Automatic Generation of Misinformation - Natural Language Generators in the Application Scenario of Wikipedia

Masterarbeit

im Studiengang Computing in the Humanities der Fakultät Wirtschaftsinformatik und Angewandte Informatik der Otto-Friedrich-Universität Bamberg

Lehrstuhl für Medieninformatik

Verfasser: David Matschat

Prüfer: Prof. Dr. Andreas Henrich

Bamberg 2026

URN: urn:nbn:de:bvb:473-irb-113231x
DOI: https://doi.org/10.20378/irb-113231

Abstract

Purpose - The spreading of misinformation increasingly poses a threat, especially in the online domain. With the advances in computational text generation techniques and in the context of Wikipedia, this thesis will answer the following question: Can contemporary Natural Language Generators be used to intentionally generate Wikipedia articles with embedded, authentic misinformation? For this purpose, the state of research about misinformation and the vulnerability of Wikipedia, as well as the foundations of Natural Language Generation (NLG) are reviewed.

Methodology - Five different, contemporary NLG models are implemented and trained on a dump of Wikipedia articles about English writers. Subsequently, they generate a variety of texts for both a quantitative and a qualitative evaluation. The quantitative outputs are then assessed by automatic NLG evaluation metrics, and the qualitative ones are reviewed manually.

Findings - Even though two of the models, GPT-2 and GPT-2 (finetuned), achieve very good results concerning word use, grammar, authenticity, and can even produce a Wikipedia-like structure, overall the five implemented models do not pose a imminent threat to the online encyclopedia as they either lack the sophistication in sentence building or the stability in their performance. However, with these already good results, the further development of new NLG models should be closely monitored.

Acknowledgment

At this point, I would like to thank the following people who supported me in getting to this point in my life and finishing my Master's thesis:

First and foremost, I want to thank my parents, Barbara and Dietmar Matschat, for their continuous help, words of advice, and also financial support. Without their encouragement, I would not have pursued this masters degree.

Furthermore, I would like to thank my dearest friend Franziska Schock, for supporting me in this challenging time, for all the advice she has given me, and in general for her calmness and reliability in our friendship.

I also want to express my gratitude for Julia Gürster, Raphael Goldmann and Benedikt Renner, who were constantly by my side in the past months and years, being a helpful and vital part of my life in any personal or academic matter.

Last but not least, I want to thank the advisor for my Master's thesis, Robin Jegan, for helping me conduct this project, rapidly responding and regularly providing help with any problem that occurred throughout the implementation and creation of this thesis.

Contents

List of Figures

Chapter 1

Misinformation with a Computer

The British referendum on leaving the European Union, the 2016 elections in the USA, or the major increase of votes for the right-wing AfD party during the 2017 German elections all have one thing in common: empirical studies imply that they have all been majorly influenced by the spread of diverse misinformation [Zimmermann and Kohring, 2020, p. 33]. In fact, statistics imply that especially the year 2016, when one of the presidential candidates lastingly coined the term "fake news", marked a turning point in the public awareness about this kind of false information. And while it might not be the initial cause for it, the trust in the media has decreased worldwide ever since, just being over 50% in 2021. It also created a somewhat contradictory connection that was observed in the United States. A majority of Americans obtain their news from social media, but at the same time also state that they encountered fake news on social media and don't see it as a trustworthy source for news. And even though social media platforms are slowly starting to take action against misinformation, this process takes time and the algorithms to detect them are still not completely matured. Yet, hindering the spread of them is also a rather urgent task, as the negative impact of fake news can be seen in the consequences they are believed to have: Over 70% of people think fake news have a negative impact on political discussions with their relatives and friends, while over 80% state that they negatively influence their countries politics [Djordjevic, 2021].

Despite these statistics only referring to misinformation in actual daily news, this term for false truths can be transferred to any kind of informative content and, considering the amount of data available online, therefore be a threat to almost any website apart from social media. One very well known platform to contain such informative content is an online encyclopedia: "Wikipedia is the world's largest encyclopedia and one of the most popular websites, with more than 500 million pageviews per day. It attracts millions of readers from across the globe and serves a broad range of their daily information needs" [Singer et al., 2017, p. 1591]. It is commonly known that Wikipedia is an open encyclopedia, meaning that its articles can by created, accessed, but also edited by any user. Considering its daily pageviews, articles that contain misinformation could reach a large number of recipients in a short amount of time, potentially having a major negative impact on those confronted with it and believing it.

But, what if the Wikipedia articles would not have to be written manually, but by an algorithm, allowing the production of text - potentially riddled with misinformation - in a simple way and even the generation of numerous texts in a short amount of time? One possible solution to this is the field of Natural Language Generation (NLG), which has the goal to produce text in a natural language for various purposes. It can be used to create descriptions of images, rewriting or expanding texts, all of which have different goals for the creation, but have to plan, determine and realize their content individually [Dong et al., 2021, p. 2]. "[...][The] field of NLG has changed drastically in the last 20 years, with the emergence of successful deep learning methods. For example, since 2014, various neural encoder-decoder models [...] have been proposed to achieve the goal by learning to map input text to output text" [Dong et al., 2021, p. 2]. Indeed, NLG has experienced a steady growth in interest, not only for scientific research, but also for the use in the corporate sector. Reports predicted that NLG would be used by a quarter of enterprises by 2022, due to increasingly available data and the expectation from consumers to be provided with personalized content. With the ongoing development in this field, the number of practical tasks for NLG is also increasing continuously: Current use cases include the generation of product descriptions in e-commerce or performance reports in the banking industry, news summarizations, chatbots in the transportation sector, but also the misuse of these systems for political propaganda and misinformation [Dilmegani, 2020]. The development team of one of the latest NLG systems mentioned an aspect about them in their system's model card, which is very important in the context of this thesis. They stated that the large-scale language models used for such systems are not able to distinguish fact from fiction, which is why they should not be used for tasks where texts need to be true [Wu, 2019]. When combining this with the intent to spread misinformation, a system that can generate authentic articles that automatically contain false information, could be a real threat for a platform like Wikipedia. Therefore, the research question for this thesis will be:

> Can contemporary Natural Language Generators be used to intentionally generate Wikipedia articles with embedded, authentic misinformation?

In the following thesis, a brief introduction into misinformation and its spreading, as well as into the chosen application scenario of Wikipedia and its vulnerability will be given. Continuing, a detailed insight into Natural Language Generation, its basics, functionality and important components will be provided, followed by an overview about state-of-the-art topics in this field. Subsequently, the third chapter will describe the plan for the practical implementation of this topic, with a concise description of important steps during the implementation and execution phase. In chapter four, the generated texts will be evaluated quantitatively, with automatic evaluation metrics, and qualitatively, with a manual review. With the results, the performance of the generated outputs will be assessed, trying to find their model's strengths and weaknesses, in order to infer implications about which contemporary model shows the best performance and what adjustments could be done in future research on this topic. Finally, the thesis will be briefly summarized and the results considered, whether the tested Natural Language Generators are sophisticated enough to qualitatively generate Wikipedia articles containing misinformation.

Chapter 2

Foundations and Context

2.1 Thesis Application Scenario

The sharing of misinformation whether by intentional deception of other or plain unawareness, has likely existed for thousands of years in human communication. However, as today's means of communication allow reaching mass audiences, misinformation has the potential to spread very fast. In order to clearly distinguish between different kinds of information, Jürgen Habermas defined the triad of concepts with information, misinformation, and disinformation. The first can simply be seen as true and correct information, while the other two are false information. Since Habermas defines truth as the product of consensus, misinformation originates from disagreement between individuals, whereas disinformation includes the deliberate spread of false information for certain reasons [Southwell et al., 2021, p. 1ff.].

2.1.1 Misinformation Research

To determine if some information can be classified as mis- or disinformation, Zimmermann and Kohring defined the six criteria of communication, actuality, claim of truth, falsehood, lack of truthfulness, and intent to deceive, although the last criterion is optional and used to differentiate between certain forms of misinformation. Misinformation obviously is a form of communication, as it is directed towards another social actor, and can therefore be any linguistic or visual message, but not of non-communicative kind, like secrecy. Additionally, the intended recipient has to interpret the message as intended information. The actuality of information does not only cover the temporal recentness of it, but also its public relevance, which is why most misinformation is usually oriented towards surprising topics that could have a major impact on society. Recent misinformation, just like regular journalism, holds a claim of truth for itself, and only with this orientation to be factual it can be declared as a misinforming message. In contrast to true information, misinformation does not fulfill this claim for truth and therefore always poses the risk of misleading its recipients. This falsehood does not necessarily refer to the communicated message itself, but to its meaning: i.e., if the answer to the question 'Where is person X?' is 'He usually is in the library at this time', the message itself might not be incorrect. However, it is misleading as it is not the concrete answer to this question. This is the falsehood criterion, which distinguishes misinformation from a rumor [Zimmermann and Kohring, 2020, p. 25ff.]. The lack of truthfulness then is a deciding factor between misin-

formation, the accidental publishing of false information due to i.e., an editorial error, and disinformation, the deliberate publishing of false information. It is important to note that this deliberate publishing does not necessarily imply bad intentions for the actor behind it, as he can simply adopt false information from others. Finally, the intention to deceive is the attempt to make the audience believe a certain wrong message is true. If this is the case, the misleading does not happen randomly, but is deliberately wanted [Zimmermann and Kohring, 2020, p. 27ff.].

As previously described, several terms related to this have been identified in recent years. Figure 1, the classification that Diego Saez-Trumper has made, shows different kinds of false information. However, in order to simplify it over the course of this thesis, all of these different forms will be referred to as misinformation.

	Authenticity	**Intention**
Disinformation	False	Bad
Misinformation	False	Unknown
Mal-Information	True	Bad
Fake News	False	Bad
Satire News	False	Not Bad
Imposter Content	False	Unknown
Fabricated Content	False	Bad
Manipulated Content	Unknown	Bad
Rumor	Unknown	Unknown

Figure 1: Different concepts of disinformation, cf. [Saez-Trumper, 2019, p. 4]

The problem with misinformation is the way humans receive it: Empirical research shows that humans initially tend to encode any given information as if it were true, and only later tag it as either true or false. They also tend to look for confirmation with others when determining the truth of information, so if their social environment initially believes some misinformation, it creates a spiral of individuals confirming the false information, increasing the number of people actually believing it. In theory, mass media would have the possibility to instantiate regulatory structures against the spreading of such misinformation. However, looking at the media in democratic countries, most regulation happens post hoc, meaning that misinformation has already been published before being detected. This of course happens in order to not impose censorship on the media, but allows false information to spread, which, even if it is corrected, can have lasting effects on an audience's attitude. Additionally, countering misinformation once it has aired is resource-intensive, as it requires extensive campaigns to create awareness about the misinformation as well as to educate and sensitize the audience for it and comprehensibly explain the correct information [Southwell et al., 2021, p. 4f.].

