experts identifies the underlying mechanisms leading to the problem. Understanding the mechanisms usually leads to locating the root cause. Once the root cause is identified, corrective measures are implemented, and the root cause is continuously monitored to prevent recurrence (Kobialka, 2022).

including process expert knowledge in the learning process of the CBNs (Kertel et al., 2023). In addition, expert knowledge can improve the learned root cause graph by identifying spurious CER (Wehner et al., 2023).

The remaining challenges for scaling CBNs to large-scale manufacturing processes are to model process expert knowledge in detail and to provide an accessible way for the process expert to interact with and improve the root cause graph (Wehner et al., 2023). In this thesis, we shall investigate those challenges by combining a KG of the manufacturing process with a CBN. The KG allows detailed modelling of large-scale manufacturing processes, incorporates the root cause graph and enables at the same time the process expert to improve the root cause graph. Overall, we interpret RCA in this thesis as a link prediction task in KGs.

1.2.2 Link Prediction in Knowledge Graphs

Link prediction, often referred to as KGC, is a ML task that learns to predict missing links in a KG. Most KGs are incomplete (Bordes et al., 2013; Hogan et al., 2022). A subset of correct but unknown triples $G_{\text{unknown}} \subseteq E \times R \times E$ is not yet modelled in the known KG. KGC models uncover these missing links by learning structural regularities within the KG, typically receiving partially specified triples (e.g., $(e^{\text{head}}, r, ?)$) and ranking the possible completions by plausibility (Rossi et al., 2021).

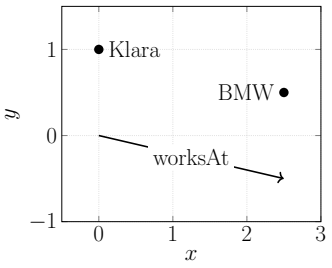
A substantial body of research (Zhang et al., 2021) explores embedding-based link prediction, with models such as RESCAL (Nickel et al., 2011), TransE (Bordes et al., 2013) (see Figure 5), and RotatE (Sun et al., 2019). These approaches

map entities and relations to vectors in $\mathbb{R}^n$, with $n \in \mathbb{N}$, then compute interaction scores via a learned interaction function:

$$\phi : E \times R \times E \to \mathbb{R}.$$

The interaction function ϕ assigns a scalar plausibility score to a triple. This enables the model to create an ordinal ranking. A higher score indicates a stronger belief by the KGC model that the triple is true. KGC models are optimized to place correct triples above corrupted ones. This is achieved by learning smooth embeddings (Hoseinnia et al., 2025) that reflect the KG's symbolic regularities. Consequently, entities exhibiting similar behaviors within the KG are represented by similar embeddings (Rossi et al., 2021; Ji et al., 2022).

Despite their predictive performance, sub-symbolic embeddings are black boxes. Thus, it is non-trivial to explain how each prediction is derived (Schramm et al., 2023; Ali et al., 2021; Rossi et al., 2021).



$$\text{Klara} = [0.0, 1.0]$$
$$\text{BMW} = [2.5, 0.5]$$
$$\vec{worksAt} = [2.5, -0.5]$$

Figure 5: Representation of the entities `Klara` and `BMW`, and the relation `worksAt` in $\mathbb{R}^2$ (with $x, y \in \mathbb{R}$) (Schramm et al., 2023).

Different approaches incorporate mechanisms that give insights into how the KGC model formed its predictions. For instance, ITransF (Xie et al., 2017) augments an embedding-based link prediction model with sparse attention vectors. The attention vectors indicate the concept dimensions most relevant to a predicted relation. Other solutions, including Anelli et al. (2019), inject semantic features derived from a KG into latent factorization machines. This enables interpretable link prediction in recommendation scenarios. CrossE (Zhang et al., 2019c) explains predictions by analogy. It identifies structurally similar paths that justify the predicted link. Furthermore, INK (Steenwinckel et al., 2022) encodes entities via the relation types connecting them to other entities in their neighbourhoods in a binary vector. This approach showed competitive performance, while being human-interpretable.

Purely symbolic KGC models, on the other hand, build upon the KG's symbolic relational structure to mine human-readable rules for link prediction. Thus, they are fully interpretable. Algorithms such as AIME+ (Galárraga et al., 2015), ScaLeKB (Chen et al., 2016), and SAFRAN (Ott et al., 2021) learn logical clauses from a KG, leveraging techniques inspired by Inductive Logic Programming (ILP) (e.g., FOIL (Quinlan, 1990; Muggleton, 1991; Pérez et al., 2006)). Early link prediction models, such as PRA (Lao et al., 2011), employ random walks to uncover path-based features, and subsequently use logistic regression to infer rules. Random-walk approaches can be computationally heavy and often underperform compared to embedding-based models. Subsequent research on such approaches improved the scalability and performance (e.g.,

SWARM (Barati et al., 2016), RLvLR (Omran et al., 2018), RDF2rules (Wang and Li, 2015), and AnyBURL (Meilicke et al., 2019, 2020)).

Neuro-symbolic link prediction KGC models leverage the expressivity of embeddings and the interpretability of logical rules. Examples include NeuralLP (Yang et al., 2017) and DRUM (Sadeghian et al., 2019), which embed relations as operators in a recurrent neural network to learn chain-like logical rules via backpropagation. Similarly, RuLES (Ho et al., 2018) expands candidate clauses guided by a KG embedding. This helps to navigate the large rule-search space efficiently. Other neuro-symbolic approaches use reinforcement learning to unify symbolic rule induction with sub-symbolic embeddings (Chen et al., 2022). TEM is another neuro-symbolic KGC model. It relies on attention vectors to establish interpretability (Wang et al., 2018). First, TEM uses a decision tree to learn decision rules. Next, they design an attention network-based embedding model that uses the decision rules as features. The combination of the decision tree and the attention network guarantees that the recommendation process is interpretable (Wang et al., 2018; Schramm et al., 2023).

Another interpretable subclass of neuro-symbolic link prediction models reformulates the KG as a MDP (Wan et al., 2020). This enables the discovery of reasoning paths. DeepPath (Xiong et al., 2017) first introduced an Reinforcement Learning (RL) agent that traverses from a head entity to a tail entity, based on links predicted by embedding-based KGC models (Bordes et al., 2013). However, because this traversal is decoupled from the actual link prediction process, the faithfulness of DeepPath's explanations can be limited. MINERVA (Das et al., 2018) addresses this by directly training an RL agent to maximize rewards by reaching the correct tail entity. This leads to interpretable paths that predict missing links in a KG. Subsequent refinements mitigate noisy supervision in incomplete KGs by using pretrained embedding-based link prediction models (e.g., ComplEx (Trouillon et al., 2016) or ConvE (Dettmers et al., 2018)) in the reward function (Lin et al., 2018), or by constraining paths under the consideration of predefined rules (Hou et al., 2021). Further work on inductive link prediction (Bhowmik and de Melo, 2020) and recommendation systems (Xian et al., 2019; Song et al., 2019; Zhang et al., 2020; Zhu et al., 2021) shows the applicability of path-based link prediction, as each navigated path is interpretable by human stakeholders (Adadi and Berrada, 2018; Bhowmik and de Melo, 2020). Users of a KGC model want the model to be correct for the correct reason. Thus, users have to understand the decision-making process of a link prediction model. If it is incorrect or correct for the wrong reasons, users want to intervene and correct it.

1.2.3 Explainability

Explainable Artificial Intelligence (XAI) aims to open the black box of complex ML models by providing a human-understandable mapping from the input of an ML model to its output and by that giving insights into the model‘s decision-making process. The goal is to justify the predictions of an ML model without sacrificing its performance. Another side of XAI is to design "white box" or inherently interpretable ML models so that the internal decision-making processes of a ML model are human-comprehensible by default (Adadi and Berrada, 2018; Schwalbe and Finzel, 2023).

In conventional ML domains such as vision, tabular, and text, post-hoc attribution methods identify which input features are most influential for a model's local decision boundary (see Figure 6). Techniques like LIME (Ribeiro et al., 2016), SHAP (Lundberg and Lee, 2017), Integrated Gradients (Sundararajan et al., 2017), Layer-wise Relevance Propagation (LRP) (Montavon et al., 2019), and DeepLIFT (Shrikumar et al., 2017) work well for providing explanations of convolutional or recurrent neural network architectures. However, these approaches do not trivially extend to sub-symbolic embeddings of triples in KGs. Mapping the latent dimensions of embedding-based KGC models back to symbolic concepts requires a specialised design, which shall be discussed later in the thesis.

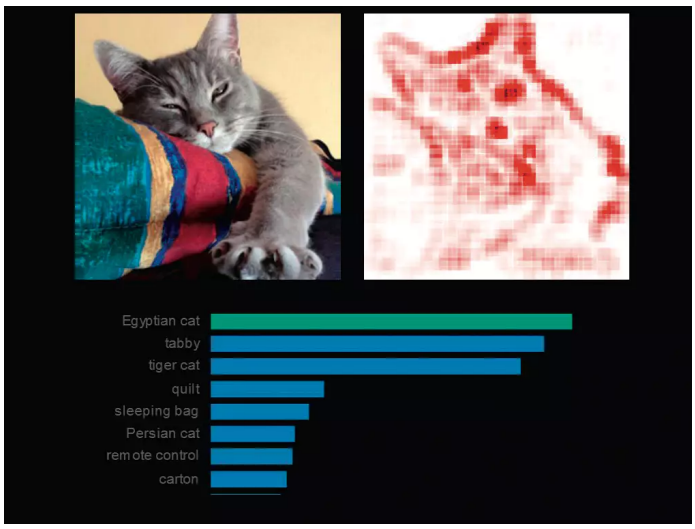


Figure 6: Illustration of LRP explaining the prediction of a convolutional neural network. The left image shows the original input (an Egyptian cat), while the right image visualises the relevance heatmap produced by LRP. LRP marks image regions that most contributed to the classifier's decision. In this case, LRP reveals that the cat's shape influenced the classifier's decision (Holzinger, 2019).

Several XAI methods have been adapted for link prediction that employs Graph Neural Networks (GNNs). However, their application is constrained to the specific architectures and mechanisms of GNNs. GraphLIME (Huang et al., 2023) leverages local perturbation and the HSIC-Lasso (Yamada et al., 2018) as a surrogate model to approximate decision surfaces within GNNs. PGMExplainer (Vu and Thai, 2020) generates a probabilistic graphical model that captures the Markov blanket of the target prediction. The method is computationally expensive due to its reliance on input perturbation. GraphLRP (Schnake et al., 2022) adapts LRP for GNNs by propagating relevance through their aggregation functions. These methods, designed to accommodate the specific operations of GNN-based link prediction models, are not universally applicable across the broader spectrum of link prediction models, which often employ frameworks other than GNN-based ones.

A different line of research leverages adversarial attacks to identify influential training facts to explain a KGC model's decision-making process. Pezeshkpour

et al. (2019) propose a gradient-based method for identifying influential training facts. Zhang et al. (2019a) explore data-poising strategies that manipulate the KG to weaken model performance to compute explanations. Similarly, Bhardwaj et al. (2021) adopt a model-agnostic instance attribution approach to select adversarial deletions. AnyBURLExplainer (Betz et al., 2022) applies rule learning and abductive reasoning to identify triples that degrade link prediction performance. This allows users to understand which triples in the training set shape the model's learned representations. These methods are mainly designed to uncover vulnerabilities in the model behaviour instead of creating post-hoc explanations. Nonetheless, they can show which facts influence the model behaviour.

Another category of XAI for link prediction relies on constructing surrogate models. Ruschel et al. (2024); Gusmão et al. (2018) and Polleti and Cozman (2019) train decoders that approximate the link prediction model's embedding space, either from a global perspective to capture the model's global behaviour or locally, around specific predictions. However, complex relationships are not always faithfully captured in these surrogates if their learning procedure is too far from the explanandum or if extensive domain heuristics are lacking. In contrast, KGEx (Baltatzis and Costabello, 2024) builds multiple surrogate models on different training subsets. Other methods, like OxKBC (Nandwani et al., 2020) and CPM (Stadelmaier and Padó, 2019), rely on heuristic templates or context paths, respectively, which can be computationally expensive at larger scales. KE-X (Zhao et al., 2023) uses information entropy on subgraphs to explain predicted links.

Other methods concentrate on interpreting entity representations. FeaBI (Ismaeil et al., 2023) learns interpretable vectors reflecting entity neighbourhoods. Chandrahas et al. (2020) utilises co-occurrence statistics to induce interpretability. Kelpie (Rossi et al., 2021) employs a perturbation-based strategy, which involves retraining the model after selective fact removal to test the significance of given triples. This is a robust but computationally demanding process. Once the model decision-making process has been made explicit to the user, the user would like to align it with their expectations of the model behaviour.

1.2.4 Artificial Intelligence Alignment

Artificial Intelligence (AI) alignment research is about anchoring human-defined goals, preferences, and human intent within AI systems (Wiener, 1960; Christian, 2020; Gabriel, 2020; Ilievski et al., 2025).

A substantial body of work aims to integrate human intent into ML methods (Mosqueira-Rey et al., 2023; Najar and Chetouani, 2021). For instance, CAIPI (Teso and Kersting, 2019) provides a model-agnostic iterative framework that allows users to provide feedback based on explanations of model decisions, thereby guiding the model toward desired behaviours. Building on CAIPI, Slany et al. (2022) refines the approach. They enable users to interact more effectively with the system by providing counterexamples generated from augmented data. Slany et al. (2024a) replace the hand-crafted data augmentation step with generative models. Furthermore, extensions of CAIPI to tabular data use logically constrained large language models to produce more targeted counterexamples (Slany et al., 2024b). Additionally, Heidrich et al. (2023) integrates fairness objectives to mitigate biases in the model's decision-making

process. Furthermore, Slany et al. (2024c) preserves and reuses human feedback within a probabilistic logic framework, thereby preventing catastrophic forgetting of expert advice during iterative learning cycles. Other work by Schramowski et al. (2020) proposes an active learning framework that incorporates "Right for the Right Reasons"-regularisation (Ross et al., 2017) into a neural network. This enables the model's learned explanations to align with human-validated rationales. Their framework leverages a Grad-CAM-based visualisation technique combined with t-distributed stochastic neighbour embedding (Maaten and Hinton, 2008) to highlight relevant regions in image data. In the text domain, Kiefer et al. (2022) utilises feedback on concept information to enhance classification.

In reinforcement learning contexts, approaches such as TAMER+RL (Knox and Stone, 2011), deep reinforcement learning from human preferences (Christiano et al., 2017), EXPAND (Guan et al., 2020), and PEBBLE (Lee et al., 2021) have demonstrated significant progress in aligning the behaviour of RL agents with human expectations, particularly in physical or simulated environments. Moreover, preference-based human-in-the-loop techniques have found successful applications outside of robotics and control, most notably in natural language processing, as seen in InstructGPT (Ouyang et al., 2022). This approach showed that interactive feedback mechanisms can steer large-scale language models towards preferred conversational styles and ethical considerations. There are no approaches known to the author that align link prediction models with human expectations. Filling this research gap is the aim of the thesis at hand.

The discussed work on RCA, link prediction in KGs, explainability, and AI alignment outlines the most relevant prior contributions in these fields. The thesis is based on these findings and aims to extend these fields with a novel approach to explainable and interactive link prediction in KGs. The following section provides a synopsis of the thesis. It outlines its structure and indicates which parts of this work have already been published by the author.

1.3 Synopsis

This thesis is structured along with the three central research questions. It moves from explanation, to feedback, and finally to iterative alignment of link prediction models. Each part builds on the previous and developes a comprehensive picture of integrating human feedback into KGC. The thesis begins with a systematic investigation of explanation modalities for link prediction, followed by an analysis of how user feedback can be elicited through those explanations. It concludes with a human-in-the-loop KGC model that aligns its behaviour with human understanding of valid reasoning.

Chapter 3 addresses Research Question 1: *How can we explain the decision-making process of a link prediction model, and how can this explanation assist a user in the downstream task of aligning the model with human intent?* The discussion begins in Section 3.1 by looking at five explanation modalities in KGC (instance-based, saliency-based, counterfactual, analogy-based, and rule-based). These are analyzed through the lens of their usability for user intervention, a perspective not previously addressed systematically in the literature. The section concludes in 3.1.6 by identifying limitations and strengths of each modality. This motivates the design of a novel explanation framework for embedding-based KGC models. KGEPrisma is introduced in Section 3.2, which integrates multi-

ple modalities to offer multi-perspective post-hoc explanations for embedding-based KGC models. In the subsequent sections 3.2.1 and 3.2.2 an in-depth description and evaluation of KGEPrisma across different benchmarks is provided. Section 3.3 concludes this part with a summary of the findings and answer to Research Question 1.

The Chapter 4 of the thesis addresses Research Question 2: *How effective are different types of feedback modalities in collecting user knowledge for the downstream task of aligning a link prediction model with human intent?* It begins with Section 4.1, which introduces RootFinder, an applied case study on RCA in manufacturing. RootFinder is a naive study to explore how user feedback on explanations can be collected and used for aligning model predictions in practice. The structure and content of the real-world manufacturing KG are detailed in Section 4.1.2, followed by a discussion of domain axioms and their role in formulating feedback in 4.1.3. A CBN is used in Section 4.1.4 to predict cause-effect relations, and the interactive user interface is presented in 4.1.5. The evaluation, quantitative and qualitative, is reported in Section 4.1.6. This motivates a broader discussion of feedback modalities in Section 4.2, including positive examples, negative examples, rules, ratings, and preferences. These are discussed in Sections 4.2.1 to 4.2.5. Finally, Section 4.3 provides a summary and answer to Research Question 2.

Chapter 5 of the thesis turns to Research Question 3: *How can user knowledge be incorporated into a link prediction model for iterative alignment with human intent?* Section 5.1 introduces LiEr, a reinforcement learning model that learns to predict missing links from preference-based feedback over model explanations. The formal foundations are laid out in Sections 5.1.1 to 5.1.4, where the KG is reformulated as an environment in a MDP, and the policy network is aligned with preference-based user feedback via a reward function that approximates human understanding of valid reasoning. Section 5.2 evaluates LiEr across a variety of benchmarks. The experimental setup is presented in 5.2.1. The MRLR metric for evaluating reasoning alignment is described in 5.2.2.

Next, the results on standard benchmark datasets are described in 5.2.3. Section 5.2.4 then introduces the *Clever Hans Countries* benchmark to evaluate LiEr in the presence of spurious correlations. Sections 5.2.5 and 5.2.6 provide further insights into collecting preference-based feedback and discuss the limitations and robustness of LiEr.

In Section 5.3 LiEr is applied to RCA in electric vehicle manufacturing to show how it can be applied to a real-world link prediction task. Section 5.3.1 describes the real-world KG used in the application, followed by a synthetic root cause generation strategy in 5.3.2. The encoding of physical measurements is explained in 5.3.3, and the evaluation of LiEr on the manufacturing task is provided in Section 5.3.4. Finally, Sections 5.3.5 and 5.3.6 discuss limitations and lessons learned from applying LiEr in this manufacturing context. The chapter concludes in Section 5.4 with a summary and answer to Research Question 3.

The final chapter 6 concludes the thesis by summarizing the findings. It revisits the three research questions, and outlines the implications of the results. The thesis is closed by identifying open problems and future research directions that arise from this work.

Table 1: Summary of contained publications with full reference, a brief summary, and where they appear in this thesis.

Reference	Summary	Section
Schramm, S., Wehner, C., and Schmid, U. (2023). Comprehensible Artificial Intelligence on Knowledge Graphs: A survey. *Journal of Web Semantics*, 79:100806. Publisher: Elsevier BV	This survey introduces Comprehensible AI as an overarching concept of XAI and Interpretable ML in the context of KGs. It presents a novel taxonomy, offers a structured overview of the field, and identifies open research questions for future work.	1, 2, 3, 4, 5
Wehner, C., Iliopoulou, C., Schmid, U., and Besold, T. R. (2024). From Latent to Lucid: Transforming Knowledge Graph Embeddings into Interpretable Structures with KGEPrisma. Version Number: 2 *(unpublished)*	The paper proposes a post-hoc method that explains Knowledge Graph Embedding (KGE) models by decoding their latent representations. It uses similarity in embedding space to uncover subgraph regularities and translate them into human-readable rules, instances, and analogies.	3, 3.1, 3.2
Besold, T. R., Akujuobi, U., Badreddine, S., Choi, J., Elshazly, H., Gifford, F., Iliopoulou, C., Maruyama, K., Nagano, K., Martin, P. S., and others (2025a). Literature-based Hypothesis Generation: Predicting the evolution of scientific literature to support scientists. In *AI4X 2025 International Conference*	The paper explores literature-based hypothesis generation, framed as KGC. It enriches the generated hypothesis with explanations via KGEPrisma, which provides context for the scientists by giving insights into why the generated hypothesis was predicted.	3.1, 3.2
Wehner, C., Kertel, M., and Wewerka, J. (2023). Interactive and Intelligent Root Cause Analysis in Manufacturing with Causal Bayesian Networks and Knowledge Graphs. In *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*, pages 1–7	The paper presents RootFinder, an interactive RCA tool for electric vehicle manufacturing that combines expert knowledge with data-driven learning. It reasons over a manufacturing KG while learning a CBN, with an interface for experts to give feedback. It serves as a kick-off for a naive exploration of feedback modalities from a user perspective.	4, 4.1, 4.2
Eirich, J., Jäckle, D., Sedlmair, M., Wehner, C., Schmid, U., Bernard, J., and Schreck, T. (2023). ManuKnowVis: How to Support Different User Groups in Contextualizing and Leveraging Knowledge Repositories. *IEEE Transactions on Visualization and Computer Graphics*, 29(8):3441–3457. Publisher: Institute of Electrical and Electronics Engineers (IEEE)	ManuKnowVis links multiple manufacturing knowledge sources for battery module production to support visual RCA. It works with feedback modalities such as annotations, edits of entities and relations, and hypothesis notes. A case study with process experts explores different feedback modalities for this applied context.	4.1, 4.2
Bahr, L., Wehner, C., Wewerka, J., Bittencourt, J., Schmid, U., and Daub, R. (2025). Knowledge graph enhanced retrieval-augmented generation for failure mode and effects analysis. *Journal of Industrial Information Integration*, 45:100807. Publisher: Elsevier BV	A KG enhanced retrieval augmented generation setup for failure mode and effect analysis. A user study validates the approach and offers insights into user feedback in the manufacturing domain.	4.1, 4.2
Poßner, L., Bahr, L., Röhl, L., Wehner, C., and Gröger, S. (2025). Anwendung von Causal-Discovery-Algorithmen zur Root-Cause-Analyse in der Fahrzeugmontage. In *Rethinking Quality - Wandel des Qualitätsmanagements durch Digitalisierung und Künstliche Intelligenz*, pages 39–59. Springer Fachmedien Wiesbaden, Wiesbaden	Application of causal discovery to assembly data to support RCA in quality management, with a comparison of algorithms and metrics.	1.2, 4.1, 5.3
Wehner, C., Bahr, L., Voigt, E., Wagner, J., Wagner, J., and Schmid, U. (2025). Aligning Path based Link Prediction with Human Understanding of Valid Reasoning *(unpublished)*	The paper introduces LiEr, a path-based link prediction method that learns valid reasoning from preference-based human feedback. LiEr is the first human-in-the-loop link prediction method that aligns its reasoning with human feedback.	5, 5.1, 5.2, 5.3

Appendices will provide information about the author's scientific contributions. Appendix A will contain contribution statements for the publications incorporated into this thesis. Appendix B will give an overview of talks, workshops, summer schools, patent and proposals that the author gave, participated in, organized, or wrote.

The chapters of this thesis are supported by several publications, which form the methodological and empirical backbone of the work. Their contributions are briefly summarized below.

All sections build on Schramm et al. (2023), which explores the conceptual foundation for the thesis. It explores what XAI and interpretable ML in the context of KGs means, and lays out open research questions. These form the literature basis in Section 1.2 of the thesis and inform the methodological framing of all research questions.

Section 3.2 and large parts of Section 3.1 are based on the authors' paper Wehner et al. (2024), which presents KGEPrisma, a post-hoc explanation framework for KGE models. The development of KGEPrisma was closely tied to the conceptual analysis in Section 3.1, as the challenges observed during its design informed the discussion of explanation modalities and vice versa.

Further, Besold et al. (2025a) applies KGEPrisma to the task of literature-based hypothesis generation. The application described in that paper influenced the exploration of explanation modalities in Section 3.1, especially with respect to how explanations can support the alignment of generated predictions with human understanding.

Chapter 4 is grounded in the work presented in Wehner et al. (2023). The paper introduces RootFinder, a decision support system for RCA in electric vehicle manufacturing. Sections 4.1 and 4.2 are based on this publication, which initiated the exploration of feedback modalities from a user-centered perspective.

The applied works Eirich et al. (2023), Bahr et al. (2025), and Poßner et al. (2025) contributed indirectly to the thesis. Although not part of the core methodology, they provided real-world contexts for developing RootFinder and LiEr. These studies offered insights into data-driven RCA, user interaction with KGs and the challenges of feedback collection in manufacturing. Their influence is visible in the Sections 4.1, 4.2, and 5.3.

Chapter 5 is based on the paper Wehner et al. (2025), which introduces LiEr, a reinforcement learning framework for aligning path-based link prediction models with human feedback. Section 5.1 is directly derived from this work. The formalism, experimental setup, and evaluation, like using the Clever Hans Countries benchmark and MRLR metric, were first developed for the paper and adapted for this chapter.

Together, these publications reflect the progression of the thesis from literature research, to method development, to applied evaluation. They document the conceptual, technical, and empirical stages of answering the research questions of this thesis. The following section outlines the methodological foundation of the thesis.

2 Foundations

The research in this thesis is founded on a wide range of work. Previous work spans from KGs, to how to find missing links in them and how to explain those missing links, to the question of how to evaluate such explanations, and finally, how to use human feedback to align the link prediction model's behaviour.

2.1 Knowledge Graphs

KGs model real-world facts as triples $(e^{\text{head}}, r, e^{\text{tail}})$, where each triple belongs to the subset $G \subseteq E \times R \times E$ of the entity set E and relation set R (see Figure 7) (Nickel et al., 2011; Hogan et al., 2022). For consideration in logic frameworks, such triples can be interpreted as grounded binary predicates $r(e^{\text{head}}, e^{\text{tail}})$, with the relation $glsrelation$ functioning as the binary predicate and the entities $e^{\text{head}}, e^{\text{tail}}$ serving as grounding constants. Each entity and relation is assigned a symbolic label (i.e., a name). A semantic schema $\zeta : E \to C$ (Hogan et al., 2022) further describes entities by categorizing them into classes C from the KG's domain. In addition, ontologies define the broader conceptual framework of the KG through axioms and logical constraints (Hogan et al., 2022; Baader, 2009). KGs can be formalized in standards such as RDF or OWL (Abiteboul et al., 2011). Those standards enable the usage of query languages (e.g., SPARQL (Prud'hommeaux et al., 2013), Cypher (Francis et al., 2018)) for deductive reasoning within the KG. However, it is non-trivial to formulate appropriate reasoning patterns in large, heterogeneous KGs where users possess incomplete domain knowledge.

KGs are interpretable by humans due to their symbolic and connected structure. Thus, KGs can serve as an interface between human experts and machine learning models, particularly when large-scale domain knowledge needs to be stored, retrieved, and updated. Early conceptual precursors to KGs are semantic networks, initially developed between 1972 and 1980 for a university project (Schneider, 1973). Later, from 1985 to 2007, applications of KG-like data structures were used in diverse fields, for example, the lexical database WordNet (Miller, 1995). During this period, research on ILP and other rule-learning algorithms (e.g., FOIL) (Quinlan, 1990; Muggleton, 1991; Pérez et al., 2006) used symbolic structures to derive human-comprehensible rules. Meanwhile, the creation of general-purpose knowledge bases, such as DBpedia (Auer et al., 2007) and Freebase (Bollacker et al., 2008), increased the usability and broadened the practical impact of large, symbolic, and interconnected knowledge representation formats. In 2012, the Google Knowledge Graph was introduced, and with it, the concept of what we now call a KG was first introduced (Singhal, 2012). Subsequently, academia and industry adopted KGs for applications ranging from information retrieval and personalization to knowledge-driven analytics (Hogan et al., 2022; Schramm et al., 2023).

Excurse: Knowledge Graph as a Quiver. KGs can be formalized in many different ways. This thesis looks at them as a quiver (Assem et al., 2006). A quiver provides a representation for edge traversal and symbol retrieval. Formally, a KG is a typed quiver $\mathcal{G} = (E, R, \Sigma_E, \Sigma_R, h, t, \ell_E, \ell_R)$ (Assem et al., 2006), where E is the set of vertices (i.e., entities/nodes), R is the set of edges (i.e., relations/links), and Σ_E and Σ_R are alphabets of vertex and edge symbols,

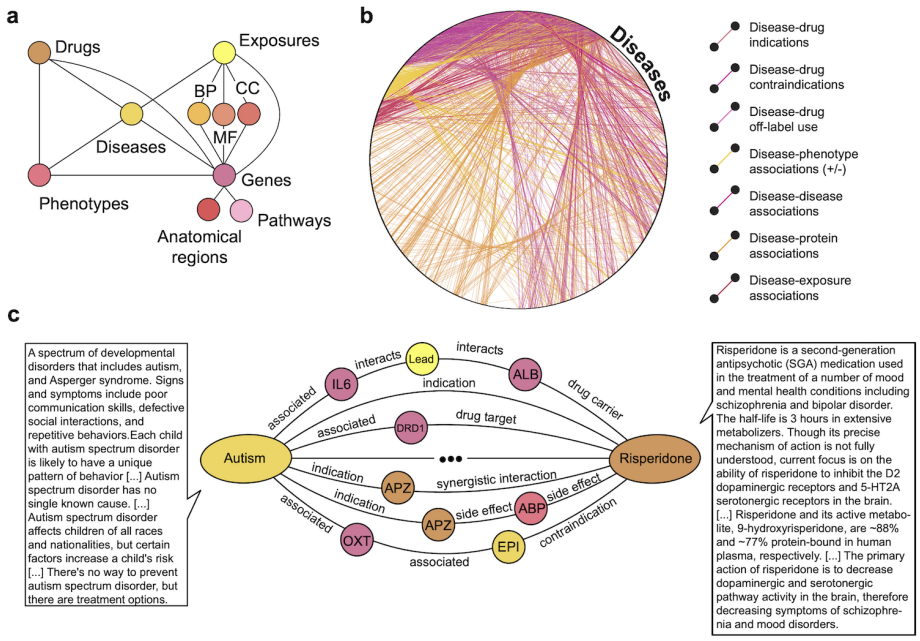


Figure 7: Example of the biomedical PrimeKG KG (Chandak et al., 2023). PrimeKG models biomedical data as a KG. This includes diseases, genes, drugs, phenotypes, and molecular pathways. (a) shows the schema of classes and their relationships. (b) visualises how densely diseases are connected in the KG. (c) illustrates an example subgraph which connects the disease Autism to the drug Risperidone.

respectively. The head mapping $h : R \to E$ associates each edge r with its head vertex e_h, while the tail mapping $t : R \to E$ associates r with its tail vertex e_t. The vertex language mapping $\ell_E : E \to \Sigma_E$ and the edge language mapping $\ell_R : R \to \Sigma_R$ assign symbolic labels to vertices and edges (Gallian, 2000; Harary et al., 1965). This formalization supports traversal (e.g., path queries) and systematic label retrieval (e.g., mapping nodes to symbolic names or measurements). Thus, this representation supports machine-driven and human-driven analyses.

The construction of KGs requires substantial domain expertise and a curation of concepts, relationships, and axioms. In practice, this leads to incomplete KGs (Hogan et al., 2022). Incomplete KGs are less useful in downstream tasks such as predictive analytics, recommendation systems, or RCA.

2.2 Link Prediction in Knowledge Graphs

As discussed before, link prediction, also referred to as KGC, is a ML task that is about predicting missing links in a KG. Most KGs are incomplete (Bordes et al., 2013; Hogan et al., 2022). A subset of correct but unknown triples $G_{\text{unknown}} \subseteq E \times R \times E$ is not modelled in the known KG G. KGC models find these missing facts by exploiting structural regularities within the KG. They receive partially specified triples (e.g., $(e^{\text{head}}, r, ?)$) as inputs and rank the possible completions (entities) by plausibility (Rossi et al., 2021).

There are three subtypes of link prediction tasks (see Table 2). They differ on which element of a triple is missing and has to be recovered by the KGC model (Hogan et al., 2022). In tail prediction, the model is given a head entity and a relation $(e^{\text{head}}, r, ?)$. The goal is to predict the most plausible tail entity e^{tail} such that the full triple $(e^{\text{head}}, r, e^{\text{tail}})$ is true (Hogan et al., 2022).

The inverse task is head prediction, where the model receives a relation and a tail entity $(?, r, e^{\text{tail}})$ and predicts the most plausible head entity e^{head} (Hogan et al., 2022).

A third variant of KGC is relation prediction, in which the head and tail entity are given $(e^{\text{head}}, ?, e^{\text{tail}})$, and the model predicts the relation r that most likely connects them (Hogan et al., 2022).

Table 2: Overview of the three link prediction tasks in KGs. Each task is about predicting one missing element in a triple based on the other two given elements.

Task Type	**Input**	**Predicted Element**
Tail prediction	$(e^{\text{head}}, r, ?)$	e^{tail}
Head prediction	$(?, r, e^{\text{tail}})$	e^{head}
Relation prediction	$(e^{\text{head}}, ?, e^{\text{tail}})$	r

In addition to the prediction target, link prediction models differ in the assumptions they make about entity availability during training and inference. In the transductive setting, all entities that appear at test time are already seen during training (Hogan et al., 2022). This is the standard setting in most benchmark datasets, where the entity set is fixed. In contrast, the inductive setting assumes that previously unseen entities may appear at inference time (Hogan et al., 2022). Here, the model must generalize to these previously unseen entities. In this thesis, we focus on tail prediction in the transductive setting. In manufacturing KGs for RCA, the set of entities (such as stations, process steps, or sensor variables) is known in advance, and the goal is to predict which known entity is the root cause of the fault of another known entity.

Root Cause Analysis as Link Prediction in Knowledge Graphs. A way to bring together knowledge-driven RCA and data-centric approaches is to see RCA as a link prediction problem within a KG. Formally, one defines an `isRootCauseOf` relation $r \in R$ between two variable nodes $var_1, var_2 \in E$ in a KG of a manufacturing process, with the goal of inferring previously unknown links of the form

$$(var_1, \texttt{isRootCauseOf}, var_1) \notin G,$$

where $G \subseteq E \times R \times E$ represents the observed triples in the KG. In this formulation, lines, stations, process steps, and measurement variables are modelled as entities (i.e., $var \in E$). For instance, a line $\in E$ is connected to multiple stations $\in E$ via `hasStation` relations, or to subsequent lines via `hasSuccessorLine`. Thereby, the structure of the manufacturing process is captured in a KG. Each station, in turn, executes a sequence of process steps linked by `hasSuccessorProcessStep`, and each process step measures variables modelled via the `measures` relation type. Variables typically carry literals, such

as numeric time-series measurements for temperature or pressure, that can be attached as attribute values in the KG.

Link prediction models are now able to assign a plausibility score to each $(var_1, \texttt{isRootCauseOf}, var_2)$ triple. Higher scores indicate a stronger causal influence of var_1 on var_2. Thus, modelling the manufacturing process as a KG enables a KGC model to uncover novel `isRootCauseOf` edges in the manufacturing process by exploiting topological patterns in G (e.g., the arrangement of lines, stations, and processes), rules (e.g., domain knowledge) and associated literal information (e.g., time-series sensor readings). We will later in this thesis discuss approaches that utilize this representation to predict root causes.

It is critical for the RCA to understand why a root cause was predicted before actions to eradicate it are taken in the physical environment, as the actions may result in significant costs. Thus, process experts must trust the predictions before acting on them. Explainability methods shed light on the decision-making behaviour of ML models, thereby creating trust in their predictions.

2.3 Explainability

XAI is about uncovering the decision-making process of complex ML models. It creates a human-understandable mapping of a ML model's input to its output (Adadi and Berrada, 2018). The goal is to justify model predictions without impairing their predictive performance. Another side of XAI is to design "white box" or inherently interpretable models so that the internal decision-making processes of a ML model are human-understandable by default (Adadi and Berrada, 2018; Schwalbe and Finzel, 2023). XAI serves four purposes:

1. **Justification:** It provides rationales that explain why the model outputs a given prediction (e.g., to judge whether the model is aligned with ethical principles).
2. **Flaw Identification:** It supports the detection of flaws, biases or spurious correlations (e.g., Clever Hans biases (Lapuschkin et al., 2019)).
3. **Model Debugging:** It enables developers to fix the discovered issues in a ML model.
4. **Rule Discovery:** It reveals novel or unforeseen domain patterns the model has learned.

A common distinction is between post-hoc explanation methods, which map inputs in a human-understandable manner to the outputs of a black-box model, and Interpretable Machine Learning (IML) models, which have a by default human-understandable decision-making process (Lahav et al., 2018; Molnar, 2019; Rudin, 2019). IML models produce inherently faithful explanations because their decision logic is explicitly encoded. On the other hand, post-hoc explanation methods approximate a black-box ML model from the outside. This also means that most XAI methods can be applied to a variety of ML models without adapting them. This makes them convenient for a wide range of ML models. Furthermore, XAI methods can be categorized by whether they generate local or global explanations. Local explanations focus on how the model arrives at a decision for a single instance. By contrast, global explanations characterize the general decision-making process that a model learned for

all instances. This provides insights into its behaviour across the entire data distribution (Schwalbe and Finzel, 2023).

One can think of many different ways to explain the same prediction when asked, "Why did the model come to this conclusion?". Some answers may be as good as others, while many are not. Judging the quality of a ML model's explanation is a core problem of XAI.

2.3.1 Metrics for Explanation Quality

The evaluation of an explanation requires the systematic measurement along multiple dimensions. The scientific literature describes several properties and associated metrics of what makes explanations good (Adebayo et al., 2018; Sundararajan et al., 2017; Montavon et al., 2019; Hedström et al., 2023). This includes the following properties:

1. **Faithfulness:** Quantifies how closely explanations mirror the model's decision-making process. Higher faithfulness implies that input features important for the ML model's decision making are similarly emphasized by the explanation. One way to test faithfulness is Region Perturbation (Samek et al., 2017). High-importance features are perturbed to observe their impact on model outputs. A deterioration in model performance implies faithful attributions.

2. **Robustness:** Reflects the stability of explanations under small input perturbations. The perturbations are kept small enough to ensure that the model predictions remain unchanged. Robustness can be tested with the Local Leibniz Estimate (Alvarez-Melis and Jaakkola, 2018). It compares explanations for neighboring points in the input space. This tests whether similar model inputs lead to similar explanations.

3. **Localization:** Measures whether an explanation pinpoints the region of interest that is given as the ground truth for the prediction. The Pointing Game (Zhang et al., 2018) tests for localization of explanations. It checks whether the explanations coincide with the intended area of interest.

4. **Explanation Complexity:** Captures how concise or sparse an explanation is. Long and overloaded explanations reduce user understanding. The Sparseness (Chalasani et al., 2020) method uses the Gini Index to judge whether an explanation consists of a small set of features. This indicates a focused explanation.

5. **Computational Complexity:** Assesses the computational overhead for generating an explanation. Low computational complexity is crucial for large-scale or real-time tasks (Montavon et al., 2019; Hedström et al., 2023).

6. **Randomization:** Tests the explanation's sensitivity to randomizing parts of the model. Randomization can be assessed with the Model Parameter Randomization Test (Adebayo et al., 2018). It randomly re-initializes network layers to see if explanations differ substantively from those of the original model. This tests whether the explanation depends on actual learned parameters.

7. **Axiomatic Compliance:** Ensures the explanation follows certain theoretical constraints. One of such constraints is Completeness (Sundararajan et al., 2017). It requires that the sum of feature attributions equals the difference in model output between a null matrix baseline input and the actual input. However, some theoretical constraints are highly method-dependent and not applicable to any XAI method.

The properties provide a multi-perspective evaluation of how good an explanation is. Once a user is presented with an explanation that satisfies the latter properties, they may discover that the model is incorrect or correct for the wrong reason. For example, it may have learned a Clever Hans bias (Lapuschkin et al., 2019). In this case, the user will want to correct the ML model. This is an alignment problem.

2.4 Artificial Intelligence Alignment

AI alignment is about anchoring human-defined goals, value judgments and human intent within AI systems (Wiener, 1960; Christian, 2020; Gabriel, 2020; Ilievski et al., 2025). In the words of Wiener (1960):

> "If we use, to achieve our purposes, a mechanical agency with whose operation we cannot interfere effectively ... we had better be quite sure that the purpose put into the machine is the purpose which we really desire."

Ortega and Maini (2018) differentiates among three types of goals embedded in an AI system. **Intended goals** represent the users' ideal vision of the AI's behaviour. These are often difficult for the user to specify in precise terms. **Specified goals** refer to the objective functions or datasets provided to the AI. These are the concrete metrics and resources that the system is explicitly trained to optimize. **Emergent goals** capture the objectives that the AI system pursues once the learning process converges. These may deviate from the specified goals due to model biases, spurious correlations, or other factors not considered in training.

Human-in-the-loop learning aims to integrate human intent and judgment directly into ML methods to make the intended, specified, and emergent goals as much overlapping as possible (Mosqueira-Rey et al., 2023; Najar and Chetouani, 2021). It utilizes human feedback to update the ML model iteratively. This enables continuous human oversight. The intervention results in reliable and user-centric ML models (Mosqueira-Rey et al., 2023).

However, human-in-loop ML is susceptible to alignment risks (Mosqueira-Rey et al., 2023):

- **Objective Misalignment:** The specified training objectives do not fully capture the user's underlying goals. This leads the AI system to optimise for metrics that deviate from the user's true requirements.
- **Biases in Human Judgments:** Humans are biased or have incomplete knowledge. Thus, the user may align the model's decision boundaries in unintended ways.
- **Quality of Human Input:** Integration of flawed or noisy human feedback may degrade overall model performance.

- **Feedback Loops:** Early misalignment can steer the model optimisation towards local minima, which are then reinforced through multiple interaction cycles. This causes the model to drift further from the user's intended outcomes.

One strategy to mitigate these risks is to align the ML model's output and internal decision-making process with human judgment. Domain experts identify and correct unintended, misspecified, and malicious emergent goals by examining the explanations of the model's decision-making process and providing feedback on them. Feedback on the decision-making process enables more targeted interventions.

Despite the existence of AI alignment in fields such as image recognition, reinforcement learning, and text generation, link prediction has remained untouched by these developments. In particular, there is no existing framework that uses human feedback on explanations of a link prediction model to align its behaviour. The contribution made in this dissertation is to utilise iterative human feedback to align the KGC model's decision-making process. This leads to more trustworthy and goal-consistent link prediction models. The following section takes the first step towards this contribution by further exploring explanation modalities of KGC models for the downstream task of model alignment.

3 Explaining Link Prediction in Knowledge Graphs

Research Question 1: How can we explain the decision-making process of a link prediction model, and how can this explanation assist a user in the downstream task of aligning the model with human intent?

To address this research question, we first explore how to explain link prediction models for KGs. This includes the discussion of how explanation modalities, such as instance-based, saliency-based, counterfactual-based, analogy-based, and rule-based approaches, enable domain experts to understand and critique the model's decision-making process. It shall be discussed which explanation modality provides actionable insights that enable user intervention to improve model alignment with human expectations and values. Based on the findings of the discussion, a novel post-hoc explainability method for embedding-based link prediction models shall be proposed.

The aim of this thesis is to investigate how explanations for KGC models can be used to align their behaviour with human intent. The first step towards this goal is to explain KGC models [1]. Explainability in KGC differs fundamentally from established XAI domains such as computer vision or natural language processing, since the input is a graph rather than an image or a sequence of words. As a result, explanation techniques must be tailored to the relational and symbolic structure of KGs. This enables unique explanation themes, such as subgraph extraction (Rossi et al., 2022b), rule induction (Betz et al., 2022), and path-based explanations (Bhowmik and de Melo, 2020), in place of, for example, the pixel- or token-based saliency maps common in other fields (Ribeiro et al., 2016; Lapuschkin et al., 2019).

This section introduces the first step toward using explainability to align link prediction models with human intent, which is explaining link prediction models. Existing explanation modalities for KGC are reviewed and discussed for the downstream task of model alignment. This exploration motivates a novel explanation method, which is presented later in detail. Once it is established how to explain KGC models, the subsequent sections illustrate how these explanations can be used to gather user feedback for iterative model alignment.

3.1 Explainability for Knowledge Graph Completion

The relational and symbolic structure of KGs imposes constraints on the modalities of explanations that can be provided for link prediction models. That means, explanations for KGC show why a model predicts the existence of a triple (e^{head}, r,e^{tail}) by referencing subgraph patterns or logical rules. Explanations in KGC fall into five modalities:

- Instance-based explanations, which identify concrete triples that influenced a given prediction.
- Saliency-based explanations, which attribute edges with weights whose perturbation would affect the model's confidence in a link.

[1] This section is adapted from the author's prior publications: Schramm et al. (2023), Wehner et al. (2024), and (Besold et al., 2025a).

- Counterfactual explanations, which describe modifications to the graph (e.g., removing or adding edges) that would flip the model's decision.
- Analogy-based explanations, which draw parallels between the target prediction and structurally similar, previously observed patterns elsewhere in the graph.
- Rule-based explanations, which extract human-readable logical rules that approximate the model's behaviour.

Each of these modalities has strengths and limitations when used to support model alignment. Additionally, it is important to note that all of these modalities are to a certain degree related. In the following, we shall draw out where they relate to each other, but more importantly, in which dimensions they deviate from each other and which strengths and limitations come with that. The following introduces each explanation modality and discusses its suitability for the downstream task of model alignment. Based on the findings of the discussion, a novel post-hoc explainability method for link prediction models is proposed.

3.1.1 Instance-Based Explanations

Instance-based explanations justify the plausibility of a predicted triple (e^{head}, r,e^{tail}) by identifying triples from the training KG that support the inference (Schramm et al., 2023).

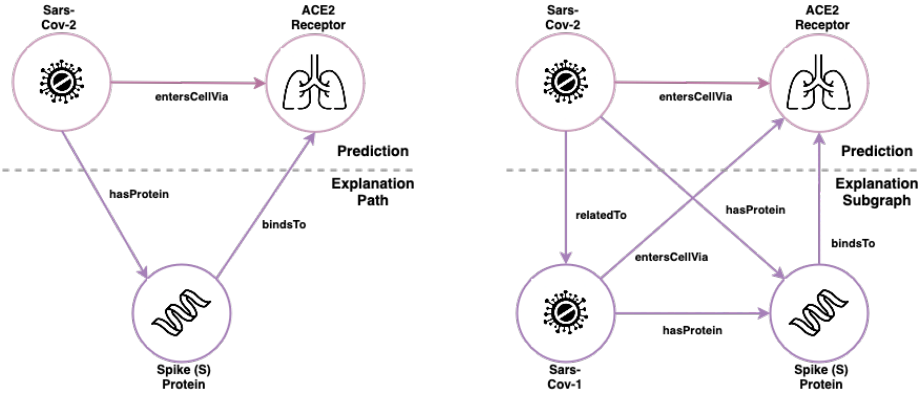


Figure 8: Left (Path Theme): In this COVID example, SARS-CoV-2 is predicted to connect to a lung cell through the Spike (S) Protein. A path in the KG justifies the predicted link. **Right (Subgraph Theme):** Here, the same predicted link is justified by additionally contextualising SARS-CoV-2 as related to SARS-CoV-1, which creates a subgraph showing that SARS-CoV-1 uses the same mechanism (Spike (S) Protein) as SARS-CoV-2 to bind to a lung cell.

There are two common themes in the instance-based explanation modality. One searches the KG for a triple-path that connects e^{head} to e^{tail}. Thereby the explanation reconstructs the predicted relation via multi-hop reasoning (Xiong et al., 2017). In the other theme, the explanation produces a subgraph of several relevant triple. Their joint configuration makes the predicted triple plausible (Zhao et al., 2023).

The **path** theme (see Figure 8) locates a sequence of triples

$$(e^{\text{head}}, r_1, e_1), (e_1, r_2, e_2), \ldots, (e_{k-1}, r_k, e^{\text{tail}})$$

that establishes a path from e^{head} to e^{tail}. The predicted triple is justified if the path aligns with domain knowledge or recognized reasoning patterns. The paths are usually generated with the help of reinforcement learning or other heuristic searches to systematically sample plausible intermediate hops (Xiong et al., 2017; Das et al., 2018; Shen et al., 2018).

The **subgraph** theme (see Figure 8) identifies a subgraph whose structure offers insight into why r likely holds between e^{head} and e^{tail} (Zhao et al., 2023). The explanation uncovers the entities and relationships community that makes the predicted triple likely. Although such subgraphs can be more complex than single paths, they tend to provide a more detailed depiction of the local context.

Table 3: Pros and cons of instance-based explanations for the downstream task of model alignment.

Pros	**Cons**
References concrete evidence from the KG.	Explanations can become extensive in complex graphs.
Intuitive for domain experts to inspect triples.	Incomplete graphs can produce misleading or shallow evidence.
Enables correction by labelling paths or subgraphs as relevant or irrelevant.	Identifying the most relevant path or subgraph is computationally expensive.

Discussion. Instance-based explanations are well-suited for the downstream task of model alignment (see Tabel 3) because they reference tangible facts. With that, users can directly examine the supporting triples to determine whether the explanation aligns with their domain knowledge. Thus, instance-based explanations serve as a bridge between black-box models and user knowledge. They enable users to inspect "where" and "how" the model is drawing its conclusions by grounding predicted links in verifiable segments of the KG. Thus, instance-based explanations are effective in downstream alignment tasks, since domain experts find it intuitive to reason about concrete examples or chains of facts (Stenning and Van Lambalgen, 2008). When valid paths are found, users can label them as positive examples that reinforce accurate reasoning. Conversely, irrelevant paths or subgraphs can be flagged to discourage the model from replicating them. This process enables direct correction of the model's decision-making. Especially when a path-based explanation aligns with well-understood causal or relational logic and has real-world applicability.

On the other hand, instance-based approaches may produce extensive explanations in complex KG domains with long paths or large subgraphs. Additionally, the strength of an instance-based explanation relies on the completeness of the underlying KG. If contextual links are missing, correct predictions are backed by shallow justifications, which in turn undermine user trust. Additionally, reliance on explicit structural evidence leads to an extensive search space

for explanations. Particularly if the KG has a high degree. In such cases, substantial computational resources are required to identify the most relevant path or subgraph.

Despite these challenges, instance-based explanations are overall a compelling foundation for feedback generation in human-in-the-loop approaches. They allow users to validate and refine model inferences through observable graph structures.

3.1.2 Saliency-Based Explanations

Saliency-based XAI methods attribute importance to specific input features. This provides insight into which parts of the input are important to a model in making its decisions. In the context of KGs, saliency-based explanations are most common for link prediction models that embed entities and relations into high-dimensional vectors. For that reason, common saliency-based approaches, like LIME (Ribeiro et al., 2016) or Integrated Gradients (Sundararajan et al., 2017), have to be reinterpreted to accommodate the discrete and relational properties of graph-structured data.

Saliency-based explanations are common in the tabular data domain, as well as in the image classification domain, to, for example, highlight regions of an image that influence a neural network's decision-making process (Schwalbe and Finzel, 2023). The *saliency* values or attributions are, for example, computed by the derivative of the model's output with respect to the input. Formally, for a model with the interaction function

$$\phi(\mathbf{x}) \in \mathbb{R},$$

a saliency map is computed as

$$\nabla_{\mathbf{x}}\, \phi(\mathbf{x}),$$

where $\mathbf{x}$ is the input space representation (e.g., the embedding of triples, or supgraphs). High gradients in $\nabla_{\mathbf{x}}$ indicate features in $\mathbf{x}$ whose perturbation leads to a decrease in the model's confidence. They are the the most salient. A heat map/saliency map can be constructed from the saliency values (see Figure 9). While the derivative of a model's output is one way to create saliency-based explanations, other methods exist to create saliency maps (Montavon et al., 2019; Ribeiro et al., 2016; Sundararajan et al., 2017).

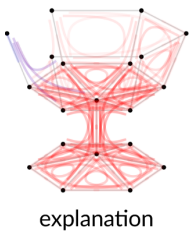


Figure 9: Saliency-based explanation in a graph generated with GNN LRP (Schnake et al., 2022). Each triple is assigned a saliency value and visualized as a heatmap. Red indicates higher importance, and blue indicates lower importance to the prediction.

A set of challenges arises when adapting saliency-based methods to KGs. First, the discrete representation of entities and relations, which are symbolic labels rather than continuous features, complicates the computation of saliency scores for individual symbols. KGC models encode symbols as latent representations. The step back, of decoding the latent representations to symbols, requires a specialized mapping to translate the latent saliency scores into human-interpretable labels or logical rules. Additionally, embedding-based approaches aggregate information across multi-hop paths, so a single entity or relation embedding encodes contextual cues from an entire local neighbourhood. Thereby, again complicating the mapping back of saliency values of latent features to discrete symbols.

Despite these challenges, a growing body of research works on saliency-based explanations for KGC models. For example, Kelpie (Rossi et al., 2022a) computes a saliency score for the triples in the training set. They do this with perturbation analysis (Rossi et al., 2022a). Perturbation analysis systematically removes edges in the local neighborhood of a predicted triple and measures the effect on the model's output. The higher the impact on the model's output, the more salient a training triple is. Although perturbation analysis is computationally intensive, it provides a stable measure of local importance.

Example. Consider a small KG with three entities (`Klara`, `Bob`, and `BMW`) and two relations (`worksAt` and `worksWith`). Suppose the model predicts that "Klara worksWith Bob". Next, an explanation is created by systematically perturbing the embeddings or edges, such as (Bob, $worksAt$, BMW) and ($Klare$, $worksAt$, BMW), and one can observe changes in the prediction score. If removing ($Klara$, $worksAt$, BMW) sharply drops the prediction score of "Klara worksWith Bob," the model learned that Alice's relationship to BMW is essential for predicting the "worksWith"-relation. This leads to a high saliency score and points users to investigate why the presence of that edge influences the model's inference.

Saliency-based explanations, are one way to uncover the decision-making processes of KGC models. Users get insights into the model's internal reasoning mechanisms by quantifying how small perturbations in embeddings or to the graph's topology affect the prediction of novel triples. The relational and symbolic nature of KGs, however, adds complexity to the generation and interpretation of saliency maps.

Discussion. Saliency-based explanations can be useful for downstream model alignment in that they visualise which entities or edges are more impactful on a prediction (see Table 4). However, these explanations are presented as diffuse numeric attributions spread across the KG. This makes it difficult for non-technical stakeholders to interpret and act upon. A saliency score of 1.1 versus 1.05, for instance, offers little intuitive guidance on how to correct or reinforce a particular triple's importance. The lack of a reference point for "ideal" saliency levels undermines the usefulness of saliency-based explanations for the feedback loop, as domain experts cannot easily determine whether a value is too high or too low for alignment purposes. Moreover, adjusting a single saliency value in isolation may have complex downstream implications due to interdependencies across embedding dimensions and multi-hop paths. Although heatmaps provide

Table 4: Pros and cons of saliency-based explanations for the downstream-task of model alignment

Pros	Cons
Visualizes which triples are most impactful on a prediction.	Numeric attributions are diffuse across the graph. This makes interpretation for non-technical stakeholders difficult.
Reveals the model's internal decision-making process.	Lack of a reference for "ideal" saliency levels hinders actionable feedback from domain experts.
	Adjusting a single saliency value can have complex downstream effects due to interdependencies.

a rough indication of important input features, it remains challenging to convert these signals into concrete model corrections in the context of KGC.

Saliency-based methods do reveal a model's internal decision-making process. However, it is prohibitively complex to derive feedback for human-in-the-loop alignment when attributions are presented in numeric form rather than, one interpretation step further, as concrete triples.

3.1.3 Counterfactual Explanations

Counterfactual explanations in KGC identify minimal modifications to a predicted triple's context that lead the model to revise its output. In practice, this involves adding and removing triples in the neighborhood of the head or tail entity, to demonstrate how local changes shift a link from plausible to implausible. For example, if the model predicts $(e^{\text{head}}, r, e^{\text{tail}})$, then introducing or deleting an edge such as $(e^{\text{tail}}, r', e_k)$ makes $(e^{\text{head}}, r, e^{\text{tail}})$ no longer the most plausible triple. These targeted edits identify the decision boundaries of the KGC model.

Counterfactual explanations differ from instance-based or rule-based methods by looking at the ranking behavior of a link prediction model. They show how the addition or removal of edges between nodes undermine or reinforce the ranking behaviour of the model. This perspective reveals the sensitivity of the model's predictions to relatively small local modifications.

Example. Consider a KG (see Figure 10) that includes information about Barack Obama, who is modelled to be the president of the USA. A KGC model predicts that Barack Obama is American. A counterfactual explanation switches the **(Barack Obama, president of America)** fact to the **(Barack Obama, president of Canada)** fact and, with this, leads the model to the prediction that Barack Obama is Canadian. The prediction "Obama is American" is replaced by "Obama is Canadian" as the highest-ranked triple, simply by modifying the neighbourhood of the "Obama" entity. Counterfactual explanations identify minimal, localized edits that redirect or negate a previous inference.

Counterfactual explanations enhance transparency of KGC models. They identify a minimal set of triples required to switch a prediction. This supports human oversight, as domain experts can assess whether the hypothetical change

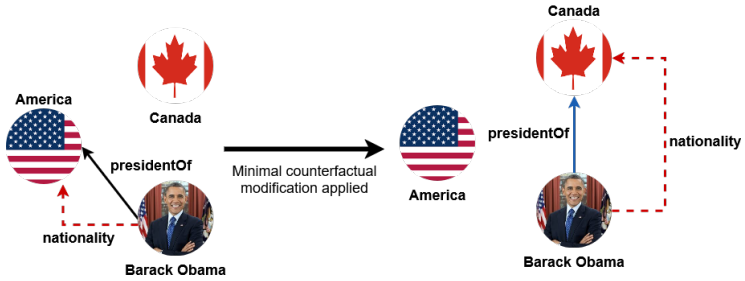


Figure 10: Visualization of a counterfactual explanation for Barack Obama's predicted nationality. The red dashed arrow represents the predicted link, while the blue arrow indicates the minimal modification that shifts the model's inference to Canadian nationality.

(e.g., Obama being president of Canada) is nonsensical or indicates correct model behaviour.

Table 5: Pros and cons of counterfactual explanations for the downstream-task of model alignment.

Pros	**Cons**
Reveals minimal edits that flip a prediction. Thus, it indicates the KGC models' decision boundary.	Requires domain knowledge and awareness of model constraints.
Offers negative examples for training.	Finding minimal edits can be computationally expensive.
Natural for users who think in concrete alternatives or hypotheticals.	Using counterfactuals can bias updates towards negative examples and miss positive validation.

Discussion. Counterfactual explanations can play a valuable role in aligning KGC models with human intent by revealing which minimal edits negate or undermine a prediction (see Table 5). From an alignment perspective, a sound counterfactual explanation can reinforce the model's decision boundary by serving as negative examples during training. Additionally, domain experts find it intuitive to think in terms of concrete alternatives or hypothetical scenarios (Wenzlhuemer, 2009). This makes counterfactuals well-suited for refining model behaviour through iterative feedback. Moreover, when these counterfactuals are treated as additional negative examples in the training or fine-tuning process, they actively steer the model away from incorrect generalizations. However, generating counterfactuals can be nontrivial, as it requires domain knowledge and that the user understands the model's operational constraints. Also, in some cases, identifying the edits is computationally expensive. Furthermore, while negative examples penalise unwanted outcomes, entirely restricting feedback to counterfactuals can bias model updates toward corrective measures. Positive confirmations of valid links are likewise necessary to reinforce predictions that align with user expectations.

Counterfactual explanations offer the user a view of the model's decision boundary. However, they should be used alongside other explanation modalities for comprehensive alignment.

3.1.4 Analogy-Based Explanations

Analogy-based explanations justify a new link by pointing to a similar, already known link. For example (see Figure 11), suppose the model predicts the triple

$$(Sars\text{-}Cov\text{-}2,\ entersCellVia,\ ACE2\ Receptor).$$

To explain this, the system finds an existing fact

$$(Sars\text{-}Cov\text{-}1,\ entersCellVia,\ ACE2\ Receptor)$$

and contextualizes the prediction with additional facts

$$(Sars\text{-}Cov\text{-}1,\ hasProtein,\ Spike\ (S)\ Protein)$$
$$\wedge\ (Spike\ (S)\ Protein,\ bindsTo,\ ACE2\ Receptor),$$

and

$$(Sars\text{-}Cov\text{-}2,\ hasProtein,\ Spike\ (S)\ Protein)$$
$$\wedge\ (Spike\ (S)\ Protein,\ bindsTo,\ ACE2\ Receptor).$$

Sars-Cov-2 and Sars-Cov-1 both have the Spike (S) Protein, which binds to the ACE2 Receptor. The analogy-based explanation suggests that Sars-Cov-1 entering the respiratory cell via the ACE2 Receptor also supports the link from Sars-Cov-2 to the ACE2 Receptor. This explanation shows how a predicted link follows the same pattern as an existing link.

The selection of an analogue triple from the known KG is KGC model dependent. Embedding-based methods use a function

$$f : triple \mapsto v_{triple} \in \mathbb{R}^d$$

to map each triple to an embedding v_{triple} (Bordes et al., 2013). A similarity score, like the Euclidean distance (Wehner et al., 2024), between two triple embeddings is defined as

$$S(triple_1, triple_2) \ = \ \|v_{triple_1} - v_{triple_2}\|_2.$$

In practice, two triples are seen as analogies if their distance S is the closest and below a threshold τ on a set of known triples (Das et al., 2022). When $S(triple_1, triple_2) < \tau$, the system treats $triple_1$ and $triple_2$ as analogous. Euclidean distance is one choice to measure the similarity. Cosine similarity or a learned distance metric are equally valid choices, depending on the embedding function f (Bellet et al., 2015; Wehner et al., 2024).

Analogy-based explanations for symbolic KGC methods explain a predicted triple (e_1, r, e_2) by finding an existing triple (e_3, r, e_4) that can be derived using the same logic rules or path pattern that produced (e_1, r, e_2). Let L denote the rule set or path template used to infer (e_1, r, e_2). The system searches for any known (e_3, r, e_4) such that applying L to (e_3, r, e_4) holds a grounding. When this reconstruction succeeds, (e_3, r, e_4) serves as the analogy. The explanation ties the new link to a known fact justified by the same inference pattern (Meilicke et al., 2019).

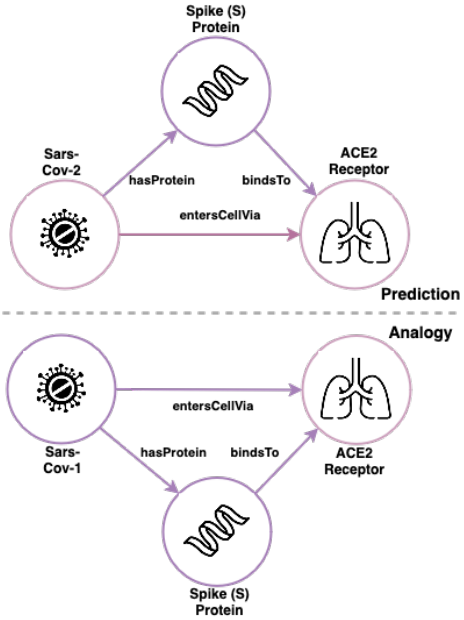


Figure 11: An analogy-based explanation in KGC utilies the analogues and known fact "SARS-CoV-1 entersCellVia ACE2 Receptor" and the shared Spike-protein binding mechanism to justify the predicted link "SARS-CoV-2 entersCellVia ACE2 Receptor."

Table 6: Pros and cons of analogy-based explanations for the downstream-task of model alignment

Pros	**Cons**
Users see which known facts are similar to a new predicted link.	Retrieving top-k analogies for each prediction is costly in large KGs.
Invalid analogies serve as negative examples to correct model overgeneralization.	Analogies can be superficial because of missing context (role, time, source).
Enables concept refinement by showing divergence from domain practice.	

Discussion. Analogy-based explanations let users see which known facts are similar and thus support a new predicted link (see Table 6). For every new triple, the system retrieves the top analogies and asks users to label them. Invalid analogies can be used to correct the model where it is overgeneralizing (e.g., it treats Influenza C as Sars-Cov-2). Thus, analogy-based explanations can be used for concept refinement to show the model where its notion of similarity diverges from domain practice. However, there are two main challenges. The first challenge is to scale analogy-based explanations. On large graphs, pulling the top-k analogies for every prediction can be expensive. Additionally, analogy-based explanations are prone to superficial parallels. Triples with a low

Euclidean distance or an exact path match can differ in unmodeled contexts (role, time, data source).

In summary, analogy-based explanations make link prediction more transparent. They allow users to confirm or reject model inferences using familiar patterns. They show how a new fact mirrors an existing one. However, they suffer from scaling issues and superficial similarities in potential analogous triples.

3.1.5 Rule-Based Explanations

Rule-based explanations for link prediction in KGs ground new inference in logical rules, thereby offering an interpretable rationale for why a predicted triple $(e^{\text{head}}, r, e^{\text{tail}})$ is considered to be true. Rule-based explanations operate directly in the symbolic domain, unlike analogy- or saliency-based approaches, which often work with embeddings or similarity measures.

Example. A KGC model predicts $(SarsCov2,\ entersCellVia,\ ACE2Receptor)$. The rule-based explanation looks as follows:

$$(a,\ entersCellVia,\ b) \leftarrow \big(a, hasProtein, c\big) \wedge \big(c, bindsTo, b\big),$$

the rule says that an entity a enters a respiratory cell via b if a has the protein c and c binds to b. The explanation shows that the predicted link holds because it follows from the known facts and the learned rule. A human stakeholder can assess whether the underlying rule is consistent with domain knowledge, question any suspicious assumptions, or identify missing constraints that should refine the rule set. With that, rule-based explanations provide transparency for link prediction. It empowers users to trace new inferences back to canonical domain statements or high-level "if–then" patterns. In domains where regulatory or ethical considerations demand that ML model decisions be verifiable, rule-based frameworks can fulfil accountability requirements by explicitly exhibiting the conditions under which novel links are predicted.

Table 7: Pros and cons of rule-based explanations for the downstream-task of model alignment

Pros	**Cons**
Surfaces rules governing predictions.	Generating and maintaining a coherent rule set requires substantial domain and model expertise.
Enables validation and targeted correction of high-level rules by domain experts.	Editing rules can introduce unintended effects, such as overfitting or misalignment elsewhere in the KG.
Mirrors expert articulation of domain logic.	Scaling is challenging as the number of potential rule clauses grows significantly.

Discussion. Rule-based explanations are effective for model alignment because they uncover the explicit constraints that govern the prediction process. This allows domain experts to see how the model reasons about a predicted

triple (see Table 7). These explanations enable validation and targeted correction. If a rule appears inconsistent with domain knowledge, human supervisors can intervene to guide the model towards better decision-making. This type of explaining KGC mirrors the way experts articulate domain logic. However, generating and maintaining a coherent set of rules may demand significant domain and model expertise, particularly in large or evolving KGs. Editing a rule can introduce unintended consequences, such as overfitting to specific cases or misalignment elsewhere in the graph. Furthermore, scaling rule-based systems is challenging as the number of potential rules increases significantly with the number of relation types and node degree. Despite these challenges, the transparency of rule-based methods remains appealing for human-in-the-loop workflows, where fine-grained adjustments to model logic can align KGC model behaviour with user expectations.

In summary, rule-based explanations uncover the rules behind predicted links. This supports users in validating and correcting model behaviour. However, they require domain expertise, can be hard to scale, and edits may cause unintended side effects.

3.1.6 Combining Explanation Modalities to Exploit Strengths and Mitigate Drawbacks

All five explanation modalities offer useful perspectives for the downstream task of model alignment in the context of link prediction in KGs, but also suffer from trade-offs.

Instance-based explanations link each prediction to known triples (subgraphs or paths) but struggle to capture broader patterns. Saliency-based explanations show the most influential entities or edges. However, they present diffuse numeric scores that users find hard to act on. Counterfactual explanations pinpoint minimal edits that alter a prediction but can mainly be used to provide negative training triples. Analogy-based explanations retrieve similar known triples that users can use to validate the predicted triple, though analogies may suffer from superficial similarity. Rule-based explanations uncover the rules relevant to prediction, but maintaining a rule set demands domain expertise.

An ideal explanation approach would provide different explanation modalities tailored to the individual and current user demand within a single, scalable framework that remains faithful to the underlying KGC model. This would allow the user to leverage the strengths of each modality while mitigating their drawbacks through complementation. In the next section, we introduce KGEPrisma. This post-hoc explainability method computes instance-based, rule-based and analogy-based explanation modalities simultaneously, scales to large KGs, and is faithful to the decision-making process of KGC models.

3.2 Generating Explanations for Knowledge Graph Embedding Models with KGEPrisma

KGs, as discussed before, are notoriously incomplete (Ji et al., 2022). This led to the investigation of link prediction techniques that infer missing relationships (Hogan et al., 2022). KGE models have become a standard tool for this task, as their high-dimensional latent representations capture complex relational patterns and semantics (Ji et al., 2022). However, these models are criticised

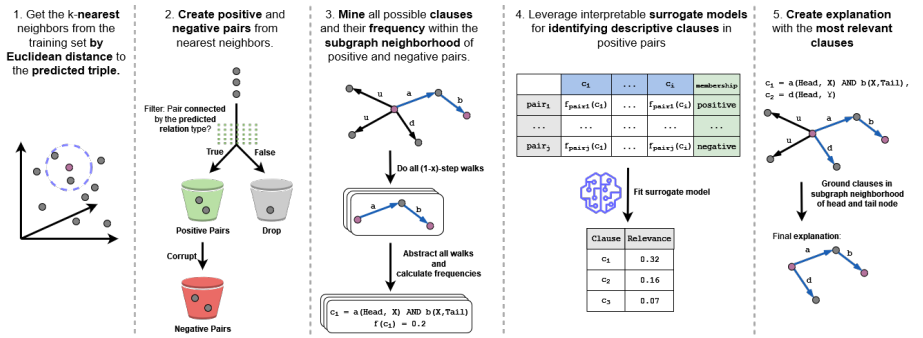


Figure 12: KGEPrisma generates post-hoc explanations of KGE models in five steps. The five steps are discussed in detail in Section 3.2.1.

for their black-box nature (Schramm et al., 2023), which prevents users from understanding how predictions are formed (Schwalbe and Finzel, 2023), an issue that is especially problematic in critical domains (Wehner et al., 2022, 2023; Besold et al., 2025a).

XAI is about uncovering the hidden decision-making process of black-box models (Schwalbe and Finzel, 2023). Traditional explainability methods, such as LIME (Ribeiro et al., 2016), SHAP (Lundberg and Lee, 2017), LRP (Montavon et al., 2019), and Integrated Gradients (Sundararajan et al., 2017), attribute relevance to specific parts of the input. However, embedding-based link prediction works differently compared to traditional ML settings like image classifications. KGE models compute plausibility scores by combining latent representations of head and tail entities with the embedding of a relation, then generate an ordinal ranking over possible predictions (Ji et al., 2022). Assigning feature importance to the embedding dimensions of entities or relations offers limited insight, since the dimensions themselves are not interpretable. The abstract, high-dimensional interactions of KGEs models thus pose unique challenges for providing faithful, user-friendly explanations.

Research Question 1 (see Section 1.1) is about the development of a technique for locally explaining a link prediction model's decision-making process. In response, this work introduces KGEPrisma, a local and post-hoc XAI method designed specifically for KGE models. KGEPrisma builds on the idea of embedding smoothness, meaning that statistical regularities are encoded into the model's latent representations (Bengio et al., 2013). Thus, entities with similar embeddings tend to exhibit similar behaviors in the KG. KGEPrisma uncovers the symbolic regularities on which the KGC model relies by extracting symbolic substructures around similarly embedded entities. These substructures, expressed as human-readable facts or rules, make the model's local decision-making process transparent. Empirical evaluations show that this approach outperforms existing methods in terms of faithfulness to the underlying decision process, and also better localises its explanations around the user's region of interest.

This work contributes three elements. First, it presents a novel local, post-hoc XAI method tailored to KGE models that generates scalable, faithful and context-specific explanations without requiring model retraining. Second, KGEPrisma provides different explanation modalities, such as rule-based, instance-

based, and analogy-based explanations. Third, through comprehensive empirical evaluations, it is demonstrated that the proposed approach is faithful to the KGC models' decision-making process and provides explanations centred on the user's region of interest.

3.2.1 From Latent Representations to Rules, Analogies and Instance-Based Explanations

KGE models abstract the structure and relationships of a KG into latent vectors. Entities that are located in similar positions in this embedding space exhibit similar behaviors in the original graph. This property is called smoothness (Bengio et al., 2013). KGEPrisma makes use of this by examining subgraph neighborhoods of closely embedded entities to uncover frequent symbolic patterns, expressed as rules or triples, on which a KGE model relies. The approach provides a local, post-hoc explanation for a predicted triple (e_p^{head}, r_p, e_p^{tail}) by mapping these latent regularities back to human-understandable clauses (Chang and Lee, 1997). Figure 12 outlines the five steps involved.

1. *k-Nearest Neighbors.* The method starts by identifying a set of triples in the latent space whose embeddings are located near the embedding of the predicted triple. Neighbouring triples in the embedding space are assumed to share structural similarities in the symbolic KG with the predicted triple.

2. *Positive and Negative Entity-Pairs.* Based on the similarly embedded triples, the method creates pairs of entities that are either true (positive) or false (negative) pairs. These pairs serve as examples and counterexamples for the later steps.

3. *Clause Mining.* Next, the subgraph neighbourhoods of these entity pairs are walked to mine conjunctive clauses (Chang and Lee, 1997). With that, the method collects a set of potential explanatory patterns and records how frequently each pattern appears among the pairs.

4. *Surrogate Modeling.* A lightweight and interpretable surrogate model ranks the candidate clauses by examining how well they discriminate positive pairs from negative pairs. Highly discriminative clauses are relevant for the KGE models' decision-making process.

5. *Grounding the Explanation.* Finally, KGEPrisma grounds the top-ranked clauses with entities from the neighborhood of the predicted or similar triple. This instantiation produces an instance-based, rule-based, and analogy-based explanation, which captures the KGE model's local decision surface. The outcome is an interpretable justification of why (e_p^{head}, r_p, e_p^{tail}) is considered plausible by the KGC model.

These steps approximate how the trained embeddings encode symbolic regularities of the KG and show how these regularities can be recovered to explain a KGC model's local decision-making process.

The output of KGEPrisma are instance-based, rule-based and analogy-based explanations. They allow users to get a complementary picture of why the KGC model predicted a link (see Table 8).

Table 8: Examples for the explanation modalities generated by KGEPrisma for the predicted triple $(Alice, knows, Bob)$. For a full description, please refer to Step 5.

Explanation Modality	Example
Rule-Based	$knows(Alice, Person) \wedge works\_with(Person, Bob) \rightarrow knows(Alice, Bob)$
Instance-Based	$knows(Alice, Tom) \wedge works\_with(Tom, Bob)$
Analogy-Based	$knows(Carol, Anja) \wedge works\_with(Anja, Dave)$

The following paragraphs describe each step in detail. KGEPrisma creates human-readable facts and rules from latent embeddings for post-hoc explainability in link prediction tasks.

Step 1: Identifying k-Nearest Neighbors. This first step takes as input the embedding of the predicted triple, denoted by $v_{predicted}$. The process of forming triple embeddings depends on the architecture of the KGE models. Section 3.2.2 describes how some common KGE models embed triples. A simple approach, feasible for example for TransE (Bordes et al., 2013), is to concatenate the head entity embedding v^{head}, the relation embedding v^r, and the tail entity embedding v^{tail} to produce

$$v_{predicted} = \left[v^{head};\, v^r;\, v^{tail}\right].$$

The method then retrieves $k \in \mathbb{N}$ embeddings $\{v_1, v_2, \ldots, v_k\}$ from the model's training triple set V_{train} based on Euclidean distance:

$$kNN(v_{predicted}) = \underset{v \in V_{train}}{\arg\min}_{\,k} ||v_{predicted} - v||_2$$

This operation identifies the embeddings in V_{train} that reside closest to $v_{predicted}$ in the latent space. The Euclidean distance is used as earlier findings showed that Euclidean distance effectively locates similar instances from the training set in other ML-settings (Hanawa et al., 2021).[2] Thus, KGEPrisma looks at the local decision surface of the KGE model to explain a triple. Step 1 isolates the embeddings in the latent space that are most likely to exhibit symbolic regularities in common with the predicted triple, due to embedding smoothness (Bengio et al., 2013). This enables an efficient computation and couples the explanation closely to the model behaviour.

Each retrieved embedding v_i is then mapped back to its symbolic triple representation $(e_i^{head}, r_i, e_i^{tail})$. The relation part r_i is dropped, and the remaining entity pairs

$$N \;=\; \left\{ \left(e_1^{head}, e_1^{tail}\right), \ldots, \left(e_k^{head}, e_k^{tail}\right) \right\}$$

are stored as the neighborhood for further analysis (see Figure 12).

Step 2: Creating Positive and Negative Entity-Pairs. The second step is to generate positive and negative examples from the nearest neighbour pairs N (see Figure 12). Let (e_i, e_j) represent one of the neighbour pairs in N. If the training knowledge graph G contains

$$\left(e_i,\, r_p,\, e_j\right) \;\in\; G,$$

[2]The cosine distance was used in an ablation study. However, no meaningful impact on the method was observed.

where r_p matches the relation type from the predicted triple, then (e_i, e_j) is added to the positive entity pair set P^+. Conversely, (e_i, e_j) is added to the negative entity pair set P^- if

$$\big(e_i,\, r_p,\, e_j\big) \;\notin\; G,$$

as a corrupted instance. Formally:

$$\begin{aligned} P^+ &= \{(e_i, e_j) \in N | (e_i, r_p, e_j) \in G\} \\ P^- &= \{(e_k, e_l) | (e_k = e_i \veebar e_l = e_j) \\ &\quad \wedge (e_k, r_p, e_l) \notin G\} \\ &, with \quad (e_i, e_j) \in P^+ \wedge e \in E \end{aligned}$$

Each $(e_k, e_l) \in P^-$ functions as a corrupted counterpart to an entity pair in P^+. In practice, one negative pair is sampled for each positive pair. This mirrors the stochastic local closed-world assumption commonly used in KGE training procedures (Ali et al., 2021). The step results in two sets. P^+ contains pairs that are connected by the predicted relation type and have a similar latent representation to the predicted triple. P^- holds pairs that are corrupted versions of the positive pairs. It is important to emphasise that P^+ holds the triples from which the KGE model learned its behaviour.

Step 3: Mining Clauses and Computing Frequencies. In the third step, the subgraph neighbourhoods of each entity pair are abstracted into conjunctive clauses (Chang and Lee, 1997), and their frequencies are computed (see Figure 12). This step extracts symbolic patterns, expressed as conjunctive clauses, that a KGE model implicitly uses when forming its predictions. The goal of this step is to describe the graph neighbourhood of each entity pair. Algorithm 1 outlines the process.

For each entity pair (e_{head}, e_{tail}) in $P = P^+ \cup P^-$, the method collects walks w of length 1 to x in G. These walks begin or end at e_{head} or e_{tail} but do not include them as intermediate nodes. Each walk $w = (e_{head|tail}, r_1, e_2, r_2, e_3, r_3, e_4, ..., e_{tail|head})$ represents a sequence of entities and relations anchored in e_{head} or e_{tail}.

Next, a schema mapping

$$\zeta : E \to C \cup \{Head, Tail\}$$

replaces entities in w with their classes. This results in conjunctive clauses (Chang and Lee, 1997) that capture class and relational patterns. The special classes $Head$ and $Tail$ anchor the abstractions to the original (e_{head}, e_{tail}). When the walk is of length 1, an additional pass replaces only e_{head} and e_{tail} with these special classes. The other entity in that walk is not mapped to its class.

This way, the method produces the following conjunctive clause types:

$$\begin{aligned} r_1(A, W) \wedge r_2(X, Y) \wedge ... \wedge r_m(Y, B) \\ r_1(A, W) \wedge r_2(X, Y) \wedge ... \wedge r_m(Y, Z) \\ r_1(X, Y) \wedge r_2(Y, Z) \wedge ... \wedge r_m(Z, A) \\ r(A, e), \end{aligned}$$

Algorithm 1 Clause Mining and Frequency Calculation

Input: Positive pairs P^+, negative pairs P^-, knowledge graph G
Parameter: Maximum walk length x
Output: Dictionary D mapping each pair to its unique clauses and frequencies
for each pair $(e_{\text{head}}, e_{\text{tail}}) \in P^+ \cup P^-$ **do**
    Initialize an empty multiset $S_{(e_{\text{head}}, e_{\text{tail}})}$ to store clauses
    for each walk w in G of length 1 to x, starting or ending at e_{head} or e_{tail}, without including them as intermediate nodes **do**
        Apply schema mapping $\zeta : E \to C \cup \{\text{Head}, \text{Tail}\}$ to abstract the entities in w
        Obtain clause c from the abstracted walk
        Add c to $S_{(e_{\text{head}}, e_{\text{tail}})}$
        if length of w equals 1 **then**
            Abstract only the head and tail entities in w to Head and Tail
            Obtain clause c' from the partially abstracted walk
            Add c' to $S_{(e_{\text{head}}, e_{\text{tail}})}$
        end if
    end for
    for each unique clause c in $S_{(e_{\text{head}}, e_{\text{tail}})}$ **do**
        Compute frequency $f_c = \frac{|\{c \in S_{(e_{\text{head}}, e_{\text{tail}})}\}|}{|S_{(e_{\text{head}}, e_{\text{tail}})}|}$
        Store (c, f_c) in $D_{(e_{\text{head}}, e_{\text{tail}})}$
    end for
end for

where $A, B \in \{Head, Tail\}$ and $W, X, Y, Z \in C$. The value $m \leq x$ is the path length.

Each conjunctive clause c is assigned a frequency

$$f_c = \frac{|c \in S_{(e_{head}, e_{tail})}|}{|S_{(e_{head}, e_{tail})}|},$$

The frequency reflects how often c appears in the set of abstracted walks S. The method saves each clause-frequency pair (c, f_c) in $D_{(e_{head}, e_{tail})}$. The frequency is used to describe each clause for a given entity pair neighbourhood. Here, the goal is to describe the graph neighbourhood of an entity pair. Arguably, other measures are possible to describe a clause in a given entity-pair neighbourhood. For example, in an ablation study, a simple appearance measure was used. If the clause appeared in the neighbourhood, then the clause had a value of 1 in $D_{(e_{head}, e_{tail})}$. The study showed worse results compared to the frequency measure. Thus, the frequency was used. However, other measures to describe the clause can be explored in future work. In Step 4, these clause-frequency pairs are used to select the symbolic patterns that explain the KGE model's local decision-making process.

Step 4: Leveraging Surrogate Models for Identifying Descriptive Clauses in Positive Entity Pairings. The dictionary D provides a structured tabular dataset where each instance corresponds to an entity pair, features

are represented by conjunctive clauses, and values are the frequencies of these clauses. The labels are binary, indicating whether the entity pair belongs to P^+ (positive) or P^- (negative) (see Figure 12). The goal of Step 4 is to identify the clauses that have the most significant influence on classifying an entity pair as positive. This is achieved through surrogate models that are used to estimate feature importances.

The goal is to assign each clause a score that ranks its relevance in classifying an entity pair as positive or negative. For KGEPrisma, three surrogate models are studied and compared: Mean Decrease in Impurity (MDI) (Nembrini et al., 2018), K-Lasso (Ribeiro et al., 2016), and Hilbert-Schmidt Independence Criterion Lasso (HSIC-Lasso) (Yamada et al., 2014). Each method offers a distinct approach to identifying feature importance. An ablation study will compare the surrogate models and shows which is the most faithful way to interpret the data.

The **Mean Decrease in Impurity** method quantifies the impact of each clause in classifying positive or negative samples in D by utilizing an ensemble of decision trees. It learns the trees by iterative data splitting based on clauses that maximize the reduction in impurity, measured by the Gini impurity (Nembrini et al., 2018). The Gini impurity for a dataset $d \subseteq D$ is defined as:

$$Gini(d) = 1 - \sum_i p(i|t)^2$$

where $p(i|t)$ represents the proportion of class $i \in \{positive, negative\}$ at tree-node (dataset) $d \subseteq D$, adjusted by weights α that reflect the Euclidean distance of a pair embedding v_i to the predicted pair's embedding v_p. The weight is defined as an exponential kernel $\alpha(v_i) = \exp(-||(v_p, v_i)||_2^2/\sigma^2)$ with kernel width σ. This assigns a higher impact to pairs that are perceived by the KGE model as similar. The Gini impurity evaluates the likelihood of mislabeling an element if randomly assigned based on the subset's label distribution. It serves as a statistical regularity indicator in D.

The impurity reduction ($\Delta Gini$) (Nembrini et al., 2018) from splitting at tree-node d on clause c, with "positive" (L) and "negative" (R) child nodes, is given by:

$$\Delta Gini(d, c) = Gini(d) - \left(\frac{M_L}{M_d} Gini(L) + \frac{M_R}{M_d} Gini(R)\right)$$

Here, M_d, M_L, and M_R represent the weighted counts of samples at the parent node and in each child node, respectively.

The MDI for a clause across the ensemble is the mean of the impurity reductions, weighted by the samples reaching the nodes where the feature splits the data:

$$MDI(c) = 1 - \frac{\gamma_c}{M} \sum_{d_c \in D} \Delta Gini(d_c, c) M_d$$

where d_c is a node split on clause c, and M_d is the total sample count in d_c. A weighted frequency co-factor γ is applied to the MDI of a clause, defined as:

$$\gamma(c) = \begin{cases} 1 & \text{if } \sum_{f_c \in D_+} f_c \geq \sum_{f_c \in D_-} f \\ -1 & \text{otherwise} \end{cases} \quad (1)$$

This co-factor weights the MDI based on whether a clause is more frequent in positive instances D_+ or negative instances D_- of the dictionary. MDI thereby assesses clause importance. It identifies clauses crucial for positive pair classifications within D, and reveals relevant patterns in similar sub-graph neighbourhoods. However, MDI may favour features with higher cardinality, such as those capturing multi-hop regularities, over property relations, due to inherent biases of MDI toward features with broader variation (Nembrini et al., 2018).