Specifically, an extreme form of misinformation, or something it can lead to, are conspiracy theories. They are usually constructed by a small group of conspiracists, which pursues destructive goals for society and manipulates events for their own good. Conspiracy theorists specifically use false or extremely biased information to achieve their goal, often trying to be as verifiable as possible to seem plausible. Throughout history, a variety of conspiracy theories has spread, about the American moon landing in 1969 not being real, or about the US government having known about the terror attacks from 11th

September 2001 in advance. The main appeal these theories have for their recipients is their simplicity and enclosed argumentation, giving them the control they think they have lost in todays' complex world. But they also frequently employ a negative image of the opposing side to reach a positive self-image. In Germany alone, almost 25% of the population are said to be vulnerable to these conspiracies, with aspects like lack of trust in the government or feeling underprivileged being driving factors for that. The problem is that once an individual has started explaining their own worldview with conspiracy theories, it is likely that they will believe other theories of this kind too. What makes this such an extreme form of misinformation is its effect on society, as it presents clear enemy stereotypes and therefore fuels fear and distrust - especially towards minorities, but also politics, the media or science - and, in the worst case, leads to violent actions in the worst case [Schneider et al., 2020, p. 284ff.]. These theories have a similar spreading pace to 'normal' information in the first hours of publishing, then however they spread slower whilst at the same time generating a higher rate of interest. Aligned with the previous statement that humans tend to look for confirmation in information, this process of selecting and sharing content related to a certain belief or narrative, combined with a specific, homogeneous social group, renders conspiracy theories highly efficient forms of misinformation. Even though many platforms try to implement solutions to detect untrustworthy information, these attempts are mostly deemed as misinformation themselves by people believing in the respective conspiracy theory [Del Vicario et al., 2016, p. 554ff.].

Concluding this, it is still rather easy to access conspiracy theories on platforms like YouTube, while arguments against these rarely are a functioning measure to counteract them. One of the most efficient options are preventive actions like mediating political and historical knowledge, or always questioning the credibility of sources, and detecting classical argumentation and manipulation schemes [Schneider et al., 2020, p. 289f.].

2.1.2 Spreading Misinformation

"While misinformation can certainly be produced from intentional deception, [it] can also stem from a misalignment between author's and viewer's vernacular or from an ambiguous context of interpretation" [Hemsley and Snyder, 2021, p. 95]. With this, it becomes clear that a variety of factors can lead to the spread of misinformation or information turning into misinformation.

Actively misinforming others can be done in diverse ways: Concerning visual information, for example, it can be done by oversimplifying or overcomplicating data, by omitting relevant variables, inappropriately scaling a graphic, or knowingly altering the data. Even though deemed unethical and morally wrong by the majority of people, this last technique still could be said to be the most obvious way of spreading misinformation. Another, still very obvious way of misinformation spread is the technical or conceptual error, which, just like intentional misinformation, can appear in many different forms. Such errors can occur due to improper data handling, software faults, or overall inappropriate design choices. In the aforementioned example of visual information, the input data usually has to be cleaned or pre-processed, mathematical figures such as medians have to be calculated and highlights identified - all of which can lead to mistakes and false information being included in the content. After that, an adequate type of visualization has to be selected, which has the challenge of selecting the correct representation for a certain content. However, it also

poses the risk of choosing completely improper forms of visualization or, with an unskilled graphic designer, occlusions, when several elements are placed on top of each other. The occurrence of errors during the process of creating such information is so prominent that designers often have to be trained in order to handle input data correctly [Hemsley and Snyder, 2021, p. 95ff.].

Some of the more unfamiliar mechanisms that spread any kind of misinformation are naive interpretations. These can happen, if the creator uses very sophisticated visualizations for the content compared to something more common like a pie chart. Without training, the audience can naively misinterpret the information, which can be a source of misinformation. Therefore, it should be important for the creator to consider the needs and knowledge of the intended audience already during conceptualization. Beyond that, some interpretive and meaning-making conventions in specific communities also have the ability to promote misinformation. Data art, for example, visualizes content based on the aesthetic and technical criteria and not on objective representations of information. Scientific visualizations, which tend to be very similar to data art, are mostly designed on objectivity and accuracy, which is why mixing these types up could lead to misinformation among certain recipients. Lastly, an important factor that determines whether misinformation occurs or not is the context of its presentation. The interpretation of information can be influenced by the location it is placed at or if it is a scientific visual that was simplified for a broader audience, since this kind of audience usually does not question the visualization. Overall, the occurrence of misinformation can be split into two groups: misinformation on the conception side of the information and misinformation at the interpretation side - independently whether it is intentional or not [Hemsley and Snyder, 2021, p. 98ff.].

In both scenarios, once this misinformation is posted or published, it can spread rapidly due to social media platforms and the ability to copy and share. Although most content that is posted nowadays remains unseen by the wider public, in case it spreads, it does so at a rapid pace and quickly reaches a diverse audience in multiple geographic locations, with different education levels and from diverse cultures. Additionally, social media allows users to alter the original content of something, by reposting it with or without added text, altering the original information, or posting it to another channel and thereby changing its context. Some users with specific influence, mostly referred to as opinion leaders, can also be successful and have a big influence on their audience's opinions. Concluding this chapter, it is therefore important to note that, even if the intention of the original creator of the information is good and it is created error-free, the recipient can still misinterpret it, leading to the creation of misinformation [Hemsley and Snyder, 2021, p. 101f.].

2.1.3 Vulnerability of Wikipedia

Even though Wikipedia has such a significant global influence with its daily pageviews and users worldwide, it remains mostly unclear what motivation or needs its user group has about why they visit the page and how they consume its content. A study among English Wikipedia readers in 2017 tried to determine the motivation, information need and prior knowledge of randomly selected users, but showed ambiguous results: even though two aspects received a higher percentage concerning the motivation, namely that something was

referred to in the media or that it came up in a conversation, the set of stated motivations was rather diverse, also ranging from school- or work-related topics to intrinsic reasons like boredom or curiosity. Similar data was observed with the information need and prior knowledge of users, making it difficult to identify overall important aspects. However, the information depth was found to correlate with the motivation of the reader. While quick look-ups are likely to be linked to extrinsic motivation such as the mentioned school-work, in-depth research mostly happens when the motivation is intrinsic. Smaller, less apparent yet still important correlations have been found between motivation and prior knowledge, as well as between prior knowledge and information need. In general, in-depth Wikipedia research does happen less often than shallow one, whereas the motivation of users remains largely stable over time [Singer et al., 2017, p. 1591ff.].

Additionally to its user's motivations, it also remains unclear why Wikipedia seems to be a rather reliable online platform. Contrary to common belief that it could not become an encyclopedia with interesting and scientific content being built by lots of contributors, it turned out to be a reliable source of information over the years. In fact, it is more reliable than anyone would have thought at its beginnings and, when put into comparison with traditional encyclopedias like the Encyclopedia Britannica, shows a remarkable result: Even though Wikipedia on average has one error more in an article, it can still be argued that the gap to the Britannica is quite narrow. When trying to answer the questions of Wikipedia's reliability, three aspects play a key role: Firstly, even though technically anyone can edit or create a Wikipedia article, and only a small portion of contributors are experts, this system does not fall into a state of chaos. Secondly, it overcomes the volatility problem, which states that content is only informative if it is present over a long period of time - against the constant possibility that anyone can modify, change or delete content on the platform. And lastly, a characteristic of Wikipedia itself makes it reliable: Studies on social epistemology imply that any actor, not only scientists, strives for success motivated by their own reputation and personal gain. Therefore, the creators of Wikipedia are overall beneficial to the whole platform [Lageard and Paternotte, 2021, p. 451ff.]. To answer the question of why its reliability is that close to the one of encyclopedia Britannica, several hypotheses have been made, such as the 'wisdom of crowds' principle, stating that the aggregation of outputs is more accurate than each of the outputs on their own, or the elite principle, which means that a small number of highly reliable contributors counterbalance a vast majority of unreliable ones. Eventually, Lageard et al. found it to be a combination of the administration of Wikipedia, which constantly bans harmful users and therefore tries to stave off sources of misinformation, and the revert option that allows reliable contributors to countermeasure the remaining small amount of harmful ones. Additional to the individual factor of honest and reliable users, these three aspects account for Wikipedia's reliability [Lageard and Paternotte, 2021, p. 455ff.]. However, the authors also state the following: "We identify the main threat for Wikipedia as the presence of negative or [unmotivated] contributors, namely disinformers and trolls, which honest contributors cannot expect to counterbalance" [Lageard and Paternotte, 2021, p. 469].

From Wikipedia's perspective, there are three policies in place to assess the information quality of an article: the NPOV (neutral point of view), which looks if an article has editorial bias or if all statements have reliable sources; the verifiability, meaning that Wikipedia users can check the information sources; and no original research, which states that no Wikipedia article can contain original research and therefore conversely any statement needs to have a source. Concrete approaches to spread false information are classified by the platform as either social or technological attacks. Social attacks either exploit the weaknesses of a social system, for example by using click farms or sock-puppeting, a technique where a single individual creates multiple online identities, or by using the lack of information on the recipient's side. Technological attacks on the other hand can be done by employing algorithms, which act and interact much faster than humans and can therefore manipulate social networks and platforms, or even imitate human behavior [Saez-Trumper, 2019, p. 5ff.]. At Wikipedia, several reports of especially social disinformation attacks could be found. The team behind the encyclopedia has repeatedly investigated complaints and banned non-compliant users, even thinking about closing some national Wikinews pages as the number of disinformation was getting too high. The challenge lies within the comprehension of when and how information in general, but disinformation in particular, has been added to Wikipedia. As this is still to be resolved, a problem arises with so called Citogenesis in Wikipedia. Information introduced to Wikipedia sooner or later becomes a reference for an external piece of information. If the source of this external piece is considered reliable and wrongfully acquires a piece of disinformation, it can later be misused as a reference to support the disinformation article on Wikipedia, thereby creating a vicious circle [Saez-Trumper, 2019, p. 7f.].