The **K-Lasso** method employs a linear model, specifically ridge regression, to weight each clause's contribution in the classification task within the dictionary D (Ribeiro et al., 2016). This method learns a weight for every feature (clause). It uses linear least squares with Euclidean regularization to optimize the model (Ribeiro et al., 2016). The objective function minimized by this model is formalized as:

$$\min \left(\sum_{(e_i,e_j) \in D} \alpha_{(e_i,e_j)} (y_{(e_i,e_j)} - \mathcal{F}^T_{(e_i,e_j)} w) + \beta ||w||_2^2 \right)$$

Here, $y \in \{1, -1\}$ is the label (positive, negative) of the entity pair $(e_i, e_j) \in D$, $\mathcal{F}$ is the feature vector holding the frequencies of all clauses of an entity pair, and w is the vector of weights corresponding to the clauses. The kernel $\alpha(v_{(e_i,e_j)}) = \exp(||(v_p, v_{(e_i,e_j)})||_2^2/\sigma^2)$ with kernel width σ scales the impact of pairs that are perceived by the model as similar. This allows to weight instances higher that are perceived by the KGE model as closer to the predicted triple. The parameter β is for the Euclidean regularisation. It penalizes the sum of squared weights to prevent overfitting.

After fitting the surrogate model, the learned weights w for each feature provide a direct measure of feature importance. These weights reflect the contribution of each clause to the prediction task, with larger absolute values indicating greater importance. This enables the identification of the most relevant clauses that contribute to classifying entity pairs as positive or negative. It provids insights into the underlying graph regularities captured by the KGE model.

Compared to MDI, K-Lasso is not biased towards features with high cardinality. However, it supports only linear feature selection, which may not capture complex relationships between features in certain datasets effectively.

The **Hilbert-Schmidt Independence Criterion Lasso** is a supervised nonlinear feature selection methodology aimed at identifying a subset of input features relevant to predicting output values (Yamada et al., 2014). As an extension of the standard Lasso, HSIC-Lasso incorporates a feature-wise kernelised Lasso to capture nonlinear dependencies between inputs and outputs. This enables it to identify non-redundant features with a statistical dependence on the output values (Yamada et al., 2014).

The optimisation problem of HSIC-Lasso is formalised as follows:

$$\min_{w_1,\ldots,w_d} \frac{1}{2} ||\mathbf{L} - \sum_k w_k \tilde{\mathbf{K}}^{(k)}||_F^2 + \lambda \sum_k |w_k|$$
$$, with \quad w_1, \ldots, w_d \geq 0$$

where $||\cdot||_F$ denotes the Frobenius norm, $\tilde{\mathbf{K}}^{(k)}$ represents the centered Gram matrix for the k-th feature, and $\mathbf{L}$ is the centered Gram matrix for the output y (Yamada et al., 2014).

After training the surrogate model, coefficients w are obtained. Positive coefficients identify clauses predominantly associated with the sub-graph neighborhood of positive entity pairs once they are multiplied by the co-factor γ (see Equation 1). The coefficients reflect how relevant each clause is to the prediction.

HSIC-Lasso is effective in capturing nonlinear relationships between features and the output. This makes the method suitable for complex datasets where linear surrogate models may fall short. However, it requires careful hyperparameter tuning of the regularization parameter λ to balance the trade-off between feature selection and model complexity.

Step 5: Generating Explanations from the Most Descriptive Clauses. After identifying the most descriptive clauses through the surrogate model, the clauses are used to generate explanations for the KGE model's predictions (see Figure 12). The approach supports three distinct modalities of explanations: rule-based, instance-based, and analogy-based. While all three explanation modalities are derived from the same set of clauses, they differ in how the clauses are grounded and presented. Table 9 provides an illustrative example of these explanation modalities.

Table 9: Comparison of explanation modalities generated from exemplary most descriptive clauses, given the predicted triple $(Alice, knows, Bob)$ and its closest positive neighbour $(Carol, Dave)$. For simplicity, the table shows only the two most relevant clauses identified by KGEPrisma.

	1st Clause	**2nd Clause**	...
Clause	$knows(Head, Person)$ $\wedge\ works\_with(Person, Tail)$	$knows(Head, Person)$ $\wedge\ sibling\_of(Person, Tail)$	...
Relevance	0.54	0.31	...
Rule-Based Explanation	$knows(Alice, Person)$ $\wedge\ works\_with(Person, Bob)$ $\rightarrow\ knows(Alice, Bob)$	$knows(Alice, Person)$ $\wedge\ sibling\_of(Person, Bob)$ $\rightarrow\ knows(Alice, Bob)$	...
Instance-Based Explanation	$knows(Alice, Tom)$ $\wedge\ works\_with(Tom, Bob)$	$knows(Alice, Pedro)$ $\wedge\ sibling\_of(Pedro, Bob)$	...
Analogy-Based Explanation	$knows(Carol, Anja)$ $\wedge\ works\_with(Anja, Dave)$	$knows(Carol, Jan)$ $\wedge\ sibling\_of(Jan, Dave)$	...

Rule-based explanations are constructed by appending an implication to the most relevant clauses, thereby forming a set of symbolic rules. These rules encapsulate the symbolic regularities that the KGE model has learned to predict a missing link. For example, if the most descriptive clause for the predicted triple $(Alice, knows, Bob)$ is $knows(Head, Person) \wedge works\_with(Person, Tail)$, the corresponding rule is formulated as:

$$knows(Alice, Person) \wedge works\_with(Person, Bob) \rightarrow knows(Alice, Bob)$$

This rule implies that if *Alice* knows someone of the class $Person$, and this $Person$ works with Bob, the triple $(Alice, knows, Bob)$ is predicted. Rule-based explanations provide the user with a justification of why the predicted triple

is believed to be true. It shows a pattern that is dominant in the subgraphs around similar embedded training triples.

Instance-based explanations are generated by grounding the most relevant clauses in the KG. Grounding, in logical terms, replaces the variables in a clause with specific constants from the domain, thereby instantiating the clause. For example, if the clause $knows(Head, Person) \wedge works\_with(Person, Tail)$ is identified as relevant for the predicted triple ($Alice$, $knows$,Bob), the grounding process results in:

$$knows(Alice, Tom) \wedge works\_with(Tom, Bob)$$

This type of explanation provides concrete triples from the KG that led the model to predict the *knows*-relation between *Alice* and *Bob*. Instance-based explanations offer tangible justification for the model's predictions.

Analogy-based explanations show how the model behaves in similar known situations by grounding the clauses with the head and tail entities of the closest positive pair in P^+. This approach shows the model's decision-making process in analogous scenarios where the prediction is confirmed to be true by the training facts. For example, if the nearest positive pair to $(Alice, knows, Bob)$ in P^+ is $(Carol, Dave)$, and the conjunctive clause $knows(Head, Person) \wedge works\_with(Person, Tail)$ is relevant, the grounding would be:

$$knows(Carol, Anja) \wedge works\_with(Anja, Dave)$$

This grounding shows an analogous situation where the model applied similar reasoning to predict the *knows*-relation between *Carol* and *Dave*. The explanation states that *Carol* knows *Dave* because *Carol* knows *Anja*, who works with *Dave*, and the model behaved similarly in the predicted case of *Alice* knowing *Bob*. Analogy-based explanations help users understand how the model generalises its predictions across similar instances. Thereby, they provide insights into its KGC model's decision-making process.

Step 5 of KGEPrisma bridges the gap between statistical patterns and concrete, interpretable explanations. It grounds the most descriptive clauses in the entities and relationships of the KG. This step results in the final explanations. It makes the KGE model's decision-making process interpretable to users. Thus, it increases the usability and trustworthiness of the KGE model in critical applications.

Limitations of KGEPrisma. While KGEPrisma is a robust method to explain KGE models, several limitations persist. First, the method requires KGE models to operate in Euclidean space. This makes it incompatible with models such as MuRe (Balažević et al., 2019) and QuatE (Zhang et al., 2019b), which utilise non-Euclidean geometries.

Second, KGEPrisma relies on how the KGE model constructs embeddings, which may vary across different architectures. This potentially limits its generalizability.

Third, its runtime scalability is influenced by the node degree of the KG, as high connectivity increases computational overhead during subgraph neighbourhood exploration in Step 3.

Fourth, the hyperparameter choices, such as the number of nearest neighbors (k) in Step 1, impact explanation quality. Overly large k values introduce noise

and dilute the locality of the explanation. Ideally, k should be set large enough to encompass all embeddings within a cluster. The elbow method (Thorndike, 1953) can be employed to determine the ideal value of k.

Fifth, KGEPrisma assumes that the KGE models' embeddings are solely based on symbolic graph structures. This may lead to reduced faithfulness for KGC models that incorporate literals (e.g., textual attributes).

Finally, KGEPrisma supports diverse explanation modalities, which reduces biases in the overall explanation. However, it does not inherently address potential biases in the underlying KG, which could propagate into explanations.

3.2.2 Evaluation

This section explains the protocol used to compare KGEPrisma with a local random baseline, a global random baseline, and state-of-the-art methods (AnyBURLExplainer (Meilicke et al., 2019), Data Poisoning (Zhang et al., 2019a), and Kelpie (Rossi et al., 2022b)). The datasets Kinship (Kemp et al., 2006), WN18RR (Bordes et al., 2013), and FB15k-237 (Toutanova et al., 2015) are used. Additionally, an empirical runtime evaluation of KGEPrisma and a qualitative evaluation of the three explanation modalities (instance-based, rule-based, analogy-based) are presented in this section.

Datasets. FB15k-237 (Toutanova et al., 2015) is built from the Freebase repository, which covers a wide range of domains such as music, films, locations, books, and people. The original FB15k contained many inverse relations. This resulted in data leakage between training and test sets. FB15k-237 addresses this by filtering out inverse relations. It consists of 310,079 triples, 14,505 entities, and 237 relation types.

WN18RR (Bordes et al., 2013) is an updated variant of WN18, drawn from the WordNet lexical database. WordNet describes semantic and lexical connections between terms, for instance, hyponyms, hypernyms, and antonyms. The older WN18 again experienced data leakage caused by pairs of inverse relations. WN18RR eliminates these pairs, yielding 93,003 triples with 40,943 entities and 11 relation types. Much of the learning challenge involves capturing hierarchical relations like hypernyms and hyponyms.

Kinship (Kemp et al., 2006) models family relationships among members of the Alyawarra tribe in Australia. This dataset contains 10,686 triples, 104 entities, and 26 kinship relation types. Many of these relations are inverse or near-inverse pairs, such as brother or sister relations. This property gives the dataset a focus on symmetrical and inverse connections. It tests how well models and explanation techniques capture reciprocal patterns. For example, if $brotherOf(Tom, Hans)$ is predicted, an effective explanation might show that $brotherOf(Hans, Tom)$ holds as well ($brotherOf$ is not a label used by the Kinship KG). This is a relevant explanation, as it reflects the social structure of the Alyawarra community.

The Knowledge Graph Embedding Models and Triple Embedding Extraction. Three KGE models TransE (Bordes et al., 2013), DistMult (Yang et al., 2015), and ConvE (Dettmers et al., 2018) are trained on these datasets. All models are implemented in PyKEEN (Ali et al., 2021), and hyperparameters follow the settings in (Ali et al., 2021).

The construction of triple embeddings varies across KGE models, as each employs distinct interaction functions to encode relationships between entities and relations. The embeddings necessary for KGEPrisma capture latent interactions between the head entity (e^{head}), relation (r), and tail entity (e^{tail}) before aggregation into a scalar plausibility score. This preserves an information-rich representation of what the model learned. Below, the embedding generation process for three widely used KGE model architectures is described.

TransE (Bordes et al., 2013) is an energy-based approach that interprets each relation as a translation in the embedding space. A fact is plausible if the tail embedding is near the sum of the head embedding and the relation embedding. Thus, TransE interprets entities and relations as linear translations in vector space:

$$\phi(e^{head}, r, e^{tail}) = -\|v^{head} + v^r - v^{tail}\|_2$$

where $v^{head}, v^r, v^{tail} \in \mathbb{R}^n$ represent embeddings of the head entity, relation, and tail entity, respectively (Bordes et al., 2013). Since no additional parameters are introduced, the triple embedding directly concatenates these components:

$$v_{triple} = [v^{head}; v^r; v^{tail}],$$

where $[;]$ denotes vector concatenation. This preserves the raw embeddings.

DistMult (Yang et al., 2015) employs a trilinear dot product. DistMult represents each relation with a diagonal matrix instead of a full matrix, as used in RESCAL (Nickel et al., 2011). Although this design reduces complexity, it cannot model antisymmetric relations. Its interaction function is defined as:

$$\phi(e^{head}, r, e^{tail}) = \sum_{i=1}^{n} v_i^{head} v_i^r v_i^{tail},$$

where $v^{head}, v^r, v^{tail} \in \mathbb{R}^n$ are entity and relation embeddings (Yang et al., 2015). Similar to TransE, the absence of learned parameters, except for the actual embeddings, leads to a straightforward triple representation:

$$v_{triple} = [v^{head}; v^r; v^{tail}].$$

ConvE (Dettmers et al., 2018) enhances the expressivity of the learned embeddings through convolutional neural networks. This approach is considered parameter-efficient and can handle nodes with high in-degree, which are common in large KGs. Unlike TransE and DistMult, it introduces learnable parameters during interaction computation. For inference, the embeddings $v^{head}, v^r \in \mathbb{R}^d$ are concatenated into a matrix $\mathbf{A} \in \mathbb{R}^{2\times d}$, and reshaped to $\mathbf{B} \in \mathbb{R}^{m\times n}$ for spatial processing. A set of 2D convolutional filters $\Omega = \{\omega_i \mid \omega_i \in \mathbb{R}^{r\times c}\}$ extracts localized features from $\mathbf{B}$. This results in feature maps that are flattened into a vector $\mathbf{v} \in \mathbb{R}^{|\Omega|}$. A linear transformation then projects this vector into the entity space:

$$v^{head,r} = \mathbf{v}^\top \mathbf{W},$$

where $\mathbf{W} \in \mathbb{R}^{|\Omega|\times d}$ (Dettmers et al., 2018). The final interaction score integrates the enriched representation with the tail embedding:

$$\phi(e^{head}, r, e^{tail}) = v^{head,r\top} v^{tail}.$$

The triple embedding for ConvE concatenates the head-relation embedding with the tail embedding:

$$v_{triple} = [v^{head,r}; v^{tail}].$$

This formulation encapsulates the convolutional transformations and the original tail representation. ConvE requires a more complex embedding creation than simpler additive or multiplicative models (Ali et al., 2021).

The Explainer. Next, let us have a look at the compared post-hoc explainers. **KGEPrisma** is applied with three different surrogate configurations (MDI, K-Lasso, and HSIC-Lasso) as described in Paragraph 3.2.1. For each configuration, the k-nearest neighbor search is set to $k = 40$ in step 1. The maximal clause length is set to 2 for FB15K-237, to 3 for WN18RR, and to 1 for KINSHIP in step 3. This hyperparameter configuration performed best in an ablation study.

Two random baselines are included for comparison. The **global random baseline** selects a path of length two randomly from the entire training graph. This choice has a high chance of selecting irrelevant paths in large graphs. The **local random baseline** selects a random path of length 2 originating or ending at the head or tail entity of the predicted triple. It thereby restricts the random explanation to graph neighbourhoods with a higher potential impact on the prediction. Both baselines use a path length of 2 for FB15K-237, of 3 for WN18RR, and of 1 for KINSHIP to align with KGEPrisma. Any actual XAI method has to outperform the random baselines in order to be effective.

AnyburlExplainer (Betz et al., 2022) is a black-box approach for generating adversarial explanations for KGE models. It identifies explanations for predicted links through the use of rule learning and abductive reasoning. The method constructs a symbolic representation of the regularities within a KG,

Given a target triple (e_p^h, r_p, e_p^t), AnyburlExplainer applies rule learning to generate a set of symbolic rules Φ, where each rule L has the form:

$$a(X, Y) \leftarrow b(X, Z) \wedge d(Z, Y),$$

and X, Y, Z are variables, $a, b, d \in R$ are relations. The body of the rule, $b(X, Z) \wedge d(Z, Y)$, is grounded using entities and triples in G to generate explanations.

The method identifies a minimal set of triples $T \subseteq G$, referred to as explanation, such that the rule L fires when grounded with the triples in T. The selection of T is guided by the confidence (conf) of the rule:

$$\text{conf}(L) = \frac{\text{correct predictions of } L}{\text{total predictions of } L}.$$

The explanation T is chosen to maximize the confidence across all firing rules.

The approach assumes that KGE models implicitly encode statistical regularities present in the KG and that symbolic rules can explicitly capture these regularities. AnyburlExplainer identifies influential triples by finding rules with a high confidence score. The method achieves competitive performance against state-of-the-art methods while requiring only access to the training data.

Kelpie (Rossi et al., 2022b) is another explainability framework designed for KGE models. It post-hoc identifies subsets of training facts that are either necessary (instance-based) or sufficient (counterfactual) to explain specific predictions.

For a given prediction (e_p^h, r_p, e_p^t)), Kelpie is able to derive two types of explanations: necessary explanations and sufficient explanations. Necessary explanations identify the minimal set of facts whose removal disables the prediction, while sufficient explanations identify the minimal set of facts required to reproduce the prediction for new entities.

A necessary (instance-based) explanation T_n^* is the smallest set of facts/triples such that removing these facts from the training set and retraining the model results in the prediction of e_p^t for $(e_p^h, r_p, ?)$ being unplausible. Formally, the necessary explanation is defined as:

$$T_n^* = \arg \min_{T_n \subseteq G_{\text{train}}} |T_n| \quad \text{such that } \phi'(e_p^h, r_p, e_p^t) < \phi'(e_p^h, r_p, e), \forall e \neq e_p^t,$$

where ϕ is the scoring function of the model after retraining with the modified training set.

A sufficient (counterfactual) explanation T_s^* is the smallest set of facts/triples such that their addition, where e_p^t is not initially predicted, enables the model to predict e_p^t for $(e_p^h, r_p, ?)$. The sufficient explanation is defined as:

$$T_s^* = \arg \min_{T_s \subseteq G} |T_s| \quad \text{such that } \phi'(c, r_p, e_p^t) > \phi'(c, r_p, e), \forall e \neq e_p^t$$

Kelpie evaluates candidate explanations efficiently through post-training. This process freezes all embeddings except for the entity being explained, which is re-embedded based on a modified set of training facts. Post-training creates mimics of the entity in question to simulate how the model's predictions would change if specific facts were added or removed. Additionally, the framework uses a so-called pre-filter to reduce the search space of possible modifications. Next, a relevance engine quantifies the impact of candidate explanations by analysing how modifications to the training data influence model predictions. The explanation builder combines these components to identify minimal and relevant explanations that satisfy the necessary or sufficient criteria.

For instance, in the case of the prediction (`Barack Obama`, `nationality`, `USA`), a set $T_n^* = \{$(`Barack Obama`, `born_in`, `Honolulu`), (`Honolulu`, `located_in`, `USA`)$\}$ constitutes a necessary explanation if removing these facts changes the prediction of nationality from USA to another entity. On the other hand, a sufficient explanation might consist of a single fact, such as $T_s^* = \{$(, `president_of`, `USA`)$\}$, which, when added to unrelated entities, causes the model to predict their nationality as USA. In this evaluation, we focus only on the necessary explanations.

Data Poisoning (Zhang et al., 2019a) is a framework designed to manipulate the plausibility of specific facts in a KGE model through adversarial modifications. For a given target triple (e_p^h, r_p, e_p^t), the objective is to modify the training set through data poisoning, either by adding or deleting specific triples, to decrease or increase the plausibility score of the target triple.

The attack strategy is formulated as follows. Given an initial training set G_{train}, the attacker introduces a limited number of perturbations, represented as T, and seeks to minimize:

$$\min_T \phi(e_p^h, r_p, e_p^t),$$

Two attack strategies exist. Direct attacks manipulate facts that explicitly involve the target entities, while indirect attacks introduce perturbations that propagate their effects through intermediate entities. This makes the attack more difficult to detect. Data poisoning leverages the KGE model's gradient to identify those facts (Zhang et al., 2019a).

Data poisoning reveals vulnerabilities in KGE models. However, the adversarial triples in T can also be used to get insights into the model behavior and, thus, to explain the model.

The Faithfulness Evaluation Protocol. Faithfulness, as defined by Hedström et al. (2023), quantifies how well explanations reflect the model's decision-making process. Thus, the most influential features should significantly impact predictions. In traditional XAI, this is validated through input perturbation, where the most relevant features are removed to observe the resulting decline in model performance. However, this technique is not applicable to KGE models, as there is no sensible way to occlude parts of the model input. Thus, model retraining is required.

To adapt perturbation-based faithfulness evaluation to KGE models, we first train a KGE model on the full dataset G_{train} and select 20 well-performing triples H from the validation set. The triples are selected to have Hits@1 (Ali et al., 2021) and MRR (Ali et al., 2021) scores of 1. For each triple in H, an explanation T is generated. The explanation consists of all grounding triples supporting the explanation rule or clause. These triples are then removed from G_{train}. This creates a modified training set $G'_{train} = G_{train} \setminus T$, on which the model is retrained from scratch. The model's performance on H is then remeasured using Hits@1 and MRR, where lower values indicate greater degradation and, consequently, a more faithful explanation (Rossi et al., 2022b; Betz et al., 2022).

As noted by Betz et al. (2022), maintaining an unaltered filter set for Hits@1 and MRR calculations after removing T from G_{train} is essential to prevent artificially lowering model performance. This ensures that observed degradation reflects the impact of the explanation.

Table 10: Results for FB15K-237. The results are the mean and variance of the MRR and Hits@1 over ten runs; the lower the MRR and Hits@1, the better. The best results are bold.

	TransE		DistMult		ConvE	
	MRR	Hit@1	MRR	Hit@1	MRR	Hit@1
Retraining	0.97 ± 0.03	0.94 ± 0.06	0.84 ± 0.12	0.78 ± 0.16	0.84 ± 0.07	0.71 ± 0.00
Global Random	0.96 ± 0.03	0.93 ± 0.05	0.82 ± 0.15	0.69 ± 0.27	0.76 ± 0.22	0.54 ± 0.43
Local Random	0.96 ± 0.03	0.93 ± 0.04	0.70 ± 0.24	0.51 ± 0.36	0.69 ± 0.38	0.64 ± 0.43
AnyBURLExplainer	0.95 ± 0.02	0.92 ± 0.05	0.41 ± 0.35	0.25 ± 0.26	0.40 ± 0.25	0.16 ± 0.29
Data Poisoning	0.96 ± 0.03	0.92 ± 0.05	0.40 ± 0.37	0.26 ± 0.29	0.41 ± 0.31	0.22 ± 0.36
Kelpie	**0.94** ± 0.03	**0.90** ± 0.06	0.37 ± 0.38	0.27 ± 0.33	0.49 ± 0.32	0.30 ± 0.36
KGEPrisma - K-Lasso	0.95 ± 0.04	0.92 ± 0.06	0.34 ± 0.08	**0.00** ± 0.00	0.41 ± 0.37	0.28 ± 0.41
KGEPrisma - HSIC-Lasso	0.95 ± 0.02	0.92 ± 0.04	**0.00** ± 0.00	**0.00** ± 0.00	**0.11** ± 0.09	**0.00** ± 0.00
KGEPrisma - MDI	0.95 ± 0.03	0.91 ± 0.05	**0.00** ± 0.00	**0.00** ± 0.00	0.32 ± 0.32	0.19 ± 0.37

Faithfulness Evaluation and Discussion. This paragraph reports the results of the faithfulness evaluation protocol. The results for **FB15k-237**, as detailed in Table 10, indicate that KGEPrisma consistently outperforms the

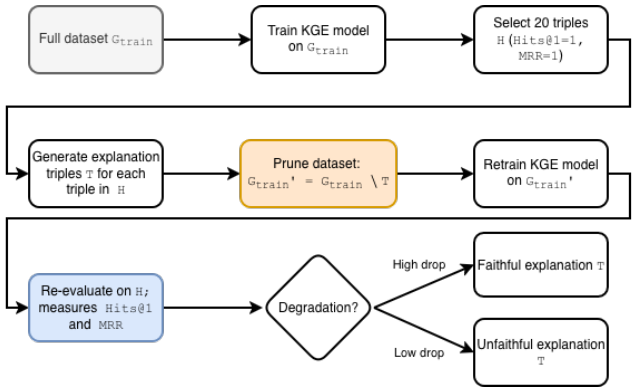


Figure 13: Flowchart of the perturbation-based faithfulness evaluation protocol for KGE models. Starting from the full dataset, the model is trained. Top-performing triples are selected and explained. The explanation triples are pruned. The model is retrained, and the resulting performance degradation on the held-out triples is measured to assess the faithfulness of the explanation.

other XAI methods across all KGE models. For TransE on FB15K-237, every method, including the random baselines, yields nearly identical MRR and Hits@1 scores. TransE's predictions on FB15K-237 rely on self-loop relations. For example, TransE learned to predict that `Kansas City` is linked to itself via `/location/hud_county_place/place` or `Virginia Tech` is linked to itselfe through `/education/educational_institution/campuses`. These self-loop relations require mostly memorization, not subgraph patterns. This suggests that TransE's scoring function falls short in capturing complex rules in FB15K-237.

DistMult on FB15K-237 shows clearer differences among methods. KGEPrisma achieves the lowest MRR and Hits@1, with minimal variance across runs. When using K-Lasso, its performance matches other approaches. KGEPrisma with HSIC-Lasso or MDI drives DistMult's scores to zero after retraining. This indicates that these variants of KGEPrisma are effective at identifying the triples most responsible for the model's predictions. An example helps to illustrate this point. DistMult predicts that `Jessica Eisenberg's` net worth is measured in `U.S. dollars`. KGEPrisma explains this prediction by pointing out that Eisenberg acted in films such as `The Social Network`, `Adventureland`, and `The Village`, all of which had budgets measured in `U.S. dollars`. This chain of reasoning offers a sensible and contextually relevant explanation for the model's prediction. In contrast, the other methods, like AnyBURLExplainer, link this prediction to an award nomination involving `Bill Hader`, a fact that appears unrelated and does not provide a coherent justification for the predicted triple. The lack of relevance in this explanation is consistent with the method's overall lower faithfulness scores and higher variance.

Results for ConvE on FB15K-237 are more mixed. While all methods show some capacity to identify high-impact explanation triples, KGEPrisma with HSIC-Lasso outperforms the others substantially. The remaining methods produce average performance scores that are somewhat comparable to one another. However, the associated variances are notably higher. This suggests that their

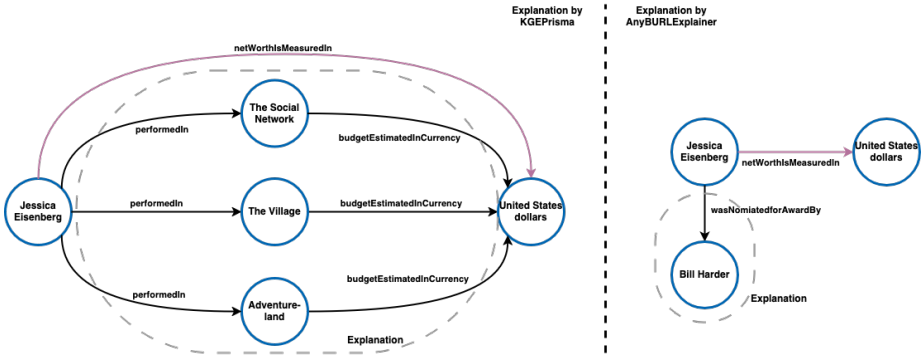


Figure 14: DistMult predicts that Jessica Eisenberg's net worth is expressed in US dollars. KGEPrisma points to her films and their budgets in dollars. AnyBURLExplainer cites her award nomination alongside Bill Hader, which is irrelevant to justify the predicted triple.

ability to find influential triples is less reliable. A closer examination of several explanation examples confirms that. Although multiple methods occasionally surface valid explanation triples, only KGEPrisma with HSIC-Lasso does so consistently across multiple runs.

Overall, KGEPrisma produces more faithful explanations on FB15K-237 compared to the other three KGE models. Its local reconstruction of the model's decision surface yields explanation subgraphs that closely mirror the embedding model's decision-making process. A similar pattern of results is observed on the WN18RR dataset.

Table 11: Results for WN18RR. The results are the mean and variance of the MRR and Hits@1 over ten runs; the lower the MRR and Hits@1, the better. The best results are bold.

	TransE		DistMult		ConvE	
	MRR	Hit@1	MRR	Hit@1	MRR	Hit@1
Retraining	0.86 ± 0.04	0.78 ± 0.07	0.92 ± 0.08	0.88 ± 0.12	0.74 ± 0.28	0.62 ± 0.31
Global Random	0.75 ± 0.20	0.61 ± 0.27	0.91 ± 0.04	0.86 ± 0.06	0.65 ± 0.25	0.57 ± 0.23
Local Random	0.76 ± 0.22	0.65 ± 0.28	0.92 ± 0.03	0.87 ± 0.05	0.67 ± 0.28	0.58 ± 0.28
AnyBURLExplainer	0.26 ± 0.06	0.16 ± 0.06	**0.22** ± 0.05	**0.14** ± 0.04	0.17 ± 0.13	0.02 ± 0.07
Data Poisoning	0.42 ± 0.20	0.32 ± 0.23	0.39 ± 0.05	0.33 ± 0.06	0.07 ± 0.05	**0.00** ± 0.00
Kelpie	0.37 ± 0.11	0.28 ± 0.11	0.23 ± 0.05	0.16 ± 0.05	0.08 ± 0.05	**0.00** ± 0.00
KGEPrisma - K-Lasso	**0.05** ± 0.05	**0.02** ± 0.06	0.37 ± 0.07	0.30 ± 0.08	0.06 ± 0.11	**0.00** ± 0.00
KGEPrisma - HSIC-Lasso	0.10 ± 0.05	0.04 ± 0.04	0.38 ± 0.04	0.31 ± 0.04	**0.01** ± 0.01	**0.00** ± 0.00
KGEPrisma - MDI	0.15 ± 0.10	0.10 ± 0.11	0.26 ± 0.06	0.20 ± 0.05	**0.01** ± 0.01	**0.00** ± 0.00

Table 11 shows the performance of KGEPrisma on **WN18RR**. KGEPrisma achieves state-of-the-art results on WN18RR. When applied to TransE on WN-18RR, KGEPrisma outperforms all other approaches across each surrogate model, with the K-Lasso surrogate performing best. Nonetheless, the low variance suggests only slight differences among the three surrogates. A side-by-side analysis of the explanations produced by KGEPrisma and the next best method, AnyBURLExplainer, shows that both approaches tend to recover identical justification patterns. For example, both approaches explain the predicted `hypernym` link between `Platichthys` and `fish_genus` by the inverse `member_meronym` relation. However, because the underlying KG lacks certain inverse facts, there are situations where the `member_meronym` edge is missing. In these cases, AnyBURL-

Explainer cannot offer any explanation. In contrast, KGEPrisma is able to find longer, multi-step reasoning chains. This can be observed for the prediction that `family_ Treponemataceae` stands in a `hypernym` relation to `bacteria_family`, where KGEPrisma constructs the following sequence of triples:

$$(order\_Spirochaetales,\ member\_meronym,\ family\_Treponemataceae),$$
$$(division\_Eubacteria,\ member\_meronym,\ order\_Spirochaetales),$$
$$(division\_Eubacteria,\ member\_meronym,\ bacteria\_family)$$

These triples establish that the Treponemataceae family is a part of the Spirochaetales order, which itself belongs to the Eubacteria division. Since the division also includes `bacteria_family`, it follows that `bacteria_family` serves as a hypernym of `family_Treponemataceae`. KGEPrisma recovers this multi-hop reasoning by focusing on the local neighborhood of the predicted triple, while AnyBURLExplainer stops short.

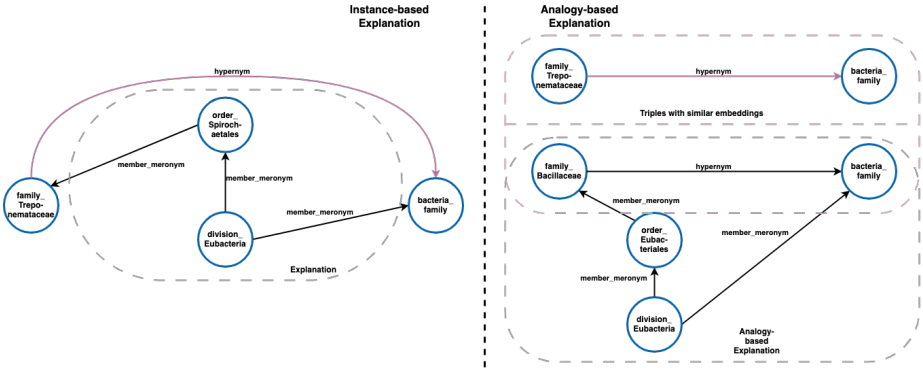


Figure 15: TransE in WN18RR predicts that `bacteria_family` serves as a `hypernym` for `family_Treponemataceae`. An instance-based explanation from KGEPrisma shows that `family_Treponemataceae` is a meronym of `order_Spirochaetales`, which belongs to `division_Eubacteria`, and that this division also includes `bacteria_family`. Thus, the explanation clarifies why `bacteria_family` is considered a hypernym of `family_Treponemataceae`. In addition to the instance-based explanation, the analogy-based explanation provided by KGEPrisma points out to the user that the triple (`family_Bacillaceae`, `hypernym`, `bacteria_family`) is similar to the predicted triple. In the context of this triple, we know that the `family_Bacillaceae` is a subset of the `order_Eubacteriales`, which in turn belongs to the `division_Eubacteria` that includes the `bacteria_family`. This analogous example reassures the user that the reasoning pattern leading to the prediction aligns with other established facts within the KG.

On WN18RR with DistMult, AnyBURLExplainer attains the highest faithfulness score. Yet, the observed variance indicates that its performance matches that of KGEPrisma with the MDI surrogate and Kelpie. The qualitative inspection of explanations confirms that all three methods often propose equivalent justifications. A similar behaviour is observable for ConvE on WN18RR. KGEPrisma achieves the highest average performance, but only by a small margin. Overall, KGEPrisma consistently outperforms AnyBURLExplainer, Kelpie, and Data Poisoning on WN18RR. A KGEPrisma shows comparable results on the Kinship KG.

Table 12: Results for KINSHIP. The results are the mean and variance of the MRR and Hits@1 over ten runs; the lower the MRR and Hits@1, the better. The best results are bold.

	TransE		DistMult		ConvE	
	MRR	Hit@1	MRR	Hit@1	MRR	Hit@1
Retraining	0.89 ± 0.08	0.75 ± 0.09	0.64 ± 0.13	0.55 ± 0.07	0.94 ± 0.05	0.92 ± 0.04
Global Random	0.83 ± 0.03	0.77 ± 0.05	0.57 ± 0.12	0.49 ± 0.08	0.86 ± 0.07	0.78 ± 0.10
Local Random	0.80 ± 0.06	0.73 ± 0.09	0.53 ± 0.13	0.045 ± 0.03	0.83 ± 0.05	0.75 ± 0.08
AnyBURLExplainer	0.60 ± 0.07	0.33 ± 0.12	**0.04** ± 0.02	**0.00** ± 0.01	0.72 ± 0.06	0.57 ± 0.08
Data Poisoning	**0.05** ± 0.03	**0.01** ± 0.04	0.10 ± 0.04	0.04 ± 0.04	0.75 ± 0.04	0.61 ± 0.06
Kelpie	**0.05** ± 0.03	**0.01** ± 0.02	0.10 ± 0.06	0.03 ± 0.05	0.72 ± 0.05	0.56 ± 0.08
KGEPrisma - K-Lasso	0.52 ± 0.07	0.28 ± 0.09	0.05 ± 0.03	**0.00** ± 0.00	**0.65** ± 0.09	**0.50** ± 0.12
KGEPrisma - HSIC-Lasso	0.58 ± 0.07	0.33 ± 0.11	**0.04** ± 0.01	**0.00** ± 0.00	0.72 ± 0.07	0.57 ± 0.10
KGEPrisma - MDI	0.51 ± 0.06	0.23 ± 0.07	0.08 ± 0.04	0.03 ± 0.05	0.70 ± 0.08	0.55 ± 0.10

In the **Kinship** KG (see Table 12), KGEPrisma's explanations lead to the lowest (best) MRR and Hits@1 for two out of three embedding models.

For TransE, trained on KINSHIP, the adversarial explainers Kelpie and Data Poisoning outperform KGEPrisma and AnyBURLExplainer. Kelpie and Data Poisoning reliably recover inverse relations, which TransE in KINSHIP depends on heavily to reconstruct missing links (e.g., recovering the relation `term2` via its inverse `term1`). The other XAI methods frequently miss these inverses and instead propose alternative paths that, while plausible, do not mirror TransE's decision-making process. Nonetheless, KGEPrisma and AnyBURLExplainer still achieve competitive faithfulness scores.

All explainability methods attain similar faithfulness for DistMult trained on KINSHIP. KGEPrisma with the HSIC-Lasso surrogate shows marginally lower variance. Qualitative inspection confirms that KGEPrisma and AnyBURLExplainer often produce identical justifications. For instance, the prediction

$$(person23,\ term15,\ person29)$$

is explained by both methods through the inverse triple

$$(person29,\ term5,\ person23).$$

Explaining ConvE trained on KINSHIP proves more difficult. Every method outperforms the retraining and random baselines. However, ConvE still reconstructs more than half of its predictions (Hit@1 > 0.50) after all explanation triples have been removed from the training graph. Example explanations across methods are qualitatively similar and not indicate why ConvE retains this level of performance.

Overall, KGEPrisma maintains strong faithfulness across all KGE models and KGs. It matches or surpasses the other XAI methods in most settings.

Runtime Evaluation. KGEPrisma exhibits competitive runtime performance compared to the other XAI methods. Figure 16 reports the mean execution time in seconds across ten runs (40 explanations per run) for AnyBURLExplainer, Kelpie, Data Poisoning, and KGEPrisma on the KINSHIP KG.

The experiment setup uses the same implementation as the faithfulness evaluation and runs on an AWS EC2 g5.12xlarge[3] instance.

[3] `https://aws.amazon.com/ec2/instance-types/g5`

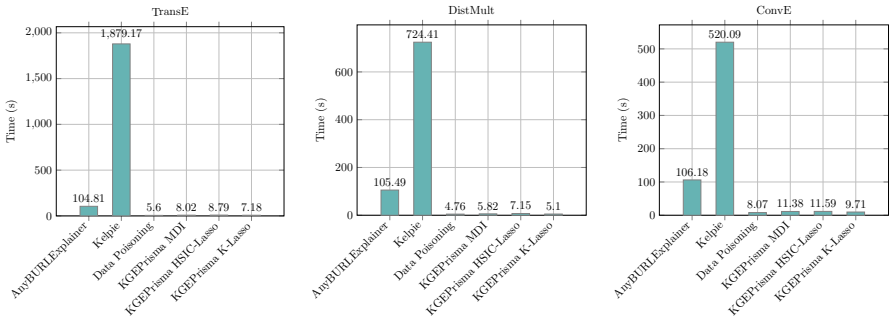


Figure 16: Mean execution time (over 10 runs on the same hardware, 40 explanations per run) for AnyBURLExplainer, Kelpie, Data Poisoning, and KGEPrisma explaining TransE, DistMult, and ConvE in the Kinship KG.

The runtime evaluation results show that KGEPrisma generates explanations for TransE, DistMult, and ConvE approximately 10 to 20 times faster than AnyBURLExplainer and about 50 to 200 times faster than Kelpie.

This speed advantage stems from KGEPrisma's localisation strategy, which samples subgraphs around the triples that the model regards as similar to approximate the KGE model's local decision surface (see Figure 12, Steps 1–3). In contrast, Kelpie applies perturbations over multiple triples, guided by heuristics, which remains computationally intensive (Rossi et al., 2022a). And AnyBURLExplainer enforces a fixed time budget independent of the embedding model. It returns the best explanation found within that window (Betz et al., 2022).

In addition to its high faithfulness, these results demonstrate that KGEPrisma is a time-efficient method for post-hoc explanation of KGE models.

Qualitative Discussion of the Three Explanation Modalities. The quantitative results show that KGEPrisma matches the state-of-the-art performance in faithfulness and also offers fast execution. Additionally, KGEPrisma produces three complementary explanation modalities, instance-based, rule-based, and analogy-based. Each modality sheds light on the same predicted triple from a different angle. This way, KGEPrisma equips users and developers with a multifaceted contextualisation of a predicted link, which, in turn, establishes trust in the model's output. What follows is a qualitative illustration of how these modalities work.

Take the prediction from the WN18RR KG that claims `bacteria_family` is a `hypernym` of `family_Treponemataceae`. The instance-based explanation (see Figure 15) shows a chain of justifying triples:

$$(order\_Spirochaetales,\ member\_meronym,\ family\_Treponemataceae),$$
$$(division\_Eubacteria,\ member\_meronym,\ order\_Spirochaetales),$$
$$(division\_Eubacteria,\ member\_meronym,\ bacteria\_family).$$

This explanation shows that the Treponemataceae family is part of the Spirochaetales order, which itself is contained in the Eubacteria division, a division that also includes the bacteria family. In this way, the instance-based explanations shows domain experts why `bacteria_family` serves as a hypernym of `family_Treponemataceae`.

While the instance-based explanation delivers example-driven context, its scope is limited to the specific entities involved. To reveal general, but still local, reasoning patterns, KGEPrisma provides rule-based explanation. For the prediction `bacteria_family` is a `hypernym` of `family_Treponemataceae`, KGEPrisma uncovered the rule:

$$\begin{aligned} hypernym(family\_Treponemataceae, bacteria\_family) \leftarrow \\ member\_meronym(X, family\_Treponemataceae) \wedge \\ member\_meronym(Y, X) \wedge \\ member\_meronym(Y, bacteria\_family). \end{aligned}$$

The rule indicates that whenever an entity X is a meronym of `family_Treponemataceae`, and some Y is a meronym of X and `bacteria_family`, one should expect the hypernym relation to hold. This rule describes that the model's decision follows a recurring structural motif across similar parts of the graph.

Finally, the analogy-based explanation draws a parallel to a known case from the trainings KG, which is similarly represented in the embedding space. KGEPrisma notes that `family_Bacillaceae` is closely embedded to `family_Treponemataceae` in the learned representation, and it presents as an analogy-based explanation:

$$\begin{aligned} (order\_Eubacteriales,\ member\_meronym,\ family\_Bacillaceae), \\ (division\_Eubacteria,\ member\_meronym,\ order\_Eubacteriales), \\ (division\_Eubacteria,\ member\_meronym,\ bacteria\_family). \end{aligned}$$

By showing that the same relational pattern holds for a similar family, the analogy reassures the user that the reasoning behind the prediction is consistent with other known facts.