This shows that Wikipedia can be very vulnerable to false information, especially to one of its most prominent forms, the hoax. Hoaxes are a type of disinformation, which contain fabricated falsehoods masqueraded as truths. What makes them so dangerous to Wikipedia is that anyone can insert information into articles on the platforms and therefore manipulate one of the most visited websites and a major information source. Examples for hoaxes on Wikipedia are plenty, from an article about the non-existent language called 'Balboa Creole French' to the fake Aboriginal god of 'Jar'Edo Wens', which had influence on a science fiction book. The difficulty with Wikipedia - and presumably any other online data basis - is the differentiation between single hoax facts and complete hoax articles. Finding the former would require tagging all facts in an article, which is rather impractical, while for the latter, it would be enough to look for articles that have been flagged as hoax by readers or algorithms [Kumar et al., 2016, p. 591f.]. Actually, the Wikipedia patrolling process controls more than 80% of new articles within an hour of creation and flags suspicious ones. If flagged, the continuing process is fairly efficient as well, since Wikipedia employees then discuss the article's fate and, if it is decided to delete them, 88% are deleted within a day of flagging and almost all of them within the first month. What also becomes clear however, is that once a hoax article survives the first day, its chances of being undetected and/or not deleted grow significantly. As their lifespan increases, the pageviews of hoax articles eventually become another problem. Even though they tend to have less pageviews than their non-hoax counterparts and also tend to get detected faster when their daily traffic increases, some thoroughly created hoaxes survive for years and attract thousands of pageviews, therefore reaching a big audience

to spread their false information to. As mentioned before, this also increases the risk for their links to be referenced on other pages, again making it more accessible for a broader audience [Kumar et al., 2016, p. 592ff.]. If an article is reviewed however, deciding if it is a hoax or not is a fairly easy task, with an accuracy up to 92%. The article's creator, or if it is referenced in other articles, are crucial factors for this. Additionally, humans are biased to suspect hoaxes with short articles, low Wikipedia-link density, or a high plain-text-to-markup ratio. The scientific knowledge about this however also means that a hoax stands a higher chance of succeeding if it is longer, and overall looks more like a typical Wikipedia article [Kumar et al., 2016, p. 598f.].

Nowadays, the biggest vulnerability of Wikipedia lies with social attacks, especially click farms. The success of such an attack fully relies on its volume and the size of the project team, with a smaller team perhaps being more sensitive towards such attacks. Concerning technological attacks however, their success completely depends on quality and sophistication of the technique. With currently available technology and the expertise of Wikipedia's patrollers, Diego Saez-Trumper of the Wikimedia Foundation does not see an imminent risk of such attacks at this time, still noting that it is important to keep tracking the development of techniques [Saez-Trumper, 2019, p. 8f.]. To tie on to this concept - even though the Wikimedia Foundation does not see an imminent threat - the question is, if contemporary means really are too underdeveloped or if they are sufficient enough to endanger Wikipedia being technologically attacked at a large scale. With computer programs or Natural Language Generators enabling the automatic generation of large bodies of text adapted to certain input data, and constant development and adaptions in this field, it can possibly become a technology to use against Wikipedia.

2.2 Natural Language Generation

Computational Linguistics is the field of processing natural language, which includes written and spoken form, with a computer. Its main goal is modeling the language knowledge onto a machine to solve practical problems, like translating a sentence into another language. To achieve these goals, Computational Linguistics has the four core tasks of developing formalisms for modeling and altering natural language, constructing methods for this, providing knowledge about different languages, and evaluating natural language systems [Antrup, 2010, p. 2ff.]. Computers have been able to communicate in natural language for years, e.g., by informing users about certain errors, but mainly do this by generating from pre-defined phrases. To figure out linguistic features and flexibly produce text is the task of Natural Language Generation. This field of computational linguistics engages with determining the content in the text to be produced, defining a structure that satisfies the purpose of the text, as well as deciding on the content of each sentence on a grammatical and lexical level [Bateman, 2010, p. 633f.].

2.2.1 Natural Language Processing

Natural Language Processing (NLP) can be seen as a hypernym to Natural Language Generation and is an almost-synonym for Computational Linguistics. It requires the computer to have a rather deep understanding of natural language, which, as of now, is still far from being achieved [Chowdhary, 2020, p. 604ff.]. In general, research in this field explores the

use of computers in understanding and manipulating natural language: "NLP researchers aim to gather knowledge on how human beings use language [...] to make computer systems understand and manipulate natural language to perform desired tasks" [Chowdhury, 2003, p. 51]. The scientific interest in NLP already started in the 1950s as an intersecting discipline between artificial intelligence and linguistics. At first, the theoretical analysis of language grammars led to the creation of specific notations used to specify grammars that were context-free. Even though this approach was deemed inadequate for natural language, it was often used in NLP practice. With natural languages being very large in size and quite ambiguous, the technique of using numerous, hand-written rules became outdated and NLP started borrowing methods from several other fields of research. A new approach to it was introduced by including machine learning into the process and using large bodies of text to train them. Statistical NLP has shown itself to be quite performant with real data, has to be seen as a complement to rather than a replacement for still used, rule-based approaches however [Nadkarni et al., 2011, p. 544f.].

The core of every NLP task is the understanding of natural language. A computer program therefore faces the challenges of thought processes, representation and meaning, and world knowledge. Beginning on a word level, an NLP system may start by conducting a morphological analysis, stemming terms in both a query and the documents in order to derive word variants, followed by lexical and syntactic processing to determine the word's characteristics, part of speech or role in a phrase. After that, it moves on to a sentence, then the overall textual and contextual level. For analyzing and extracting meaning from natural language, researchers suggest distinguishing among phonetic, morphological, lexical, syntactic, semantic, discourse, and pragmatic levels [Chowdhury, 2003, p. 55f.]. For example, the task of the syntax analysis is to determine the structure of given sentences and then regularizing them by, e.g., omitting certain words. The semantic analysis on the other hand determines the meaning of a sentence, in order to unambiguously translate a text into a formal language. Continuing, the discourse analysis looks at the meaning not only on a sentence level, but of the entire text. It analyzes the connections between sentences, which usually are implicit and therefore difficult to determine [Chowdhary, 2020, p. 608ff.].

The major problem of NLP, which has already been mentioned above, is that natural languages are complex constructs, containing many ambiguous words and sentences that are only determined by their respective contexts. Compared to a human reader, NLP algorithms do not have abundant information about a situational background, making their understanding of a given text limited. The syntax may help to decide which words are being combined for a larger meaning, but only in combination with semantics and pragmatics can an internal representation of information be created [Chowdhary, 2020, p. 604f.]. Currently important fields of research in NLP are the manipulation of texts for knowledge extraction and text production in a certain format, known as natural language text processing. Taking some text as input and translating it, this task is designed to transform potentially ambiguous queries or texts to be unambiguous, so that they can be used for further processing [Chowdhury, 2003, p. 59]. Other tasks in NLP include detecting sentence boundaries or part-of-speech tagging, while on a higher level also identifying spelling errors or Named Entity Recognition [Nadkarni et al., 2011, p. 545f.].

2.2.2 Foundations

A natural language generator is a computer program that, based on a communicative intention, determines the content of what it will say, selects words and organizes them into written text - mimicking the actions of a human speaker. Natural Language Generation (NLG) first appeared in the 1950s in the context of machine translation, with the first dynamically generated sentences from questions on a database program being produced 20 years later. However, it was not until the 1980s that it became an independent scientific field [McDonald, 2010, p. 121f.]. NLG has long been a niche interest, with much of the scientific focus going toward other areas of NLP, especially Natural Language Understanding. It has only been recently that NLG has seen increasing scientific and commercial interest. The attempts to use it commercially can be tracked back to the 1990s, when CoGenTex tried to bilingually generate English and French texts. Then, in the 2000s and early 2010s, five companies now leading in the commercial use of NLG were founded, trying to either utilize NLG models to provide services to customers, or use them as self-service toolkits for third-party developing. As selling new technology products can be a challenging task, the companies tried to sell it as low-priced as possible, trying not to underbid, and at the same time kept extensive documentation details for themselves to retain their market advantages [Dale, 2020, p. 481f.]. As the number of potential integrations that NLG vendors tried to leverage grew over the years, functional transparency into model documentations also improved significantly. Looking at currently available products for NLG, they show a rather coherent picture of mainly earlier, template-based approaches being used today. This leads to the assumption that most currently employed NLG techniques are not based on any of the more sophisticated approaches but rely on the quality and ease of use of 'older', already proven models. However, such sophisticated models already exist, and the major trend is neural text generation. The existing neural models, despite not producing completely authentic texts in every attempt and not yet being in commercial use, show a promising utility for future use: augmenting human authoring. Even though technology for next word prediction or alternative suggestions sounds more obvious and already exists, the application of neural text generation could switch the current use from the programs offering suggestions on how to say or do something towards them offering suggestions about what to say or where to go, with the user still being in control [Dale, 2020, p. 482ff.].

An NLG system is typically defined by making choices, whether high-level ones about text content or low-level ones about the use of certain words. Even though decision making is also an important part of NLP, it is a lot more substantial for NLG. Additionally, such a system does not only have to choose between correct and incorrect options for text, but also select one of several correct alternatives based on diverse criteria, which might be the genre of the produced text or the language proficiency of the intended recipient [Reiter, 2010, p. 574f.]. According to Ehud Reiter, "NLG is thus largely the study of choice making, including analyses of individual choices; architectures and systems that can be used to make sets of choices; and methodologies for creating and evaluating new choice making rules" [Reiter, 2010, p. 575]. However, Reiter already observed the practical problem of NLG stated years before: it has not been extensively used in real-world applications. Even though a few systems have been employed in a very limited way, there has not yet been a widespread, long-term use of NLG systems compared with other areas

of NLP. An example of a system with limited use would be SumTime, a generator that uses data about temperature, percipitation or wind speed to produce weather forecasts then read by human journalists [Reiter, 2010, p. 575ff.].

To generate text, information has to be mapped from some non-linguistic source, which requires diverse knowledge about the domain, language, or text types. Furthermore, it also requires knowledge about the engineering, like decomposing or representing information, and about the end users, as their knowledge or habits might impose constraints. "Much of the complexity of NLG arises out of the fact that producing language is a knowledge-intensive, flexible, and highly context-sensitive process" [Bateman and Zock, 2012, p. 285f.]. Essentially, it requires the two questions of 'What to write?' and 'How to write it?' to be answered, which will define the input to an NLG system and the text's language and structure. The problem with the former is the need to define how elaborate the input should be, as insufficiently detailed input can lead to limited options or problems with forming correct sentences [Siddiqui and Tiwary, 2008, p. 210f.]. The information to be expressed with an NLG has to be selected, a text structure to satisfy the purpose of the text must be defined, the content of each sentence determined, and grammatical and lexical decisions made which can be simplified if the application scenario for the generation is known. As it is barely relevant to find all potential realizations of some content in practice, refraining from a comprehensive approach and just using texts of a certain kind is possible. This makes the aforementioned application of weather forecasts a fitting example, as they take raw data that is incomprehensible to humans and use a very narrowly defined language to create texts. Furthermore, text generation can be classified as either deep approaches, which employ knowledge-based text and sentence planning, and shallow approaches, which logically proceed the template-based NLG described in the architecture chapter of this thesis (2.2.4). Application-oriented systems are often developed in a bottom-up manner, meaning that every application creates a new situation with examination of language usage and new linguistic resources. This is contrary to approaches of the late 20th Century, where large, universal linguistic resources were focused to make a system reusable. Since such resources are very time and memory intensive, as well as complex when it comes to changes, they were mostly abandoned in recent decades [Bateman, 2010, p. 634ff.].

2.2.3 Components

No matter the approach to natural language generation, its process involves three central tasks, the discourse planning, which comprises textual organization like order and structure, information selection and an overall situation perspective; the text planning, to define details of the utterance like words, phrases, or different structural options; and surface realization, to finally produce text in written or spoken form. Besides that segmentation of tasks, there are several subordinate tasks like lexicalization which, depending on the application case, will be allocated to one of the three [Siddiqui and Tiwary, 2008, p. 211]. According to this task split, three generator components are usually defined to attend them: for the discourse planning, the application program maintains a model of the situation and provides the source on which the other two operate. The text planner receives units from the application program and defines a structure for the final utterance, whilst for the realization the linguistic component implements the text planner's output as concrete language. Even though not a part of the generator, it is important to note that the

application program defines the scene for the text to be generated, and therefore must be designed in concert with the generator to achieve good results [McDonald, 2010, p. 127ff.].

For the first task of discourse planning, also sometimes referred to as macro planning, the application programs' goal is to correctly interpret a certain communicative intention, given specific context and assumptions about the recipient, which can then be used to build a specification for the text plan. This step requires an adeqaute knowledge base that is used in two ways: on the one hand, it is used for content determination, selecting the elements to fulfill the communicative intention, and on the other hand, it is used for content organization, to sort these elements into a structure suitable for the recipient [Horacek, 2010, p. 437]. Discourse planning can be done either by using text schemata or with incremental planning with discourse relations. The former takes pre-defined representations of stereotypical structures, which supports the necessary coherence with specifications concerning the use of main and subordinate clauses as well as the variability of single elements. The other approach builds a discourse structure tree by continuously connecting text segments, whose nodes are the relations and leafs the concrete expressions. The discourse relations are linked with certain rules for the combination of segments, to also ensure coherence for this approach. However, the results of the discourse planning, which are usually manifested in a rhetorical tree structure, are too vague in many ways to directly enable generating text, leaving the need for the text planning component [Horacek, 2010, p. 441ff.]. One of the first approaches in this was schema-based, meaning that based on observations about humans presenting text, schemata were built in order to capture the regularities contained within them. These were then used to form templates for the text to be produced. Yet this approach proved to have major shortcomings, like not being able to specify the goal of a schema as a whole. Due to this, the rhetorical structure theory was applied to this task, which identifies every coherent sentence as decomposable into a recursive structure of phrases, whose relations allow not only for the definition of certain predicates like cause or purpose, but also for the separation of the phrase into core and supportive structures. With using this approach to macro planning, the communicative goal is being decomposed to acquire a plan library with a set of operators to achieve the overall goal and a planning method. The macro planning phase ends when all communicative goals are expanded to steps in the plan with primitive subgoals [Bateman and Zock, 2012, p. 290ff.].

For micro planning, one of the central tasks is to determine the words, their meaning and syntactic constraints, the so-called lexicalization, to formulate and express the parameters defined in the macro planning. The elements of the target language have to be sorted in the right structure, as well as consider the situative context of the text generation. For that, the connotation of words, the right perspective on the situation, repetitions, or the general context have to be considered. Also important for micro planning is the aggregation, which is the composition of phrases to complete sentence representations, while at the same time keeping redundancies as compact as possible. A third objective to consider are expressions that refer to certain objects. These referring expressions can be represented as descriptions, names, or simple pronouns, and have to be balanced between containing enough information to identify the referred object, while not containing too much irrelevant information and considering the knowledge of the communication's recipient [Horacek, 2010, p. 453ff.]. In order to finding suitable reference words for ele-

ments and avoid repeated naming, the text planner takes the output of the macro planner, groups this information into blocks of sentence-like length and selects representations for their contents. It again generates a tree-like structure, where each leaf contains a sentence plan with all information to build a sentence [Siddiqui and Tiwary, 2008, p. 216f.]. With the document plan as input, micro planning has to make several choices concerning this lexicalization, referring expressions for entities, syntactic structures and the number of expressed messages in each sentences. The produced output is a text specification, yet another tree structure with specifications on document structure and, as leaves, the syntactic sentence structure. Important for this is that an NLG system can not simply generate high-quality text based on a corpus analysis of human written text, but also on studies about human comprehension, as the language produced by humans, especially concerning referring expressions, is not necessarily the most suitable form for any recipient. Today's microplanners, except for the already mentioned referring expressions, mostly rely on empirical analysis of the use of language in certain domains and genres [Reiter, 2010, p. 581ff.].

The linguistic component then takes the output from the text planner and transforms it into some form, in accordance to a grammar, which it then employs to form a syntactically structured sequence of words as the output of the generator. Every approach to this surface realization is an implementation of a recognized grammatical formalism, providing it with rules, constraints, and representations, whilst at the same time being quite incomplete. This component strongly depends on the form of the grammar it uses. If it consists of a set of combinable elements, the component has to select and assemble the elements into a desired representation, while it has to navigate through the grammar and accumulate the basis of the text to produce it all at once, if the grammar is a single structure. In practical use, several grammars have prominently been used, like systemic or functional unification grammars. Systemic grammars can be visualized as decision trees, where choices between usually distinctive alternatives have to be made, based on prior and influencing later choices. Traversing this tree eventually accumulates the final utterance. Functional unification grammars are also traversed, however they use a unification process to merge the component's input with the grammar to get a fully specified surface structure [McDonald, 2010, p. 134ff.]. The surface realization can be done by using a systemic grammar represented in an and-/or-graph, which sufficiently defines the grammar of a sentences. Specifications made concerning mood, transitivity and topic of a sentence contain realization instructions, which are constraints to further specify the final form of the sentence. Alternatively, a functionally unified grammar can be used, as it composes sentences from a unified formalism existing of clauses, noun and verb phrases on the top level and provides functional descriptions, containing details for the desired sentences, for its components [Siddiqui and Tiwary, 2008, p. 219ff.]. A special form of this task is a realizor for overgeneration and selection, which generates several forms of a sentence and uses a mechanism to select one of them. Such overgenerate-and-select architectures supposedly simplify the realization process and enable an NLG to be used for different text genres, yet make it more difficult to assure quality or perform testing. "[...][The] state of the art in realization is sufficiently advanced (compared to other aspects in NLG) that we can seriously consider software engineering issues such as quality assurance, maintenance, ease of software integration, and documentation quality" [Reiter, 2010, p. 584f.].

Concluding this chapter, it has to be noted that even for this basic definition of NLG components, the naming of them is rather inconsistent between different authors. As mentioned above, some split them into macro- and micro-planning and some into discourse and text planning; Ehud Reiter uses the term 'document planner' for the macro planning [Reiter, 2010, p. 579f.]. Additionally, the term text planning is used to describe to different phases: while Siddiqui and Tiwary [2008] uses it for the micro planning, Bateman and Zock [2012] or McDonald [2010] use it as a summarizing term for both macro and micro planning. Either way, all of them roughly describe the process in the same way, as explained above.

2.2.4 Architectures

Commonly, architectures in NLG can be split into three types: modular, planning-based and integrated. This split, however, is not inherent to a specific method, as it is possible for a system to utilize parts of all these architectures. "In summary, there are at least two orthogonal ways of classifying NLG systems, based on their [design] or on the [methods] adopted in their development" [Gatt and Krahmer, 2018, p. 82f.].

Modular approaches are among the oldest NLG architectures and have been challenged by the other two approaches in recent years. The 'standard' in this approach is the pipeline architecture, which basically mirrors the steps explained in the components chapter above and goes from macro to micro planning and then to realization. Different modular approaches that diverge from the pipeline also use a similar structure of deciding the 'what' to say before the 'how' to say it. Even though some proposals for this approach differ in their specifications, they all respect the principle of keeping the tasks in NLG well-defined and distinguished. The pipeline architecture, however, has two major challenges: early decisions in the process, as they can have unforeseen effects on components further down the pipeline, and establishing constraints, such as a certain length on the text to be generated, since following them is rather difficult in an early, pre-linguistic stage of the pipeline [Gatt and Krahmer, 2018, p. 83ff.].

Planning-based approaches view NLG as a planning problem, where the original goal has to be split up into several sub-goals. In this context, text generation is seen as the execution of these actions to achieve a certain communicative goal. The planning-based approaches can cut across the strictly distinguished tasks of the pipeline architecture and incorporate a strategic aspect to the process, for example, combining 'what' to say and 'how' to say it as a set of operations. It can be done by planning through grammar, using the interpretation of certain linguistic formalisms, or by stochastic planning, which takes uncertainties into account and sees the planning of a suitable output of a text generator as a stochastic optimization problem. "A practical advantage to planning-based approaches is the availability of a significant number of off-the-shelf planners. Once the NLG task is formulated in an appropriate plan description language [...] it becomes possible in principle to use any planner to generate text" [Gatt and Krahmer, 2018, p. 86ff.].

Integrated approaches are currently the dominant trend in NLG, which are mostly data-driven methods taking advantage of the availability of big data sources. Depending on the field of application, a variety of different approaches for the integrated architecture have been made, like viewing it as a sequential or stochastic process, applying deep learning methods or encoder-decoder structures. Viewing text generation as a sequential

or stochastic process (if data and text are aligned), modeling the NLG process can be done with statistical alignment to inform content selection and deploy NLP techniques for sentence planning and realization. An example of this approach would be Markov-based language modeling, which has been a prominent approach to data-driven NLG since the early 2000s. The first models of this kind involved content planning and realization, based on a dialogue corpus, and is therefore known as two-step process. As these approaches rely on standard language models that are founded on local history assumptions, prior selections influencing current ones is rather limited here [Gatt and Krahmer, 2018, p. 92]. Another integrated approach for data-driven NLG can also be done using deep learning methods like neural networks. This kind of architecture has profited from growing interest in recent years due to hardware improvements that allowed for the handling of resource intensive learning problems. Neural networks exploit the principle of backpropagation, allowing them to learn representations at increased levels of abstraction and therefore enabling them to capture grammatical and semantic generalizations very well. Neural networks have been notably successful in recent years in many variations, like feedforward networks or recurrent neural networks [Gatt and Krahmer, 2018, p. 97]. A third integrated approach and advancement from the previous one are encoder-decoder architectures that use two separate recurrent neural networks encoding input into vector representation and decoding it into the desired output. This splitting enables it to be used in other NLP tasks as well and is suitable, for example, for the so-called sequence-to-sequence text generation. For NLG, its most common use is response generation or dialogue generation [Gatt and Krahmer, 2018, p. 98f.].

2.2.5 Corpora

Generally in computer linguistics, the corpus is a collection of spoken or written language and is usually pre-edited to be machine-parsible, independent of its later use. Texts that have not been edited and for example contain HTML-code from web pages can also be called corpora if they are used for (computer) linguistic purposes. It is a rather important aspect for any NLP program, specifically for NLG, as the generators need it for training, developing, and testing [Zinsmeister, 2010, p. 482ff.]. A typical corpus consists of three parts: the basic data from texts, which can be voice recordings or their transcripts, annotations of this data, and the meta data. The basic data is not only defined by whether it is from a written or spoken source. Its appearance concerning size, color, font, or the distribution on one or several pages are also features that can be encoded in the corpus. The annotation part of a corpus consists of several layers, beginning with the segmentation that decomposes the data into linguistic units like words or sentences, which can be followed by semantic, discourse- or even emotion-oriented annotation. In practical use, this part is often done using part-of-speech tagging with XML. The last part of a corpus is the meta data, which contains descriptions about the basic data and annotations as well as contextual information about publication dates or annotation guidelines [Zinsmeister, 2010, p.483ff.].