Together, these three explanation modalities, instance-based, rule-based, and analogy-based, offer complementary insights. The different contextualisations help users to understand and trust the KGE model explained by KGEPrisma.

This section has presented KGEPrisma, a post-hoc explainability framework designed for KGE models. KGEPrisma exploits embedding smoothness to translate latent representations into human-readable rules, facts, and analogies by extracting symbolic patterns from the local neighborhoods of similarly embedded entities. This approach operates without retraining the underlying KGE model. It is faithfulness to the KGE model's behaviour, and supports three complementary explanation modalities.

Evaluation on benchmark KGs (WN18RR, Kinship, and FB15k-237) has shown that KGEPrisma matches or surpasses current state-of-the-art methods in faithfulness, while achieving 10–200× faster runtimes. Its ability to generalize recurring patterns and draw analogies in the embedding space is robust across diverse KGE architectures.

Future work will apply KGEPrisma in domains where transparent model behaviour is critical, like in drug repurposing with biomedical KGs, so that explanations enhance decision-making and build user trust (Besold et al., 2025a). KGEPrisma makes KGE models more understandable and reliable in high-stakes applications. It identifies symbolic rules and facts that guide embedding-based link prediction.

3.3 Summary and Answer to Research Question 1

Research Question 1 asked how the decision-making of link prediction models can be explained to support domain experts in aligning model behaviour with

human intent. To address this, the section first looked at existing explanation modalities (instance-based, saliency-based, counterfactual, rule-based, and analogy-based), discussing each modality's advantages and downsides in terms of interpretability and usability for the downstream task of model alignment.

The discussion concluded that an ideal explanation approach must provide multiple explanation modalities tailored to different user needs within a single, scalable framework that remains faithful to the underlying link prediction model. Such a unified solution enables users to leverage the strengths of different modalities while mitigating their drawbacks. Motivated by these requirements, the section introduced KGEPrisma, a post-hoc explainability method capable of computing multiple complementary explanation modalities simultaneously, scaling to large graphs, and preserving faithfulness to the internal decision-making process of KGE model.

KGEPrisma decodes embedding-based predictions by extracting symbolic patterns from the local subgraph neighborhoods of similarly embedded triples. Its five-step pipeline (k-nearest neighbors, positive/negative pair selection, clause mining, surrogate modelling, and grounding) generates three complementary explanation styles:

- **Instance-based explanations**, which assemble concrete triples that directly influenced a prediction;
- **Rule-based explanations**, which generalize recurring subgraph patterns into human-readable symbolic rules;
- **Analogy-based explanations**, which identify embedding-space neighbors and present parallel reasoning chains from known facts.

Empirical evaluation on FB15K-237, WN18RR, and KINSHIP demonstrated that KGEPrisma explanations achieve state-of-the-art faithfulness while running 10–200× faster than competing methods. Qualitative examples confirmed that instance-based, rule-based, and analogy-based explanations provide insightful and user-friendly examples.

KGEPrisma provides a multi-modal explanation framework that answers Research Question 1 by generating faithful and efficiently computed explanations for embedding-based link predictions. This enables users to understand model behavior and paths the way to critique it.

4 On Feedback Modalities for Aligning Link Prediction in Knowledge Graphs

Research Question 2: How effective are different types of feedback modalities in collecting user knowledge for the downstream task of aligning a link prediction model with human intent?

Here, the focus is on identifying, formalizing and discussing the diverse types of feedback modalities users can provide on model explanations, including rules and preferences, for the downstream task of aligning a link prediction model. This research question investigates how effective different feedback modalities impact the model's training or inference processes. Thus, with Research Question 2, we will answer how different feedback modalities can be used to align with human intent from a user's perspective.

The literature on human-in-the-loop machine learning describes different feedback modalities, such as preference elicitation (Christiano et al., 2017), rating-based supervision (Knox and Stone, 2011), and demonstration of correct behaviors (Teso and Kersting, 2019). In domains like reinforcement learning and image classification, feedback is obtained by evaluating observed behaviours or visual outputs. In contrast, KGs are inherently graph-structured, with entities and relations. This forms a web of interdependencies. Such graph-structured data does not offer the same intuitive visual cues or behavioural patterns that facilitate direct feedback in other domains. That is why aligning link prediction models with human feedback poses a challenge. Within KGs, user knowledge can be used to guide the KGC models' decision-making behaviour to follow patterns within the graph-structure. Guiding the model's decision-making process requires feedback on the explanations that underlie the predictions. In this context, the second research question focuses on discussing the effectiveness of different feedback modalities for link prediction in KGs.

This section begins by investigating a real-world manufacturing use case where a Bayesian network predicts missing cause-effect relationships in a manufacturing process KG. The use case provides a concrete example of aligning link prediction with naive negative examples. It thus provids a setting to explore the challenges of collecting and integrating user feedback. Qualitative domain experts interviews motivate a discussion about which feedback forms (e.g., explicit rules, preferences, ratings, or example-based corrections) are most valuable for the downstream task of model alignment [4].

The primary focus of this section is to understand how and in what form user feedback on model explanations can improve link prediction performance. The investigation discusses whether certain feedback modalities, when applied to the KG structure and corresponding link prediction model, lead to more robust and useful feedback for the downstream task of model alignment.

Several types of feedback modalities are considered in this research:

- **Positive Examples**: Valid instances with explanations that demonstrate correct model behaviour.

[4] This section is adapted from the author's prior publications: Wehner et al. (2023), Eirich et al. (2023), Schramm et al. (2023), (Poßner et al., 2025), and Bahr et al. (2025).

- **Negative Examples**: Incorrect or misleading instances with explanations that indicate flaws in the model's behaviour.
- **Rules**: Logical statements that define valid relational structures in the KG domain.
- **Preferences**: User preferences over instances with explanations. Prefered instance demonstrate more desirable model behaviour.
- **Ratings**: Quantitative user assessment of instances and explanations that captures how well the model's justification aligns with domain knowledge.

The section discusses the benefits and limitations of feedback modalities for human-in-the-loop link prediction in KGs. Let us start by looking at the use case of aligning predicted cause-effect relations for RCA in electric vehicle manufacturing with naive negative example-based human feedback.

4.1 Interactive Root Cause Analysis in Electric Vehicle Manufacturing with Causal Bayesian Networks and Knowledge Graphs

Electric vehicle production makes use of sensor networks that monitor thousands of process variables in real-time. This generates large amounts of data hour by hour. The data includes quality control records that identify faults detected in various process steps. However, detecting a fault in a highly complex manufacturing process does not automatically reveal the process step where the fault was induced. Understanding the CERs between manufacturing variables and diagnosed defects is the task of RCA (Oliveira et al., 2022).

Traditionally, RCA is a manual, knowledge-intensive, and costly process (Serrat, 2017; Tay and Lim, 2008; Ruijters and Stoelinga, 2015). It requires experts to design and evaluate controlled experiments to analyze the propagation of faults. Given the complexity of modern production systems, only experienced process experts can conduct RCA effectively within a reasonable timeframe. Thus, the expert labour shortage and the scale of electric vehicle manufacturing create an increasing demand for intelligent decision-support tools to make RCA efficient, data-driven, and less dependent on individual process knowledge (Oliveira et al., 2022).

CBNs are a common approach for automating RCA by learning CERs between manufacturing sensor variables (Pradhan et al., 2007). These networks construct a graph, in the latter called a root cause graph, over the manufacturing process. They learn causal dependencies as labeled and directed edges between sensor variable nodes (Kirchhof et al., 2020). However, learning CBN in large-scale manufacturing environments is computationally prohibitive due to the explosion of possible cause-effect relations. Incorporating process expert knowledge into the learning process is a way to mitigate this challenge (Kertel et al., 2023). Expert input can constrain the search space and improve the learned root cause graph by identifying spurious or misleading causal inferences.

This use case investigates the challenges of scaling CBN for RCA in electric vehicle manufacturing with the help of expert feedback. Specifically, the challenge lies in modelling process expert knowledge in a structured and scalable manner to enable intuitive expert interaction with the learned root cause graph.

And how can the interaction in turn refine link predictions and align them with user expectations through an explanation-driven feedback loop.

To address these challenges, this work integrates a KG with a CBN. It proposes a hybrid approach where the KG serves as a structured representation of the manufacturing process, able to capture expert knowledge, in which the CBN predicts missing cause-effect relations. The approach enables:

- The CBN learns to predict cause-effect relationships as links between variables in the manufacturing KG. It makes use of prior domain knowledge to constrain the search space. This enables learning on large-scale KGs.

- The manufacturing KG provides an interface where domain experts can inject feedback to refine causal inferences by explaining where and how the model should improve.

- Expert annotations on explanations are used to iteratively refine the link prediction process in the KG.

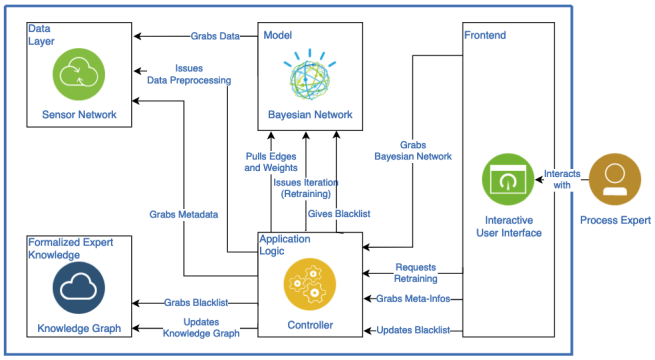


Figure 17: The system architecture of the interactive root cause analysis tool RootFinder (Wehner et al., 2023).

Figure 17 illustrates the proposed interactive RCA system. It is called RootFinder. The system combines link prediction with an expert-driven feedback loop. That way, the learned causal relationships fit the sensor data and align with expert domain-specific constraints.

Relevance to Research Question 2. The RCA use case presents a scenario for studying expert feedback modalities for model alignment. The feedback modality in the use case is naive, meaning it is a simple annotation of explanations. Still, the user interactions focus on the mechanisms that underlie the inferred cause-effect relationships. The research objectives in this use case are to analyse how users perceive the naive feedback modality and how it impacts the learned model. Through this use case, an empirical foundation is established for discussing feedback modalities in explanation-driven model alignment, specifically for link prediction tasks. Thus, the insights gained from expert interactions with the RCA system will inform a broader discussion on feedback modalities in human-in-the-loop KGC.

4.1.1 System Overview

The system for interactive RCA consists of five components (see Figure 17), each implemented as an independent microservice. These components work together to enable the iterative alignment of the Bayesian Network for KGC with expert intent through explanations.

At the system's core, a **Controller** coordinates all communication between the components. It serves as the central processing unit, receiving requests, triggering computations, and ensuring that results are delivered between the different modules.

The **Data Layer** provides access to sensor data collected from a vehicle manufacturing line. The sensor data undergoes preprocessing. Preprocessing ensures that the raw sensor measurements are structured to enable learning with the CBN. The transformed data is then stored in the data layer. This allows fast retrieval and usage by other components. Details on this preprocessing pipeline are further elaborated in Subsection 4.1.6.

The **Knowledge Graph** is key to structuring expert knowledge about the electric vehicle manufacturing process. It encodes relationships between stations, process steps, and sensor variables. Additionally, it captures domain knowledge that constrains the link prediction process and finally models the predicted CERs. The structured representation within the KG enables automated extraction of constraints that define which sensor variables are likely to influence others. It thereby improves the efficiency of causal discovery. Further details on the construction and functionality of the KG are discussed in Subsection 4.1.2.

The **Causal Bayesian Network** is the ML model responsible for learning and predicting CERs between the sensor variables of the KG. It takes the preprocessed sensor data from the data layer as input to infer directed causal dependencies between variables, which are then added as links to the KG. The KG helps in this process by drastically reducing the search space. This ensures that causal learning remains computationally feasible even in large-scale manufacturing settings. The resulting output is a learned root cause graph, which holds the inferred causal relationships between sensor variables. This process is described in more detail in Subsection 4.1.4.

To enable expert interaction, the **Interactive User Interface** provides a visual representation of the root cause graph. Experts can explore the inferred causal pathways and validate the explanations provided by the model. Additionally, the interface enables domain experts to inject feedback directly into the system. Experts can modify causal links, add new relationships to the KG, or assign specific sensor variable types. By iterating on this process, the model continuously improves its alignment with expert feedback. The functionalities of the user interface are discussed in Subsection 4.1.5.

This architecture enables a dynamic, feedback-driven approach to RCA where expert knowledge is iteratively integrated into the model through explanation-based interaction. The system engages domain experts in refining the reasoning process itself. This ensures that inferred causal dependencies align with statistical evidence and expert knowledge. The following section describes the KG of the manufacturing process in detail.

4.1.2 The Manufacturing Knowledge Graph

The KG of the RCA tool formalizes expert knowledge of the electric vehicle manufacturing process. It structures domain knowledge in a machine-readable format and enables the automatic, efficient, and large-scale integration of expert knowledge into the learning process of the CBN. This structured representation ensures that the KGC is data-driven and also incorporates human expertise to refine and constrain the search for cause-effect relationships.

The expert knowledge of the manufacturing process is modelled as depicted in Figure 18.

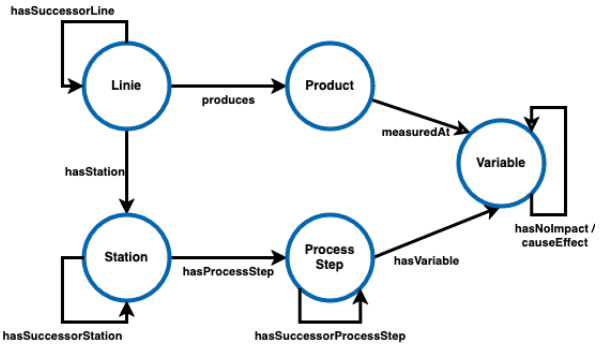


Figure 18: Schema of the RootFinder manufacturing KG (Wehner et al., 2023).

The KG represents the manufacturing process of electric vehicles as a structured sequence of production components (see Figure 18). The production is organized into lines, each consisting of multiple stations. The stations themselves are also modelled as a sequence. This ensures that the temporal flow of the manufacturing process is represented in the KG. Each station executes multiple process steps, and the KG captures the sequential dependencies between these steps with the `hasSuccessorProcessStep` relation. Within each process step, sensor data is collected as sensor variables, which provide quantitative measurements of the production process. Finally, a line produces products, and variables are measured for a product.

Central to the KG is the `causeEffect` and `hasNoImpact` relation between sensor variables. The `causeEffect` relation models the learned CER relations between two sensor variables. The `hasNoImpact` relation is used to constrain link prediction by marking sensor variables that are known to have no causal influence on each other. The evaluation will show that integrating this domain knowledge significantly reduces the search space for learning the CBN. This makes link prediction more efficient and less prone to spurious correlations (Wehner et al., 2023).

Additionally, each sensor variable is classified into one of three subclasses:

- `Root Variables`: Sensor variables that are at the start of a CER chain in a manufacturing process.
- `Leaf Variables`: Sensor variables that are at the end of a CER chain in a manufacturing process.
- `Irrelevant Variables`: Sensor variables that experts agree on to have no meaningful impact on fault propagation.

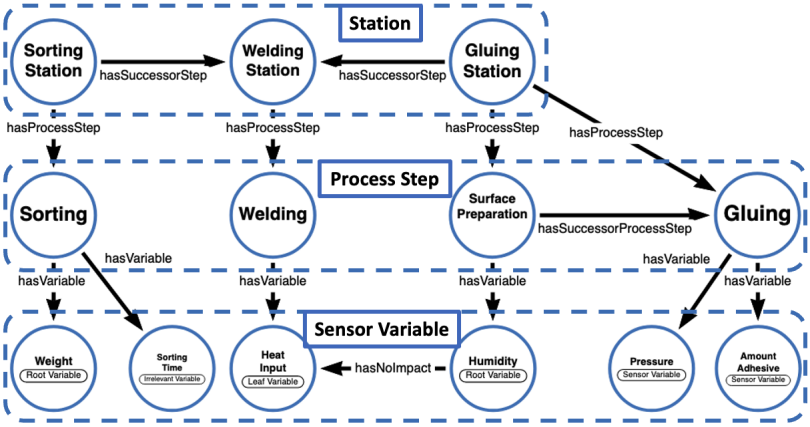


Figure 19: Example of the RootFinder manufacturing KG (Wehner et al., 2023).

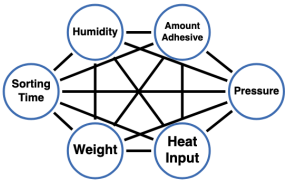


Figure 20: The possible CERs between the sensor variables of the manufacturing KG (see Figure 19), disregarding the structural topology of the manufacturing process (Wehner et al., 2023).

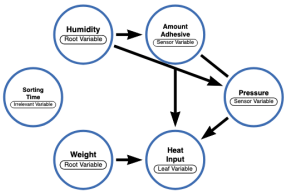


Figure 21: The possible CERs between the sensor variables in the manufacturing KG (see Figure 19), considering the structural topology of the manufacturing process (Wehner et al., 2023).

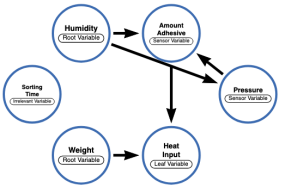


Figure 22: The true root cause graph among the sensor variables, where its edges are a subset of the edges depicted in the potential CER graph in Figure 21 (Wehner et al., 2023).

The structured representation of manufacturing knowledge in the KG enables automated reasoning over the production process. The system leverages this knowledge from two dimensions for reasoning.

First, during the learning process of the CBN, the KG provides structural constraints that help prune the search space. This ensures that causal relationships inferred by the model align with established domain knowledge, prevents the model from learning spurious correlations, and improves the interpretability of the resulting root cause graph.

Second, the KG serves as a structured interface for expert feedback. The KG acts as a continuously evolving representation of manufacturing knowledge. It enables experts to modify relationships, refine classifications, and correct inferred links. Experts can iteratively inject corrections into the system. The corrections are then incorporated into subsequent learning iterations of the CBN. This aligns its inferences closely with user knowledge.

This way, the interactive RCA tool ensures that RCA is data-driven, explainable and continuously refined based on expert feedback.

4.1.3 Temporal-Spatial Relations and Variable Subclass Properties

Temporal-Spatial Relations Between Variables. As discussed above, the KG constructs a hierarchical topology of the manufacturing process by modelling the line as a sequence of stations, each consisting of a sequence of process steps. At the lowest level of this hierarchy, sensor variables are measured in real-time. It provides a structured representation of how data is captured throughout the manufacturing process. And this structured representation, in turn, enables automatic reasoning to derive a partial ordering over the sensor variables. The partial orderings support the CBN in learning CERs that adhere to known temporal and spatial constraints.

The KG models process dependencies through the `hasSuccessorProcess Step` relation, which defines the sequential execution order of process steps. Since sensor variables are measured within process steps, the temporal structure of the manufacturing process imposes a constraint on causal inference. A sensor variable measured in a later process step cannot causally impact a sensor variable measured in an earlier process step.

Formally, for any two process steps x_1 and x_2 and for any two sensor variables y_1 and y_2:

$$\forall x_1 \, \forall x_2 \, \forall y_1 \, \forall y_2$$

$$\Big(\texttt{ProcessStep}(x_1) \land \texttt{ProcessStep}(x_2) \land$$

$$\texttt{Variable}(y_1) \land \texttt{Variable}(y_2) \land$$

$$\texttt{hasSuccessorProcessStep}(x_1, x_2) \land \texttt{hasVariable}(x_1, y_1) \land \texttt{hasVariable}(x_2, y_2)$$

$$\rightarrow \texttt{hasNoImpact}(y_2, y_1) \Big)$$

This rule ensures that any variable measured in a later process steps cannot causally impact a variable from an earlier step. It enforces the temporal structure of the manufacturing process. By integrating this constraint, the CBN is prevented from learning spurious CER that contradict known process dependencies.

Consider the KG depicted in Figure 19. The structured topology of the manufacturing process allows for the inference that sensor variables such as `Weigh` and `Sorting Time` may causally impact the sensor variable `Heat Input` due to their position in earlier process steps. However, `Heat Input` cannot influence earlier variables, and it does not have any impact on `Humidity`, `Amount Adhesive`, or `Pressure`, as they are constrained by their respective spatial-temporal dependencies.

The formalized reasoning over manufacturing process topology significantly reduces the number of candidate CER that must be considered while learning the CBN. Thus, a substantial number of CER are excluded a prior. This allows the learning process to focus on plausible causal interactions. The approach results in a drastic reduction of the search space compared to naive methods that attempt to infer causal dependencies solely from tabular sensor data (see Figures 20 and 21).

Properties of Variable Subclasses. The KG further models expert knowledge by classifying sensor variables into subclasses: root, leaf, and irrelevant variables. These subclass properties, introduced through expert feedback, serve as additional constraints that reduce the search space of the CBN.

Root variables represent external influences that cannot be impacted by any other sensor variables, but they may impact downstream variables:

$$\forall x \, \forall y \left(\texttt{RootVariable}(x) \land \texttt{Variable}(y) \rightarrow \texttt{hasNoImpact}(y, x) \right)$$

For example, in Figure 19, the sensor variable `Humidity` is classified as a root variable since air humidity is an external factor independent of the manufacturing process. Thus, it is not impacted by any other variable in the process. However, `Humidity` may impact process-relevant variables, such as those involved in material adhesion.

Leaf variables function as the inverse of `RootVariables`. Other sensor variables may influence them, but cannot impact any downstream variables:

$$\forall x \, \forall y \left(\texttt{LeafVariable}(x) \land \texttt{Variable}(y) \rightarrow \texttt{hasNoImpact}(x, y) \right)$$

In Figure 19, the `Variable Heat Input` is an example of a `LeafVariable`, as it is measured in the final process step and does not influence any subsequent measurements.

Irrelevant variables do not participate in meaningful causal interactions within the manufacturing process:

$$\forall x \, \forall y \left(\texttt{IrrelevantVariable}(x) \land \texttt{Variable}(y) \right.$$
$$\left. \rightarrow \texttt{hasNoImpact}(x, y) \land \texttt{hasNoImpact}(y, x) \right)$$

These variables are often part of data from the manufacturing process as artifacts of unclear requirements, highly specific use cases, or regulatory constraints. For instance, the `Variable Sorting Time` in Figure 19 is an example of an `IrrelevantVariable`, as it does not impact any other process-related sensor measurements. Such variables are excluded from RCA to prevent spurious causal inferences.

The formalization of variable subclass properties within the KG help in aligning the CBN with expert knowledge. The CBN uses these properties while learning. It ensures that inferred CERs reflect domain knowledge. Furthermore, assigning subclasses to variables provides a structured mechanism for expert feedback. It allowes iterative refinement of the model's reasoning. Experts can modify variable subclasses, add missing links, or adjust incorrect inferences. This is the basis for a continuous improvement loop that aligns the learned root-cause graph with manufacturing knowledge.

Figures 20 and 21 illustrate how incorporating temporal-spatial reasoning and subclass constraints for sensor variables can narrow down the search space of the CBN learning algorithm. In Figure 20, every fault detected in any sensor variable is potentially attributable to any other. This results in 30 possible cause-effect relationships among the six sensor variables, since the sensor network's topology, as defined in Figure 19, is not taken into account. However, when the sensor network topology is considered, as shown in Figure 21, the number of potential CER is reduced to seven. This example illustrates that automated reasoning based on the expert knowledge can significantly prune the search space for the CBN learning algorithm.

4.1.4 Causal Bayesian Network

Next, the concept of CBNs is introduced, and it is described how they are used for link prediction of CERs between sensor variables. A CBN is a probabilistic model that learns dependencies among variables such that a change in one variable alters the value of another while all remaining variables are held constant (Kertel et al., 2023). In this use case, the CBN is used to predict links that represent the causal influence between two sensor variables. The model is trained on sensor data collected from the manufacturing process and integrates expert knowledge from the manufacturing KG (Wehner et al., 2023).

CERs capture the direct impact that one variable has on another. For example, consider the true causal graph shown in Figure 22. A change in `Pressure` influences `Heat Input` through the path `Pressure` → `Amount Adhesive` → `Heat Input`. If all sensor variables except `Heat Input` are fixed after an increase in `Amount Adhesive`, then the influence of `Pressure` on `Heat Input` is blocked. Such direct effects define the causal order among the sensor variables. In the running example, one possible causal order is: `Sorting Time`, `Weight`, `Humidity`, `Pressure`, `Amount Adhesive`, `Heat Input` (Wehner et al., 2023).

CBNs derive the unknown root cause graph from sensor data. Traditional approaches to causal graph derivation involve design of experiments, which is labor-intensive and expensive. In contrast, the present method leverages the manufacturing lines‘ sensor network for data. However, relying exclusively on sensor data introduces challenges:

1. **Confounding:** A correlation between two sensor variables is due to a third variable that influences both.
2. **Causal Order:** Pure sensor data does not tell whether a variable is the cause or the effect.
3. **Complexity:** The number of possible causal graphs increases rapidly with the number of sensor variables.

The KG, with its domain knowledge, mitigates these issues. The integration of the KG into the training process of the CBN constrains the search space by incorporating domain knowledge. This reduces cofounder, the complexity and provides the causal ordering.

Learning the CBN and Incorporating Expert Feedback. Expert feedback is used to constrain the learning of the CBN for link prediction of CERs between sensor variables. The KG encodes the sensor networks‘ topology, which establishes a partial ordering on the variables. As discussed before, this partial order limits the candidate causal graphs by allowing only edges that follow the established order.

The CERs are learned with the help of Structural Equation Models (SEMs) (Ullman and Bentler, 2012). In the SEM framework, each variable x is modeled as a function of its causal parents $\mathrm{PA}(x)$ and an independent noise term ϵ_x:

$$x = f_x\big(\mathrm{PA}(x), \epsilon_x\big).$$

For example, the `Heat Input` variable may be modeled as

$$\texttt{Heat Input} = f\big(\texttt{Weight}, \texttt{Amount Adhesive}, \epsilon_{\texttt{Heat Input}}\big).$$

This model captures the transformation of input variables through the process while accounting for external influences through the noise term.

Additionally, the identification of the causal graph relies on the assumption that the data follows a Causal Additive Model (CAM) (Bühlmann et al., 2014). This specifies the SEM framework, such that each sensor variable is modeled as an additive function of its causes plus an independent noise term. For example, the `Heat Input` variable is now modeled as

$$\texttt{Heat Input} = f_1(\texttt{Humidity}) + f_2(\texttt{Amount Adhesive}) + \epsilon_{\texttt{Heat Input}},$$

where the noise is normally distributed and f_1 and f_2 are non-linear and learned functions. If an f_1 and f_2 can be found such that $\epsilon_{\texttt{Heat Input}}$ is under a given threshold, a CER from `Humidity` and `Amount Adhesive` to `Heat Input` is diagnosed (Kertel et al., 2023). By checking each possible combination of variables, the true root cause graph can be recovered from observed data. However, the possible search space for combinations of variables is vast (Wehner et al., 2023).

Therefore, before learning the CAM, a causal ordering must be determined. First, a partial ordering from the KG is mined. The expert feedback from the KG supports this by:

- Providing a partial ordering of `Variables` based on the manufacturing process's topology.
- Excluding specific CERs that are implausible.
- Limiting the possible positions of variables in the causal order through subclass assignments.

For the variables where the KG provides a complete ordering. The CAM can be immediately estimated. For the other variables, the ordering is approximated.

Assume that we have n observations of m sensor variables, represented as $(x_{l1}, \ldots, x_{lm})$ for $1 \leq l \leq n$. To derive the causal ordering, one searches for a permutation π on $\{1, \ldots, m\}$ that minimizes the score

$$Z(\pi) := \sum_{k=1}^{m} \log \left[\sum_{l=1}^{n} \left(x_{lk} - \sum_{\pi(j)<\pi(k)} \widehat{f_{kj}^{\pi}}(x_{lj}) \right)^2 \right],$$

where the functions $\widehat{f_{kj}^{\pi}}$ are estimated by maximum likelihood (Kertel et al., 2023). For the resulting orderings, CAMs are estimated. If the noise term ϵ is under a certain threshold, an CER is diagnosed. This results in the final root cause graph and, thus, in the manufacturing KG to complete it.

In this way, expert feedback is integrated into CBN learning. It reduces the number of candidate graphs and improves link-prediction accuracy for CER, as will be shown later.

4.1.5 Interactive User Interface

The Interactive User Interface displays the learned CBN (see Figure 23). The interface is designed to enable process experts to inspect and adjust the root cause graph without writing any code.

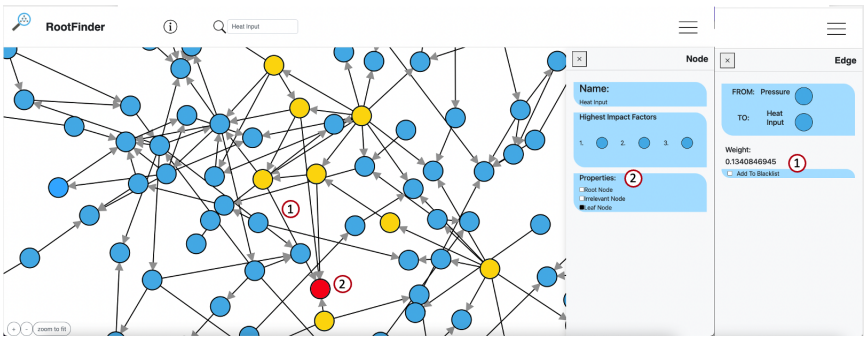


Figure 23: The user interface of the interactive and intelligent root cause analysis tool. The fault is marked in red, sensor variables that are part of the learned root cause path are yellow, and the numbers in red circles denote interactive elements (Wehner et al., 2023).

Root Cause Analysis Workflow. A process expert initiates the analysis by searching for and locating the sensor variable with the diagnosed fault. All candidate root cause paths associated with that sensor variable are then highlighted. In this manner, the expert is provided with information on which other sensor variables influence the faulty variable. This directed visualization allows the expert to efficiently inspect the relevant process steps in the real-world manufacturing process. It thereby reduces the time required to investigate and resolve the fault (Wehner et al., 2023).

Expert Feedback Integration. The user interface provides several ways to interact with the root cause graph. For instance, the process expert may select a specific product under analysis. Since sensor variable measurements are product-dependent, this selection refines the root cause graph. Additionally, a specific RCA timeframe can be set so that the analysis is focused on the period when the fault occurred.

When a spurious CER is detected, the expert wants to intervene. Suppose the CBN infers a CER between sensor variables x_i and x_j that is inconsistent with domain knowledge. The expert can select the corresponding edge and mark it by choosing the option "Add to Blacklist" (see Figure 23 1). This operation is formalized by adding a `hasNoImpact` relation to the manufacturing KG:

$$(x_i \texttt{ hasNoImpact } x_j) \cup KG$$

The feedback is incorporated into subsequent iterations of learning the CBN. It updates and improves the CBN with domain knowledge.

Furthermore, the subclass assignments of sensor variables (e.g., root, leaf, and irrelevant variable) can be modified via the interface (see Figure 23 2). For example, if it is observed that the classification of a variable does not align with its causal role, the expert can adjust its subclass membership. Such adjustments further constrain the learning process by reducing the number of feasible causal orderings and candidate edges.

Through iterative feedback, the root cause graph is refined gradually. The process expert's corrections are incorporated into each subsequent learning it-

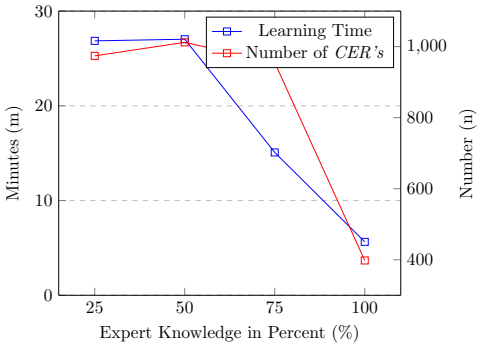


Figure 24: Decrease of learning time and number of CER with an increase in expert knowledge (Wehner et al., 2023).

eration, which drives the inferred causal structure closer to the true underlying causal graph.

4.1.6 Evaluation

The evaluation uses data from a real-world electric vehicle manufacturing process. One production line is selected as a case study to evaluate the functionality of the interactive and intelligent RCA tool and to gather input from users on the feedback modalities.

Data. Two data sources, the KG and the data layer, are used in this evaluation. The KG is implemented in Neo4J (Lal, 2015), which enables the retrieval of expert knowledge through Cypher (Francis et al., 2018), a query language designed for deductive reasoning. This large-scale KG contains 100,015 nodes and 417,944 relationships. The evaluated production line consists of 15 products, 53 stations, 96 process steps, 1683 sensor variables, and 2143 relationships connecting them.

Data for the CBN is prepared from the data layer by first linking sensor measurements to individual output products. The preprocessing steps involve removing columns with a single unique value and eliminating one column from each pair of columns with a correlation above 0.95 to reduce collinearity. Additionally, columns and rows where more than 50% of the values are missing are discarded. Missing values in the remaining data are imputed using the column mean. This preprocessing approach is designed to be generic and is adaptable to various product types and time periods. For this evaluation, the focus is on one day's data for a specific product type (Wehner et al., 2023).

Experiments. The aim of the experiments is to provide a real-world demonstration of how expert knowledge, modelled in a large-scale KG, can be used to improve the performance of CBNs for RCA in manufacturing. Three experiments are presented to illustrate the benefits of integrating expert knowledge into the learning process.

Reduction in Causal Bayesian Network Learning Time. The inclusion of expert knowledge reduces the training time of the CBN. In the ex-

periment, five runs were conducted in which 25%, 50%, 75%, and 100% of the information available from the KG were randomly selected and used to train the CBN. The results, as shown in Figure 24, indicate a decrease in learning time as the percentage of expert knowledge increases. Specifically, a decline in training time of approximately 79.0% was observed when moving from lower to higher proportions of expert knowledge. This reduction in training time shows that the integration of expert knowledge optimizes the learning process in large-scale manufacturing settings. It unlocks previously prohibitively extensive manufacturing processes for data-driven RCA.

Reduction of Spurious CERs. In addition to the reduction in training time, the use of expert knowledge reduces the number of spurious CERs learned by the CBN. The same experimental setup as described in the previous paragraph was employed. As more expert knowledge was incorporated, a decrease in the number of inferred CERs was observed, with a pruning of approximately 60.6% of spurious relationships. This reduction leads to a more accurate and reliable root cause graph, thereby enhancing the overall performance of the RCA tool.

Impact of Expert Feedback on Causal Bayesian Network Performance. To assess the contribution of expert feedback, the performance of the CBN was additionally evaluated using the adapted Structural Hamming Distance (aSHD) (Bookstein et al., 2002; Kertel et al., 2023) metric. This metric quantifies the deviation of the learned CBN from a partially known root cause graph. A lower aSHD value indicates a closer alignment. In this experiment, RootFinder was applied to 100 random samples, each containing eight sensor variables arranged in a temporal-spatial order. Initially, the mean aSHD was 0.47 with a standard deviation of 0.74. After an expert added a single `hasNoImpact` relation to the KG, the mean aSHD decreased to 0.44, and the standard deviation declined to 0.70. This improvement of nearly 6% in the aSHD and the reduction in variability indicate that the inclusion of expert feedback improves the accuracy of the CBN and also stabilizes its learning process. These results exemplify one iteration of the interactive RCA tool, where expert interventions drive the model closer to the true causal graph.

Qualitative Expert Interviews. In addition to the quantitative evaluation, four process experts at a German car manufacturer were asked to test the tool and provide qualitative feedback. Each interview was structured similarly. Initially, the experts were introduced to the interface and allowed to explore the tool freely while asking clarifying questions. Next, they were tasked with identifying the root cause of a fault in a specific variable like (`css_welding_status`) within a given timeframe, and were encouraged to add expert feedback directly to the displayed root cause graph while verbalizing their observations. After completing the analysis, they were asked whether they would use the tool in practice and to comment on the feedback mechanism.

The feedback was generally positive. One expert commented, that there is no tool like this available at their workplace. The experts quickly understood the interface's minimalist design. However, some issues were also noted. It was mentioned that the time frame represented in the root cause graph was not immediately clear. Additionally, several experts found the root cause graph overly complex due to its high number of nodes and edges. This makes it challenging to use for a free CERs exploration. Despite this, the search function enabled each expert to quickly locate the target variable and its corresponding

root cause path. The participants pointed out that the root cause path seemed plausible and could be easily translated into an actionable inspection of the corresponding production process. During the session, one expert identified a variable that should have been classified as a leaf variable and updated this information via the interface.

Furthermore, in terms of feedback modality, the process experts noted that since each edit affects at max two nodes, the scalability of the feedback process is limited, especially in production graphs containing thousands of nodes. Finally, the experts also pointed out that their limited familiarity with all variables in the production process makes it challenging to specify hard constraints for each individual variable. One expert proposed a change log and crowd-based validation mechanism to mitigate the impact of false annotations.

Overall, the experts agreed that the tool is practical for RCA, as it accelerates the investigation process compared to traditional, individual, script-based analyses. They suggested that a more concise depiction of the root cause graph, showing only nodes relevant to the fault along with their associated station information, would enhance usability. Additionally, a more scalable and fault-tolerant feedback mechanism was pointed out to be beneficial for future interactive RCA tools. A major concern was the long retraining time (15–30 minutes) required after expert feedback was added to the graph, which was viewed as a bottleneck. An incremental approach to model updates could help, such that the impact of expert feedback can be observed more quickly.

4.2 Discussion on Feedback Modalities for Downstream Model Alignment

The evaluation of RootFinder points to two criteria for effective feedback modalities in the context of KGC model alignment. These are scalability and the ability to deal with limited domain knowledge. In practice, the feedback mechanisms must support efficient input from users even when detailed domain knowledge is unavailable, and must scale well to a large number of instances, to minimize the required feedback.

This motivates the need to evaluate different feedback modalities that can be employed to collect user knowledge. Let us recall Research Question 2:

> *How effective are different types of feedback modalities in collecting user knowledge for the downstream task of aligning a link prediction model with human intent?*

The feedback modalities discussed are positive examples, negative examples, rules, ratings, and preferences. Each modality varies in complexity, specificity, and practical feasibility. A **positive example** provides correct instances supported by explanatory subgraphs, whereas a **negative example** provides incorrect instances paired with a subgraph that provides a justification for their incorrectness. The **rule-based** feedback modality works with logical constraints intended to generalize over numerous instances simultaneously. Meanwhile, the **rating** feedback modality assigns numerical plausibility scores. Lastly, the **preference** feedback modality requires users to indicate comparative judgments between pairs of instances with explanation.

Throughout the following discussion, the analysis of each feedback modality is guided by the previously identified alignment risks (see Section 2.4). Specifically, the modalities are assessed based on their susceptibility to objective misalignment, biases inherent in human judgment, feedback quality, and risks associated with feedback loops.

4.2.1 Positive Example as Feedback Modality

Positive examples as a feedback modality are true, previously observed or unobserved triples. They are accompanied by explanations, in the form of subgraphs that demonstrate why the triples are true.

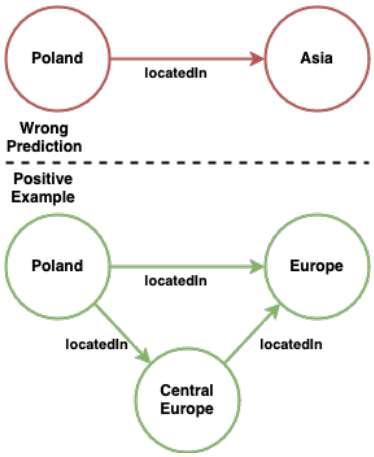


Figure 25: The incorrect prediction that `Poland` is in `Asia` prompted a user to correct it with a positive example. The user notes that `Poland` is located in `Europe`, specifically in `Central Europe`, which is part of the continent `Europe`. The positive feedback is used in a later step to align the KGC model's predictive behaviour.

Example. Consider the task of predicting the continent of a country in a KG of countries, regions, and continents (see Figure 25). A positive example as feedback for KGC model alignment provides the model with a training triple and an ideal explanation. It is triggered by an incorrect prediction observed by the user. In this case, the model wrongfully predicted, that `Poland` is located in `Asia`. It triggers the user feedback with the positive example of `Poland` being located in `Europe`. The explanation subgraph provided as feedback supports that `Poland` is located in `Europe` by pointing out that `Poland` is located in `Central Europe` and that `Central Europe` is located in `Europe`.

Postive examples for model alignment are useful in two ways. First, it potentially augments the training set with a new instance. This enhances model performance through increased data volume. Second, the explanation subgraph serves as the ideal decision-making process that the model is expected to learn during training.

Discussion. At first glance, providing a positive example appears to be an efficient approach for aligning KGC models with human feedback. Nevertheless,

Table 13: Alignment risk assessment of the positive example feedback modality.

Alignment Risk	Risk Level
Objective misalignment	Low
Human bias	High
Quality of human input	High
Feedback loops	High

in complex domains with a high number of relation types, this approach requires a thorough understanding of the domain, which limits its applicability. The challenge becomes more apparent when the user is asked to generate explanation subgraphs, as the number of potential subgraphs increases combinatorially in a graph with a high node degree. Consequently, the search for an appropriate explanation subgraph becomes prohibitively expensive for the user. Moreover, a single positive example is not sufficient for model alignment. Complex link prediction tasks require hundreds or thousands of examples before measurable alignment is achieved. This limits the scalability of positive examples.

The following reiterates the feedback modality on the alignment risks described in Section 2.4. The positive example modality is useful for reducing objective misalignment because its precise specification by the explanation leaves little room for the model to deviate from the target. The model must optimize towards the supplied true fact, with the help of the given explanation. Nonetheless, objective misalignment remains possible if the true explanation cannot be fully expressed by the user due to limitations in the KG's vocabulary. This leads to under- or misspecified explanation subgraphs.

Furthermore, this modality is vulnerable to bias in human judgment. Since the provision of positive examples is entirely controlled by the user, any bias in their feedback will be directly transmitted to the model. Thus, the model will align with human biases.

Additionally, positive examples depend on high-quality human input, due to the specificity of positive examples. High complexity domain's will lead to low-quality human inputs. In such domains, user expertise is likely insufficient to create high-quality explanation subgraphs or to correctly identify missing predicted links.

Lastly, feedback loops pose a risk in complex KG domains. Users tend to provide feedback in areas where they have specialized knowledge (Moser, 2017). This leads to an overrepresentation of feedback in certain niches where the user feels comfortable. Selective feedback can create loops that distort the model's learning process.

Overall, the positive example feedback modality is not prone to objective misalignment. However, its implementation must carefully consider scalability, the quality of human input, and the potential for bias-induced feedback loops. A similar observation can be made for negative examples.

4.2.2 Negative Examples as Feedback Modality

Negative examples are generated by a human providing known false triples and corresponding explanation subgraphs that show inconsistent information, which

invalidates the triples. This feedback modality is typically triggered when the user observes a wrong prediction and then marks it as incorrect.