Classification of different corpora can be done using content specific aspects like the language of the primary data, their mono-, bi-, or multilinguality, or if they contain different data in several languages or translations of the same source data. Alternatively, aspects independent from the content like medium on which the data lies, annotation

type, size, or persistence can be used. For the development of corpus-based programs, it is important to separate the corpus into three parts. The majority of the data will be used as a training corpus to train the program about rules and probabilities. Two smaller shares of it will be used as a development corpus to test the program during its development, and as a testing corpus, to test the final product [Zinsmeister, 2010, p. 486ff.].

A major advantage for NLP corpora has been the rise of social networks in the last decades. It brought the potential to counteract the knowledge acquisition problem in computer linguistics, as it offers an enormous and rich data base, which again can be used to define task specific corpora of different languages, domains, etc. for natural language processing. Using the World Wide Web as a corpus can be done with programming interfaces to access statistical information of search engines, but poses the problem of needing a significant amount of memory or computing capacity. Therefore it is important to remove irrelevant content, duplicates and spam, all while considering legal restrictions of use for the data [Gurevych, 2010, p. 544ff.]. In the application example of Wikipedia, the guidelines for creating an article are mainly focused on encyclopedic interest, therefore containing mostly nouns as titles, with few adjectives and verbs that mainly refer to the noun articles via links. A source of lexical-semantic relations in Wikipedia is usually the first paragraph of an article, as it contains a short definition of the topic and several related terms. Combined with all other outgoing links of that article, an article graph can be built. Furthermore, Wikipedia employs a system of semantic tags to categorize the articles, with ambiguous terms having a separate page for word-sense disambiguation [Gurevych, 2010, p. 547f.].

2.2.6 Challenges and Problems

Even though NLG has been a topic of scientific research for years and several advances and different architectures have been developed, this field still has several problems and challenges that text generation systems encounter regularly. Firstly, a general problem in theoretical research on NLG is the comprehension of the three main tasks. The final part of the classical pipeline-architecture, the realization, generally has a fairly deep, scientific understanding. For the microplanning, the understanding of approach and process is not as distinct, but still reasonable. It might be comprehensible for tasks like generating noun phrases for referring objects, whilst being unintelligible for aggregation or affective issues in lexical choice. Currently, the weakest understanding of all concerns the document planning, as it is mostly based on empirical work in one domain and not on theoretical models [Reiter, 2010, p. 586].

A major difficulty in the practical use of natural language generation is, as David McDonald calls it, the 'relative stupidity' of computer programs, which tend to rather work on templates than to process emotional or rhetorical attitudes of the people using them and therefore do not consider enough information to generate a natural language. Another difficulty is, where the starting point of these generators is. This comes from a lack of information from its human counterpart, as linguistics still do not know where people start to produce utterances. Due to this inconsistency of the generator source, recent comparative evaluation of systems has strongly focused on sub-problems like referring expressions, rather than on the overall generation [McDonald, 2010, p. 123f.]. Going further from this, "[the] speaker as an application program carrying out a task, has a pragmatically complete

but conceptually impoverished model of what it wants to relate to its audience. Concepts that must be explicit in the text are implicit but unrepresented in the application's code and it remains to the generator [...] to make up the difference" [McDonald, 2010, p. 130f.].

Among the architectural issues with natural language generators, the so-called generation gap is an often-discussed example. It occurs due to the fact that text generation is no task of linguistic nature, which creates the need for ensuring the ability of composition when transitioning between non-linguistic and linguistic elements in both text and sentence planning. As developers usually implement a unidirectional flow from macro planning to the realization - either due to ease of implementation or because of other reasons - the system is in danger of running out of resources to express a plain message and becoming deadlocked. The factor of control adds another challenge to this, as the number of possible ways to express something grows continuously, and concerns decisions on when to use certain clauses, when to terminate the system, or how to apply conflict management [Bateman and Zock, 2012, p. 300][Horacek, 2010, p. 438f.].

One final problem with NLG might not directly be connected to the generator itself, but to its output. With increasing text quality and performance of these generators, the risk of for example generating and accidentally sending a text with wrong or even malicious content also grows. If it happens due to laziness or inattentiveness, the question will be whether or not 'it was the computer, not me' will be an acceptable excuse [Dale, 2020, p. 486].

2.3 State of Research

The Proceedings of the annual conference on Empirical Methods in Natural Language Processing (EMNLP) give a very profound insight into current trends and research focus in NLG. Based on a brief review of them, four NLG topics - among others - have been strongly present in the conference in recent years and will shortly be explained in the following.

2.3.1 Data-to-Text Generation

Data-to-text generation can be seen as creating textual descriptions from structured input data and is a task employed by many real world applications, i.e., the aforementioned weather forecast generator or dialogue response systems. It is a rather difficult task, as it requires generating the text from structured data, which is why the typical system in this field is built upon a template-based approach. With the developments in the field of deep learning, scientific focus has shifted towards end-to-end neural generation models, a more performant approach to this with existing large datasets [Chen et al., 2020, p. 8635f.].

The first selected study in this field sees the problem that, despite data-to-text generators using deep learning, the recent models are focused on fully supervised learning. According to Chen et al., current models already achieve good results in this task, but usually are set in a fully supervised setting with sufficient amount of structured data and therefore are hard to adopt for real-world applications. To solve this, the authors propose a knowledge-grounded pre-training for the text generation, a model that can deliver strong results with only a small number or no labeled data at all. The data pairs are constructed

by crawling hyperlinks in Wikipedia and then linking these to their source Wikipedia article, which is then used to build a knowledge graph. The combination of this graph and the selected texts provides certain distant supervision for the corpus it produces. This corpus is then used in pre-training the proposed model, which is subsequently finetuned for the task it is applied to. To prove the effectiveness of this approach, the authors tested it with three different table-to-text tasks on different datasets. The results showed that the knowledge-grounded pre-training did not only outperform prominent models like sequence-to-sequence generators in its standard, fully supervised setting, but also in few- and zero-shot settings. Additional human evaluation supported these results, stating that pre-training enhances the model's understanding, therefore reducing the problem of the model generating non-existent facts, also known as over-generation [Chen et al., 2020, p. 8635ff.].

In the study of partially-aligned data-to-text generation, Fu et al. focused on the fact that this kind of text generation is usually based on well-aligned data, which is difficult and expensive to obtain. Their solution is to use partially-aligned data that can be produced automatically and therefore is easier to obtain, to solve the data scarcity problem. It allows for application in a broader array of domains, as this kind of data does not require each part of input and output to be aligned with one another. The problem with using partially-aligned data is that current generators tend to over-generate text, adding sentences completely unrelated to the input data. To tackle this problem, the authors also introduce a distant supervision to their approach [Fu et al., 2020, p. 9183f.]. Their system is based on a sequence-to-sequence model, with the distant supervision consisting of a supportiveness estimator, the generator, a supportiveness adaptor, and a rebalanced beam stream. The supportiveness estimator is pre-trained to construct a supportiveness vector for estimating whether each target word is describing a knowledge base triple of the input data. The sequence-to-sequence generator then calculates the generation loss, which the supportiveness adaptor then combines with the previously built vector. Eventually, the rebalanced beam stream combines all of this with the probability distribution of the words to get a better generation result. Experimental results show that implementing that stream can alleviate the over-generation problem, especially with the last to components of the distant supervision, overall generating better sentences compared to other data-to-text models [Fu et al., 2020, p. 9184ff.].

Data-to-text generation can also be done with the nearest neighbor principle, where neighbors usually are examples of qualitative text from a database paired with the source information of a sample that is used for concrete generation, in order to simplify the generation or at least make it more controllable. Adapting this, Wiseman et al. tried to generate text by directly splicing together segments from the text from retrieved neighbors, which has several advantages such as a more interpretable generation due to clarity from which neighbor a piece of generated text derives, and gives more control over the generated text, for example preventing it from being harmful or biased. This is achieved by training a policy that replaces or inserts spans of neighboring text and implementing an insert function to realize this policy. It follows a principle of defining an oracle-like sequence of insert actions for each training example, while the proposed insertion function adapts a generation scheme, making only minor changes to some existing text, for

example copying a neighbor sentence whilst only replacing names or dates. In order to use this function for generation, a parameterized distribution has to be learned, which is done with the aforementioned oracle policy. This basically means that for each true dataset output, an oracle sequence of canvases paired with corresponding oracle actions has to be provided [Wiseman et al., 2021, p. 4283ff.]. The experiment to this approach was evaluated with standard automatic metrics, showing the oracle policy performs better than other baseline models with pre-training. The human evaluation also proved that the proposed model is as faithful as the most advanced generator in this field, although it generated information contradicting the sources more frequently. For controllability, the authors conducted a separate case-study, which indicated that the oracle policy allows full control when imposing certain generation constraints [Wiseman et al., 2021, p. 4288ff.].

A final study about this topic engages with the output of existing models: whilst they are able to produce fluent and coherent sentences mostly for short texts, they fail to capture semantic structure and the diverse surface forms of such texts, when it comes to longer and more diverse texts however. This is because they are not able to model input data dynamically during generation, lack a high-level planning - which leads to not very well capturing of inter-sentence coherence - and can only express low-level variations that are lacking diversification. The proposed solution for this is a planning-based hierarchical variational model for better input modeling and alleviating the incoherence problem between sentences. This model does a planning phase at first, segmenting the input data into groups, before generating sentences based on the corresponding group and preceding generated sentences. With this approach, long text generation is decomposed into dependent sentence generation tasks. To also ensure the diversity of expressions, the model implements latent variables in both the planning and realization phases [Shao et al., 2019, p. 3257f.]. The performance of this model was tested on two long-text generation tasks, namely advertising text and recipe generation. The automatic evaluation resulted in mixed outcomes: overall, the approach received good scores, excelling with several evaluation metrics. However, with two of them, it received very poor results, which the authors tried to explain with the model's diversity of expressions, leading to a lot of different ways to express the same content. Additional human evaluation supported the models good performance, resulting in significantly higher scores regarding grammar and coherence [Shao et al., 2019, p. 3262ff.].

2.3.2 AMR-to-Text Generation

"Abstract Meaning Representation [...] is a linguistically-grounded semantic formalism that represents the meaning of a sentence as a rooted directed graph, where nodes are concepts and edges are semantic relations. As AMR abstracts away from surface word strings and syntactic structure producing a language neural representation of meaning, its usage is beneficial in many semantic related NLP tasks [...]" [Ribeiro et al., 2019, p. 3181]. Even though it technically is a subtask to data-to-text generation and has been very prominently mentioned in the EMNLP proceedings of recent years, it will now be explained separately.