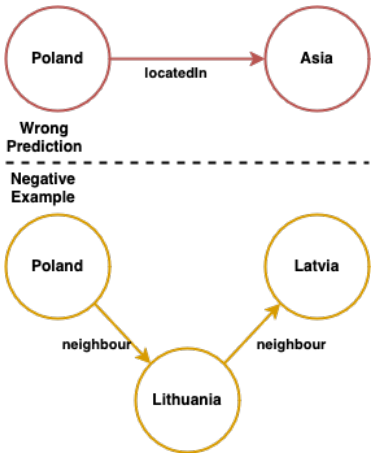


Figure 26: The incorrect prediction that `Poland` is in `Asia` prompted a user to mark it as a negative example. Additionally, the user provides the invalid explanation that `Poland` is neighbour of `Lithuania` and `Lithuania` is neighbour of `Latvia`. It demonstates an invalid reasoning pattern and reinforces why the incorrect triple should not be accepted. The negative feedback can be used in a later step to align the KGC model's predictive behaviour.

Example. For instance, in a KG of countries, regions, and continents, a negative example is provided by a false triple `Poland` is located in `Asia` (see Figure 26). An explanation subgraph clarifies that `Poland` is a neighbour of `Lithuania` and `Lithuania` is a neighbour of `Latvia`. The explanation is irrelevant to the predicted triple. It is noise that reinforces why the incorrect triple should not be accepted.

Table 14: Alignment risk assessment of the negative example feedback modality.

Alignment Risk	Risk Level
Objective misalignment	High
Human bias	High
Quality of human input	High
Feedback loops	High

Discussion. This modality offers a twofold benefit. On one hand, it supplies the model with a new negative training instance. This can be used for contrastive learning approaches that learn decision boundaries between correct and incorrect links. On the other hand, the explanation subgraph serves as a reasoning pattern for what constitutes a false inference, which can be incorporated during training to further penalize the model if this type of decision-making is encountered.

Despite these advantages, several challenges are inherent to the negative example modality (see Table 14 and Section 2.4). Firstly, the candidate space for potential true links remains extensive even after excluding a given negative triple, which allows for specification gaming during KGC model optimization. The provided explanation mitigates this risk by providing punishable incorrect behaviour. Objective misalignment still occurs if the domain vocabulary is insufficient to capture reasoning patterns. In such cases, the explanation subgraphs are under- or misspecified.

Moreover, the negative example modality is sensitive to biases in human judgment. Since users are responsible for providing false facts and explanations, any bias in their understanding or perspective is expressed in the feedback. This can lead the model to align with subjective biases.

The complexity of a KG domain further increases the issue, as insufficient domain expertise results in low-quality feedback that fails to produce adequate incorrect explanation subgraphs or triples.

Additionally, feedback loops are a potential risk in complex KG. Similar to the positive feedback modality, users tend to provide feedback in areas where they possess specialized knowledge (Moser, 2017), which may result in an overemphasis on certain aspects of the KG. This selective feedback can reinforce existing model biases and further distorts what the model learns.

Overall, negative examples are conceptually straightforward and can contribute to model alignment by reducing false positives. However, their practical application is limited and must address challenges related to scalability, bias, and the complexity of a KG‘ domain. The following paragraph discusses a more scalable form of feedback.

4.2.3 Rules as Feedback Modality

Rules provide a general form of feedback by defining constraints that apply within the KG domain. A rule is typically expressed as a logical statement. Users express these rules when encountering a prediction that violates a constrain within a domain they are familiar with. The rule is then used to align the KGC model.

$$\forall x \in X,\ \forall y \in Y,\ \forall z \in Z : \big(locatedIn(x,y) \wedge locatedIn(y,z)\big) \ \rightarrow\ locatedIn(x,z) \tag{2}$$

Example. For instance, in a KG of countries X, regions Y, and continents Z, a user supplies a rule stating that if a country is located in a region and the same region is located in a continent, then the country must also be located in the continent (see Equation 2). This rule applies to every country, every region, and every continent without the need to list each case individually.

Rules have a large impact because they generalize across many instances. A small number of rules can induce substantial behavioural changes in link prediction methods. However, formulating effective rules requires comprehensive domain knowledge. Furthermore, rules may not account for exceptions or edge cases that fall outside the general pattern.

Table 15: Alignment risk assessment of the rule-based feedback modality.

Alignment Risk	Risk Level
Objective misalignment	Low
Human bias	High
Quality of human input	High
Feedback loops	Low

Discussion. This feedback modality is susceptible to alignment risks (see Table 15). However, rules as feedback modality exhibit a low potential for objective misalignment. Rules are specifically formulated. They leave little room for specification gaming in strictly rule-based environments. For instance, in a KG modelling family relationships, the rule stating that "the father of my father is my grandfather" will always hold true. A model that aligns with this rule when predicting grandfather relationships has no space to generate misaligned behaviour against the expressed human intent. However, the situation changes if the vocabulary of the KG is restricted. For example, if the KG only includes a general `hasParent` relation instead of a specific `hasFather` relation, the rule would state that "a parent of my parent is my grandfather." This formulation leads to misinterpretation. It allows a model to traverse the `hasParent` relationship to a grandmother instead of a grandfather. The KGC model thus satisfies the rule but arrives at an incorrect prediction.

Additionally, rule-based feedback is vulnerable to bias in human judgment. The strictness of rules, combined with their capacity to generalize across many instances, results in the KGC model aligning closely with any bias in a rule expressed by a biased user.

Consequently, the effectiveness of rule-based feedback depends on the quality of the rules. The formulation of high-quality rules that provide appropriate behavioural guidance to the model requires comprehensive knowledge from the domain expert of the KG domain. If this expertise is lacking, there is the risk of including instances the domain expert did not intend to cover.

However, feedback loops are unlikely to occur because they are straightforward to detect. If a new rule overlaps with an existing rule, conflict resolution strategies can be applied. However, some KG subdomains may still have a higher feedback rule coverage due to the user's specialization in this particular subdomain (Moser, 2017).

Overall, rules serve as an effective feedback mechanism when the vocabulary of the KG is sufficiently specific to address the link prediction task and when the domain is well understood by the user. However, if these conditions are not met, the rule-based feedback modality has a high risk to undermine the alignment process by generalizing to unintent instances. The following describes a more robust feedback modality.

4.2.4 Ratings as Feedback Modality

In the rating feedback modality, a user is presented with a specific instance that includes the classification and a corresponding explanation. The user is then asked to provide a numerical score that reflects the plausibility of the instance. A high scores (e.g., close to 10) indicate high plausibility and low scores

(e.g., near 0) indicate low plausibility. The KGC model is then conditioned to score instances similar to a high-rated instance higher than instances similar to a low-rated instance. Collecting ratings is straightforward. Instances with explanations where the KGC model is uncertain how to classify are shown to a user for feedback. The ratings feedback modality requires less detailed domain expertise from the user compared to modalities where the user has to generate rules or examples. This modality is particularly useful in domains characterized by uncertainty, where the line between true and false facts is not sharply defined.

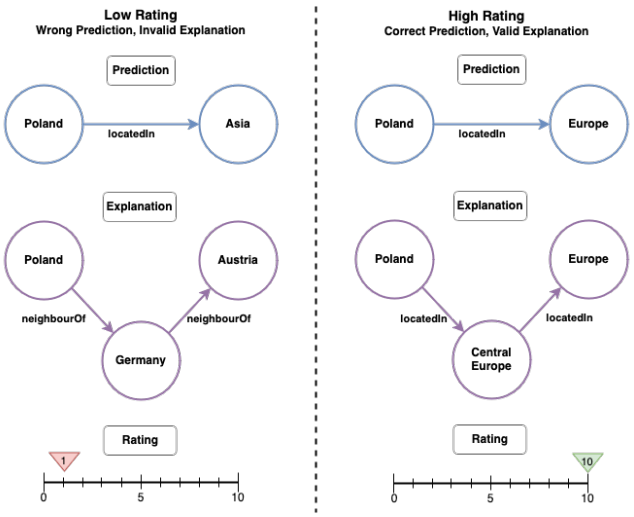


Figure 27: The KGC model proposes two candidate continent links for `Poland`. The incorrect link (`Poland-Asia`), supported by an irrelevant explanation subgraph (`Poland neighborOf Germany` and `Germany neighborOf Austria`), is rated 1/10 for low plausibility. In contrast, the correct link (`Poland`–`Europe`), justified via the valid explanation `Poland locatedIn Central Europe` and `Central Europe locatedIn Europe`, is rated 10/10. These ratings guide the model's ranking behaviour during alignment.

Example. Consider again the task of predicting the continent of a country in a KG of countries, regions, and continents (see Figure 27). The model proposes the candidate triple `(Poland, locatedIn, Asia)` and provides an explanation subgraph showing that `Poland` is a neighbour of `Germany` and `Germany` is a neighbour of `Autria`. This reasoning pattern does not support the prediction. The user assigns it a low plausibility score (e.g. 1/10). In contrast, the correct triple `(Poland, locatedIn, Europe)` with its valid explanation subgraph (via `Poland` → `Central Europe` → `Europe`) is rated highly (e.g. 10/10) by the user. The KGC model is than trained to score high–rated instances above low–rated ones. It reinforces the correct predictions and explanations without requiring the user to craft explicit rules or examples.

Discussion. The rating modality collects feedback for one triple at a time. This does not scale to large KGs. However, the risks (see Table 16) associated with objective misalignment are comparatively low with this modality, as the

Table 16: Alignment risk assessment of the rating feedback modality.

Alignment Risk	Risk Level
Objective misalignment	Low
Human bias	Low
Quality of human input	Low
Feedback loops	Low

numerical ranking provides a specification that the model must adhere to during training. Nonetheless, objective misalignment still occurs if the explanation provided is under- or misspecified due to limitations in the KG's vocabulary. In such cases, the model is forced to optimize based on an incomplete or inaccurate reasoning pattern.

Furthermore, the rating modality is less prone to capturing user biases. Although subjective biases can influence ratings, the ordinal nature of numerical scores allows for finer distinctions. It offers a degree of freedom that is not available with binary truth or strict rule-based feedback. As a consequence, even if an instance is rated high despite reflecting an undesirable bias, there remains the freedom for the model to learn a higher rank for the truly correct instance.

In a similar manner, the quality of feedback is less significant because it influences the ranking rather than the precise truth values. This means that an incorrect instance can be wrongfully ranked high. Meanwhile, the model still has the freedom to mitigate this error by ranking the actual correct answer higher based on dominant patterns in the rest of the collected feedback.

The rating modality also reduces the likelihood of feedback loops. The rating modality requires a selection mechanism to identify representative instances from a training set. Without such a mechanism, gathering ratings for all possible instances becomes infeasible. This heuristic-driven selection process, limits the user's ability to steer where feedback is given and allows the model to decide where to provide feedback based on where it is most needed. It thereby reduces the likelihood of feedback loops.

In summary, the rating modality is a robust feedback modality with a low risk of objective misalignment and individual biases. Its ordinal nature provides flexibility in uncertain domains. However, the modality has challenges related to instance selection and scalability. The following section presents another ordinal feedback modality, which is more scalable, as it gathers feedback for two instances simultaneously.

4.2.5 Preferences as Feedback Modality

For the preference-based feedback modality, the user is asked to compare two instances and their associated explanations to indicate which one more closely aligns with their understanding of correctness. This comparative judgment is expressed using preferences such as $x_1 \succ x_2$, indicating a preference for x_1 over x_2, $x_1 \sim x_2$, indicating indifference between x_1 and x_2, or $x_1 \prec x_2$, indicating a preference for x_2 over x_1. It allows users to provide relative judgments and reduces the need for detailed domain-specific knowledge, as tacit understanding is generally sufficient to judge which prediction is more valid in a given domain. The final ranking is then approximated by the KGC model while aligning with

the expressed preferences. Also, this modality is targeted towards uncertain domains, where the line between true and false facts is not sharply defined.

The number of possible instance pairs on which the user could give feedback increases quadratically with the number of instances. This results in a large space for potential comparisons, which leads to scalability issues. To prevent scalability issues, the preference-based feedback modality requires sampling of instances for feedback collection. The sampling utilizes heuristics to choose those pairs for which the KGC model exhibits high uncertainty (Christiano et al., 2017). That way, feedback focuses on areas where the model is uncertain about how to classify correctly. The feedback automatically targets the KGC model's blind spots.

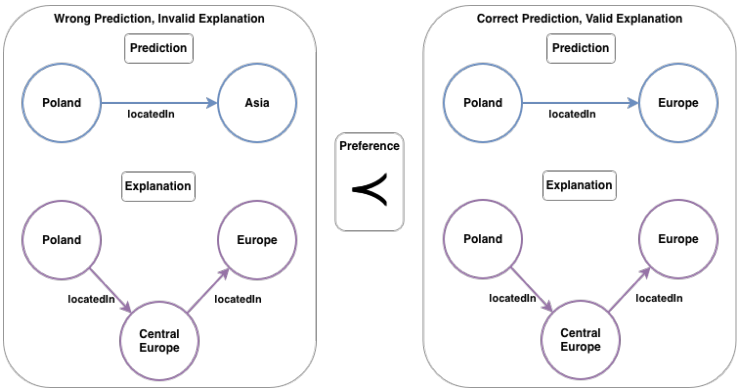


Figure 28: The KGC model proposes two candidate continent links for `Poland`. The incorrect link (`Poland locatedIn Asia`), supported by an irrelevant and invalid explanation subgraph (`Poland neighbourOf Germany` and `Germany neighbourOf Austria`), is presented alongside the correct link (`Poland locatedIn Europe`), justified via the relevant and valid explanation (`Poland locatedIn Central Europe` and `Central Europe locatedIn Europe`). The user expresses a preference $(\texttt{Poland}, \texttt{locatedIn}, \texttt{Europe}) \succ (\texttt{Poland}, \texttt{locatedIn}, \texttt{Asia})$. It incentivises the model to score the preferred and correct triple above the unpreferred and incorrect one.

Example. Consider once more the task of predicting the continent of a country in a KG of countries, regions, and continents (see Figure 28). The user is shown two candidate triples with their explanation subgraphs: `(Poland, locatedIn, Asia)` justified by `(Poland neighbourOf Germany)` and `(Germany neighbourOf Austria)`, and `(Poland, locatedIn, Europe)` supported by `(Poland locatedIn Central Europe)` and `(Central Europe locatedIn Europe)`. Because the latter candidate is valid, the user indicates a preference:

$$(Poland, locatedIn, Europe) \succ (Poland, locatedIn, Asia)$$

This comparative judgment guides the KGC model to score the valid triple (`Poland located in Central Europe`) and its explanation above the invalid one. It reinforces intended decision-making behaviour without requiring explicit numerical scores, example construction, or knowledge about domain-specific constraints.

Table 17: Alignment risk assessment of the preference feedback modality.

Alignment Risk	Risk Level
Objective misalignment	Medium
Human bias	Low
Quality of human input	Low
Feedback loops	Low

Discussion. Several alignment risks are associated with the preference feedback modality (see Table 17). First, the risk of objective misalignment is notable because the preference orderings provided by a user can be inconsistent. This leads to cycles (Von Neumann and Morgenstern, 2007). For example, it is possible that a user expresses a set of preferences where x_1 is preferred over x_2, x_2 is preferred over x_3, yet x_3 is preferred over x_1. Such cyclic preferences are exploited during optimisation by the KGC model. It results in unintended outcomes. Additionally, suppose the true explanations cannot be fully expressed due to vocabulary limitations within the KG. In that case, the model is forced to optimize based on incomplete or inaccurate reasoning patterns.

The feedback modality based on preferences is also subject to biases inherent in human judgments. Since users are responsible for the comparative assessments, their subjective biases are reflected in the feedback. However, because the pairing of instances is typically determined algorithmically rather than by the users themselves, the impact of biases is mitigated. Users have no governance over the feedback modality, except to express their preferences, which prevents them to focus their feedback on specific domains within the KG where they feel most comfortable.

Furthermore, the ordinal nature of the preference modality reduces the dependency on the absolute quality of human input, as an incorrect but preferred instance can be balanced against other preferred and correct instances. This reduces the impact of potential errors in feedback expression.

Finally, the potential for feedback loops is minimal with preference feedback because the selection of instance pairs is automated. This limits the control users have over where and when feedback is gathered. Nevertheless, cycles in the expressed preferences still occur, and such loops need to be managed to avoid reinforcing errors in the KGC model.

In summary, the preference feedback modality offers a robust, easy-to-use, and scalable method for aligning model predictions with user judgment by leveraging relative comparisons.

4.3 Summary and Answer to Research Question 2

Research Question 2 asked how effective different types of feedback modalities are in collecting user knowledge to align a link prediction model with human intent. To answer this, the chapter began with a concrete use case in electric-vehicle manufacturing. There, we integrated a causal Bayesian network with a KG, to observe how users interact via naive positive example feedback. From that case, two major criteria for feedback modalities were identified: scalability of feedback and minimal dependence on domain or model knowledge.

Building on these insights, the chapter systematically evaluated five feedback modalities (positive examples, negative examples, rules, ratings, and preferences) against four alignment risks (objective misalignment, human bias, feedback quality, and feedback loops):

- **Positive examples** offer precise corrections but demand heavy domain expertise and scale poorly. This makes them vulnerable to bias and looping.
- **Negative examples** likewise require detailed domain knowledge and suffer from specification and bias challenges.
- **Rules** generalize broadly with medium misalignment risk, because they depend on comprehensive vocabulary and expert rule quality.
- **Ratings** provide straightforward numerical feedback.This reduces bias and misalignment risk. However, it requires careful instance selection to remain scalable.
- **Preferences** enable users to compare pairs of predictions. This makes use of tacit knowledge with minimal specification effort. Automated pair selection limits bias and loops. However, cyclic preferences must be managed.

In summary, preference-based feedback provides the best balance. It scales through targeted uncertainty sampling, requires little model or domain knowledge, uses users' intuitive judgments, and has relatively low alignment risk. This conclusion directly motivates Research Question 3, which introduces LiEr, a human-in-the-loop link prediction framework that aligns model reasoning through preference-based feedback.

5 Iteratively Aligning Link Prediction with Human Intent

Research Question 3: How can user knowledge be incorporated into a link prediction model for iterative alignment with human intent?

With this research question, we investigate the feasibility and effectiveness of a link prediction model that incorporates user feedback on the model's behaviour to refine the model iteratively. We aim to determine whether such a model can align with human feedback while maintaining or improving its predictive performance.

This section builds on the findings from the first two research questions, which lead to the novel human-in-the-loop link prediction method LiEr.[5] First, let us recall that among the explanation modalities, the instance-based explanation offers a reliable fit for the downstream task of KGC model alignment. It is especially suitable to use for RCA with link prediction in manufacturing KGs, where faults propagate step by step through the production process. It forms fault-propagation paths through the KG that resemble instance-based explanations in the graph. Still, choosing the most relevant path is difficult. Many possible paths may exist, and the model must identify which steps truly explain the fault.

Second, the discussion of feedback modalities showed that preference-based feedback provides a balance between scalability and ease of use. Users can judge which of the two candidate paths better reflects their knowledge without having to write formal rules or hold extensive model knowledge to construct examples on their own. Preference feedback carries low alignment risk and leverages existing domain expertise effectively.

Regardless, one can think of many possible methods to align link prediction in KG with human feedback. Combinations could include fitting a supervised reward model from demonstration or using gradient-based fine-tuning on explanation scores (Mosqueira-Rey et al., 2023). One could also integrate rule-based constraints derived from user knowledge (Wehner et al., 2023). However, these approaches either demand extensive labelled examples, require experts to formalise rules, or risk misalignment. In contrast, LiEr brings together instance-based paths with preference feedback. It enables users to compare two candidate explanations at a time, without having to craft rules or provide demonstrations of positive or negative examples. This approach guides the KGC model towards plausible results and keeps alignment risk low. The proposed method balances between usability, data efficiency, and predictive performance.

Research Question 3 brings the previous findings together in LiEr[6] (**L**earning **i**nteractively by **E**xplanations to **r**eason). LiEr frames link prediction as an MDP on the KG. This allows a reinforcement agent to walk paths through the KG that are optimised to satisfy a reward learned from preference-based feedback over instance-based explanations. It steers the agent towards behaviour

[5] This section is adapted from the author's prior publications: Wehner et al. (2023), Schramm et al. (2023), and Wehner et al. (2025).

[6] It is commonly believed that Li Er is the birth name of the semi-legendary philosopher Laozi, the founder of Taoism (Seidel and von Falkenhausen, 2008)

aligned with human understanding of valid reasoning. This approach enables users to review candidate paths (e.g. explanations), to provide preferences, and to refine the policy iteratively. Finally, LiEr is validated on various benchmark datasets and a real-world RCA problem in the manufacturing line of electric vehicles.

This section introduces LiEr to answer research question 3. First, an alternative mathematical notion for tail link prediction is presented, which is required to formalise the optimisation of LiEr later. Next, it is shown how to translate a KG into an MDP. The architecture of LiEr's policy network is introduced, and it is described how the policy network stepwise solves the MDP. The policy network is trained to maximise a reward. The reward itself is an ensemble of learners trained to approximate a human's preference ordering, thus providing human-like rewards. LiEr is then evaluated on various benchmark datasets. Finally, the approach is applied to the use case of RCA in a manufacturing line. LiEr learns from feedback on explanations.

5.1 Aligning Path-based Link Prediction with Human Understanding of Valid Reasoning with LiEr

Path-based link prediction methods learn sequences of relations in a KG that reconstruct missing links between entities (Das et al., 2018; Wan et al., 2020). These paths serve as explanations, since each step contributes to a human-interpretable mapping from inputs to the predicted link (Bhowmik and de Melo, 2020). For instance, to predict that `Berlin` is in `Germany`, a path such as

$$\texttt{Berlin} \xrightarrow{\texttt{locatedInState}} \texttt{Brandenburg} \xrightarrow{\texttt{partOfCountry}} \texttt{Germany}$$

provides a sequence of facts whose composition justifies the conclusion. Formally, we say a reasoning path is **valid** if the truth of each reasoning step makes it impossible for the conclusion to be false (Copi, 1954). This means that each reasoning step is relevant to the conclusion. The model is correct for the correct reason. In a KG, each edge in the path is factually correct. The overall inference may still be **invalid** reasoning if one or more steps are semantically unrelated to the target relation (Copi, 1954). Invalid reasoning in KGs arises from spurious connections that happen to reconstruct the link in the data but are not reflected in human knowledge over the semantic concept underlying the predicted link. The model is in-/correct for the wrong reason.

Let us have a look at the example in Figure 29. A link prediction model predicts that `Hungary` is in `Europe`. It believes so because `Hungary` consumes pepper from `Europe`. The premise and the conclusion are true. However, pepper consumption does not guarantee that a country is located on a continent. A valid reasoning pattern instead connects `Hungary` to `Central Europe` and then to `Europe`. Each step is relevant to the conclusion.

When invalid paths appear frequently in the training and validation data, standard path-based KGC models learn these spurious patterns. This leads to a high accuracy on held-out splits. At the same time, they fail to generalise in practice (Marconato et al., 2023). This misalignment between model reasoning and human intent is a form of **Clever Hans bias** (Lapuschkin et al., 2019). Developers and users expect link predictors to capture their knowledge about the semantic concept underlying a relation (e.g. what it means for a country to

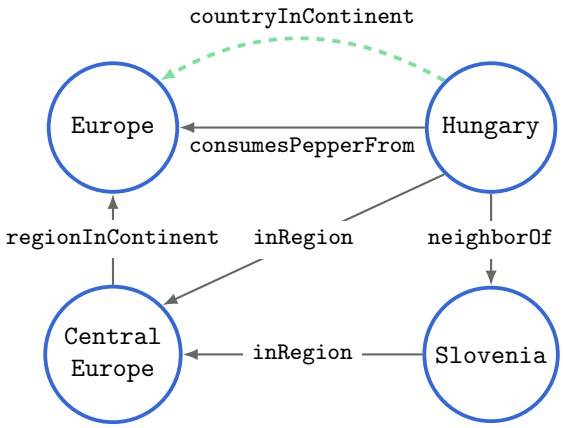


Figure 29: This Figure shows a minimal example of a KG with countries, regions, and continents (Wehner et al., 2025). A path-based link predictor infers the missing `countryInContinent` relation from `Hungary` to `Europe`. It does so by following a learn reasoning path through intermediate nodes.

lie within a continent). However, this knowledge is not enforced during training. As a result, models exploit dataset-specific cues which may not align with valid reasoning.

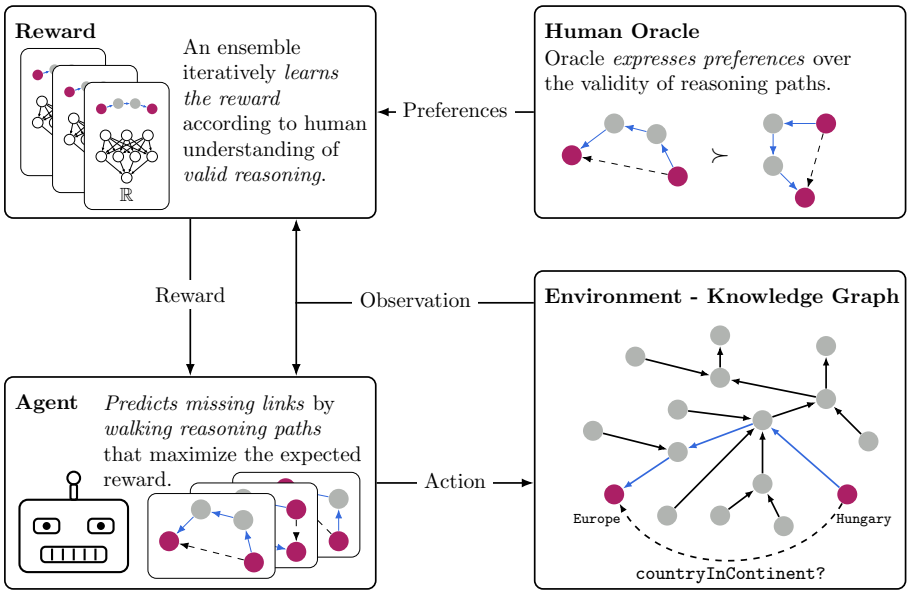


Figure 30: Schematic overview of LiEr (Wehner et al., 2025): a reinforcement-learning agent predicts missing links by walking from a given head entity through the KG. LiEr selects paths using an ensemble of reward estimators trained on human preference feedback. This guides the walk toward explanations that align with human understanding of valid reasoning.

To address this, we propose LiEr. LiEr frames link prediction as an MDP (Bellman, 1957) on the KG and uses a reinforcement-learning agent to generate

candidate paths (Figure 30). An ensemble of reward estimators is trained on pairwise preferences provided by a human oracle. The agent learns to favour paths that reflect valid reasoning.

LiEr makes two main contributions:

- A reinforcement learner for path-based link prediction that incorporates preference-based feedback over reasoning paths into its reward to align the agent's path selection with human judgments of valid reasoning.
- Empirical validation on benchmark datasets and a controlled Clever Hans' Countries KG, on which LiEr demonstrates improved generalisation when spurious patterns are present.

5.1.1 Definitions

First, the formal notions that ground LiErs' design are introduced. This includes the definition of valid and invalid reasoning, a brief recall of the KG as a typed quiver, and the formalisation of the tail link prediction task. Together, these concepts are the basis for path-based link prediction with human-aligned reinforcement learning.

Valid and Invalid Reasoning. Path-based link prediction methods predict missing links by chaining relations in a KG. The relation chains are explanations for the predictions at the same time. However, not all paths are equally meaningful. A model may learn to traverse sequences that connect the source and target entities without them reflecting human knowledge about the semantic concept underlying the predicted link.

To prevent this, we rely on a formal notion of valid reasoning. A reasoning path is **valid** if the truth of its steps makes it impossible for the conclusion to be false (Copi, 1954). In practical terms, this means that each step must be relevant to the query relation. Paths that link the right entities but include irrelevant or spurious steps are considered **invalid**. This distinction is critical. In a KG, every edge represents a true fact. Still, combining true facts into a path does not guarantee that the reasoning behind the conclusion makes sense to a human. The KGC model might be right for the wrong reason.

Consider the query $(\texttt{Hungary}, \texttt{inContinent}, ?)$. Two candidate reasoning paths are:

$$\text{Path A:}\quad \texttt{Hungary} \xrightarrow{\texttt{inRegion}} \texttt{Central Europe} \xrightarrow{\texttt{regionInContinent}} \texttt{Europe}$$

$$\text{Path B:}\quad \texttt{Hungary} \xrightarrow{\texttt{consumesPepperFrom}} \texttt{Europe}$$

Path A reflects valid reasoning, since each step supports the conclusion that `Hungary` is in `Europe`. Path B is factually correct but semantically unrelated to geographic inclusion, and thus constitutes invalid reasoning.

Knowledge Graph as a Typed Quiver. To formalise the structure in which these reasoning paths are constructed, we recall the definition of a KG as a typed quiver (Assem et al., 2006; Hogan et al., 2022; Das et al., 2018):

$$G = (E, R, \Sigma_V, \Sigma_R, h, t, \ell_E, \ell_R)$$

Here, E is the set of entities, R the set of relations, and Σ_E, Σ_R alphabets of entity and relation types. The mappings h, t assign direction to edges, and ℓ_E, ℓ_R label nodes and edges. This KG representation retains the symbolic expressivity of relational data while enabling graph-based traversal. It also prepares the ground for translating the KG into a sequential decision process.

Consider a minimal KG with three vertices to illustrate the formal definition of a KG as a typed quiver:

$$E = \{e_1, e_2, e_3\}$$

Let the vertex labels be given by:

$$\ell_E(e_1) = \texttt{Poland}, \quad \ell_E(e_2) = \texttt{Central Europe}, \quad \ell_E(e_3) = \texttt{Europe}$$

Define two directed edges $r_1, r_2 \in R$, with:

$$\begin{aligned} h(r_1) = e_1, \quad t(r_1) = e_2, \qquad & \ell_R(r_1) = \texttt{inRegion} \\ h(r_2) = e_2, \quad t(r_2) = e_3, \qquad & \ell_R(r_2) = \texttt{regionInContinent} \end{aligned}$$

Together, the composition of r_1 and r_2 is a path that allows traversal from `Poland` to `Europe` via the intermediate node `Central Europe`:

$$\texttt{Poland} \xrightarrow{\texttt{inRegion}} \texttt{Central Europe} \xrightarrow{\texttt{regionInContinent}} \texttt{Europe}$$

This enables multi-hop traversal through the graph structure.

Tail Link Prediction. The prediction task of LiEr is tail prediction. Given a head entity and a relation type (i.e. query), the model must infer the most plausible tail entity. Formally, tail prediction is about learning a mapping:

$$\rho : E \times \Sigma_R \to E$$

This mapping is trained to approximate the tail mapping t for edges, even if the edge is not observed during training.

The space of possible mappings $\rho \in \mathcal{P}$ is large. In practice, the mapping shall maximize predictive accuracy over a validation set:

$$\arg\max_{\rho \in \mathcal{P}} |\{r \in R_{\text{validation}} \mid t(r) = \rho(h(r), \ell_R(r))\}|$$

Suppose we query $(\texttt{Poland}, \texttt{locatedIn}, ?)$. A direct `locatedIn` edge to `Europe` is missing. However, the following reasoning path between `Poland` and `Europe` exists:

$$\texttt{Poland} \xrightarrow{\texttt{inRegion}} \texttt{Central Europe} \xrightarrow{\texttt{regionInContinent}} \texttt{Europe}$$

The path-based link prediction model p shall learn to return `Europe` by internally composing these reasoning steps.

$$\rho(\texttt{Poland}, \texttt{locatedIn}) = \texttt{Europe}$$

This formulation makes explicit the challenge that motivates LiEr. There are many paths can connect the head and tail, but only a subset expresses valid reasoning. Human feedback is required to identify which of these paths align with human knowledge of the underlying semantic concept. The following section describes how LiEr integrates such feedback into the learning process.

5.1.2 From Knowledge Graph to Markov Decision Process

We reformulate the link prediction task as a sequential decision process to enable learning from feedback over paths. Sequential decision processes are commonly modelled as a MDP (Bellman, 1957). Here, each decision step corresponds to following an edge in the graph. This allows the use of reinforcement learning to optimize the traversal policy. The traversal policy results in reasoning chains that predict tail entities.

This approach follows prior work on multi-hop reasoning in KGs (Das et al., 2018; Lin et al., 2018; Wan et al., 2020). In this line of work, the reward is maximised, by selecting at each decision step an outgoing edge that has the highest change of leading to the correct final node.

Let us recall that our goal is to learn from feedback over reasoning paths. These paths serve as explanations of the model's behaviour and are the basis for the preference-based feedback. In each iteration, the user is presented with two candidate paths and asked to express which one better reflects the intended reasoning. Thus, the model must construct explicit paths and present them during training.

We frame path construction as a MDP (Bellman, 1957), where each step corresponds to traversing an edge in the graph. Finally, the policy is optimised with the help of a reward function that rewards a traversal policy in accordance with human judgment of valid reasoning.

Formally, the MDP (Bellman, 1957) is defined as a tuple:

$$\text{MDP} = (\mathcal{S}, \mathcal{A}, \delta, \mathcal{R})$$

where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, δ the transition function, and $\mathcal{R}$ the reward function. The structure of the MDP is directly derived from the underlying KG.

Each state $s_i \in \mathcal{S}$ at step $i \in \mathbb{N}$ is a tuple:

$$s_i = (e_i, e_h^q, \sigma_r^q)$$

where e_i is the current vertex, and (e_h^q, σ_r^q) are fixed components of the link prediction query q. The task of the agent is to traverse the graph starting from e_h^q, guided by σ_r^q, in order to reach the correct tail entity.

At each state s_i, the agent selects an action from the available set $\mathcal{A}_{s_i}$, defined as:

$$\mathcal{A}_{s_i} = \{(e_i, r, e) | \forall r \in R, \forall e \in E : h(r) = e_i \wedge t(r) = e\} \cup \{(e_i, r_{loop}, e_i)\}$$

The first component allows the agent to walk to a neighbouring node via a directed edge r. The second component is a loop action, useful when no further traversal improves the expected reward.

The transition function $\delta : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is deterministic and updates the current state based on the selected action:

$$\delta(s_i, a_i) = s_{i+1} = (e_{i+1}, e_h^q, \sigma_r^q)$$

where e_{i+1} is the tail of the selected edge in action $a_i \in \mathcal{A}_{s_i}$.

Finally, the MDP includes a reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, which assigns numeric rewards to individual reasoning steps. The reward incentivises the

agent to prefer paths that reflect valid reasoning over those that do not. Since the notion of valid reasoning is not encoded in the KG itself, the reward must be learned from user feedback. Section 5.1.4 describes how $\mathcal{R}$ is parameterised and trained to approximate human preferences over paths.

This formulation supports multi-hop path construction through the graph by chaining actions. Once the MDP is defined, the next step is to learn a policy that solves it.

5.1.3 Solving the Markov Decision Process

The MDP is solved by a policy. The policy defines the behaviour of the agent as it walks the graph. At each step i, a non-stationary, history-dependent policy $\pi = (d_1, \ldots, d_I)$ selects an action based on the current state and the path history. In other words, at each time step $i \in \{1, \ldots, I\}$, a reinforcement agent observes its current state s_i, updates its internal representation of the path history, and selects the next action $a_i \in \mathcal{A}_{s_i}$. The agent continues walking until it reaches a maximum number of steps I. At this point, it outputs the last visited node e_I as the predicted tail entity.

The policy is implemented as a recurrent model that incrementally builds a latent representation of the path. Figure 31 provides a schematic overview of the policy network architecture. At each step i, the policy receives the current state embedding ς_i and the embedding α_{i-1} of the previously selected action a_{i-1}, and updates a hidden state η_i^d with a layer-normalised Long Short-Term Memory (LSTM) (Ba et al., 2016):

$$\eta_i^d = \text{lnLSTM}(\eta_{i-1}^d, [\alpha_{i-1}; \varsigma_i])$$

The latent path representation η_i^d summarises the walk so far. It holds structural and query information.

To select the next action, the policy computes a query- and history-dependent context vector by passing $[\eta_i^d; \varsigma_i]$ through two dense layers (Goodfellow et al., 2016) with leakyRelu (Redmon et al., 2016) activation functions:

$$z_i = \text{leakyReLU}(W_2 \cdot \text{leakyReLU}(W_1 \cdot [\eta_i^d; \varsigma_i] + B_1) + B_2)$$

Here, W_1, W_2 and B_1, B_2 are trainable weights and biases. The resulting vector z_i acts as an attention query for scoring available actions.

Let $A_i \in \mathbb{R}^{|\mathcal{A}_{s_i}| \times d}$ be the matrix of embeddings for all available actions at state s_i, where each row corresponds to an action embedding. The policy computes the probability distribution over actions using a softmax (Goodfellow et al., 2016) over the dot product between the action embeddings and the context vector:

$$d_i = \text{Softmax}(A_i \cdot z_i)$$

At training time, the agent samples actions from d_i to encourage exploration. At test time, the agent deterministically selects the action with the highest probability:

$$a_i = \arg\max d_i$$

The agent continues walking until a fixed number of steps I is reached. The final node e_I is returned as the predicted tail entity. The full walk $w = (\kappa_1, \ldots, \kappa_I)$, with $\kappa_i = (s_i, a_i)$, is the explanation for the prediction.

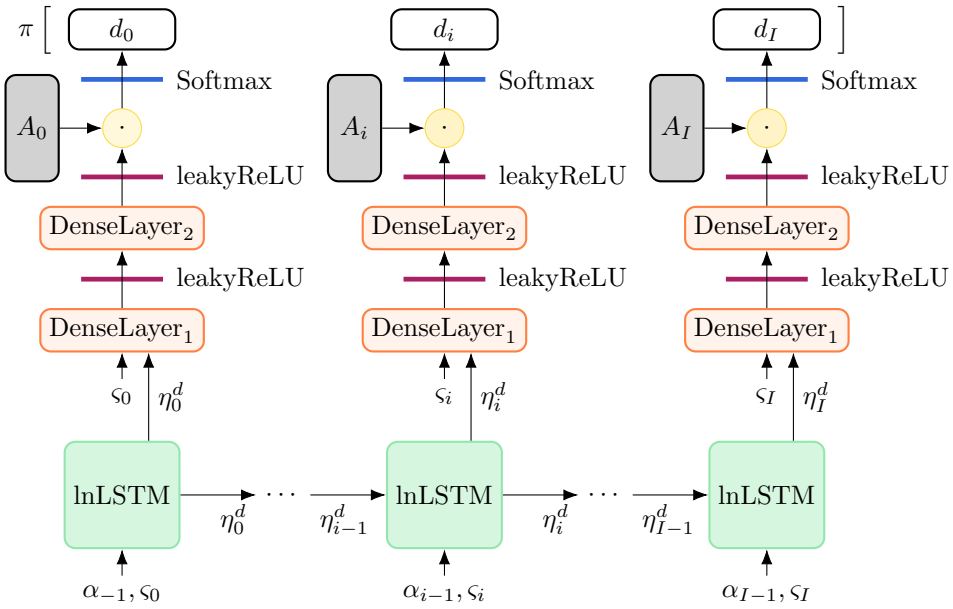


Figure 31: The architecture of the policy network. The model updates a latent path representation via a layer-normalised LSTM and uses a two-layer perceptron to score available actions. The selected action guides the next step in the reasoning path.

Graph-Constrained Reasoning. The policy network operates in a sub-symbolic space. Nevertheless, the available actions are constrained by the symbolic structure of the KG. This ensures that every step follows a true edge in the graph. However, while every individual step is true, this does not necessarily imply validity. The path may lead to the correct answer for the wrong reasons. This means that the agent may still learn to follow spurious but frequent paths that are not aligned with human judgment of valid reasoning.

To address this, the policy is trained to maximize a reward function aligned with human preferences. The following section introduces this reward function and describes how it is learned from pairwise feedback over reasoning paths.

5.1.4 Aligning the Policy Network with Human Understanding of Valid Reasoning via the Reward

In this section, a reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is introduced, which learns reward according to human judgments of valid reasoning. The reward function is trained from pairwise comparisons between reasoning paths. This enables human supervision at the level of explanations.

Preference-Based Feedback. Domain experts possess an understanding of valid reasoning within their field. For example, an "expert" may know that `Hungary` is in `Europe` because `Hungary` is in `Central Europe`, and `Central Europe` is in `Europe`. However, they may not know how this knowledge is described in the structure of the KG. For example, the fact that `Hungary` is in

`Central Europe` may be represented via the `inRegion` or `locatedIn` relation. This level of detail, such as the naming of relation types, is typically hidden from non-technical users.

In addition, some domain knowledge is tacit and difficult to express as explicit rules. However, domain experts are usually able to judge a reasoning path's validity when shown concrete examples.

Consider the previously introduced candidate reasoning paths for the query (`Hungary`, `inContinent`, ?):

$$\text{Path A:} \quad \texttt{Hungary} \xrightarrow{\texttt{inRegion}} \texttt{Central Europe} \xrightarrow{\texttt{regionInContinent}} \texttt{Europe}$$

$$\text{Path B:} \quad \texttt{Hungary} \xrightarrow{\texttt{consumesPepperFrom}} \texttt{Europe}$$

Both paths lead to the correct tail entity `Europe`, but only Path A reflects valid reasoning. Path B, while factually correct, is semantically unrelated to the concept of geographic containment. We ask the user to express preferences between the two model-generated reasoning paths. Here, the user prefers the valid and correct reasoning Path A over the invalid reasoning Path B. This makes the feedback mechanism independent of the user knowing how to formulate rules or examples, given the KG. It results in intuitive human supervision, which enables the training of a reward function that captures human-aligned notions of valid reasoning. The following paragraphs will describe how this feedback-based alignment is enforced over the reward.

Reward Ensemble. The reward is parameterised as an ensemble of recurrent networks (Dietterich and others, 2002). Each ensemble member x estimates the scalar reward τ_i^x for a reasoning step $\kappa_i = (s_i, a_i)$. The reward architecture is similar to the architecture of the policy network. The computation is given by:

$$\begin{aligned}
\eta_i^\tau &= \text{lnLSTM}(\eta_{i-1}^\tau, \varkappa_i) \\
\tau_i^x &= \text{batchNorm}\left(\text{leakyReLU}\left(W_2 \cdot \text{leakyReLU}(W_1 \cdot [\eta_i^\tau; \varkappa_i] + B_1) + B_2\right)\right)
\end{aligned}$$

Here, $\varkappa_i = [\varsigma_i; \alpha_i] \in \mathbb{R}^{l+n}$ is the concatenation of the state embedding ς_i with length l and the action embedding α_i with length n. The history embedding η_{i-1}^τ captures prior reasoning context relevant to the reward. The reward is batch-normalized to stabilize training (Goodfellow et al., 2016).

The final reward assigned to a reasoning step is the average across all $M \in \mathbb{N}$ ensemble members:

$$\hat{\tau}_i = \frac{1}{M} \sum_{x=1}^{M} \tau_i^x$$

We rely on annotations provided by a human oracle Λ, to align the reward with human preferences.

Preference Collection. An active selection strategy for preference queries is used to train the reward ensemble. The strategy aims to minimize the effort of the oracle Λ. At each training epoch, a set of path pairs $(w_{\sigma_r}^1, w_{\sigma_r}^2)$ is sampled according to the variance in their predicted rewards across ensemble members (Christiano et al., 2017; Hüllermeier and Waegeman, 2021).

This variance is used as an approximation for predictive uncertainty (Hüllermeier and Waegeman, 2021). Intuitively, paths with high disagreement in reward between ensemble members indicate epistemic uncertainty in the ensemble's understanding of valid reasoning. Getting feedback on such pairs eradicates these blind spots and improves the reward function where it is most needed. The targeted feedback approach reduces the overall annotation burden.

After selecting path pairs with a high variance in reward, they are presented to the oracle Λ. The Oracle expresses a preference over each pair:

$$\succ_\Lambda \quad \text{(preferred)}, \quad \prec_\Lambda \quad \text{(dispreferred)}, \quad \sim_\Lambda \quad \text{(indifferent)}$$

The pairs and their associated preferences are stored in a database $\mathcal{D}$. The database serves as the training set for the reward ensemble. Figure 32 shows a simple interface used for collecting this feedback.

The preference data guides the training of $\hat{\tau}$. It learns the preference ranking behavior of Λ.

The properties we assume, by which Λ's preference ranking is guided, are formalised in the following paragraph.

Properties of Λ. The human oracle Λ expresses preferences over reasoning paths based on two key criteria:

1. *Correctness*: The predicted link is correct.
2. *Validity*: The truth of the reasoning steps is relevant to the truth of the conclusion (Copi, 1954).

The examples in Table 18 illustrate how these criteria guide human judgment. Each path attempts to answer the query `(Hungary, inContinent, ?)`:

Table 18: Three reasoning paths for the query `(Hungary, inContinent, ?)`. The correct answer is `Europe`. *Ex.1* is correct and valid, *Ex.2* is correct but invalid, and *Ex.3* is incorrect and invalid.

(*Ex.1*)	`inContinent(Hungary, Europe)` ← `inRegion(Hungary, Central Europe)` ∧ `inContinent(Central Europe, Europe)`
(*Ex.2*)	`inContinent(Hungary, Europe)` ← `consumesPepperFrom(Hungary, Europe)`
(*Ex.3*)	`inContinent(Hungary, Austria)` ← `hasNeighbor(Hungary, Austria)`

Ex.1 uses reasoning that is relevant to the conclusion. `Hungary` is in `Central Europe`, and `Central Europe` is in `Europe`. This reasoning is correct and valid. *Ex.2* reaches the correct answer `Europe`. However, it does so through an unrelated fact about pepper consumption. The path is correct, but not valid. *Ex.3* derives that `Hungary` is in `Austria`, which is incorrect and invalid. The human oracle expresses a preference ordering that reflects these distinctions: *Ex.1* $\succ$ *Ex.2* $\succ$ *Ex.3*. That is, correct and valid reasoning is preferred over correct but invalid reasoning, which is in turn preferred over incorrect and invalid reasoning.

The reward model $\hat{\tau}$ is trained to approximate this ordering. It assigns higher cumulative rewards to paths that are preferred by the oracle. In this way, $\hat{\tau}$ rewards the validity of the model's reasoning.

Preference Alignment. The annotated path pairs $(w^1_{\sigma_r}, w^2_{\sigma_r})$ are used to train the reward. This enables the alignment with human understanding of valid reasoning. Let $\omega^1_{\sigma_r} = (\varkappa^1_1, \ldots, \varkappa^1_I)$ and $\omega^2_{\sigma_r} = (\varkappa^2_1, \ldots, \varkappa^2_I)$ be the embedded paths $w^1_{\sigma_r}$ and $w^2_{\sigma_r}$. The learned reward $\hat{\tau}$ must satisfy:

$$\begin{aligned}
\sum_{\varkappa^1 \in \omega^1_{\sigma_r}} \hat{\tau}(\varkappa^1) &> \sum_{\varkappa^2 \in \omega^2_{\sigma_r}} \hat{\tau}(\varkappa^2) \quad \text{if } w^1_{\sigma_r} \succ_\Lambda w^2_{\sigma_r} \\
\sum_{\varkappa^1 \in \omega^1_{\sigma_r}} \hat{\tau}(\varkappa^1) &< \sum_{\varkappa^2 \in \omega^2_{\sigma_r}} \hat{\tau}(\varkappa^2) \quad \text{if } w^1_{\sigma_r} \prec_\Lambda w^2_{\sigma_r} \\
\sum_{\varkappa^1 \in \omega^1_{\sigma_r}} \hat{\tau}(\varkappa^1) &\approx \sum_{\varkappa^2 \in \omega^2_{\sigma_r}} \hat{\tau}(\varkappa^2) \quad \text{if } w^1_{\sigma_r} \sim_\Lambda w^2_{\sigma_r}
\end{aligned}$$

These constraints are enforced using the Bradley–Terry model (Bradley and Terry, 1952), following Christiano et al. (2017). Given the pair (w^1, w^2), the probability that the model prefers w^1 is:

$$\hat{P}[w^1 \succ w^2] = \frac{\exp\left(\sum_{\varkappa^1 \in \omega^1} \hat{\tau}(\varkappa^1)\right)}{\exp\left(\sum_{\varkappa^1 \in \omega^1} \hat{\tau}(\varkappa^1)\right) + \exp\left(\sum_{\varkappa^2 \in \omega^2} \hat{\tau}(\varkappa^2)\right)}$$

The reward model is then trained with the following loss:

$$\mathcal{L}_{\text{reward}} = - \sum_{(w^1, w^2) \in \mathcal{D}} \mu \cdot \log(\hat{P}[w^1 \succ w^2]) + (1 - \mu) \cdot \log(\hat{P}[w^2 \succ w^1]) \tag{3}$$

where the preference weight μ encodes the human feedback:

$$\mu = \begin{cases} 1.0 & \text{if } w^1 \succ_\Lambda w^2 \\ 0.0 & \text{if } w^1 \prec_\Lambda w^2 \\ 0.5 & \text{if } w^1 \sim_\Lambda w^2 \end{cases}$$

This way, the reward ensemble is trained to approximate Λs' preference ordering, over what is valid reasoning in the KG domain at hand.

It is now fully described how LiEr predicts links between two vertices. It does so by constructing and evaluating reasoning paths that align with human judgment of valid reasoning. The model learns to favour correct, valid reasoning paths through a combination of symbolic constraints, recurrent policies, and preference-aligned reward learning. The following section presents the experimental results and evaluates the quality of the learned reasoning behaviours.

5.2 Evaluating LiEr‘s Predictive Performance and Reasoning Alignment

This section evaluates LiEr with respect to Research Question 3: *Can link prediction be aligned with human understanding of valid reasoning through interactive feedback?* First, it assesses whether the model achieves predictive performance comparable to that of existing link prediction methods, despite being trained on preference-based feedback. Second, it tests whether LiEr produces reasoning paths that better reflect human judgment of valid reasoning, even when spurious but predictive reasoning paths exist.

To that end, the evaluation considers three settings:

- Controlled reasoning benchmarks (Countries (Bouchard et al., 2015; Rocktäschel and Riedel, 2017)) to evaluate multi-hop reasoning behaviours.
- More realistic graphs (Family (Yang et al., 2017), Locations (Mitchell et al., 2018), and Sports) (Mitchell et al., 2018) to test generalisation to noisy, less structured domains.
- A deliberately misaligned benchmark (Clever Hans Countries) to test whether LiEr avoids learning spurious but predictive patterns when these conflict with valid reasoning.

Together, these evaluations assess whether LiEr's decision-making process is consistent with human judgments of valid reasoning. They also enable us to assess the amount of feedback required and how LiEr performs across tasks that vary in complexity and alignment risk.

Additionally, this section examines the practical aspects of collecting preference-based feedback. For that reason, a user interface for collecting preferences is presented. Furthermore, the robustness of LiEr is tested under noisy feedback. It is tested how the quality of predictions degrades as feedback noise increases. These results provide insights into the amount of feedback required and the system's tolerance to annotation errors.

The evaluation directly addresses Research Question 3, which asks if user knowledge can be incorporated into a link prediction model to iteratively align its reasoning with human intent, without compromising predictive performance. The next sections describe the experimental setup, metrics, and benchmark results in detail.

5.2.1 Experimental Setup

Implementation Details. LiEr is implemented in `PyTorch 1.10` and executed on a workstation with an AMD Ryzen Threadripper 2920X CPU, two NVIDIA GeForce RTX 2080 Ti GPUs, and 64 GB RAM. The codebase is publicly available on GitHub[7], hyperparameters and in-depth results are also published there.

The policy network described in Section 5.1.3 is trained with the REINFORCE algorithm (Williams, 1992). Minor adaptations are added to customise the algorithm for multi-hop reasoning following the strategy introduced in Das et al. (2018). The first adaptation is an additive control variate baseline. It reduces the variance in the gradient estimates (Evans et al., 2000; Hammersley, 2013). Additionally, an entropy regularisation term is added to the loss to encourage diverse exploration of actions while training. During training, dropout is applied to all layers and to the available actions to prevent overfitting and to encourage diverse exploration again at training time (Goodfellow et al., 2016).

The reward ensemble (see Section 5.1.4) is optimised via the ADAM optimiser (Kingma and Ba, 2015), with early stopping based on the Bradley-Terry loss introduced in Equation 3. All networks in the ensemble are regularised with dropout after each layer. Hyperparameters for the policy network and the reward ensemble are tuned using the Optuna framework (Akiba et al., 2019) and the TPESampler (Bergstra et al., 2011).

[7] `https://github.com/ChWehner/LiEr`

Simulating the Oracle with Regions of Interest. As described before, LiEr samples paths during each training epoch. The path pairs with the highest disagreement in reward across reward ensemble members are selected for human preference annotation. These annotations are used to update the reward model via the Bradley-Terry loss (see Section 5.1.4). The policy is then updated based on the current reward signal using REINFORCE (Williams, 1992). This iterative loop continues until convergence.

In this evaluation, the feedback process is simulated using predefined Region of Interests (ROIs) for each relation type in the evaluated KGs. Analogous to ROIs in image processing (Brinkmann, 2008), ROIs in KGs are structured reasoning paths that define valid multi-hop reasoning patterns. The patterns are considered valid for the prediction task. Thus, each ROI is reasoning template that reconstructs a missing link between a head and tail entity in a way that aligns with human understanding of valid reasoning.

For instance, in the COUNTRIES dataset, an ROI for the `locatedIn` relation is:

$$\texttt{Country} \rightarrow \texttt{locatedIn} \rightarrow \texttt{Region} \rightarrow \texttt{locatedIn} \rightarrow \texttt{Continent}$$

Alternatively, another ROI uses neighbouring countries as an intermediate reasoning step:

$$\texttt{Country} \rightarrow \texttt{neighbourOf} \rightarrow \texttt{Country} \rightarrow \texttt{locatedIn} \rightarrow \texttt{Continent}$$

The ROIs are derived from domain knowledge. The simulated feedback is generated based on those predefined ROIs. The feedback is expressed such that reasoning paths that match an *ROI* are preferred over those that do not. Furthermore, paths leading to the correct tail entity are preferred over those that lead to an incorrect tail entity. This allows for reproducible generation of preference annotations that reflect validity and predictive correctness.

It is important to note that the construction of ROIs is only feasible for relation types with well-defined underlying rules in structured domains. This constraint influenced the selection of the benchmark datasets used throughout the evaluation.[8]

Comparison Models. LiEr is evaluated against a set of neuro-symbolic and sub-symbolic link prediction models. The selection provides a diverse basis for comparing LiEr's predictive performance and reasoning behaviour.

As a neuro-symbolic baseline, we include MINERVA (Das et al., 2018), a reinforcement-learning based model that, like LiEr, walks paths in the graph to predict missing links. However, MINERVA is trained solely on ground-truth edges without incorporating feedback over reasoning paths. We also compare against Neural Theorem Provers (NTP) and its improved variant NTP-λ (Rocktäschel and Riedel, 2017), which use differentiable logical inference for end-to-end training. Additionally, Neural Logic Programming (NeuralLP) (Yang et al., 2017) is included in the evaluation. It combines rule learning with neural sequence models. All three neuro-symbolic methods produce interpretable reasoning chains and serve as widely used state-of-the-art baselines for interpretable link prediction.

[8] A complete list of the predefined ROIs is available here: `https://github.com/ChWehner/LiEr`

On the other side, there are the non-interpretable embedding-based link prediction methods, which are also state-of-the-art in certain link prediction tasks (Ali et al., 2021). Here TransE (Bordes et al., 2013), DistMult (Yang et al., 2015), and ConvE (Dettmers et al., 2018) are included in the evaluation as comparison models. As their name suggests, they learn embeddings of entities and relations. The embeddings are fed to interaction functions to score facts. The scores are used to judge the plausibility of unknown facts. These models were also used in the previous evaluation of KGEPrisma and are standard choices for link prediction due to their scalability and empirical performance (Ali et al., 2021).

The neuro-symbolic baselines were evaluated using the authors' official implementations[9] with default settings, unless dataset-specific configurations were published. Embedding-based models were trained using the `PyKEEN` framework (Ali et al., 2021)[10], which provides reproducible pipelines for a wide range of KGE models. The hyperparameters were optimized in an ablation study.

Together, these baselines cover interpretable and black-box KGC models. This ensures a comprehensive evaluation of LiEr with respect to predictive accuracy and alignment with valid reasoning. The following section provides a detailed definition of the evaluation metrics.

5.2.2 Evaluation Metrics

The evaluation uses two metrics to quantify predictive accuracy and reasoning quality. This is done with the help of the metrics Mean Reciprocal Rank (MRR) and MRLR.

Mean Reciprocal Rank (MRR). MRR is a standard metric for evaluating link prediction models (Fuhr, 2018; Ali et al., 2021). It measures how the correct entity is ranked among all candidate entities. Let us recall that a link prediction model produces for each query q a ranked list of predictions. MRR gives a measure on the scale 1 to 0 of how correct answers were ranked. A MRR close to 1 indicates that correct answers tend to be ranked near the top.

Formally, MRR is computed as:

$$\text{MRR} = \frac{1}{U} \sum_{q=1}^{U} \frac{1}{\text{rank}_q}$$

where rank_q is the position of the correct entity in the ranked list for query q. Q is the total number of test queries. This metric reflects a model's predictive ability but does not consider how the answer was derived.

Mean Reciprocal Localisation Rank (MRLR). We additionally introduce the MRLR metric to evaluate if predictions are made for the right reasons. The formulas of the MRR and MRLR are similar. Thus, they share similar properties, such as a rapid deterioration of the score in the top ranks. However, unlike MRR, MRLR looks at the reasoning path used to reach a prediction instead of the correctness of the predicted entity.

[9] MINERVA: `https://github.com/shehzaadzd/MINERVA`, NTP/NTP-λ: `https://github.com/uclnlp/ntp`, NeuralLP: `https://github.com/fanyangxyz/Neural-LP`

[10] `https://github.com/pykeen/pykeen`

The KGC model outputs a ranked list of candidate reasoning paths for each query q. Rank by rank, MRLR checks if a path matches one of the predefined ROIs. The rank of the first valid path is used for q in the calculation of the MRLR. Importantly, MRLR ignores whether the final entity predicted by the path is correct. MRLR only measures whether the reasoning aligns with what a human considers valid.

Formally, the MRLR is defined as:

$$\text{MRLR} = \frac{1}{Q} \sum_{q=1}^{Q} \frac{1}{\text{valid\_rank}_q}$$

where valid_rank_q is the rank of the top ranked path for query q that matches any ROI. KGC models have to produce reasoning paths so that this metric can be computed.

Together, MRR and MRLR allow us to separate two aspects of model performance. MRR captures how often a model predicts the correct entity. MRLR measures how often the model reasons in a way that matches a predefined notion of valid reasoning.

5.2.3 Benchmark Comparison

The first set of experiments evaluates if LiEr matches the reasoning capabilities of existing state-of-the-art link prediction models, despite being trained solely on human preference feedback. For this, we use the COUNTRIES benchmark (Bouchard et al., 2015; Rocktäschel and Riedel, 2017), a synthetic KG specifically constructed to test the multi-hop reasoning capability of link prediction methods in a controlled setting.

The COUNTRIES KG consists of 244 `Countries`, 23 `Regions`, and 5 `Continents`. Entities are connected via two relation types: `locatedIn` and `neighbourOf`. The test and validation data are about predicting the `Continent` of a `Country`. Direct links between `Countries` and `Continents` are removed to enforce reliance on multi-hop reasoning. The tasks are designed to isolate reasoning capabilities and eliminate confounding factors such as data noise or relation ambiguity found in more realistic KG.

Three tasks are defined over this KG. These tasks are denoted as **S1**, **S2**, and **S3**. The tasks get progressively more complex by requiring deeper multi-hop reasoning chains. This also means that each task requires a KGC model to learn a different composition of relations to answer queries correctly. The valid reasoning chains for each task are summarised in Table 19. Each valid reasoning chain also defines a ROI used for evaluating reasoning alignment.

Table 19: Reasoning patterns required to solve the COUNTRIES tasks. Each pattern defines a ROI for the `locatedIn` relation.

(*S1*)	`locatedIn(Country, Continent) ← locatedIn(Country, Region) ∧ locatedIn(Region, Continent)`
(*S2*)	`locatedIn(Country X, Continent) ← neighbourOf(Country X, Country Y) ∧ locatedIn(Country Y, Continent)`
(*S3*)	`locatedIn(Country X, Continent) ← neighbourOf(Country X, Country Y) ∧ neighbourOf(Country Y, Country Z) ∧ locatedIn(Country Z, Continent)` `locatedIn(Country X, Continent) ← neighbourOf(Country X, Country Y) ∧ locatedIn(Country Y, Region) ∧ locatedIn(Region, Continent)`

S1. In the first task, the KGC model must predict the `locatedIn` relation between countries and continents. A model has to infer the missing relation through a two-step transitive reasoning chain. It has to go from the country to its region, and from the region to the continent. This setting tests the model's ability to perform basic multi-hop inference.

LiEr achieves perfect MRR and MRLR score. The neuro-symbolic models NeuralLP and MINERVA perform equally well. NTP performs slightly worse. These results show that LiEr is capable of learning valid two-hop transitive reasoning patterns from preference feedback.

By contrast, all embedding-based models fail to solve this task. Their performance collapses due to the semantic overloading of the `locatedIn` relation. This relation connects countries to regions and continents, and regions to continents. Embedding methods like TransE, DistMult, and ConvE are unable to represent this. It leads to mixed signals while training and to poor generalisation. This failure pattern will recur across other tasks that require structured reasoning.

S2. The second task increases the complexity by requiring models to predict a country's continent based on the location of a neighbouring country. Thus, a valid reasoning chain must first traverse a `neighbourOf` relation from a country to another country before using a `locatedIn` link to the correct continent.

Here, symbolic methods such as NeuralLP, NTP, and NTP-λ perform best. They achieve near-perfect MRR and MRLR scores. These models learn the intended two-hop inference pattern.

LiEr and MINERVA follow closely behind. While their scores are slightly lower compared to NeuralLP and NTP, the models still perform well across metrics. Examining their explanations shows that the minor drop in performance is due to selecting the wrong neighbouring country. Not all neighbouring countries are connected to their continent. Selecting the wrong neighbouring country out of a potentially large number of options prevents the model from reaching the correct continent. Nonetheless, their high MRLR scores indicate that the majority of predicted paths align with ROIs. This indicates that correct reasoning is learned in most cases.

Embedding-based models again fail to generalize in this setting due to similar reasons as in the first task. Their representations lack the expressiveness required to model multi-hop inference chains in this setting.

S3. The third task increases the complexity of the task once more (see Table 19). The KGC models must follow a valid three-hop reasoning chain to answer queries correctly. For example, a model first has to traverse a `neighbourOf` edge to reach a neighboring country, then use a `locatedIn` relation to identify an intermediate region, and finally take another `locatedIn` edge to reach the correct continent. Alternatively, models can follow two consecutive `neighbourOf` edges before reaching a country that is directly linked to the correct continent. The reasoning templates learned in *S1* and *S2*, which only require two hops, are not sufficient to solve this task.

As shown in Table 20, all embedding-based models struggle on task three. Their low performance confirms earlier observations about their inability to learn predictive embeddings when relations are overloaded, such as is the case

for the `locatedIn` relation. These models cannot resolve the multi-hop dependencies required by the task.

Neuro-symbolic models such as NTP, NTP-λ, and NeuralLP show a divergence between the MRR and MRLR metrics (see Table 21). Their MRR scores are low. This indicates limited predictive accuracy. However, their MRLR values are high. On closer inspection, these models tend to rely on reasoning patterns similar to those required to solve *S1* and *S2*, which, although valid, are not sufficient to reach the correct answer in the *S3* setting. The resulting paths are aligned with the ROIs and thus rewarded under MRLR. However, they do not result in the correct predictions.

MINERVA and LiEr better cope with the increased reasoning depth required by this task. MINERVA shows moderate performance with higher variance across runs. This instability stems from its bias towards shorter paths. During training, these shorter two-hop paths are rewarded, but they fail to generalize to the test set. As a result, MINERVA follows incomplete or redundant paths, for instance, it reaches a dead end at a region or fails to continue to a continent. It results in invalid trajectories and reduced MRLR.

In contrast, LiEr achieves the highest performance on both metrics. It is able to identify valid three-hop reasoning patterns that were enforced while training. These results show the benefits of using interactive feedback to guide a model towards human-aligned reasoning behaviours.

Summary. Across all three reasoning tasks in Countries, LiEr performs on par with or better than state-of-the-art link prediction models. It learns to generate accurate predictions and valid reasoning paths from preference-based feedback on its explanations. These results validate the claim that preference-based feedback on explanations can support generalizable link prediction in structured and controlled tasks.

Table 20: MRR (↑) across tasks *S1*, *S2*, and *S3* in the Countries KG. Mean and variance are computed over 10 runs.

Model	Countries S1	Countries S2	Countries S3
ConvE	0.05 ± 0.00	0.05 ± 0.00	0.05 ± 0.01
TransE	0.03 ± 0.00	0.06 ± 0.00	0.03 ± 0.00
DistMult	0.04 ± 0.00	0.03 ± 0.00	0.04 ± 0.00
NTP	0.80 ± 0.10	0.95 ± 0.01	0.28 ± 0.00
NTP-λ	0.91 ± 0.07	0.98 ± 0.00	0.10 ± 0.01
NeuralLP	**1.00** ± 0.00	**1.00** ± 0.00	0.05 ± 0.00
MINERVA	**1.00** ± 0.00	0.95 ± 0.00	0.49 ± 0.11
LiEr	**1.00** ± 0.00	0.90 ± 0.02	**0.85** ± 0.01

Table 21: MRLR (↑) for tasks *S1*, *S2*, and *S3* in Countries KG. Mean and variance across 10 runs.

Model	Countries S1	Countries S2	Countries S3
NTP	0.77 ± 0.15	**1.00** ± 0.00	**1.00** ± 0.00
NTP-λ	**1.00** ± 0.00	**1.00** ± 0.00	**1.00** ± 0.00
NeuralLP	**1.00** ± 0.00	**1.00** ± 0.00	**1.00** ± 0.00
MINERVA	**1.00** ± 0.00	0.95 ± 0.00	0.33 ± 0.01
LiEr	**1.00** ± 0.00	0.93 ± 0.01	0.88 ± 0.01

The Countries benchmark demonstrates that LiEr can learn in a controlled setting. However, Countries is the KG equivalent of computer vision datasets such as MNIST (Deng, 2012) or Cats vs. Dogs (Parkhi et al., 2012). It helps to diagnose the generalization capability of a model, but it is not complex and messy enough to be representative of real-world data. We next evaluate LiEr on the Family KG (Yang et al., 2017)[11] to move one step closer to realistic graph structures.

Family. Family is a benchmark KG with 3007 entities and 12 relation types. It models kinship relations, such as `father`, `uncle`, `aunt`, `daughter`, and `niece`.

[11] `https://github.com/fanyangxyz/Neural-LP/tree/master/datasets/family`

Naturally, a kinship relation can be reconstructed using multi-hop compositions over other kinship relations. These compositions form a complete and known set of valid ROIs. They enable the evaluation of reasoning quality. For example:

$$\texttt{daughter(X,Y)} \leftarrow \texttt{wife(X,Z)} \wedge \texttt{daughter(Z,Y)}$$
$$\texttt{uncle(X,Y)} \leftarrow \texttt{brother(X,Z)} \wedge \texttt{father(Z,Y)}$$
$$\texttt{aunt(X,Y)} \leftarrow \texttt{niece(Y,X)}$$

Despite its structured nature, two challenges make this benchmark non-trivial. First, the KG exhibits a high degree of structural redundancy. Most nodes share similar patterns in their local neighbourhoods. Each child has two parents. Each parent has one to three kids. Each child has an aunt and so forth. This leads to oversmoothing in embedding-based models (Hoseinnia et al., 2025). Nodes become indistinguishable in the latent space, and predictions collapse. Second, several relation types in the KG show inconsistent directionality. For example, the `father` relation sometimes points from parent to child, and in other cases the reverse. This complicates rule induction and generalisation.

The results in Table 22 reflect these challenges. Embedding-based models such as TransE, ConvE, and DistMult fail almost entirely. They show near-zero MRR scores. Their vector representations collapse under structural repetition. They are not able to capture the multi-hop reasoning chains.

More surprisingly, NTP and NTP-λ also perform poorly. They do recover syntactically correct rules, such as `aunt(X,Y)` $\leftarrow$ `niece(Y,X)` or `daughter(X,Y)` $\leftarrow$ `niece(X,Z)` $\wedge$ `brother(Z,Y)`. However, these rules often fail to generalize due to confusion over relation directionality. An inspection of the learned rules shows mismatches in the ordering of arguments. This leads to failure in grounding the rules to identify the correct tail.

In contrast, NeuralLP achieves high MRR and MRLR scores. Its rule-learning mechanism appears to be more robust. It learns correct and valid rules covering both directionalities.

MINERVA and LiEr show moderate performance on this benchmark KG. The models struggle with resolving the directionality correctly. However, LiEr slightly outperforms MINERVA in terms of MRR and MRLR. LiEr is better at reproducing valid paths. This is because LiEr was explicitly trained to do so by supervision via preference-based feedback on explanations.

The Family benchmark introduces noise and inconsistencies and thus presents an increase in difficulty compared to Countries. LiEr does not solve the task perfectly. However, it performs comparably to most baselines, especially in terms of MRLRs, where its preference-based learning encourages reasoning paths that align with ROIs. This shows that LiEr can learn valid reasoning behaviour in structured real-world KGs.

Locations. While Family introduces symbolic complexity, it is still curated. We now turn to a subset of the Never-Ending Language Learning (NELL) KG Mitchell et al. (2018) to further test if LiEr generalises to more realistic KGs. The Locations benchmark KG includes geographical facts extracted from textual sources across the internet. Unlike Countries or Family, the data is less structured and introduces natural noise.

The KG consists of 445 entities across four classes, `city`, `state`, `country`, and `capital` and includes five relationtypes: `concept_citylocatedincountry`,

Table 22: MRR (↑) metric and variance for the FAMILY, SPORTS and LOCATIONS KGs. Mean and variance of 10 training runs.

Model	Family	Locations	Sports
ConvE	0.04 ± 0.00	0.02 ± 0.00	0.01 ± 0.00
TransE	0.03 ± 0.00	0.02 ± 0.00	0.01 ± 0.00
DistMult	0.03 ± 0.00	0.01 ± 0.00	0.01 ± 0.00
NTP	0.03 ± 0.00	0.58 ± 0.00	0.72 ± 0.00
NTP-λ	0.03 ± 0.00	0.63 ± 0.00	0.75 ± 0.00
NeuralLP	**0.70** ± 0.00	**0.99** ± 0.00	**0.84** ± 0.00
Minerva	0.24 ± 0.00	0.49 ± 0.00	0.79 ± 0.00
LiEr	0.28 ± 0.00	0.52 ± 0.00	0.79 ± 0.00

Table 23: MRLR (↑) metric and variance for the FAMILY, SPORTS and LOCATIONS KGs. Mean and variance of 10 training runs.

Model	Family	Locations	Sports
NTP	0.06 ± 0.00	0.34 ± 0.02	0.75 ± 0.12
NTP-λ	0.07 ± 0.00	0.29 ± 0.01	**0.88** ± 0.05
NeuralLP	**0.95** ± 0.00	0.00 ± 0.00	0.01 ± 0.00
Minerva	0.68 ± 0.00	0.34 ± 0.00	0.76 ± 0.00
LiEr	0.77 ± 0.00	**0.37** ± 0.00	0.79 ± 0.00

`concept_citylocatedinstate`, `concept_statelocatedincountry`, `concept_citycapitalofcountry`, and `concept_statehascapital`. The relations are composable into valid multi-hop reasoning chains. For example:

$$\texttt{concept\_citylocatedincountry(X,Y)} \leftarrow \texttt{concept\_citylocatedinstate(X,Z)} \wedge \texttt{concept\_statelocatedincountry(Z,Y)},$$
and
$$\texttt{concept\_statehascapital(X,Y)} \leftarrow \texttt{concept\_statelocatedincountry(X,Z)} \wedge \texttt{concept\_citycapitalofcountry(Y,Z)}.$$

These chains define the ROIs used to assess reasoning quality. However, compared to FAMILY, the reasoning space is wider and noisier. Thus, the benchmark is a closer to real-world link prediction tasks.

As reported in Table 22, embedding-based models again underperform. Their scores remain near zero. This is because of their limited capacity to model multi-hop dependencies. The inability to represent entity roles and relational composition leads to collapsed predictions with little generalization.

NTP, NTP-λ, and MINERVA achieve moderate MRR scores. Their reasoning paths are often valid. LiEr matches these models in MRR, but outperforms them on MRLR. Thus, LiEr is better at reproducing valid, human-aligned reasoning paths. A typical path learned by LiEr is:

$$\texttt{concept\_citylocatedincountry(X,Y)} \leftarrow \texttt{concept\_statehascapital(Z,X)} \wedge \texttt{concept\_statelocatedincountry(Z,Y)}$$

This path reflects correct domain reasoning: if a city is the capital of a state, and that state is located in a given country, the city must be in that country as well.

NeuralLP achieves almost perfect performance on MRR while scoring zero on MRLR. An analysis of its learned rules reveals that it tends to exploit statistical shortcuts, like:

$$\texttt{concept\_citylocatedincountry(C,A)} \leftarrow \texttt{concept\_citylocatedincountry(B,A)} \wedge \texttt{concept\_citylocatedincountry(C,B)}$$

This rule is invalid. However, it results in the correct prediction. Cities cannot be located in other cities. The rule still works in some cases, because

the concept_citylocatedincountry relation is loosely interpreted by this KG. It connects two cities in the same country, in addition to connecting a city with its country. This inconsistency in the KG points to poor data modelling practices. On a side note, that this rule produces a correct result is an error in the benchmark KG and should be fixed.

LiEr provides a middle ground of predictive accuracy and valid reasoning. It achieves the highest MRLR among all models. Preference-based learning encourages validity, even in real-world settings with ambiguous or noisy data.

Sports. The final real-world benchmark KG is SPORTS, another subset of the NELL KG (Mitchell et al., 2018). This KG models athletes and sports organizations and has a larger number of entities and relation types compared to LOCATIONS. It includes 1039 entities from the classes athletes, sports teams, universities, sports leagues, and organizations, connected by the relation types concept_athleteledsportsteam, concept_athleteplaysforteam, concept_coachesteam, and concept_personbelongstoorganization.

These relations can be reconstructed through valid reasoning paths. For instance:

$$\texttt{concept\_personbelongstoorganization(X,Y)} \leftarrow \texttt{concept\_coachesteam(X,Y)}$$

Such patterns define the ROIs used for preference simulation and serve as a basis for evaluating the alignment of reasoning.

As seen in Table 22, embedding-based models again fail to recover meaningful structure. Their predictive performance is near zero. This continues the trend observed in LOCATIONS and points to a consistent weakness of embeddings in tasks that are reasoning-heavy.

Neuro-symbolic models perform better. NTP, NTP-λ, MINERVA, and LiEr achieve high MRR and MRLR scores. NTP-λ shows the highest reasoning validity. LiEr also performs strongly. It produces paths that are accurate and align with valid reasoning.

As in the LOCATIONS benchmark, NeuralLP reaches high MRR but fails to produce meaningful explanations. A common pattern induced by NeuralLP is:

$$\texttt{concept\_personbelongstoorganization(X,Y)} \leftarrow \texttt{concept\_personbelongstoorganization(X,Z)} \wedge \texttt{concept\_personbelongstoorganization(Z,Y)}$$

The rule is invalid. Meanwhile, it predicts correct tails. The rule work, because in some cases, the concept_personbelongsto- organization relation is loosely interpreted by this KG. It connects two persons in the same organisation, in addition to connecting a person with its organisation. This inconsistency in the KG points to poor data modelling practices. Here and in SPORTS, NeuralLP overfits to such regularities at the expense of valid reasoning. Again, that this rule produces a correct result is an error in the KG and should be fixed.

LiEr, in contrast, benefits from feedback that penalises such invalid patterns. Its reasoning paths are aligned with the ROIs. In doing so, LiEr demonstrates that preference-based supervision on explanations guides link prediction models away from statistically convenient but logically flawed solutions, towards human-understanding of valid reasoning.

Across the three benchmarks (Family, Locations, and Sports), LiEr shows predictive performance comparable to or better than existing neuro-symbolic models, while maintaining better alignment with valid reasoning paths. LiEr also outperforms embedding-based KGC models by relying on interpretable reasoning and user feedback. These results confirm that LiEr is effective on realistic, noisy, and structurally diverse KGs.

The next section evaluates LiEr under deliberately misleading conditions. It will be tested if LiEr resists learning spurious shortcuts when predictive structure is misaligned with valid reasoning.

5.2.4 Evaluating LiEr's Alignment with Valid Reasoning

The previous benchmarks focused on whether LiEr learns valid reasoning patterns from preference-based feedback. However, in the previous KGs, correct answers can mostly only be found if the reasoning is valid. The model can basically not be correct for the wrong reasons. This section evaluates a spurious scenario. Does LiEr align with human reasoning even when the training data contains spurious shortcuts?

The Clever Hans Countries KG benchmark is introduced to answer that question. This KG is a modified version of Countries S3. It is designed to isolate and test the robustness of link prediction models against spurious relations. The benchmark KG is inspired by the Clever Hans effect (Lapuschkin et al., 2019).

Benchmark Design. Two changes are applied to the Countries S3 KG in order to derive the Clever Hans Countries KG. First, all `locatedIn` edges are replaced with the relation types `inRegion`, `regionInContinent`, and `countryInContinent`. This eradicates the semantic overload of the `locatedIn` relation. Second, the new relation type `consumesPepperFrom` is injected. It links each country to its correct continent during training and validation. However, in the test set, this relation is systematically corrupted to point to the wrong continent.

As a result, models that rely on the `consumesPepperFrom` relation to predict `countryInContinent` perform well during validation. However, they fail at test time. This setup tests if a model has learned spurious correlations or robust, valid reasoning.

Table 24: MRR (↑) metric and variance for the *Clever Hans Countries* KG. Mean and variance of 10 training runs.

Model	Clever Hans Countries
ConvE	0.39 ± 0.02
TransE	0.35 ± 0.01
DistMult	0.29 ± 0.03
NTP	0.08 ± 0.00
NTP-λ	0.08 ± 0.03
NeuralLP	0.08 ± 0.00
Minerva	0.23 ± 0.07
LiEr	**0.86** ± 0.08

Table 25: MRLR (↑) metric and variance for the *Clever Hans Countries* KG. Mean and variance of 10 training runs.

Model	Clever Hans Countries
NTP	0.14 ± 0.13
NTP-λ	0.05 ± 0.05
Neural LP	0.00 ± 0.00
Minerva	0.22 ± 0.07
LiEr	**0.86** ± 0.08

Table 26: Exemplary top three reasoning paths and their predicted probabilities for the query `(Israel, countryInContinent, ?)`, comparing the behavior of MINERVA, NeuralLP, and LiEr. Reasonings are extracted from representative training runs.

MINERVA	0.91 countryInContinent(Israel, Africa) ← consumesPepperFrom(Israel, Africa) 0.07 countryInContinent(Israel, Africa) ← neighbourOf(Israel, Egypt) ∧ consumesPepperFrom(Egypt, Africa) 0.02 countryInContinent(Israel, Asia) ← neighbourOf(Israel, Lebanon) ∧ inRegion(Lebanon, West Asia) ∧ regionInContinent(West Asia, Asia)
NeuralLP	0.72 countryInContinent(Country, Continent) ← consumesPepperFrom(Country, Continent) 0.18 countryInContinent(Country, Continent) ← neighbourOf(Country, Country) ∧ consumesPepperFrom(Country, Continent) 0.08 countryInContinent(Country, Continent) ← neighbourOf(Country, Country) ∧ neighbourOf(Country, Country) ∧ consumesPepperFrom(Country, Continent)
LiEr	0.78 countryInContinent(Israel, Asia) ← neighbourOf(Israel, Lebanon) ∧ inRegion(Lebanon, West Asia) ∧ regionInContinent(West Asia, Asia) 0.17 countryInContinent(Israel, Asia) ← neighbourOf(Israel, Syria) ∧ neighbourOf(Syria, Iraq) ∧ countryInContinent(Iraq, Asia) 0.04 countryInContinent(Israel, Asia) ← neighbourOf(Israel, Jordan) ∧ inRegion(Jordan, West Asia) ∧ regionInContinent(West Asia, Asia)

Results. LiEr outperforms all baselines, as shown in Table 24 and Table 25. It has the highest MRR and MRLR of all compared methods. Looking at LiEr's explanation showes that it does not rely on the spurious `consumesPepperFrom` relation to predict missing `countryInContinent` relations. Instead, it learns paths aligned with valid reasoning. Since the preference-based feedback never rewards reasoning steps involving the misleading relation, the model is discouraged from using it, despite its high predictive utility during training.

In contrast, the link prediction methods NTP, NTP-λ, and NeuralLP perform poorly. Manual inspection of their learned rules reveals a heavy reliance on the `consumesPepperFrom` relation to predict missing `countryInContinent` relations. While rules with the `consumesPepperFrom` relation are predictive in the training distribution, they fail on the test set. This issue is what motivated LiEr. Rule-based systems are susceptible to spurious relations (Marconato et al., 2023) and can not distinguish between valid and invalid reasoning if no alignment with human expectations is enforced.

MINERVA performs moderately. However, its MRR and MRLR is below LiEr's. While its policy occasionally explores correct reasoning chains, it tends to favour short paths that include the spurious relation.

Qualitative Analysis. Table 26 shows examples of the reasoning patterns learned by LiEr, MINERVA, and NeuralLP. The latter two are confident in paths that include the `consumesPepperFrom` relation. For instance, MINERVA predicts that `Israel` is in `Africa` based on a direct `consumesPepperFrom` link, even though this relation is intentionally corrupted in the test set.

LiEr prefers longer and valid chains, like traversing `neighborOf` links, inferring the region, and then reaching the correct continent through a three-hop reasoning path. These inferences are more complex and less frequent in the training set, but are actively reinforced by the preference-based feedback signal. Because LiEr optimizes for alignment with feedback on explanations, it avoids fitting to spurious statistical patterns and performs better at test time.

These experiments confirm that LiEr performs well in settings where valid reasoning is aligned with statistical signals, as well as under deliberately misleading, spurious relations. It avoids shortcuts and learns robust reasoning aligned with human judgment of valid reasoning. This is possible through its human-in-the-loop training paradigm. LiEr is designed to reproduce human-preferred explanations. Thus, LiEr's value lies in generalizing robustly in deceptive environments.

5.2.5 Remarks on the Collection of Preference-based Feedback

As discussed before, LiEr's main feature is its ability to learn from human preference-based feedback on explanations. A lightweight command-line interface (CLI) was designed for collecting feedback from a human oracle (user) (see Figure 32). The interface displays reasoning paths side by side, together with the associated queries and answers, and prompts the human to indicate which path is more plausible.

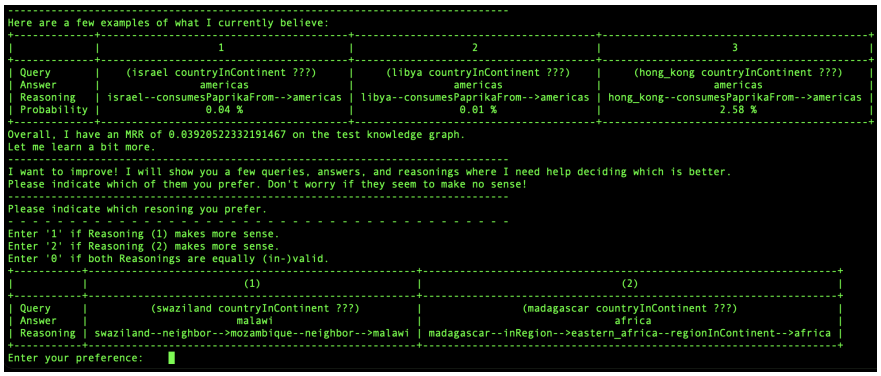


Figure 32: Screenshot of the CLI used to collect preference feedback for LiEr. The CLI presents queries, predicted answers, and candidate reasoning paths, and asks the human to choose the most plausible one.

The CLI is intentionally simple. Feedback is given by pressing a single key '1', '2', or '0' to indicate whether reasoning path (1), (2), or neither is preferred. No special annotation format or prior knowledge of the model internals is required. In practice, this form of interaction was simple to learn and easy to use, even for users without machine learning expertise. The time required for a decision varies depending on the complexity of the reasoning paths and the user's familiarity with the domain.

Learning from Comparisons Between Incorrect Paths. While testing, it was observable that LiEr benefits from preferences over incorrect and invalid paths. For example, in the early stages of training on the Countries S3 KG, the model often presents invalid and incorrect paths to the user for feedback. However, when a user consistently prefers paths that contain a partially correct structure, like an occurrence of the `locatedIn` relation to regions, LiEr begins to bias its sampling process towards paths that involve regions. Once LiEr reaches a region, the probability is high that it discovers a path to the correct continent. In this way, even weak or noisy feedback can help steer the model toward promising parts of the search space.

Robustness to Erroneous Preferences. Human feedback is not always accurate. Users may prefer flawed reasoning paths, either due to a lack of attention, incomplete knowledge, or misleading naming conventions in the KG. To study how LiEr handles such human errors, its performance is evaluated under increasing levels of deliberately injected preference noise.

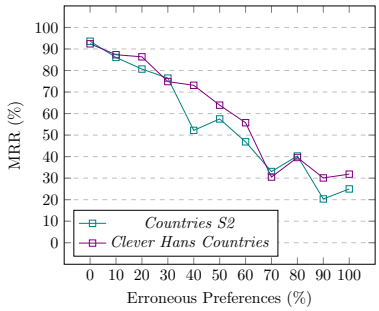


Figure 33: Performance of LiEr with increasing levels of erroneous preferences. LiEr maintains predictive performance up to 30% randomly incorrect feedback, though with an increase in variance.

Figure 33 shows the results. The robustness was tested on Countries S2 and Clever Hans Countries. For the test, a percentage of preference decisions was flipped at random, and LiEr's performance over 20 runs per noise level was measured. The KGC model retains most of its accuracy, up to 30% erroneous preferences. However, the variance increases. After this point, performance starts to degrade significantly.

The experiment shows that LiEr is resilient to a moderate level of feedback error. However, it also shows that its robustness depends on the task. As the task's complexity increases, as in Clever Hans Countries, the model becomes more sensitive to incorrect feedback. Additionally, the test assumes a random distribution of human labelling errors. In reality, human mistakes are biased. The type and severity of the bias will also influence model deterioration.