The problem with AMR-to-text generation is capturing the complex structural information stored in graph-based data. Graph neural networks have been a recent approach to deal with this problem, which is why in 2019, Ribeiro et al. proposed a graph-to-sequence

approach for this issue. It is inspired by pre-neural generation algorithms that promote a combination of top-down and bottom-up approaches as the most performant way for graph traversal. Therefore, the input graph for this style of generation is represented by a top-down and a bottom-up structure. Early approaches in this field were driven by linearization and put the focus on in-depth traversal and several other graph-to-sequence models with convolutional networks that failed to perform sufficiently. Contrary to that, the proposed approach eases the burden on a neural model during encoding a single vector representation, while also eliminating the need for additional positional information in the graph. The dual graph encoder, one for the top-down graph and one for the bottom-up, encodes structural information and learns node representations from both perspectives. This setup makes the approach similar to a bidirectional recurrent neural network, which benefits from propagation in both directions and therefore learns representations [Ribeiro et al., 2019, p. 3183ff.]. For the experimental setup, the authors employed two AMR corpora, in which each instance contains an AMR graph and a sentence. They extracted vocabularies from the corpora and trained three graph neural network models with their approach. In the automatic evaluation with BLEU and METEOR metrics, their model significantly outperformed the baseline model and other state of the art results of the test datasets. In contrast to the other results of the datasets, this approach leverages neither supplementary syntactic information, nor anonymization pre-processing. Human evaluation accredited the model a mediocre performance concerning meaning similarity and readability, which still outperformed the baseline model [Ribeiro et al., 2019, p. 3188ff.].

Ribeiro and his colleagues refined their approach in 2020, again focusing on the problem that pre-trained language models cannot directly be applied to AMR- or graph-to-text generation due to the structure of the input in this task. The existing linearized approaches suffer from the limitation that these AMR graphs have a different structure than natural language, which is why pre-training knowledge is not fully transferred. Additionally, the linearized representations weaken the structural information of the graphs, resulting in the language model's need to infer graph edge connections. Their new approach, called STRUCTADAPT, tries to encode a graph into a pre-trained language model by adding layer-wise modules to extract information and employing a graph convolution to learn representations built upon the graph. This allows for deep integration of both linguistic and graph knowledge. To realize it, a graph-to-text model based on an encoder-decoder architecture is used, which is finetuned and then trained in an adapter module. STRUCTADAPT then injects structural inductive bias into the pre-trained model for the graph-to-text task with only a small number of parameters. The model's performance proved to be substantial with evaluation metrics, with slight improvements compared to the baselines models. Especially the proximity in meaning of the generated text to its reference or input sentences was a strong indicator for its performance [Ribeiro et al., 2021b, p. 4269ff.].

With a focus on the decoder side in encoder-decoder architectures in AMR-to-text, Bai et al. tried to enhance it with online back-parsing. With this, the AMR graph is built through autoregressive sentence construction, saving the need for a separate AMR parser, better preserving structural information of the graph. The used model in this study is a graph Transformer model, which exploits a graph Transformer encoder and a standard Transformer decoder for text generation. The decoder is then enhanced with back

parsing, employing graph prediction to reconstruct the AMR graph, by jointly predicting the nodes and projected relations when generating a word. Additionally, the online part of back parsing is integrated with the decoder using the last node and edge predictions to better inform the generation of the next word. The proposed model did not achieve results as good as some externally fed baseline models but outperformed any of them in a scenario with no external data. It also achieved strong results concerning fluency and concept preservation rates [Bai et al., 2020, p. 1206ff.].

Similarly, another study also attempts AMR-to-text generation using a Transformer-based model by using a structure-aware self-attending encoding approach in order to incorporate structural information better. Zhu et al. built their generator based on a standard Transformer model consisting of stacked encoder and decoder layers. Each of these has two sub-layers, a self-attention layer that employs multiple attention heads, whose results are concatenated and transformed to be the output of this layer, and the feed forward layer. To model the graph structures into the Transformer, the original AMR graph is extended to be compatible with sub-words and an extended version of conventional self-attention architecture is used to encode the relation of an element pair in the model. These concept pairs can be done feature-based, leading to a large number of different features; average-based, using the average of all label embeddings; self-attention-based to obtain hidden states by getting the sequence of the word embedding; or with a convolutional neural network. Again, this approach based on a Transformer architecture proved itself to be more performant than other baselines models, especially when using self-attention or the convolutional network [Zhu et al., 2019, p. 5459ff.].

Finally, an important topic in AMR-to-text is multilingualism. The basic problem is that structured data such as graphs are lacking certain parts of language, i.e., function words, which leads to a gap between input data and the natural language to be generated. The multilingualism is an additional challenge since training datasets only exist in English, and other languages differ from it concerning morphological and word order properties. Fan and Gardent's proposal for this is to create multilingual training data with AMR to annotate the English data used to generate text. They evaluated recent advances in NLP like cross-lingual embeddings to improve the text quality and use this combination of techniques as a possibility to fluently and multilingually conduct AMR-to-text generation. The used method applies a neural sequence-to-sequence model, which uses automatically derived English text input and is pre-trained, before generating natural language. For the encoding, the AMR graph is first linearized and modeled with graph embedding, providing additional information to the encoder by in-depth traversal. The model is trained on the AMR and on text in different languages, to enable the decoder for the generation of multilingual text with the right variations in word order and morphology [Fan and Gardent, 2020, p. 2889ff.]. Experiments showed the model to perform good results throughout different languages, achieving similar results to a hybrid approach of first NLG and then machine translation, but with a simpler approach and less data needed. Even though AMR was designed to describe the meaning of English sentences and is therefore strongly biased towards this language, this shows that it can definitely be used for multilingual generation [Fan and Gardent, 2020, p. 2894ff.].

Reibero et al. again conducted a study concerning this, supporting Fan and Gardent's findings. They also note that whilst studies already found parsers to be capable of

transforming multilingual text into English and the possibility to do it vice versa, multilingual AMR training data is still scarce and, for a study in this field, has to be generated synthetically. To easily acquire large amounts of this data, they propose providing an analysis of different augmentation techniques, parsing English sentences into AMRs, and then using machine translation to acquire text in a target language from the AMR-to-text corpus. The authors chose an English-centric AMR graph as input and employed an encoder-decoder architecture, where the encoder learns language agnostic representations to be used in multilingual setups, which the decoder is then required to turn into text in different languages. The selected techniques for the data augmentation use multilingual parallel corpora to generate AMRs for the respective English sentences and the encoder-decoder architecture as a translation model with the same parallel sentences. On their own, the latter options significantly outperform the former but, as they are meant to complement each other, achieve the best results when combined. The combination of them can overcome the problem with the lack of training data, achieving state-of-the-art performance in multilingual text generation [Ribeiro et al., 2021a, p. 742ff.].

2.3.3 Low-Resource Text Generation

When generating text, most methods use hand-crafted rules leading to rather monotonic sentences, while also needing a substantial amount of programming or engineering effort. Even though recent methods have improved this, they still have one problem: "[...] these methods are typically developed for particular domains. Moreover, they are often data-intensive to train" [Mi et al., 2019, p. 3151]. High annotation costs and the time-intensive data acquisition lead to a lot of developers not building new NLG systems from scratch. Therefore, training a natural language generator that can be generalized for several domains with a reasonable amount of data is desirable and referred to as low-resource text generation. In this field, it is usually attempted to finetune pre-trained systems with small amounts of training data, in order to prepare the generator for new tasks. Mi et al. tried to achieve this from a meta-learning perspective with three categories: metric-based, which uses metric to compare the low-resource samples to rich training data; model-based, which attempts to update the original learner with few training samples; and the optimization-based perspective, which attempts to achieve good parameter initialization for easy finetuning. To build this model, the idea is to repeatedly simulate auxiliary meta NLG tasks that mimic the finetuning process. These meta tasks consist of two independent utterance pairs used for the pre-training. They are constructed according to the principles of task generalization, meaning each task follows the same modality as the target task, and low-resource adaption, which limits the utterance subsets in every meta task to a rather small size. The NLG's parameters are then optimized based on the first, and then the second subset of each meta task. This approach was tested based on a long short-term memory generator, implemented in four different model settings. Generated utterances on several application scenarios were shown to human annotators, with the proposed meta NLG achieving high scores in informativeness and naturalness [Mi et al., 2019, p. 3152ff.].

Tran and Nguyen examined the performance problem that low-resource NLG systems tend to have. They offer two solutions to this by adapting the domain to learn from a sufficient source and designing the model for a low-resource setting. The first solution is implemented with a multi-domain, variational autoencoder that is based on a standard

autoencoder employing variational neural inference, a principle using a neural network to approximate the distribution of a latent variable, and reparameterization, which alters this variable instead of using a Gaussian noise variable for sampling. This is integrated into a standard encoder-decoder model and, combined with a convolutional neural network, tackles both problems of domain and text similarity. The approach was tested in settings of training from scratch or with a pre-trained and then finetuned model, resulting in good scores from the automatic evaluation metrics throughout all domains and training scenarios [Tran and Nguyen, 2021].

To avoid degenerate text, modern generators require a variety of parameter settings and techniques, creating a kind of mismatch between training and testing conditions. The problem with this is that these models are trained to maximize the likelihood of the observed text, leading to either a deterministic search for the most probable sentence that results in a rather repetitive text, or to stochastic sampling with many implausible words when the input data is cut short. Additionally, language modeling's most commonly used metric, the perplexity, cannot handle sparse input and therefore cannot compare different strategies for truncating data. Martins et al. propose the so-called entmax transformation for this in order to train and generate text from a relatively sparse language model [Martins et al., 2020, p. 4252f.]. "Entmax transforms a vector of scores into a [sparse probability distribution], preventing implausible words from receiving [any] probability mass" [Martins et al., 2020, p. 4253]. Usually, language models generate sentences word by word, sampling from the learned probability distribution. Every word in the vocabulary receives some probability above zero, leading to the mentioned degenerate text that researchers try to counteract with additional parameters. With the entmax approach, sparse probability distributions can be generated with several, improbable words receiving a probability of zero. Therefore, the model is also highly adaptive to the degree of uncertainty in a specific domain. The authors tested their method on the three tasks of language modeling, story completion, and dialogue generation. Evaluation metrics and human evaluation both resulted in high scores in all three tasks, with the entmax approach reaching a high diversity in words, almost matching human-written texts, and good results concerning fluency and coherence [Martins et al., 2020, p. 4254ff.].