5.2.6 Discussion and Limitations

The results from the latter section show that LiEr is a viable alternative to conventional KGC models, especially when reasoning quality is as important as predictive accuracy. Using preference-based feedback over reasoning paths aligns LiEr with human judgment of valid reasoning. This makes it well-suited for domains where constraints on valid reasoning exist, due to legal, pedagogical, or domain-specific requirements, or where standard supervision is noisy or misaligned with the desired inference.

LiEr learns to align its predictions with human preference-based feedback over instance-based model explanations. This creates an interface between symbolic structure and human judgment. The model is steered via preferences over explanations. In this way, LiEr supports link prediction that can be adapted to reflect broader objectives, like legal validity or conceptual correctness in tutoring systems.

At the same time, the preference-based feedback strategy introduces new challenges. One challenge is the effort required to collect feedback from humans. LiEr depends on comparisons between reasoning paths. In our experiments, this feedback was based on manually defined ROIs. This allowed for efficient feedback simulation. However, this strategy only scales to domains where ROIs are available. In less structured domains, humans must provide feedback. LiEr

is a human-in-the-loop method by design. So this is intended. Still, collecting sufficient amounts of feedback from humans may be expensive or slow.

One direction for future work is automating parts of this process. For example, by prompting large language models to provide preferences over reasoning patterns. This make preference-based training more scalable. However, it is an open question how closely preferences expressed by a language model align with those of real users.

Another limitation lies in early-phase sensitivity. If the first feedback pairs are poorly selected or ambiguous, LiEr can be pushed toward invalid reasoning subspaces. From which it is difficult to recover. In practice, early-phase instability can be mitigated by introducing a warm-up phase. In the warm-up phase, the training of the reward is monitored. If no significant reduction in reward uncertainty can be observed after the warm-up phase, the run is reinitialized at a new seed. This reduced overall variance and improved robustness.

It can also be observed that the performance on real-world KGs improved when LiEr was pretrained with standard correctness-based supervision before switching to preference-based learning. This hybrid training strategy helped the model build an initial understanding of the KG structure, which was later refined through human-aligned preferences. The pretraining makes exploration during early feedback phases less noisy and more consistent, and leads to better results.

LiEr enables interactive training of a link prediction model that reflects the user's knowledge about the semantic concept underlying a missing link. It is most effective in settings where reasoning quality is a requirement.

5.2.7 LiEr and Research Question 3

In this thesis, the aim is to investigate whether link prediction models can be trained to predict missing facts by providing feedback on their explanations. Thereby, enforcing a model behaviour that aligns with human expectations. To address this, LiEr was introduced, a link prediction model that learns from preference-based feedback on model explanations.

Unlike existing approaches, LiEr does not treat reasoning as a byproduct of prediction. Instead, it makes reasoning itself, and thus the explanation, part of the training objective. Through an interface that allows users to express pairwise preferences over explanations, the model learns to give answers reached through valid reasoning steps.

LiEr was evaluated across seven benchmarks. The benchmarks reached from fully synthetic KGs to real-world, noisy, and structurally diverse KGs. LiEr shows that preference-based feedback on instance-based explanations provides a viable training signal. It achieves competitive predictive accuracy while producing reasoning paths that consistently align with human judgments of valid reasoning. On the Clever Hans Countries benchmark, where spurious but predictive patterns dominate the training set, LiEr was the only model to maintain predictive performance.

These findings provide an answer to the research question 3. It is possible to train a link prediction model with preference-based feedback over instance-based explanations to align its behaviour with human expectations. Preference-based feedback offers a viable mechanism to guide the model's behaviour according to what makes sense to humans.

At the same time, this work shows the challenges of preference-based human-in-the-loop KGC. For example, the feedback collection is expensive, and the training dynamics are more sensitive to early signals.

This points to future work. Pretraining on correctness signals, followed by preference-based refinement, could reduce variance. Moreover, lightweight interfaces and targeted comparisons could make human-in-the-loop training more feasible in practice. Also, automating the feedback loop using language models to simulate user judgments at scale could reduce the burden on human annotators. Another interesting idea is to apply LiEr to domains such as biomedical link prediction, where explanation quality is mission-critical.

The following section will pick up on one of those ideas and apply LiEr to the domain of RCA in manufacturing, where explanation quality is mission-critical.

5.3 Applying LiEr to Root Cause Analysis in Electric Vehicle Manufacturing

The previous evaluation established LiEr's capabilities on benchmarks KGs. The following evaluation turns to the real-world application of RCA in the production line of electric vehicles. It is shown how preference-based human-in-the-loop link prediction can be applied to manufacturing data, with the goal of identifying causes of product faults.[12]

Let us shortly revisit RCA. RCA is the task of identifying the underlying cause of a detected fault. In manufacturing, a typical workflow includes an end-of-line test where the product is evaluated across a range of quality metrics. At a large German vehicle manufacturer, faults occur sporadically at this stage. However, if a fault appears with notable frequency in a batch of manufactured units, it signals a possible issue in one of the upstream production steps. It is important to identify this step to restore product quality, prevent recurring faults, reduce scrap rates, and ensure the safety of drivers, passengers, and all traffic participants.

In practice, RCA is a highly manual process. When a fault occurs, a task force is assembled. The task force consists of process engineers, data scientists, and domain experts. Their analysis spans a wide range of tools. It includes static documentation in knowledge bases, process diagrams, logs of past faults, self-written scripts, and firsthand observations on the production floor. This approach is slow, resource-intensive, and difficult to scale.

We already presented the RootFinder framework in this thesis to make RCA data-driven. It supports process experts with the RCA and makes it less resource-intensive. However, a user study identified two limitations of RootFinder. The integration of new expert knowledge required manual rule updates and reconfiguration, and it is not robust to noise in the feedback or partial information. This limits its usefulness in production environments.

In this context, LiEr offers an alternative approach that addresses these limitations. It learns from expert feedback through preferences over model explanations. This enables experts to steer the model iteratively towards the correct model behaviour. Moreover, the preference-based reward structure makes

[12]This evaluation was largely conducted by Philipp Kosel as part of his Master's Thesis under close guidance and supervision by Christoph Wehner.

LiEr robust to noise, ambiguity, and conflicting signals, which are common in high-volume production lines with shifting operators.

In the following, we demonstrate how LiEr is applied to RCA with real-world data from the MP07 battery production line at a large German vehicle manufacturer. First, the data schema and graph construction are discussed. Next, minor adaptations to the LiEr architecture are described, which enable LiEr to consider numeric sensor inputs. Finally, the experimental results are presented, and we discuss the system's practical impact and limitations in a production environment.

5.3.1 Manufacturing Data as a Knowledge Graph

The production data is based on the MP07 manufacturing line at a large German vehicle manufacturer, located in Dingolfing. In this line, high-voltage battery systems used in electric vehicles are assembled through a largely automated process. The assembly process involves packing individual cells into modules, which are then combined to form a battery. Cooling components, insulation layers, and the battery management system are added to finalise the high-voltage battery system.

For that reason, the MP07 line consists of numerous processing steps, ranging from cell and module assembly to the integration of complete packs and end-of-line testing. The steps are carried out at stations equipped with sensor. These sensors record physical measurements during each subprocess. They record how individual battery packs are produced.

The line is represented as a KG to structure the symbolic data from the production process. The MP07 KG encodes the process flow, from assembly to testing, and captures where each measurement occurs. The KG representation provides a queryable abstraction of the manufacturing process. Moreover, it enables LiEr to trace faults from their detection to their origin.

The original MP07 KG consists of 1,584 entities and 7,401 relations (see Figure 34). For the purposes at hand, we reduce this to a smaller graph focused on numeric sensor data relevant to RCA. Less relevant entities, such as `App` or `Language`), are filtered out. Entities associated with variables, process steps and stations are kept. The filtered KG contains 375 entities of 6 types, connected by 488 relations from 5 relation types (see Tables 27 and 28).

Table 27: Entity types and their number of occurrences in the MP07 KG.

Entity Type	Count
`Line`	1
`ProcessStep (AssemblyStep)`	42
`ProcessStep (TestingStep)`	11
`SubProcessStep`	57
`Station`	88
`Variable`	176
Total	**375**

Table 28: Relation types and their number of occurrences in the MP07 KG.

Relation Type	Count
`implementsProcessStep`	53
`hasSubProcessStep`	57
`hasSuccessorProcessStep`	106
`occursInStation`	96
`isMeasuredAtSubProcessStep`	176
Total	**488**

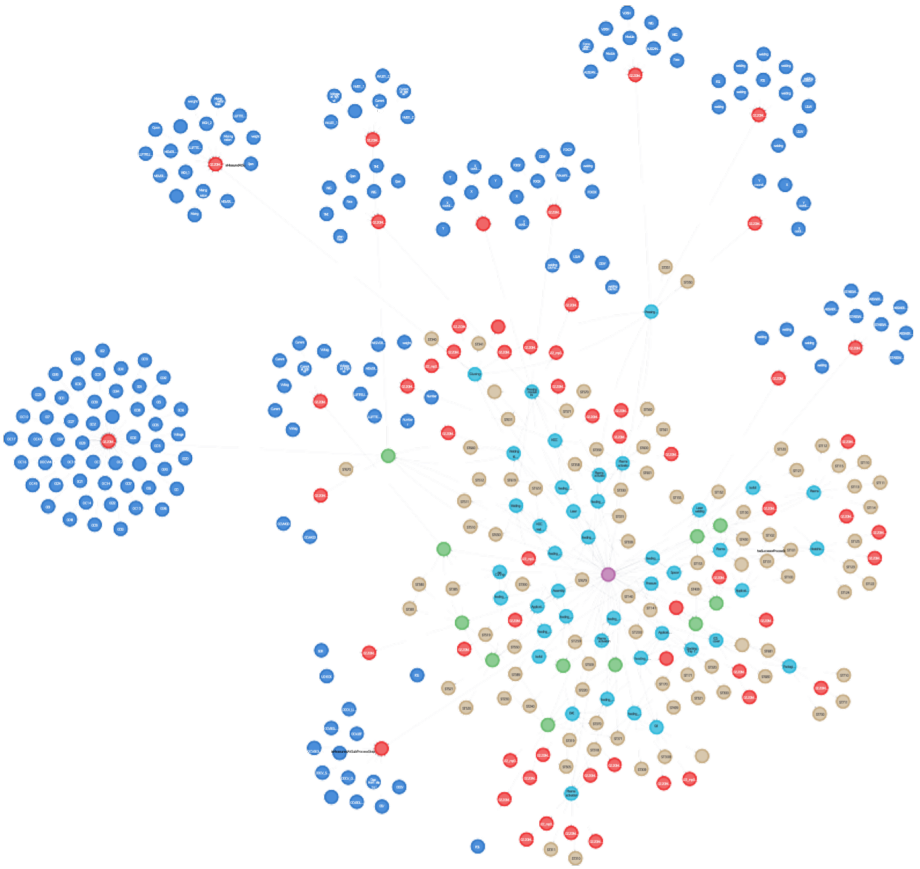


Figure 34: An excerpt of MP07 represented as a KG. Different types of entities are illustrated in different colors, i.e., `Line` (purple), `AssemblyStep` (turquoise), `TestingStep` (green), `SubProcessStep` (red), `Station` (tan), and `Variable` (blue).

Figure 35 shows the schema used in the following experiments. A `Line` has `ProcessSteps`. Each `ProcessStep` has `SubProcessSteps`. The `ProcessStep` nodes are connected to `Station` nodes through `occursInStation`. Numeric `Variable` nodes are linked to `SubProcessSteps` via `isMeasuredAtSubProcessStep`. Successor relations define temporal precedence between steps.

5.3.2 Synthetic Root Cause Data Generation

Manufacturing processes are designed to minimise faults. They operate at fault rates measured in parts per million. As a result, the dataset we had access to contained almost no real faults. We therefore introduce *synthetic faults* and their corresponding *root causes* into the MP07 KG.

After consultation with process experts, two main patterns were defined to model how synthetic faults and root causes propagate through the production line. All synthetic faults are located at `Variable` entities of `TestingSteps`. The associated root causes are placed either at `Variable` entities of `AssemblySteps` or at `Station` entities. This reflects real-world RCA hypotheses where a fault

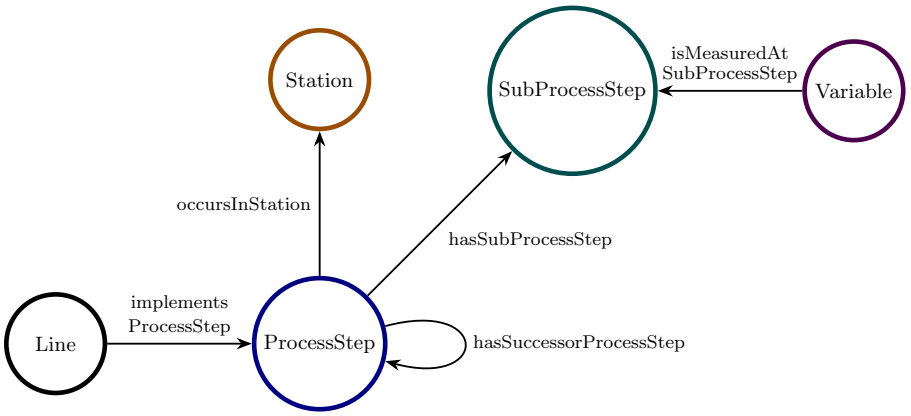


Figure 35: The schema of the production line represented as a KG.

originates in a sensor or at a physical workstation. Let us have a look at two example path patterns.

Path 1: Variable to Variable (5 hops). This path models faults whose root causes are traced to a sensor variable associated with an upstream assembly step. It consists of five hops and ends at a `Variable` entity:

$$\begin{aligned}
&\texttt{Variable} \xrightarrow{\texttt{isMeasuredAtSubProcessStep}} \texttt{SubProcessStep} \\
&\xrightarrow{\texttt{hasSubProcessStep}^{-1}} \texttt{ProcessStep} \\
&\xrightarrow{\texttt{hasSuccessorProcessStep}^{-1}} \texttt{ProcessStep} \\
&\xrightarrow{\texttt{hasSubProcessStep}} \texttt{SubProcessStep} \\
&\xrightarrow{\texttt{isMeasuredAtSubProcessStep}^{-1}} \texttt{Variable}
\end{aligned}$$

Path 2: Variable to Station (4 hops). This path captures faults that are traced to a station. It consists of four hops and ends at a `Station` entity:

$$\begin{aligned}
&\texttt{Variable} \xrightarrow{\texttt{isMeasuredAtSubProcessStep}} \texttt{SubProcessStep} \\
&\xrightarrow{\texttt{hasSubProcessStep}^{-1}} \texttt{ProcessStep} \\
&\xrightarrow{\texttt{hasSuccessorProcessStep}^{-1}} \texttt{ProcessStep} \\
&\xrightarrow{\texttt{occursInStation}} \texttt{Station}
\end{aligned}$$

Both paths incorporate the inverse of the `hasSuccessorProcessStep` relation. It ensures that all root causes temporally precede the observed fault and mirrors causal fault propagation in real-world RCA scenarios.

Root Cause Types. We define nine types of synthetic root causes, each for a different upstream root cause factor (e.g., welding quality, temperature, or humidity). Table 29 lists the types and their counts.

Finally, we create a perturbation in the numeric values of variables along error–root cause paths for each root cause relation. It makes the synthetic faults

Table 29: Types of synthetic root causes and their number of occurrences.

Root Cause Type	Count
`hasWeldingOkRootCause`	186
`hasLSLWRootCause`	186
`hasNumberOfStationRunsRootCause`	186
`hasTemperatureRootCause`	104
`hasOpenTimeRootCause`	104
`hasMixingRationShareRootCause`	156
`hasWeightRootCause`	156
`hasHumidityRootCause`	104
`hasMisVerAnteilRootCause`	156
Total	**1,338**

and root causes detectable in the sensor data. Specifically, we add twice the standard deviation to the affected sensor measurements. This creates outliers at positions corresponding to injected errors. The hypothesis is that LiEr can pick up on this signal and use it to distinguish between normal and faulty conditions, thereby guiding it to the correct root cause. These synthetic triples are used as ground truth links between faults and their causes.

An important aspect of this dataset is that the underlying manufacturing graph is static. It encodes the physical layout of the production line, its stations, process steps, and sensors. However, the synthetic root cause relations are dynamic. They represent failure cases that occur only for specific faulty batches.

This distinction introduces two constraints. First, because the root cause links are not part of the static production graph, LiEr cannot traverse them directly. It must infer them by reasoning over the manufacturing process and sensor data. Second, each root cause relation is tied to a unique measurement snapshot. These snapshots contain numeric data for 100 faulty parts with the physical sensor measurements that hold a fault and its cause. As noted before, we perturb the numerical values of the `Variable` nodes along each fault-to–cause path, to simulate observable effects. Specifically, we add two standard deviations to the affected sensor values. This creates detectable outliers. It enables LiEr to use sensor variation as a cue to traverse the inverse fault propagation path.

5.3.3 Measurement-Based Feature Construction

Each synthetic root cause relation in the KG is associated with a batch of 100 faulty parts. These batches serve as measurement snapshots. They represent a context in which a fault consistently appears. From each snapshot, we extract a set of features designed to enable LiEr to identify paths that reflect underlying faults.

Snapshot Transformation into Features. A feature vector ψ is computed for each node and each measurement snapshot. These vectors are part of the input representation seen by LiErs' policy network. We consolidated process experts to decide on the features to use. They advised us to use a combination of signal variability, fault likelihood, and constraint violations in a way that mirrors

how human experts interpret faults. For each node, the following features are calculated:

- **Standard deviation**: Measures the variability of a variable within the faulty batch. High values suggest instability of the variable.
- **Out-of-bounds ratio**: The ratio of measurements of a variable outside their defined upper or lower bounds. This feature indicates constraint violations. High value indicates a large number of constraint violations.
- **FaultStation/TotalStation ratio**: The frequency of faulty measurements relative to all measurements of all variables measured at that station. It represents the overall station reliability. A lower value suggests a better reliability.

All features are normalised across the snapshot to ensure comparability across nodes.

Adapting the State s. The next step is to enable LiEr to access the features at each node. To achieve this, we extend the state s, which is how the MDP formally represents the current state at each step of the path. In Section 5.1.2, s_i encodes the current node and the query. We extend the state as follows to account for the physical measurements at each node:

$$s_i = \big(e_i,\ \psi(e_i),\ e_\eta^q,\ \sigma_r^q\big)$$

where $\psi(e_i) \in \mathbb{R}^3$ is appended to the state. The feature vector $\psi(e_i)$ holds the standard deviation, out-of-bounds ratio, and FaultStation/TotalStation ratio of node e_i. This allows the policy network to incorporate numerical evidence from the faulty batch when deciding how to traverse the KG.

This approach enables LiEr to use features based on physical sensor measurements to guide its reasoning. Each measurement snapshot provides context-specific evidence, embedded at the node level, which supports LiEr in identifying root causes of faults.

This setup brings LiEr closer to how RCA is performed in practice. It tracks structure and links structural paths to measurement anomalies.

5.3.4 Evaluation

In this section, the evaluation of LiEr on the task of RCA for manufacturing data from the production line MP07 is presented. The goal is to demonstrate that LiEr's human-in-the-loop approach can be transferred to an industrial use case.

All experiments are implemented in PyTorch version 2.0 and executed on a workstation with an Intel i5-8400 CPU, NVIDIA RTX 4060 Ti, and 16 GB RAM. The hyperparameters are selected via grid search and are listed in Table 30.

Training Protocol. First, a stratified split of the 1,338 synthetic root cause triples into training, validation, and test sets is created. The training set comprises 1,070 triples, while the validation and test sets each contain 134 triples. At each step, LiEr receives a query of the form `(faulty_variable, root_cause, ?)`. A simulated oracle is queried for 10 preferences at each epoch for the first 12 epochs.

Table 30: Final hyperparameter configuration used for evaluation of LiEr on the MPO7 KG.

Hyperparameter	**Value**
Embedding size	64
LSTM hidden size	256
Dense layer size	128
Dropout (LSTM)	10%
Batch size	64
Learning rate	0.001
Rollouts per query (train/test)	20
Action dropout	10%
Number of Feedback	120
Number of Epochs	50
Number of Hops	5

Results. Table 31 shows the average performance across ten training runs. The model ranks correct root causes at the top in a significant number of cases. It achieves an MRR and Hits@1 of 43.75%. These results confirm that LiEr can learn to identify root causes from preference-based feedback.

Table 31: Evaluation results of LiEr for RCA. Results are averaged over 10 runs.

	Hits@1	MRR
LiEr	0.4375	0.4375

Additionally, we tracked the average reward over 50 epochs of training to assess learning behaviour. The largest improvement was observed in the first epochs, when the presence-based feedback was collected. Another observation is that the reward is unstable while additional feedback is collected. This indicates that the model requires some time to adapt to the new reward information. Once no additional feedback is added, an upward trend in the reward can be observed. Figure 36 shows overall a stable improvement over time. LiEr learns from preference-based feedback to identify root cause paths.

This evaluation shows that LiEr can be applied to industrial domains such as electric vehicle manufacturing. It is capable of using symbolic structure, sensor measurements and preference-based feedback to learn paths that correspond to root causes of faults. This supports its suitability for interactive root cause analysis under real-world constraints.

5.3.5 Limitations and Lessons Learned

The results show that LiEr can be applied to an industrial RCA problem. However, several limitations exist and point to further research directions.

In this section, we focused on establishing that LiEr is applicable to RCA in a real-world manufacturing setup. We did not perform a systematic comparison against alternative RCA approaches on the same dataset. A complete picture of LiErs' performance for RCA will require side–by–side evaluation with baselines

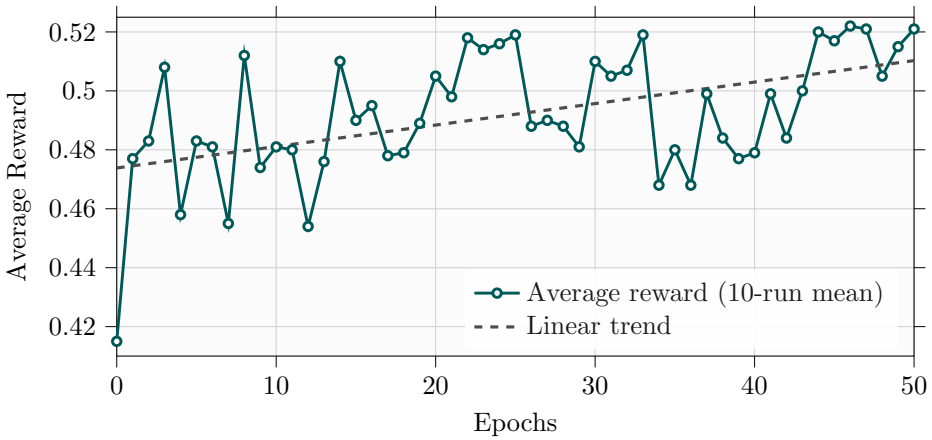


Figure 36: Rewards over 50 training epochs of LiEr on the MP07 KG, averaged over 10 runs. A least-squares linear trend (dashed) summarizes the trajectory.

representative of current practice, e.g., causal discovery pipelines tailored to manufacturing. However, this was not the scope of research question 3. Here we wanted to show that data-driven RCA can be iterative and interactive.

Additionally, the preference signal on MP07 for the evaluation is simulated via ROIs. A quantitative user study with process engineers is missing. Important open questions include annotation time per comparison, inter-rater agreement, how preference quality evolves as the interface proposes harder comparisons, and a more in-depth analysis of how quickly model performance improves as a function of labelled comparisons.

Furthermore, the achieved MRR of ≈ 0.44 indicates meaningful ranking ability but leaves room for improvements. A manual inspection of the results indicated at least two remaining error sources. First, near-ties among plausible upstream root cause variable candidates within the same process neighbourhood tend to result in the model choosing the wrong candidate. This could be addressed by experimenting with different node-level features (e.g., confidence on bounds, sensor health, maintenance logs) to break the tie in favour of the correct root cause variable. Secondly, LiEr struggles with long-range root cause paths. This issue could be addressed by improving the temporal context at each step and incorporating stronger regularizers in the policy to mitigate vanishing and exploding gradients.

Finally, the synthetic fault generation might be oversimplified. We inject root cause links for faulty batches and perturb numeric variables by 2 standard deviations along fault–cause paths. This creates learnable signals, but it underrepresents several real phenomena, such as correlated drifts across stations, intermittent sensor failures, and confounding from product variants. Future iterations should use more realistic values for physical measurements. For example, by incorporating time-dependent perturbations, physics- or control-limits to shape plausible faults, or collecting a large amount of retrospective fault incidents when available. This improves the quality of the evaluation.

Despite these limitations, the study provides a proof of concept. It showes how a manufacturing line of high-voltage batteries for electric vehicles can be

represented as a KG. It attaches measurement snapshots as node features and learns link prediction for RCA based on expert preferences.

5.3.6 Discussion of Applying LiEr to Root Cause Analysis

We presented LiEr to three process experts and data specialists at a large German vehicle manufacturer. The participants had previously used RootFinder. The feedback was collected in informal, think–aloud demo sessions.[13]

Participants appreciated that LiEr requires no rule writing or representation engineering, as was the case with RootFinder: "I don't need to encode a rule, I just pick the path that makes more sense." The preference interface was seen as cognitively light. The act of choosing between two explanations felt closer to the way engineers debate hypotheses on the shop floor. Additionally, preferences influenced path rankings within a few iterations. This creates a sense of steering the model. Participants described LiEr as "a good way to capture what we already know and make it reusable," especially across shifts or teams where senior experts are not always available.

A common critique was that LiEr entirely relies on human feedback. In practice, this means a novel fault still needs to be discovered through conventional means (e.g., alarms, dashboards, ad–hoc analysis) before LiEr can be trained to reproduce and propagate that knowledge. Thus, LiEr helps codify and disseminate expertise. However, it is not effective at discovering new failure modes on its own. A second concern was coverage. Engineers asked how many comparisons are "enough" to guarantee that a behaviour is installed in the model. Finally, a user requested richer temporal context in explanations (e.g., "Show me the time window where the station went out of bounds.").

The demonstration sessions suggest further research directions. A different unsupervised or weakly supervised approach for RCA could be used to discover root cause candidates. Next, the candidates are used to simulated feedback while pretraining LiEr. Finally, human preferences are used to align and fine-tune LiEr with user judgment in an iterative process. Additionally, various features should be explored to represent the measurement snapshots, such as those that capture trends, persistence, and co-movement of features. Finally, preferences should be logged and versioned, which helps to track expressed preference sets and make them shareable so that expertise becomes a portable asset.

Process experts valued LiEr as an efficient mechanism for capturing and sharing human expertise. It is a low-friction way to turn "this is the reasoning we trust" into a model that others can reuse. As such, LiEr fits into RCA workflows where expert time is scarce and alignment and transparency are critical.

5.4 Summary and Answer to Research Question 3

Let us recall Research Question 3: *How can user knowledge be incorporated into a link prediction model for iterative alignment with human intent?* This chapter addresses the question by introducing LiEr, a path-based link prediction method that learns to behave in alignment with preference-based feedback over its explanations.

The method treats path construction as a sequential decision process in the KG. It collects pairwise preferences between alternative reasoning paths and uses

[13]The sessions were in the German language, and all citations are translated freely.

them to train a reward function. The reward approximates the users‘ preference function and guides the policy towards explanations that reflect valid reasoning. In doing so, LiEr produces correct answers for the right reasons.

The link prediction task in KG is formulated as a traversal problem represented by an MDP. A recurrent policy selects relations step by step, conditioned on the query and history. Human preference-based feedback entered through a reward model trained to assign a higher cumulative reward to preferred paths. This introduced an inductive bias towards reasoning that aligns with human expectations.

Additionally, the chapter introduced MRLR alongside the standard MRR to evaluate reasoning quality. While MRR captured whether the correct entity was retrieved, MRLR measured how early a valid path appeared in the ranking. This allowed us to evaluate the quality of the models‘ decision-making behaviour.

Empirical results on, for example, Countries, Family, and Clever Hans Countries showed that LiEr performed on par with baselines in terms of prediction performance, and surpassed them in reasoning quality. It avoided shortcuts through spurious relations and instead preferred valid paths. This resulted in improved generalisation when spurious links were present at test time.

The method was then applied to RCA in a manufacturing setting. A process-level KG connected stations, process steps, and measurements. LiEr produced candidate root cause paths that engineers considered plausible and helpful. LiEr improved on RootFinder by not requiring hand-crafted rules, offered transparent reasoning, and faster iterative improvements over runs.

In summary, this chapter answered Research Question 3 by providing a proof of concept that user feedback over explanations can be operationalized through preferences over paths and translated into a learned reward. This reward shaped a policy that produces predictions supported by reasoning paths aligned with human judgements of valid reasoning within the KG domain. The approach is effective on synthetic benchmarks and an industrial use case. LiEr enables iterative alignment with human feedback in a practical and scalable way.

6 Conclusion

The research presented in this thesis is about exploring how explanations of KGC models can be used for model alignment with human feedback. The central idea of this thesis is that model explanations are not one-way messages, but can serve as a base for collecting and using human feedback to improve the model. Explanations allow experts to correct and refine the underlying decision processes of a KGC model. In the conclusion, the motivation is revisited, the research questions are summarised, and the findings are discussed. Finally, an outlook on future research directions is provided.

6.1 Summary

The motivation for this thesis comes from a practical and conceptual challenge. In industrial contexts such as RCA, link prediction models can identify potential root-cause relations among process variables, but often fail to use expert knowledge or justify their internal reasoning. This lack of human oversight limits its trust in data-driven RCA. In this thesis, we show how explanations can be used to make KGC models interactive, in the sense that they iteratively align with human feedback. This allows for the incorporation of user knowledge and leads to increased user trust and oversight. The work, therefore, focused on understanding how explanations can be used to iteratively guide a KGC model's decision-making process toward alignment with human feedback. The thesis was structured around three research questions.

Research Question 1: *How can we explain the decision-making process of a link prediction model, and how can this explanation assist a user in the downstream task of aligning the model with human intent?* This question was addressed by first discussing explanation modalities for KGC, particularly for the downstream task of model alignment. The takeaway here was that there is no one-size-fits-all solution. Different explainability modalities have different strengths and different drawbacks. For the downstream task of model alignment, it is best to combine modalities to give the user the most complete view of the KGC model's internal decision-making process. This need was addressed by developing KGEPrisma, a post-hoc, local explanation framework for embedding-based link prediction models. It translates the sub-symbolic representations learned by KGE models into human-readable representations, such as instance-based, analogy-based, and rule-based explanations. The evaluation demonstrated that these explanations can expose valid and spurious reasoning patterns, thereby forming a foundation for human interventions.

Research Question 2: *How effective are different types of feedback modalities in collecting user knowledge for the downstream task of aligning a link prediction model with human intent?* This question shifted the focus from explaining to interacting. The RootFinder system was introduced to explore how feedback modalities are perceived in the applied RCA domain. It kicked off a discussion on how different feedback modalities, ranging from positive and negative examples to logical rules, ratings, and preferences, affect model correction from a user perspective. The findings showed that the expressivity of the feedback mechanism determines its usefulness. While simple annotations guide local corrections, structured feedback, such as rules or preferences, captures domain knowledge at a more abstract level. Overall, preference-based feedback

was identified as the modality with the best balance. It scales through targeted uncertainty sampling, requires little model or domain knowledge, uses users' intuitive judgments, and has relatively low alignment risk.

Research Question 3: *How can user knowledge be incorporated into a link prediction model for iterative alignment with human intent?* We answered this question by introducing LiEr, a reinforcement learning model that treats link prediction as a sequential reasoning problem. LiEr reformulates the KG as a MDP and integrates preference-based feedback directly into the reward function. This allows the model to adapt its policy based on user evaluations of explanation validity. The proposed MRLR metric quantified the degree to which the model's reasoning paths align with valid reasoning patterns. Empirical results on standard benchmark KGs, a highly spurious KG, and applied to RCA confirmed that LiEr can iteratively improve predictive accuracy and reasoning validity through preference-based user feedback over model explanations.

Taken together, these findings contribute a methodological pathway from explanation to interaction and from interaction to alignment. In this thesis, we demonstrate that explanations can serve as interfaces for human feedback to iteratively align link prediction. The research lays the groundwork for interactive and explainable approaches to KGC. It enables KGC models to be right for the right reasons.

6.2 Discussion

This section reflects on the implications and limitations of the thesis. It discusses findings on explanation, feedback, and alignment in the KGC domain, and connects them to broader questions in XAI and alignment research. The work began with the challenge of explaining link prediction decisions. However, explanation alone is not sufficient. In high-stakes domains, users do not just want to know why a model made a prediction. They want to correct it when it is wrong. We showed that explanation can serve as an interface for feedback. The methodological progression from KGEPrisma to RootFinder and finally LiEr shows this shift from understanding to interaction, and from interaction to alignment.

One insight was that a single explanation modality alone is not sufficient for all alignment tasks. Instance-based explanations provide concrete examples but offer limited abstraction. Rule-based methods are scalable but require expertise. Saliency and counterfactuals tend to be unintuitive for non-technical or non-domain-expert users. The design of KGEPrisma addressed this challenge by providing complementary modalities within a single framework. In doing so, it allowed users to get a more complete picture of the model's internal decision-making process.

RootFinder served as a case study to learn that the usability of feedback modalities varies with user expertise and task complexity. A formal discussion concluded that preference-based feedback is particularly effective as a feedback modality. It reduces cognitive load, requires no formal rule-writing, and fits human judgment. However, it also introduces new challenges, which include ambiguity in early comparisons and the need for scalable preference collection.

Another contribution of the thesis is treating explanations of KGC models as a trainable objective. This is a common idea in AI alignment. Trustworthy systems must be right for the right reasons. LiEr uses this idea by aligning

its behavior with valid reasoning paths via human feedback. This enables the integration of domain knowledge and user expectations into the learning process of the KGC model itself.

Several limitations remain and were thoroughly discussed in the previous sections. For example, feedback collection requires effort and domain context. Simulated preferences via ROIs are not a substitute for real user interaction, and early in the learning process, mislocated feedback can lead to unstable learning dynamics.

Despite these limitations, the symbolic structure of KGs provides a viable foundation for model explanation. When combined with human feedback, it enables model alignment. We showed in this thesis that explanations are not simple mirrors of model behaviour. They can serve as a basis for aligning KGC models and making them more useful.

6.3 Open Potentials

This thesis introduces a conceptual and technical pipeline that uses the explanations of a KGC model to align it with human feedback. This final section outlines open potentials for future work. One direction is to apply the methods developed in this thesis to other high-stakes domains where explainability and valid reasoning are critical. For example, biomedical KGs support critical decision-making in drug repurposing or disease diagnosis. In such settings, reasoning chains should be aligned with established domain knowledge. LiEr, extended to biomedical KGs, could align predictions with expert judgment, supporting them in drug repurposing tasks.

Another research direction is automating the feedback loop. A major bottleneck is the collection of human feedback at scale. Large language models could mitigate this bottleneck by simulating human preferences over explanations, especially in common-knowledge domains with clear reasoning templates. This could make LiEr trainable without continuous access to human feedback. The question is how well LLM-based preferences approximate real and potentially diverse user feedback and how they can be integrated reliably.

Another idea is to explore multi-modal feedback. In practice, users want to formulate rules, provide positive examples, or express preference. One open challenge is learning from multiple feedback modalities simultaneously. Combining preferences, rules, and positive or negative examples within a unified learning framework could increase alignment accuracy while reducing user effort. This would require new training objectives and model architectures that can deal with heterogeneous feedback modalities.

More broadly, future research could explore generalising the alignment framework proposed in this thesis. While LiEr was designed as a specific instance of preference-based learning over sequential reasoning paths, the core idea of aligning a KGC model's internal decision-making process via explanation feedback would improve any KGC model. A general alignment framework could enable plug-and-play alignment strategies that are agnostic to the underlying KGC model.

Several other questions remain open. For example, how should competing user preferences be resolved when multiple users interact with the same model? Can alignment be versioned and shared as a resource, so that human knowledge becomes a transferable asset?

KGC should be able to reason in ways that reflect patterns in the data and also align with the knowledge and intent of their users. This thesis offers a first step towards that goal. Let us build KGC models that are explainable and interactive.

References

Abiteboul, S., Manolescu, I., Rigaux, P., Rousset, M.-C., and Senellart, P. (2011). Ontologies, RDF, and OWL. In *Web Data Management*, pages 143–170. Cambridge University Press.

Adadi, A. and Berrada, M. (2018). Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access*, 6:52138–52160.

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., and Kim, B. (2018). Sanity Checks for Saliency Maps. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Ahn, G., Hur, S., Shin, D., and Park, Y.-J. (2019). A Graphical Model to Diagnose Product Defects with Partially Shuffled Equipment Data. *Processes*, 7(12).

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pages 2623–2631, New York, NY, USA. Association for Computing Machinery. event-place: Anchorage, AK, USA.

Ali, M., Berrendorf, M., Hoyt, C., Vermue, L., Galkin, M., Sharifzadeh, S., Fischer, A., Tresp, V., and Lehmann, J. (2021). Bringing Light Into the Dark: A Large-scale Evaluation of Knowledge Graph Embedding Models under a Unified Framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–1.

Alvarez-Melis, D. and Jaakkola, T. S. (2018). On the Robustness of Interpretability Methods.

Anelli, V. W., Di Noia, T., Di Sciascio, E., Ragone, A., and Trotta, J. (2019). How to Make Latent Factors Interpretable by Feeding Factorization Machines with Knowledge Graphs. In Ghidini, C., Hartig, O., Maleshkova, M., Svátek, V., Cruz, I., Hogan, A., Song, J., Lefrançois, M., and Gandon, F., editors, *The Semantic Web – ISWC 2019*, volume 11778, pages 38–56. Springer International Publishing, Cham.

Assem, I., Skowronski, A., and Simson, D. (2006). *Elements of the Representation Theory of Associative Algebras: Techniques of Representation Theory*, volume 1 of *London Mathematical Society Student Texts*. Cambridge University Press.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). DBpedia: A Nucleus for a Web of Open Data. In Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., and Cudré-Mauroux, P., editors, *The Semantic Web*, pages 722–735, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer Normalization. Version Number: 1.

Baader, F. (2009). Description Logics. *Reasoning Web: Semantic Technologies for Information Systems, 5th International Summer School 2009*, 5689:1–39.

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. (2015). On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *PLOS ONE*, 10(7):1–46.

Bahr, L., Wehner, C., Wewerka, J., Bittencourt, J., Schmid, U., and Daub, R. (2025). Knowledge graph enhanced retrieval-augmented generation for failure mode and effects analysis. *Journal of Industrial Information Integration*, 45:100807. Publisher: Elsevier BV.

Balažević, I., Allen, C., and Hospedales, T. (2019). Multi-relational poincaré graph embeddings. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA. Curran Associates Inc.

Baltatzis, V. and Costabello, L. (2024). KGEx: Explaining Knowledge Graph Embeddings via Subgraph Sampling and Knowledge Distillation. In *Proceedings of the Second Learning on Graphs Conference*, pages 27:1–27:13. PMLR. ISSN: 2640-3498.

Barati, M., Bai, Q., and Liu, Q. (2016). SWARM: An Approach for Mining Semantic Association Rules from Semantic Web Data. In Booth, R. and Zhang, M.-L., editors, *PRICAI 2016: Trends in Artificial Intelligence*, pages 30–43, Cham. Springer International Publishing.

Bellet, A., Habrard, A., and Sebban, M. (2015). *Metric Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Springer International Publishing, Cham.

Bellman, R. (1957). A Markovian Decision Process. *Indiana University Mathematics Journal*, 6(4):679–684.

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. (2011). Algorithms for hyper-parameter optimization. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, NIPS'11, pages 2546–2554, Red Hook, NY, USA. Curran Associates Inc. event-place: Granada, Spain.

Besold, T. R., Akujuobi, U., Badreddine, S., Choi, J., Elshazly, H., Gifford, F., Iliopoulou, C., Maruyama, K., Nagano, K., Martin, P. S., and others (2025a). Literature-based Hypothesis Generation: Predicting the evolution of scientific literature to support scientists. In *AI4X 2025 International Conference*.

Besold, T. R., Akujuobi, U., Badreddine, S., Choi, J., Nishikawa, K., Wehner, C., Kumari, P., Hwang, H.-C., Elshazly, H. M. A. E., Iliopoulou, C. T., and Sanchez, P. (2025b). Method, apparatus, and computer-readable medium for generating a hypothesis from a knowledge graph.

Betz, P., Meilicke, C., and Stuckenschmidt, H. (2022). Adversarial Explanations for Knowledge Graph Embeddings. In Raedt, L. D., editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 2820–2826. International Joint Conferences on Artificial Intelligence Organization.

Bhardwaj, P., Kelleher, J., Costabello, L., and O'Sullivan, D. (2021). Adversarial Attacks on Knowledge Graph Embeddings via Instance Attribution Methods. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8225–8239, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bhowmik, R. and de Melo, G. (2020). Explainable Link Prediction for Emerging Entities in Knowledge Graphs. In Pan, J. Z., Tamma, V., d'Amato, C., Janowicz, K., Fu, B., Polleres, A., Seneviratne, O., and Kagal, L., editors, *The Semantic Web – ISWC 2020*, pages 39–55, Cham. Springer International Publishing.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. Association for Computing Machinery.

Bookstein, A., Kulyukin, V. A., and Raita, T. (2002). Generalized Hamming Distance. *Information Retrieval*, 5(4):353–375.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O. (2013). Translating Embeddings for Modeling Multi-relational Data. In Burges, C. J., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 26, pages 1–9. Curran Associates, Inc.

Bouchard, G., Singh, S., and Trouillon, T. (2015). On Approximate Reasoning Capabilities of Low-Rank Vector Spaces. In *Knowledge Representation and Reasoning: Integrating Symbolic and Neural Approaches, AAAI Spring Symposium Series*, pages 1–4.

Bradley, R. A. and Terry, M. E. (1952). Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika*, 39(3/4):324–345.

Brinkmann, R. (2008). *The Art and Science of Digital Compositing: Techniques for Visual Effects, Animation and Motion Graphics*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2 edition.

Bühlmann, P., Peters, J., and Ernest, J. (2014). CAM: Causal additive models, high-dimensional order search and penalized regression. *The Annals of Statistics*, 42(6).

Chalasani, P., Chen, J., Chowdhury, A. R., Wu, X., and Jha, S. (2020). Concise Explanations of Neural Networks using Adversarial Training. In *Proceedings*

of the 37th International Conference on Machine Learning, pages 1383–1391. PMLR.

Chandak, P., Huang, K., and Zitnik, M. (2023). Building a knowledge graph to enable precision medicine. *Scientific Data*, 10(1):67.

Chandrahas, ., Sengupta, T., Pragadeesh, C., and Talukdar, P. (2020). Inducing Interpretability in Knowledge Graph Embeddings. In Bhattacharyya, P., Sharma, D. M., and Sangal, R., editors, *Proceedings of the 17th International Conference on Natural Language Processing (ICON)*, pages 70–75, Indian Institute of Technology Patna, Patna, India. NLP Association of India (NLPAI).

Chang, C.-L. and Lee, R. C.-T. (1997). *Symbolic Logic and Mechanical Theorem Proving*. Academic Press, Inc., USA, 1st edition.

Chen, L., Jiang, S., Liu, J., Wang, C., Zhang, S., Xie, C., Liang, J., Xiao, Y., and Song, R. (2022). Rule mining over knowledge graphs via reinforcement learning. *Knowledge-Based Systems*, 242:108371.

Chen, Y., Goldberg, S., Wang, D. Z., and Johri, S. S. (2016). Ontological Pathfinding. In *Proceedings of the 2016 International Conference on Management of Data*, SIGMOD '16, pages 835–846, New York, NY, USA. Association for Computing Machinery.

Christian, B. (2020). *The Alignment Problem: Machine Learning and Human Values*. WW Norton.

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., and Amodei, D. (2017). Deep Reinforcement Learning from Human Preferences. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30, pages 1–9. Curran Associates, Inc.

Copi, I. M. (1954). Introduction to Logic. *Revue de Métaphysique et de Morale*, 59(3):344–345. Publisher: Presses Universitaires de France.

Das, R., Dhuliawala, S., Zaheer, M., Vilnis, L., Durugkar, I., Krishnamurthy, A., Smola, A., and McCallum, A. (2018). Go for a Walk and Arrive at the Answer: Reasoning Over Paths in Knowledge Bases using Reinforcement Learning. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, pages 1–18. OpenReview.net.

Das, R., Godbole, A., Naik, A., Tower, E., Zaheer, M., Hajishirzi, H., Jia, R., and Mccallum, A. (2022). Knowledge Base Question Answering by Case-based Reasoning over Subgraphs. In *Proceedings of the 39th International Conference on Machine Learning*, pages 4777–4793. PMLR. ISSN: 2640-3498.

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142. Publisher: IEEE.

Dettmers, T., Minervini, P., Stenetorp, P., and Riedel, S. (2018). Convolutional 2D Knowledge Graph Embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Dietterich, T. G. and others (2002). Ensemble learning. *The handbook of brain theory and neural networks*, 2(1):110–125.

Dwivedi, R., Dave, D., Naik, H., Singhal, S., Omer, R., Patel, P., Qian, B., Wen, Z., Shah, T., Morgan, G., and Ranjan, R. (2023). Explainable AI (XAI): Core Ideas, Techniques, and Solutions. *ACM Computing Surveys*, 55(9).

Eirich, J., Jäckle, D., Sedlmair, M., Wehner, C., Schmid, U., Bernard, J., and Schreck, T. (2023). ManuKnowVis: How to Support Different User Groups in Contextualizing and Leveraging Knowledge Repositories. *IEEE Transactions on Visualization and Computer Graphics*, 29(8):3441–3457. Publisher: Institute of Electrical and Electronics Engineers (IEEE).

Evans, M., Swartz, T., and Swartz, A. (2000). *Approximating Integrals Via Monte Carlo and Deterministic Methods*. Oxford statistical science series. Oxford University Press.

Fan, S.-K. S., Lin, S.-C., and Tsai, P.-F. (2016). Wafer fault detection and key step identification for semiconductor manufacturing using principal component analysis, AdaBoost and decision tree. *Journal of Industrial and Production Engineering*, 33(3):151–168.

Francis, N., Green, A., Guagliardo, P., Libkin, L., Lindaaker, T., Marsault, V., Plantikow, S., Rydberg, M., Selmer, P., and Taylor, A. (2018). Cypher: An Evolving Query Language for Property Graphs. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, pages 1433–1445, New York, NY, USA. Association for Computing Machinery.

Fuhr, N. (2018). Some Common Mistakes In IR Evaluation, And How They Can Be Avoided. *SIGIR Forum*, 51(3):32–41. Place: New York, NY, USA Publisher: Association for Computing Machinery.

Gabriel, I. (2020). Artificial Intelligence, Values, and Alignment. *Minds and Machines*, 30(3):411–437.

Gallian, J. (2000). A Dynamic Survey of Graph Labeling. *Electronic Journal of Combinatorics, Dynamic Surveys*, 19.

Galárraga, L., Teflioudi, C., Hose, K., and Suchanek, F. M. (2015). Fast rule mining in ontological knowledge bases with AMIE+. *The VLDB Journal*, 24(6):707–730.

Goodfellow, I. J., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press, Cambridge, MA, USA.

Guan, L., Verma, M., Guo, S., Zhang, R., and Kambhampati, S. (2020). Widening the Pipeline in Human-Guided Reinforcement Learning with Explanation and Context-Aware Data Augmentation. In *Neural Information Processing Systems*.

Guidotti, R. (2022). Counterfactual explanations and how to find them: literature review and benchmarking. *Data Mining and Knowledge Discovery*.

Gusmão, A. C., Correia, A. H. C., Bona, G. D., and Cozman, F. G. (2018). Interpreting Embedding Models of Knowledge Bases: A Pedagogical Approach. *ICML Workshop on Human Interpretability in Machine Learning (WHI). Stockholm, Sweden*.

Hammersley, J. (2013). *Monte Carlo Methods*. Monographs on Statistics and Applied Probability. Springer Netherlands.

Hanawa, K., Yokoi, S., Hara, S., and Inui, K. (2021). Evaluation of Similarity-based Explanations. In *International Conference on Learning Representations*.

Harary, F., Norman, R. Z. R. Z., and Cartwright, D. (1965). *Structural models: an introduction to the theory of directed graphs*. Wiley.

Hedström, A., Weber, L., Krakowczyk, D., Bareeva, D., Motzkus, F., Samek, W., Lapuschkin, S., and Höhne, M. M.-C. (2023). Quantus: An Explainable AI Toolkit for Responsible Evaluation of Neural Network Explanations and Beyond. *Journal of Machine Learning Research*, 24(34):1–11.

Heidrich, L., Slany, E., Scheele, S., and Schmid, U. (2023). FairCaipi: A Combination of Explanatory Interactive and Fair Machine Learning for Human and Machine Bias Reduction. *Machine Learning and Knowledge Extraction*, 5(4):1519–1538.

Ho, V. T., Stepanova, D., Gad-Elrab, M. H., Kharlamov, E., and Weikum, G. (2018). Rule Learning from Knowledge Graphs Guided by Embedding Models. In Vrandečić, D., Bontcheva, K., Suárez-Figueroa, M. C., Presutti, V., Celino, I., Sabou, M., Kaffee, L.-A., and Simperl, E., editors, *The Semantic Web – ISWC 2018*, pages 72–90, Cham. Springer International Publishing.

Hogan, A., Blomqvist, E., Cochez, M., D'amato, C., Melo, G. D., Gutierrez, C., Kirrane, S., Gayo, J. E. L., Navigli, R., Neumaier, S., Ngomo, A.-C. N., Polleres, A., Rashid, S. M., Rula, A., Schmelzeisen, L., Sequeda, J., Staab, S., and Zimmermann, A. (2022). Knowledge Graphs. *ACM Computing Surveys*, 54(4):1–37.

Holzinger, A. (2019). Interpretierbare KI: Künstliche Intelligenz verstehen können.

Hoseinnia, F., Ghatee, M., and Chehreghani, M. H. (2025). Mitigating oversmoothing in Graph Neural Networks for node classification through Adaptive Early Embedding and Biased DropEdge procedures. *Knowledge-Based Systems*, 320:113615.

Hou, Z., Jin, X., Li, Z., and Bai, L. (2021). Rule-aware reinforcement learning for knowledge graph reasoning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4687–4692.

Huang, Q., Yamada, M., Tian, Y., Singh, D., and Chang, Y. (2023). GraphLIME: Local Interpretable Model Explanations for Graph Neural Networks. *IEEE Transactions on Knowledge and Data Engineering*, 35(7):6968–6972.

Hüllermeier, E. and Waegeman, W. (2021). Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning*, 110(3):457–506.

Ilievski, F., Hammer, B., van Harmelen, F., Paassen, B., Saralajew, S., Schmid, U., Biehl, M., Bolognesi, M., Dong, X. L., Gashteovski, K., Hitzler, P., Marra, G., Minervini, P., Mundt, M., Ngomo, A.-C. N., Oltramari, A., Pasi, G., Saribatur, Z. G., Serafini, L., Shawe-Taylor, J., Shwartz, V., Skitalinskaya, G., Stachl, C., van de Ven, G. M., and Villmann, T. (2025). Aligning generalization between humans and machines. *Nature Machine Intelligence*, 7(9):1378–1389. Publisher: Nature Publishing Group.

Ismaeil, Y., Stepanova, D., Tran, T.-K., and Blockeel, H. (2023). FeaBI: A Feature Selection-Based Framework for Interpreting KG Embeddings. In Payne, T. R., Presutti, V., Qi, G., Poveda-Villalón, M., Stoilos, G., Hollink, L., Kaoudi, Z., Cheng, G., and Li, J., editors, *The Semantic Web – ISWC 2023*, pages 599–617, Cham. Springer Nature Switzerland.

ISO/IEC (2019). Risk management — Risk assessment techniques. Standard, International Organization for Standardization, Geneva, CH.

Ji, S., Pan, S., Cambria, E., Marttinen, P., and Yu, P. S. (2022). A Survey on Knowledge Graphs: Representation, Acquisition, and Applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):494–514.

Kemp, C., Tenenbaum, J. B., Griffiths, T. L., Yamada, T., and Ueda, N. (2006). Learning systems of concepts with an infinite relational model. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, AAAI'06, pages 381–388, Boston, Massachusetts, USA. AAAI Press. event-place: Boston, Massachusetts.

Kertel, M., Harmeling, S., Pauly, M., and Klein, N. (2023). Learning Causal Graphs in Manufacturing Domains Using Structural Equation Models. *International Journal of Semantic Computing*, 17(04):511–528.

Kiefer, S., Hoffmann, M., and Schmid, U. (2022). Semantic Interactive Learning for Text Classification: A Constructive Approach for Contextual Interactions. *Machine Learning and Knowledge Extraction*, 4(4):994–1010.

Kingma, D. P. and Ba, J. (2015). Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Kirchhof, M., Haas, K., Kornas, T., Thiede, S., Hirz, M., and Hermann, C. (2020). Root Cause Analysis in Lithium-Ion Battery Production with FMEA-Based Large-Scale Bayesian Network.

Knox, W. B. and Stone, P. (2011). Augmenting Reinforcement Learning with Human Feedback. In *ICML 2011 Workshop on New Developments in Imitation Learning*.

Kobialka, H.-U. (2022). AI based root cause analysis ≫ Lamarr Institute.

Lahav, O., Mastronarde, N., and Schaar, M. v. d. (2018). What is Interpretable? Using Machine Learning to Design Interpretable Decision-Support Systems. *Machine Learning for Health (ML4H) Workshop at NeurIPS 2018*, abs/1811.10799.

Lal, M. (2015). *Neo4j Graph Data Modeling*. Packt Publishing.

Lao, N., Mitchell, T., and Cohen, W. W. (2011). Random Walk Inference and Learning in A Large Scale Knowledge Base. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 529–539. Association for Computational Linguistics.

Lapuschkin, S., Wäldchen, S., Binder, A., Montavon, G., Samek, W., and Müller, K.-R. (2019). Unmasking Clever Hans predictors and assessing what machines really learn. *Nature Communications*, 10(1):1096.

Lee, K., Smith, L. M., and Abbeel, P. (2021). PEBBLE: Feedback-Efficient Interactive Reinforcement Learning via Relabeling Experience and Unsupervised Pre-training. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, pages 6152–6163.

Lin, X. V., Socher, R., and Xiong, C. (2018). Multi-Hop Knowledge Graph Reasoning with Reward Shaping. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3243–3253, Brussels, Belgium. Association for Computational Linguistics.

Lokrantz, A., Gustavsson, E., and Jirstrand, M. (2018). Root cause analysis of failures and quality deviations in manufacturing using machine learning. *Procedia CIRP*, 72:1057–1062.

Lundberg, S. M. and Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 4768–4777, Red Hook, NY, USA. Curran Associates Inc.

Ma, Q., Li, H., and Thorstenson, A. (2021). A big data-driven root cause analysis system: Application of Machine Learning in quality problem solving. *Computers and Industrial Engineering*, 160:107580.

Maaten, L. v. d. and Hinton, G. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605.

Marconato, E., Teso, S., Vergari, A., and Passerini, A. (2023). Not all neuro-symbolic concepts are created equal: analysis and mitigation of reasoning shortcuts. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NeurIPS '23, Red Hook, NY, USA. Curran Associates Inc. event-place: New Orleans, LA, USA.

Meilicke, C., Chekol, M. W., Fink, M., and Stuckenschmidt, H. (2020). Reinforced Anytime Bottom Up Rule Learning for Knowledge Graph Completion. *ArXiv*, abs/2004.04412.

Meilicke, C., Chekol, M. W., Ruffinelli, D., and Stuckenschmidt, H. (2019). Anytime Bottom-Up Rule Learning for Knowledge Graph Completion. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 3137–3143. International Joint Conferences on Artificial Intelligence Organization.

Miller, G. A. (1995). WordNet: A Lexical Database for English. *Communication ACM*, 38(11):39–41.

Mitchell, T., Cohen, W., Hruschka, E., Talukdar, P., Yang, B., Betteridge, J., Carlson, A., Dalvi, B., Gardner, M., Kisiel, B., Krishnamurthy, J., Lao, N., Mazaitis, K., Mohamed, T., Nakashole, N., Platanios, E., Ritter, A., Samadi, M., Settles, B., Wang, R., Wijaya, D., Gupta, A., Chen, X., Saparov, A., Greaves, M., and Welling, J. (2018). Never-ending learning. *Commun. ACM*, 61(5):103–115. Place: New York, NY, USA Publisher: Association for Computing Machinery.

Molnar, C. (2019). *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. https://christophm.github.io/interpretable-ml-book/.

Montavon, G., Binder, A., Lapuschkin, S., Samek, W., and Müller, K.-R. (2019). Layer-Wise Relevance Propagation: An Overview. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 193–209. Springer International Publishing, Cham.

Moser, K. S. (2017). The Influence of Feedback and Expert Status in Knowledge Sharing Dilemmas. *Applied Psychology*, 66(4):674–709.

Mosqueira-Rey, E., Hernández-Pereira, E., Alonso-Ríos, D., Bobes-Bascarán, J., and Fernández-Leal, A. (2023). Human-in-the-loop machine learning: a state of the art. *Artificial Intelligence Review*, 56(4):3005–3054.

Muggleton, S. (1991). Inductive logic programming. *New Generation Computing*, 8(4):295–318.

Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., and Yu, B. (2019). Definitions, Methods, and Applications in Interpretable Machine Learning. *Proceedings of the National Academy of Sciences*, 116(44):22071–22080.

Najar, A. and Chetouani, M. (2021). Reinforcement Learning With Human Advice: A Survey. *Frontiers in Robotics and AI*, 8.

Nandwani, Y., Gupta, A., Agrawal, A., Chauhan, M. S., Singla, P., and Mausam (2020). OxKBC: Outcome Explanation for Factorization Based Knowledge Base Completion. In *Automated Knowledge Base Construction*.

Nembrini, S., König, I. R., and Wright, M. N. (2018). The revival of the Gini importance? *Bioinformatics*, 34(21):3711–3718. _eprint: https://academic.oup.com/bioinformatics/article-pdf/34/21/3711/48920785/bioinformatics_34_21_3711.pdf.

Nickel, M., Tresp, V., and Kriegel, H.-P. (2011). A Three-Way Model for Collective Learning on Multi-Relational Data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 809–816, Madison, WI, USA. Omnipress.

Oliveira, E. e., Miguéis, V. L., and Borges, J. L. (2022). Automatic root cause analysis in manufacturing: an overview and conceptualization. *Journal of Intelligent Manufacturing*.

Omran, P. G., Wang, K., and Wang, Z. (2018). Scalable Rule Learning via Learning Representation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, pages 2149–2155. AAAI Press.

Ortega, P. A. and Maini, V. (2018). Building safe artificial intelligence: specification, robustness, and assurance.

Ott, S., Meilicke, C., and Samwald, M. (2021). SAFRAN: An interpretable, rule-based link prediction method outperforming embedding models. In *3rd Conference on Automated Knowledge Base Construction*, pages 1–18.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. (2022). Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NeurIPS '22, Red Hook, NY, USA. Curran Associates Inc. event-place: New Orleans, LA, USA.

Parkhi, O. M., Vedaldi, A., Zisserman, A., and Jawahar, C. V. (2012). Cats and dogs. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3498–3505.

Pezeshkpour, P., Tian, Y., and Singh, S. (2019). Investigating Robustness and Interpretability of Link Prediction via Adversarial Modifications. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3336–3347, Minneapolis, Minnesota. Association for Computational Linguistics.

Phillips, P. J., Hahn, C., Fontana, P., Yates, A., Greene, K. K., Broniatowski, D., and Przybocki, M. A. (2021). Four Principles of Explainable Artificial Intelligence.

Polleti, G. and Cozman, F. (2019). Faithfully Explaining Predictions of Knowledge Embeddings. In *Anais do XVI Encontro Nacional de Inteligência Artificial e Computacional*, pages 892–903, Porto Alegre, RS, Brasil. SBC.

Poßner, L., Bahr, L., Röhl, L., Wehner, C., and Gröger, S. (2025). Anwendung von Causal-Discovery-Algorithmen zur Root-Cause-Analyse in der Fahrzeugmontage. In *Rethinking Quality - Wandel des Qualitätsmanagements durch Digitalisierung und Künstliche Intelligenz*, pages 39–59. Springer Fachmedien Wiesbaden, Wiesbaden.

Pradhan, S., Singh, R., Kachru, K., and Narasimhamurthy, S. (2007). A Bayesian Network Based Approach for Root-Cause-Analysis in Manufacturing Process. In *2007 International Conference on Computational Intelligence and Security (CIS 2007)*, pages 10–14.

Prud'hommeaux, E., Harris, S., and Seaborne, A. (2013). SPARQL 1.1 Query Language. Technical report, W3C.

Pérez, J., Arenas, M., and Gutierrez, C. (2006). Semantics and Complexity of SPARQL. In Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., and Aroyo, L. M., editors, *The Semantic Web - ISWC 2006*, pages 30–43, Berlin, Heidelberg. Springer Berlin Heidelberg.

Quinlan, J. R. (1990). Learning Logical Definitions from Relations. *Machine Learning*, 5(3):239–266.

Rabold, J., Siebers, M., and Schmid, U. (2018). Explaining Black-Box Classifiers with ILP – Empowering LIME with Aleph to Approximate Non-linear Decisions with Relational Rules. In Riguzzi, F., Bellodi, E., and Zese, R., editors, *Inductive Logic Programming*, pages 105–117, Cham. Springer International Publishing.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). "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*, KDD '16, pages 1135–1144, New York, NY, USA. Association for Computing Machinery.

Rocktäschel, T. and Riedel, S. (2017). End-to-end Differentiable Proving. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30, pages 1–15. Curran Associates, Inc.

Ross, A. S., Hughes, M. C., and Doshi-Velez, F. (2017). Right for the Right Reasons: Training Differentiable Models by Constraining their Explanations. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2662–2670.

Rossi, A., Barbosa, D., Firmani, D., Matinata, A., and Merialdo, P. (2021). Knowledge Graph Embedding for Link Prediction: A Comparative Analysis. *ACM Transactions on Knowledge Discovery from Data*, 15(2).

Rossi, A., Firmani, D., Merialdo, P., and Teofili, T. (2022a). Explaining Link Prediction Systems based on Knowledge Graph Embeddings. In *Proceedings of the 2022 International Conference on Management of Data*, SIGMOD '22, pages 2062–2075, New York, NY, USA. Association for Computing Machinery.

Rossi, A., Firmani, D., Merialdo, P., and Teofili, T. (2022b). Kelpie: an explainability framework for embedding-based link prediction models. *Proceedings of the VLDB Endowment*, 15(12):3566–3569.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Ruijters, E. and Stoelinga, M. (2015). Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools. *Computer Science Review*, 15-16:29–62.

Ruschel, A., Gusmão, A. C., and Cozman, F. G. (2024). Explaining answers generated by knowledge graph embeddings. *International Journal of Approximate Reasoning*, page 109183.

Sadeghian, A., Armandpour, M., Ding, P., and Wang, D. Z. (2019). DRUM: End-to-End Differentiable Rule Mining on Knowledge Graphs. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 1–11, Red Hook, NY, USA. Curran Associates Inc.

Samek, W., Binder, A., Montavon, G., Lapuschkin, S., and Müller, K.-R. (2017). Evaluating the Visualization of What a Deep Neural Network Has Learned. *IEEE Transactions on Neural Networks and Learning Systems*, 28(11):2660–2673.

Schnake, T., Eberle, O., Lederer, J., Nakajima, S., Schütt, K. T., Müller, K.-R., and Montavon, G. (2022). Higher-Order Explanations of Graph Neural Networks via Relevant Walks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7581–7596.

Schneider, E. W. (1973). *Course Modularization Applied: The Interface System and Its Implications For Sequence Control and Data Analysis*. Distributed by ERIC Clearinghouse.

Schramm, S., Wehner, C., and Schmid, U. (2023). Comprehensible Artificial Intelligence on Knowledge Graphs: A survey. *Journal of Web Semantics*, 79:100806. Publisher: Elsevier BV.

Schramowski, P., Stammer, W., Teso, S., Brugger, A., Herbert, F., Shao, X., Luigs, H.-G., Mahlein, A.-K., and Kersting, K. (2020). Making deep neural networks right for the right scientific reasons by interacting with their explanations. *Nature Machine Intelligence*, 2(8):476–486.

Schwalbe, G. and Finzel, B. (2023). A comprehensive taxonomy for explainable artificial intelligence: a systematic survey of surveys on methods and concepts. *Data Mining and Knowledge Discovery*.

Seidel, A. and von Falkenhausen, L. (2008). The Emperor and His Councillor Laozi and Han Dynasty Taoism. *Cahiers d'Extrême-Asie*, 17:125–165. Publisher: École française d'Extrême-Orient.

Serrat, O. (2017). The Five Whys Technique. In *Knowledge Solutions: Tools, Methods, and Approaches to Drive Organizational Performance*, pages 307–310. Springer Singapore, Singapore.

Shen, Y., Chen, J., Huang, P.-S., Guo, Y., and Gao, J. (2018). M-Walk: Learning to Walk over Graphs using Monte Carlo Tree Search. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31, pages 0–1. Curran Associates, Inc.

Shrikumar, A., Greenside, P., and Kundaje, A. (2017). Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 3145–3153. JMLR.org.

Singhal, A. (2012). Introducing the Knowledge Graph: things, not strings.

Slany, E., Ott, Y., Scheele, S., Paulus, J., and Schmid, U. (2022). CAIPI in Practice: Towards Explainable Interactive Medical Image Classification. In Maglogiannis, I., Iliadis, L., Macintyre, J., and Cortez, P., editors, *Artificial Intelligence Applications and Innovations. AIAI 2022 IFIP WG 12.5 International Workshops*, pages 389–400, Cham. Springer International Publishing.

Slany, E., Scheele, S., and Schmid, U. (2024a). Bayesian CAIPI: A Probabilistic Approach to Explanatory and Interactive Machine Learning. In Nowaczyk, S., Biecek, P., Chung, N. C., Vallati, M., Skruch, P., Jaworek-Korjakowska, J., Parkinson, S., Nikitas, A., Atzmüller, M., Kliegr, T., Schmid, U., Bobek, S., Lavrac, N., Peeters, M., van Dierendonck, R., Robben, S., Mercier-Laurent, E., Kayakutlu, G., Owoc, M. L., Mason, K., Wahid, A., Bruno, P., Calimeri, F., Cauteruccio, F., Terracina, G., Wolter, D., Leidner, J. L., Kohlhase, M., and Dimitrova, V., editors, *Artificial Intelligence. ECAI 2023 International Workshops*, pages 285–301, Cham. Springer Nature Switzerland.

Slany, E., Scheele, S., and Schmid, U. (2024b). Explanatory Interactive Machine Learning with Counterexamples from Constrained Large Language Models. In Hotho, A. and Rudolph, S., editors, *KI 2024: Advances in Artificial Intelligence*, pages 324–331, Cham. Springer Nature Switzerland.

Slany, E., Scheele, S., and Schmid, U. (2024c). Hybrid Explanatory Interactive Machine Learning for Medical Diagnosis. In Maglogiannis, I., Iliadis, L., Macintyre, J., Avlonitis, M., and Papaleonidas, A., editors, *Artificial Intelligence Applications and Innovations*, pages 105–116, Cham. Springer Nature Switzerland.

Song, W., Duan, Z., Yang, Z., Zhu, H., Zhang, M., and Tang, J. (2019). Ekar: Explainable Knowledge Graph-based Recommendation via Deep Reinforcement Learning. *ArXiv*, abs/1906.09506.

Stadelmaier, J. and Padó, S. (2019). Modeling Paths for Explainable Knowledge Base Completion. In Linzen, T., Chrupała, G., Belinkov, Y., and Hupkes, D., editors, *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 147–157, Florence, Italy. Association for Computational Linguistics.

Steenwinckel, B., Vandewiele, G., Weyns, M., Agozzino, T., Turck, F. D., and Ongenae, F. (2022). INK: knowledge graph embeddings for node classification. *Data Mining and Knowledge Discovery*, 36(2):620–667.

Stenning, K. and Van Lambalgen, M. (2008). *Human Reasoning and Cognitive Science*. The MIT Press.

Sun, Z., Deng, Z.-H., Nie, J.-Y., and Tang, J. (2019). RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Sundararajan, M., View Profile, Taly, A., View Profile, Yan, Q., and View Profile (2017). Axiomatic attribution for deep networks. *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, pages 3319–3328.

Tay, K. M. and Lim, C. P. (2008). On the use of fuzzy inference techniques in assessment models: part II: industrial applications. *Fuzzy Optimization and Decision Making*, 7(3):283–302.

Teso, S. and Kersting, K. (2019). Explanatory Interactive Machine Learning. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '19, pages 239–245, New York, NY, USA. Association for Computing Machinery.

Thorndike, R. L. (1953). Who Belongs in the Family? *Psychometrika*, 18(4):267–276.

Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P., and Gamon, M. (2015). Representing Text for Joint Embedding of Text and Knowledge Bases. In Màrquez, L., Callison-Burch, C., and Su, J., editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Lisbon, Portugal. Association for Computational Linguistics.

Trouillon, T., Welbl, J., Riedel, S., Gaussier, E., and Bouchard, G. (2016). Complex Embeddings for Simple Link Prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 2071–2080. JMLR.org.

Turek, M. (2018). Explainable artificial intelligence (XAI). *Defense Advanced Research Projects Agency. http://web. archive. org/web/20190728055815/https://www. darpa. mil/program/explainable-artificial-intelligence*.

Ullman, J. B. and Bentler, P. M. (2012). Structural Equation Modeling. In *Handbook of Psychology, Second Edition*. Wiley, 1 edition.

Von Neumann, J. and Morgenstern, O. (2007). *Theory of Games and Economic Behavior (60th Anniversary Commemorative Edition):*. Princeton University Press.

Vu, M. and Thai, M. T. (2020). PGM-Explainer: Probabilistic Graphical Model Explanations for Graph Neural Networks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12225–12235. Curran Associates, Inc.

Wan, G., Pan, S., Gong, C., Zhou, C., and Haffari, G. (2020). Reasoning Like Human: Hierarchical Reinforcement Learning for Knowledge Graph Reasoning. In Bessiere, C., editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 1926–1932. International Joint Conferences on Artificial Intelligence Organization.

Wang, X., He, X., Feng, F., Nie, L., and Chua, T.-S. (2018). TEM: Tree-enhanced Embedding Model for Explainable Recommendation. In Champin, P.-A., Gandon, F., Lalmas, M., and Ipeirotis, P. G., editors, *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1543–1552. ACM.

Wang, Z. and Li, J.-Z. (2015). RDF2Rules: Learning Rules from RDF Knowledge Bases by Mining Frequent Predicate Cycles. *CoRR*, abs/1512.07734.

Wehner, C., Bahr, L., Voigt, E., Wagner, J., Wagner, J., and Schmid, U. (2025). Aligning Path based Link Prediction with Human Understanding of Valid Reasoning.

Wehner, C., Iliopoulou, C., Schmid, U., and Besold, T. R. (2024). From Latent to Lucid: Transforming Knowledge Graph Embeddings into Interpretable Structures with KGEPrisma. Version Number: 2.

Wehner, C., Kertel, M., and Wewerka, J. (2023). Interactive and Intelligent Root Cause Analysis in Manufacturing with Causal Bayesian Networks and Knowledge Graphs. In *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*, pages 1–7.

Wehner, C., Powlesland, F., Altakrouri, B., and Schmid, U. (2022). Explainable Online Lane Change Predictions on a Digital Twin with a Layer Normalized LSTM and Layer-wise Relevance Propagation. In Fujita, H., Fournier-Viger, P., Ali, M., and Wang, Y., editors, *Advances and Trends in Artificial Intelligence. Theory and Practices in Artificial Intelligence*, pages 621–632, Cham. Springer International Publishing.

Wenzlhuemer, R. (2009). Counterfactual Thinking as a Scientific Method. *Historical Social Research / Historische Sozialforschung*, 34(2 (128)):27–54. Publisher: GESIS - Leibniz Institute for the Social Sciences.

Wiener, N. (1960). Some Moral and Technical Consequences of Automation. *Science*, 131(3410):1355–1358.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256.

Xian, Y., Fu, Z., Muthukrishnan, S., de Melo, G., and Zhang, Y. (2019). Reinforcement Knowledge Graph Reasoning for Explainable Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, pages 285–294, New York, NY, USA. Association for Computing Machinery.

Xie, Q., Ma, X., Dai, Z., and Hovy, E. (2017). An Interpretable Knowledge Transfer Model for Knowledge Base Completion. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 950–962, Vancouver, Canada. Association for Computational Linguistics.

Xiong, W., Hoang, T., and Wang, W. Y. (2017). DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 564–573, Copenhagen, Denmark. Association for Computational Linguistics.

Yamada, M., Jitkrittum, W., Sigal, L., Xing, E. P., and Sugiyama, M. (2014). High-dimensional feature selection by feature-wise kernelized lasso. *Neural computation*, 26(1):185–207. Publisher: MIT Press.

Yamada, M., Tang, J., Lugo-Martinez, J., Hodzic, E., Shrestha, R., Saha, A., Ouyang, H., Yin, D., Mamitsuka, H., Sahinalp, C., Radivojac, P., Menczer, F., and Chang, Y. (2018). Ultra High-Dimensional Nonlinear Feature Selection for Big Biological Data. *IEEE Transactions on Knowledge and Data Engineering*, 30(7):1352–1365.

Yang, B., Yih, W.-t., He, X., Gao, J., and Deng, L. (2015). Embedding Entities and Relations for Learning and Inference in Knowledge Bases. *CoRR*, abs/1412.6575.

Yang, F., Yang, Z., and Cohen, W. W. (2017). Differentiable Learning of Logical Rules for Knowledge Base Reasoning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30, pages 1–11. Curran Associates, Inc.

Zhang, H., Zheng, T., Gao, J., Miao, C., Su, L., Li, Y., and Ren, K. (2019a). Data poisoning attack against knowledge graph embedding. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, pages 4853–4859. AAAI Press.

Zhang, J., Bargal, S. A., Lin, Z., Brandt, J., Shen, X., and Sclaroff, S. (2018). Top-Down Neural Attention by Excitation Backprop. *International Journal of Computer Vision*, 126(10):1084–1102.

Zhang, J., Chen, B., Zhang, L., Ke, X., and Ding, H. (2021). Neural, symbolic and neural-symbolic reasoning on knowledge graphs. *AI Open*, 2:14–35.

Zhang, S., Tay, Y., Yao, L., and Liu, Q. (2019b). Quaternion knowledge graph embeddings. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA. Curran Associates Inc.

Zhang, W., Paudel, B., Zhang, W., Bernstein, A., and Chen, H. (2019c). Interaction Embeddings for Prediction and Explanation in Knowledge Graphs. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 96–104. Association for Computing Machinery.

Zhang, Y., Xu, X., Zhou, H., and Zhang, Y. (2020). Distilling Structured Knowledge into Embeddings for Explainable and Accurate Recommendation. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, WSDM '20, pages 735–743, New York, NY, USA. Association for Computing Machinery.

Zhao, D., Wan, G., Zhan, Y., Wang, Z., Ding, L., Zheng, Z., and Du, B. (2023). KE-X: Towards subgraph explanations of knowledge graph embedding based on knowledge information gain. *Knowledge-Based Systems*, 278:110772.

Zhu, Y., Xian, Y., Fu, Z., de Melo, G., and Zhang, Y. (2021). Faithfully Explainable Recommendation via Neural Logic Reasoning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3083–3090, Online. Association for Computational Linguistics.

Appendix

A Peer-Reviewed Publications

A.1 Comprehensible Artificial Intelligence on Knowledge Graphs: A survey

Full reference: Schramm, S., Wehner, C., and Schmid, U. (2023). Comprehensible Artificial Intelligence on Knowledge Graphs: A survey. *Journal of Web Semantics*, 79:100806. Publisher: Elsevier BV

Type: Journal Paper

My scientific contribution:
Simon Schramm and I contributed equally to all aspects of the conceptualization, methodology, formal analysis, and investigation. Ute Schmid advised on the conceptualization.

My writing contribution:
Simon Schramm and I were jointly responsible for the original draft, review and editing, and visualization. Ute Schmid provided feedback, which we integrated.

A.2 Interactive and Intelligent Root Cause Analysis in Manufacturing with Causal Bayesian Networks and Knowledge Graphs

Full reference: Wehner, C., Kertel, M., and Wewerka, J. (2023). Interactive and Intelligent Root Cause Analysis in Manufacturing with Causal Bayesian Networks and Knowledge Graphs. In *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*, pages 1–7

Type: Conference Paper with Presentation

My scientific contribution:
This work, led by me, was carried out with Maximilian Kertel and Judith Wewerka. I contributed to all aspects of conceptualization, methodology, formal analysis, investigation, implementation, and evaluation. Maximilian Kertel contributed to methodology, formal analysis, implementation, and evaluation, while Judith Wewerka supported the conceptualization.

My writing contribution:
I was responsible for the original draft, review and editing, and visualization. Maximilian Kerte contributed to the sections on the Bayesian network and evalua-

tion. Judith Wewerka provided feedback, which we integrated.

A.3 ManuKnowVis: How to Support Different User Groups in Contextualizing and Leveraging Knowledge Repositories

Full reference: Eirich, J., Jäckle, D., Sedlmair, M., Wehner, C., Schmid, U., Bernard, J., and Schreck, T. (2023). ManuKnowVis: How to Support Different User Groups in Contextualizing and Leveraging Knowledge Repositories. *IEEE Transactions on Visualization and Computer Graphics*, 29(8):3441–3457. Publisher: Institute of Electrical and Electronics Engineers (IEEE)

Type: Journal Paper

My scientific contribution:
This joint work was led by Joscha Eirich and involved Dominik Jäckle, Michael Sedlmair, Ute Schmid, Jürgen Bernard and me. I contributed to the conceptualization, methodology, implementation, and evaluation related to KGs and the integration of user feedback.

My writing contribution:
The main draft was written by Joscha Eirich. I contributed to the sections on KGs and user feedback integration and provided reviews.

A.4 Literature-based Hypothesis Generation: Predicting the evolution of scientific literature to support scientists

Full reference: Besold, T. R., Akujuobi, U., Badreddine, S., Choi, J., Elshazly, H., Gifford, F., Iliopoulou, C., Maruyama, K., Nagano, K., Martin, P. S., and others (2025a). Literature-based Hypothesis Generation: Predicting the evolution of scientific literature to support scientists. In *AI4X 2025 International Conference*

Type: Conference Paper with Presentation

My scientific contribution:
Tarek R. Besold led this joint work. It involved Uchenna Akujuobi, Samy Badreddine, Jihun Choi, Hatem Elshazly, Frederick Gifford, Chrysa Iliopoulou, Kana Maruyama, Kae Nagano, Pablo Sanchez Martin, Thiviyan Thanapalasingam, Alessandra Toniato, and me. I contributed to the conceptualisation, methodology, implementation, and evaluation related to explainability and the integration of user feedback.

My writing contribution:
Tarek R. Besold wrote the main draft. I provided reviews.

A.5 Knowledge graph enhanced retrieval-augmented generation for failure mode and effects analysis

Full reference: Bahr, L., Wehner, C., Wewerka, J., Bittencourt, J., Schmid, U., and Daub, R. (2025). Knowledge graph enhanced retrieval-augmented generation for failure mode and effects analysis. *Journal of Industrial Information Integration*, 45:100807. Publisher: Elsevier BV

Type: Journal Paper

My scientific contribution:
This research was led by Lukas Bahr. I contributed to the conceptualization, methodology, and validation, with a focus on all KG-related topics and the evaluation, including the user study. Lukas Bahr was responsible for the conceptualization, methodology, implementation, data curation, and evaluation. Judith Wewerka contributed to the validation and investigation. José Bittencourt supported the validation, data curation, and funding acquisition. Ute Schmid contributed to the conceptualization and funding acquisition. Rüdiger Daub supervised the project and contributed to the conceptualization.

My writing contribution:
Lukas Bahr wrote the main draft and created the visualizations. I contributed to the original draft and the revisions. Judith Wewerka and Rüdiger Daub contributed to the review and editing process.

A.6 Anwendung von Causal-Discovery-Algorithmen zur Root-Cause-Analyse in der Fahrzeugmontage

Full reference: Poßner, L., Bahr, L., Röhl, L., Wehner, C., and Gröger, S. (2025). Anwendung von Causal-Discovery-Algorithmen zur Root-Cause-Analyse in der Fahrzeugmontage. In *Rethinking Quality - Wandel des Qualitätsmanagements durch Digitalisierung und Künstliche Intelligenz*, pages 39–59. Springer Fachmedien Wiesbaden, Wiesbaden

Type: Conference Paper with Presentation

My scientific contribution:
Lucas Poßner led this joint work. It involved Lukas Bahr, Leonard Röhl, Sophie Gröger and me. I con-

tributed to the conceptualisation and methodology related to RCA.

My writing contribution:

Lucas Poßner wrote the main draft. I contributed to the introduction and related work. Additionally, I provided reviews.

B Scientific activities

B.1 Non-Peer-Reviewed Publications

B.1.1 From Latent to Lucid: Transforming Knowledge Graph Embeddings into Interpretable Structures with KGEPrisma

Full reference: Wehner, C., Iliopoulou, C., Schmid, U., and Besold, T. R. (2024). From Latent to Lucid: Transforming Knowledge Graph Embeddings into Interpretable Structures with KGEPrisma. Version Number: 2

Type: Journal Paper

My scientific contribution:
I led this research and was responsible for conceptualization, methodology, software development, validation, formal analysis, investigation and data curation. Tarek R. Besold contributed to the conceptualization, provided resources, and supervised the project. Chrysa Iliopoulou contributed to the investigation, and Ute Schmid supported the formal analysis.

My writing contribution:
I wrote the original draft, revised and edited the manuscript, and created the visualizations. Chrysa Iliopoulou, Ute Schmid, and Tarek R. Besold contributed to the review and editing process.

B.1.2 Aligning Path-based Link Prediction with Human Understanding of Valid Reasoning

Full reference: Wehner, C., Bahr, L., Voigt, E., Wagner, J., Wagner, J., and Schmid, U. (2025). Aligning Path based Link Prediction with Human Understanding of Valid Reasoning

Type: Journal Paper

My scientific contribution:
I led this research and was responsible for conceptualisation, methodology, software development, validation, formal analysis, investigation, and data curation. Lukas Bahr contributed to the conceptualisation, methodology, and formal analysis. Erik Voigt and Janne Wagner supported the software development and data curation. Jakob Wagner and Ute Schmid helped with the methodology, and formal analysis.

My writing contribution:
I wrote the original draft, revised and edited the manuscript, and created the visualisations. Lukas Bahr, Erik Voigt, Janne Wagner, Jakob Wagner, and Ute Schmid

contributed to the review and editing process. Additionally, Lukas Bahr, Erik Voigt, and Jakob Wagner provided support with the visualisation.

B.2 Reviews

I reviewed for the following conferences and journals:

- 26th European Conference on Artificial Intelligence (ECAI 2023), 30 September – 4 October 2023, Kraków, Poland
- 46th German Conference on Artificial Intelligence (KI 2023), 26–29 September 2023, Berlin, Germany
- 20th International Conference on Principles of Knowledge Representation and Reasoning (KR 2023), 2–8 October 2023, Rhodes, Greece
- 47th German Conference on Artificial Intelligence (KI 2024), 25–27 September 2024, Würzburg, Germany
- *Journal of Cognitive Systems Research*
- *Journal of Neurosymbolic Artificial Intelligence*
- *Journal of Computers in Industry*

B.3 Advised Thesis

Kosel, Philipp (2024). "Human-in-Loop-Link Prediction for Root Cause Analysis in Manufacturing". MSc thesis. Supervision by Christoph Wehner, Judith Wewerka, and Timo Kaufmann.

Fehr, Julian Holger (2023). "Text Literal Supported Link Prediction in Knowledge Graphs Utilizing the MINERVA Algorithm". MSc thesis. Supervision by Christoph Wehner and Ute Schmid.

Soykök, Bartu (2022). "Evaluating the Faithfulness of XAI Attribution Methods – End-To-End Testing Pipeline of Relevance-Based Attribution Methods on Computer Vision Foundation Models." MSc thesis. Supervision by Christoph Wehner and Ute Schmid.

B.4 Other Activities

During my PhD, I was involved in a range of scientific activities, including writing grant proposals, patent, giving talks, attending conferences and summer schools, co-organizing a workshop, and completing a research internship.

- Schmid, Ute; Wehner, Christoph; Zimmermann, Patrick; Schramm, Simon; and Bahr, Lukas (2025), for Otto-Friedrich-Universität Bamberg and BMW Group. "KIWIS – Entwicklung eines integrierten KI-gestützten Wissensmanagementsystems zur Optimierung der Produktion von Elektrofahrzeugkomponenten und Marktanalysen im Automotive-Bereich." BayVFP, Bayerisches Staatsministerium für Wirtschaft, Landesentwicklung und Energie (StMWi)

- Patent Besold, T. R., Akujuobi, U., Badreddine, S., Choi, J., Nishikawa, K., Wehner, C., Kumari, P., Hwang, H.-C., Elshazly, H. M. A. E., Iliopoulou, C. T., and Sanchez, P. (2025b). Method, apparatus, and computer-readable medium for generating a hypothesis from a knowledge graph
- Talk with Tarek R. Besold at the Neuro-symbolic AI Group at the Alan Turing Institute on *AI for (the Creative Part of) Scientific Discovery*, London, England, August 2024
- Research internship at Sony AI, Barcelona, Spain, November 2023 – April 2024
- Co-organizer of the *AI for Production Workshop* at KI2024, Würzburg, Germany, September 2024
- *Dialogtag* of the BMW Doctoral Program, Munich, Germany, September 2022
- HCAI-LAB Symposium "Digital Transformation for Smart Farm and Forest Operations", Tulln an der Donau, Greater Vienna Area, Austria, August 2022
- DeepLearn Summer School 2022, Las Palmas de Gran Canaria, Spain, July 2022
- ICPRAI 2022 Doctoral Consortium, Paris, France, May 2022