A more specific approach for low-resource generation is pursued by Liu et al., who applied it to knowledge-grounded dialogue, a task of generating a response based on the dialogue history and external knowledge. This is a rather complex task, heavily relying on the training data and therefore being difficult to transfer to a new domain with limited input data. Consequently, most current generators perform poorly in this field. The proposed solution is a so called three-stage learning framework, which attempts to divide the used parameters into dialogue- and knowledge-integration-related ones. The first stage uses supervised learning for pre-training the dialogue parameters with general dialogue, followed by a domain-adaptive pre-training for the knowledge-related parameters. The second stage matches a set of pseudo-knowledge for each dialogue to construct a knowledge-grounded dataset, and then further pre-trains the parameter sets of the first stage on this data. In the last stage, the model is finetuned on the respective low-resource data of the target domain [Liu et al., 2021, p. 2262]. A Transformer-based generator called KAT (Knowledge-Aware Transformer) is used for this approach and accepts not only

dialogue history, but also external knowledge as additional input. Based on this three-stage framework, experiments were conducted on two public knowledge-grounded datasets. The results showed that KAT itself is a state-of-the-art model for dialogue generation. In combination with the three-stage learning framework, even with no additional resources in the final stage, it surpasses the performance of other dialogue systems and, with increasing input, generates more fluent and contextually coherent responses compared to the baseline models [Liu et al., 2021, p. 2265ff.].

2.3.4 Dialogue Generation

Continuing the application of the previous study, dialogue generation is a commonly researched topic in NLG in recent years. In practical use, especially generation-based chatbots have achieved significant outcomes, even reaching human-like levels on specific datasets. As these are mostly based on pre-trained models and usually restricted to generate only recent or specific topics, this leads to poor performance with input terms that are not part of the training data [Cui et al., 2021, p. 2328].

One problem with current chatbots is their knowledge base. In the example of the social network Reddit, the natural dialogue on this platform is so diverse that almost half of all topic words are not covered in the knowledge base, with another quarter being ambiguous. As these generators heavily depend on the coverage of their knowledge base, a non-existent or ambiguous word might lead to very poor performance. Cui et al. tried to solve this with a knowledge-enhanced finetuning method, which at first tries to interpret a word's meaning by its context, and then generates a hypernym for that word. This forces the model to learn semantic information from the knowledge base and therefore generates a better understanding of the input utterance. The idea of knowledge-grounded dialogue generation has already been researched for a few years, with proposals of using profile information, posterior information or reinforced learning with unlabeled data being examples of the solution approaches. The proposed framework, however, tries to use both context and knowledge as input. If an input utterance contains an unknown word, it will be marked as masked and a definition, i.e., retrieved from Wikipedia, then guides the model to understand the background knowledge of the masked word. At the same time, topic words in the input utterance will be replaced by their hypernyms with the support of the application WordNet in order to guide the model to understand semantic information of unseen words [Cui et al., 2021, p. 2329ff.]. The experimental setup in this was based on the test sets of Wizard of Wikipedia and Reddit Trendings and, even though it was designed for settings with unavailable knowledge, the approach performed competitively in a setting with available knowledge. In its intended setting, it outperformed all comparison baseline models, since most of them cannot be applied without available knowledge at all, though not quite achieving a human-like performance [Cui et al., 2021, p. 2332ff.].

Taking this approach further, with the growing amount of data in terms of human conversations on social media, the interest in research on open-domain systems for dialogue generation has increased over the years. The problem with this is that these open domain systems usually tend to generate very generic and bland responses, especially when trying to get into a specific topic. In recent research, two approaches have emerged to counter this quality issue. The first approach is to apply pre-trained natural language generators to dialogue generation which, even though these enormous language models can capture

a lot of data, only produces average semantics and still results in bland answers. The second approach is to use extra knowledge from unstructured data like Wikipedia articles, which leads to the problem of finding enough dialogues that are based on these documents to train a model. Zhao et al. suggest using the already mentioned knowledge-grounded models to acquire a model with good knowledge and generalization capabilities. As pre-trained models usually have constraints on the maximum number of tokens generated, they equipped the model with a knowledge selection module that can slim down the redundant knowledge to meet the constraints. Additionally, they added an unsupervised approach, where learning the knowledge selection and finetuning of the pre-trained model are jointly done [Zhao et al., 2020, p. 3377f.]. This approach employs a GPT-2 model and, in order to apply it to the task of knowledge-grounded dialogue generation, faces challenges such as modeling the correlation between context and external knowledge. The authors therefore defined a context-aware knowledge encoder to take advantage of the pre-training, the interaction of dialogue context and associated knowledge, as well as a sequential knowledge selector to choose relevant knowledge depending on the context. Like the previous study, the experiment here was done using the Wizard of Wikipedia dataset, as well as a second one called CMU Document Grounded Conversations. The effectiveness of large-scale pre-training on language models is proved in the task of dialogue generation, as it achieves good results on both datasets. Concerning text fluency, human evaluation found it to be comparable to the baseline models, however exceeding them in terms of context coherence and knowledge relevance [Zhao et al., 2020, p. 3380ff.].

A rather similar approach to this is the one from Wu et al., who also view knowledge enhancing as the best possibility to improve dialogue generation and claim that existing work about this usually solely relies on a single source of knowledge. To make an NLG system have diverse conversations, a single knowledge base is too limited, also given that each knowledge base has its own advantages and disadvantages. Humans, however, utilize knowledge from different sources in a conversation, which leads to the assumption that dialogue generation could benefit from multiple sources. Yet, doing so leads to the challenge of dealing with irrelevant or conflicting information and altered word distributions. Therefore, this study proposes the multi-score heterogeneous knowledge-enhanced model, which uses additional common sense, text and info box knowledge to improve the dataset coverage. It also imposes a multi-reference selection mechanism to alleviate the conflict with irrelevant or conflicting topics, and a multi-reference generation mechanism to construct a unified dynamic vocabulary. The multi-reference selection is supported by the implementation of a relevance gate, checking each reference concerning their relevance and conflicts with other references. Following that, the multi-reference generation utilizes a dynamic vocabulary, which consists of all words appearing in the fixed vocabulary, and the reference set to eliminate conflicts brought up by different word distributions of the references [Wu et al., 2021, p. 2286ff.]. In the automatic evaluation, the proposed model achieved fairly satisfactory results concerning relevance, diversity, and knowledge. Extensive human evaluation also proved the model's outputs to perform well in the categories of appropriateness, with regards to fluency and context, as well as in informativeness - all whilst using fewer parameters and less training data than the models it was compared to. Therefore, using heterogeneous knowledge from multiple sources has proved itself to be advantageous in knowledge-enhanced dialogue generation [Wu et al., 2021, p. 2290ff.].

Recent studies on dialogue generation tend to split this task into the two sub-tasks of knowledge selection and knowledge-aware response generation. Even the use of variational models, which have proven themselves successful, has two major flaws. On the one hand, current models emphasize the aspect of knowledge selection without acknowledging that this is inherently coupled with the knowledge-aware response generation. On the other hand, while knowledge selection diversity is continuously improved, the diversity for response generation is still being neglected. To improve this, Zhan et al. employed a collaborative latent variable model, which, during generation, draws a knowledge candidate from latent space conditioned on the dialogue text, then samples a response from another latent space conditioned on context and knowledge. This latent variable is constructed based on a latent space that depends on the knowledge latent space conditioned on the dialogue context. In fact, the model employs two of these content-dependent variables: one for the dialogue response, and one for the dialogue knowledge [Zhan et al., 2021, p. 2250ff.]. The study's experimental setup was built on the two knowledge-grounded dialogue datasets of the already mentioned Wizard of Wikipedia and Holl-E. Results of the automatic evaluation show that the collaborative latent variable model performs significantly better than many of its baselines, also achieving better knowledge selection performance. Therefore, the authors conclude that the model has the ability to close the gap between knowledge and response as it improves the performance of the knowledge selection. In line with that, human evaluation also testifies it to produce significantly more qualitative responses that contain a high level of diversity concerning its given knowledge [Zhan et al., 2021, p. 2256ff.].

Finally, a completely different issue for this task is seen by Zhu et al.: "[Dialogue generation] suffers from data insufficiency problem as there may exist many potential responses for a given dialogue history" [Zhu et al., 2020, p. 3438]. The best solution to solve this problem would be to explore potential responses by directly chatting with a user, which is rather impractical due to time or labor constraints. Employing a simulator would help with this, however, it can only approximate real user behavior. Therefore, a different approach would be - given a specific response in a dialogue - to consider the effect another response would have had. This retrospectively created response is also called a counterfactual response, as it is the result of posterior reasoning about the outcome of alternative actions whilst leaving every other aspect unchanged. The authors utilized this approach and proposed a counterfactual off-policy training to explore potential responses. This method employs a structural causal model which describes generation with scenarios, a random noise variable to capture unobserved environment aspects, and causal mechanisms, a function that combines a scenario and dialogue history to a response. The latter can be fed with an observed response to generate a counterfactual response, which then leads to the generator optimizing itself accordingly and a higher quality output than in standard dialogue generation. Counterfactual reasoning has its origin in psychology which, combined with a structural causal model, is proven to improve the performance of reinforced learning [Zhu et al., 2020, p. 3438f.]. The basic generator for this approach is a standard sequence-to-sequence model which uses a recurrent neural network to read the dialogue history during generation. With a so-called Gumble-Max trick, the generator is turned into a structural causal model, which - again - views every response as being

generated within a scenario. The generator as well as a discriminator are firstly trained on an adversarial learning framework, where the generator tries to produce human-like responses and the discriminator tries to distinguish whether an utterance has been human- or model-generated. Concrete pre-training of the discriminator then depends on the specific model it is applied to. The experimental setup in this study differs from the previous ones in so far, that several different generator models were implemented with and without this counterfactual off-policy training. Automatic evaluation showed mixed results, as two of the models did not improve at all with this extension, which the authors tried to explain with the used dataset not being big enough to effectively train the discriminator. With the other two models, however, the extension showed significant improvements that were partially confirmed by human evaluation [Zhu et al., 2020, p. 3440ff.].

Chapter 3

Implementation

Following the state of the art in NLG, this chapter now provides a detailed description of the practical implementation in this thesis. Following a short explanation and plan for the implementation, the selected NLG methods are introduced, briefly going into the concrete implementations with their characteristics and potential implementation problems. After that, the practical execution is described step by step, beginning with the used software and data basis, the pre-processing of the data, and the training and generation with the NLG systems.

The plan for the implementation follows the research question mentioned in the introduction and the emphasis set in the second chapter. With the use of NLG systems, several texts are supposed to be generated which resemble the structure and content of a Wikipedia article but contain false information. As mentioned before, a variety of different contemporary NLG systems exist. Because of that, various approaches to NLG will be implemented, to allow for a comparison regarding their performance in this task, but also to give insight into the shortcomings of each of them. Once implemented, the systems will be trained with data from Wikipedia articles, and then generate the text. To allow for good qualitative and quantitative evaluation of the generated texts, this generation has to be repeated several times and the texts saved to dedicated files.

3.1 Implementation of Natural Language Generators

Within the scope of this Master thesis, five Natural Language Generators have been implemented according to the integrated approaches mentioned in the 'Architectures' section and following the article of Wigmore [2021] concerning state of the art methodologies for NLG. This chapter explains the source and specific characteristics of each implementation, as well as problems and adaptions made during the implementation phase. A detailed explanation of their functionality and user instructions will be provided in the code documentation, which is uploaded to the project's GitLab, a copy of which is on the CD attached to the back of this thesis.

3.1.1 Markov chain

The first integrated approach to be implemented was the Markov chain. Being a common principle in NLP, it works as follows for text generation: It predicts each following word solely based on the current one by considering the likelihood of the connection between these words, not looking at the text generated before. This approach has been especially used in early cell phone technology for word-prediction while texting. The strength of Markov chains is significant with the use of correctly labeled data, where it significantly outperforms deep learning methods. This is why, contrary to the current trend of neural networks and Transformers that ousted Markov chains as top interest of research, they are still an effective and state of the art approach to text generation [Kumar and A, 2021, p. 43][Abdelwahab and Elmaghraby, 2018, p. 252ff.].

The implementation of this approach takes up a special role, as it is bipartite. A web research showed that the majority of available Markov chain implementations were generating text randomly based on the training input. Additionally, the implementations that allowed for some form of input could be trained on individual data, but mostly only took one word as input for the generation. For this, the implementation from an anonymous author with the pseudonym "b" was used, which from now on will be referred to as the Markov chain (one word) model. To be more specific, the author described this model, or Markov chains in general, as being simulators for a sequence of events and elaborates that, even though they are not the most reliable method for predicting these events, they are used especially in situations with fixed probabilities. Taking the training input, this method therefore counts every word in it and generates a distribution of words dependent on the preceding word. For the exemplary implementation, a collection of speeches from Donald Trump in his 2016 campaign was used as training input, containing around 166,000 words [n.a., 2017]. As mentioned before, the only change made during implementation was concerning the input word for generation: By default, this implementation started generating on the basis of a word randomly selected from the training corpus. To allow for a more targeted generation, this was changed to being a custom word.

With regards to the following models, which take complete sentences as input, and also to evaluate how a Markov chain NLG performs with different amounts of input, a second model with the capability to generate from sentences was implemented. The implementation for this from Ryan Thelin has a more complicated approach to building Markov chains than the previous one as it works on a character level. It uses multiple words as input and is thereby compliant to the other models, it will be referred to as the standard Markov chain model from now on. It firstly constructs a lookup table with the occurrences of all characters, looking at a specific number of them at once and their following character, before converting these frequencies into probabilities and, building the Markov chains. Then, the text is sampled by looking at the previous characters and combining it with the model to generate text character by character. Compared to the other Markov model, this implementation was exemplarily trained on an undisclosed political speech that only contained around 800 words [Thelin, 2020]. Again, this model did not need to be adapted significantly. It is finely divided into step-by-step methods and contains a lot of intermediate steps and informational outputs. It has been condensed into one method and the unnecessary outputs were removed.

3.1.2 LSTM with RNN

The third implemented NLG system is based on a recurrent neural network (RNN). This approach tries to imitate human brain activity by passing all information through a feed-forward structure. It employs a cyclic approach, which stores the words encountered in the past in every cycle, ascribes a likelihood for each word depending on its predecessor, and selects the one with the highest likelihood. Compared to basic neural networks, the RNN's network loops allows it to keep information for subsequent computations, which can be useful in tasks where the context of previous words is needed to predict the following ones. Additionally, this approach has the ability to process sequences of any length. Standard RNNs, however, do have a problem with long-range conditions between words, the so called exploding gradient problem, making them barely used in practice. This issue is solved with an adaption of the RNN, the long short-term memory (LSTM) that controls the amount of information passed on, and hence addresses long-term dependencies. A LSTM's central component is the cell state carrying information from one cell to the other. Information is altered between cells using gates: the forget gate, which controls the amount of information kept in subsequent computations, and the input gate that decides what new information will be stored in a cell. This extension allows the cells in the neural network to retain their state over a long period of time, but also to dismiss old and add new information continuously. Contrary to basic RNNs, the LSTM is already a widely applied approach in speech and natural language processing [Kumar and A, 2021, p. 43] [Balaji et al., 2016, p. 3ff.].

The implemented model for this is from an online instruction provided by Abdou Rockikz and is the most extensive of all implemented models. It starts by splitting the training data into single characters, therefore obtaining a vocabulary with the total number of characters as well as the number of unique characters, which are then encoded into integers. This dictionary of encoded characters is then used to encode the complete training text. For the actual training, the method now analyzes the training input in sequences, examining a given sequence of characters, saving which character succeeds this sequence, and then moving the sequence further by one character, thereby creating training samples. The trained LSTM model is built and then saved as an external file, which can be used for generation. Finally, the dictionaries as well as the model have to be loaded, which then allows for generating text of variable length and, as it is a character-based approach, it is not limited to English text like other NLG systems. The author trained his model based on an abstract of Lewis Carroll's "Alice in Wonderland" with almost 30,000 words [Rockikz, 2019].

It is important to note that this implementation differs from most NLG implementations, as they do not generate the sentences from a word-based, but from a character based level, sharing this feature with the second, standard Markov chain model. A reason behind this approach can be seen with the two main shortcomings a word-based model can have: considering every word in the training data would make the number of possible output classes huge, leading to an optimization problem and the need for a significantly bigger training input. Additionally, the set of possible output would be limited to words in the input and therefore limit the creativity of the text, whilst character-based models can build any word from scratch [Lippi et al., 2019, p. 3328].

3.1.3 GPT-2

A new approach for natural language generation can be found in Transformer-based systems, which use a pile of encoders for preparing contributions of any length and a pile of decoders to yield produced sentences. Contrary to LSTM, which performs a number of interleaved steps, Transformers work with a small, steady number of steps. "In contrast to past models, the Transformer utilizes the portrayal of all words in setting without packing all the data into a solitary fixed-length portrayal that permits the framework to deal with longer sentences without the soaring of computational prerequisites" [Kumar and A, 2021, p. 44]. Today, one of the most advanced models is the series of the generative pre-trained Transformers (GPT), with the most recent version being GPT-3. It is trained on 175 billion parameters, making it the largest language model currently available, and its output is difficult for human readers to identify as machine generated. Other Transformer-based models include the Bidirectional Encoder Representations from Transformers BERT, which is used as a pre-training technique, or XLNet, which tries to overcome the shortcomings of GPT and BERT with the ability to model bidirectional contexts by maximizing the expected likelihood and using autoregressive formulation [Topal et al., 2021, p. 2].

Staying with the GPT series, however, the dominant issue that the developers had with current machine learning systems was the fact that they were used for a specific task, which required the collection of a training dataset for this domain, training the system with it and then only applying it for this task. With their wish for generalization, the developers wanted a system to be able to perform many different tasks, even with the same input, and thereby shifting the focus of the system's conditioning from not only the input but also the desired task. In contrast to many other generators, which are being trained on a single domain like news articles or Wikipedia, the developers of the GPT series tried to build the dataset as large and diverse as possible. For GPT-2, the predecessor to the current version, they obtained this dataset by defining a custom web scraper that emphasized text quality and used the platform Reddit as a starting point. It scraped all outbound links, reaching an accumulated number of 45 million URLs, and resulting in a final data set of roughly eight million documents after de-duplication and basic learning. The final model uses around 1.5 billion parameters and has been tested on a variety of tasks, including summarization, translation and question answering, delivering good results in all of them. The authors decided not to share any information about the dataset, the training code, or any of the model weights due to concerns about them being misused for deceptive purposes. Yet, for the application scenario of this thesis, it is an important aspect that they stated that all references from Wikipedia were removed from this dataset [Radford et al., 2019a] [Radford et al., 2019b, p. 1ff.].

The concrete implementation of the GPT-2 NLG was done following an article of James Briggs. This version basically loads the pre-trained model and tokenizer, the latter of which is used to split the input text and translate it into numeric indices, before generating the text. [Briggs, 2020].

3.1.4 GPT-2 (finetuned)

As discussed in the previous explanations, the major problem with the generic GPT-2, or pre-trained models in general, is their approach to be generally applicable systems, no matter the task. Their key success factor is their use of very large training sets, which are predominantly collected from websites or social media. However, taking the example of value aligned text as a task, this approach fails as the dataset potentially contains text not abiding to social norms, like texts describing harmful acts or antagonistic behavior. Considering the fact that the generated texts might have significant impact on its recipients, it could be important to block an NLG from generating text with such content. Therefore, the task is to train the already pre-trained GPT-2 model to avoid such topics or, more generally speaking, to lead the text generator in a certain direction. This process is also called finetuning [Peng et al., 2020, p. 1f.]. Looking at the application scenario, finetuning the generic GPT-2 model could be a useful improvement for two reasons: Firstly, as mentioned above, in the training dataset of the GPT-2, Wikipedia articles and any content related to them were specifically removed. This stands in opposition to the research objective of this thesis to generate Wikipedia-like articles. Secondly, as the data aggregation originated on Reddit, a known platform that allows users to upload any kind of content, it is very likely that most of the data is not of long and informative kind that usually appear in Wikipedia articles - even though this remains an assumption as the training data is not available.

To incorporate these improvements, a GPT-2 finetuned model was implemented in the smallest form of 124 million parameters, with the approach provided by data scientist Max Woolf who already implemented a variety of different text generators. Its functionality is rather similar to the generic GPT-2 model: It builds a custom tokenizer based on the finetuning input and then builds a model based on this custom tokenizer and the pre-trained GPT-2 model configuration. This refined model is then saved and can be reloaded for concrete generation. As a reference, Woolf finetuned this implementation with a collection of plays by William Shakespeare, which contained around 203,000 words [Woolf, 2019].

This implementation had two special characteristics, which the other ones did not have and which made changes necessary: In the documentation available in the model's GitHub repository it is mentioned that finetuning GPT-2 models cannot be done on CPUs and most standard GPUs. Therefore, the training had to be outsourced into a Google Colaboratory project provided by Woolf, which is also the reason for finetuning the model in its smallest parameter setting. Additionally, the standard text generating function of this model directly printed the text out on the console, not allowing for controlled printing of it or saving it to a file. However, the model also provides a "generate_to_file" option, which stores the text to a numerically named file in the model's directory.

3.2 Execution

3.2.1 Software

The implementation of these models required a variety of software and modules, which will be illustrated in this chapter. Precise information about where they were used can be found in the project's code and the code documentation.

Overall, all coding was done on computers with the Windows 10 operating system, using the IntelliJ IDEA development environment. After a brief research on existing NLG implementations, the Python 3.9 version was chosen as the programming language. For general scripts, the Python modules OS and RE were used. The OS module provides an interface to the operating system, allowing for the manipulation of file paths or directories stored outside of a project's repository. It was mainly used to access and search through the files of the Wikipedia dump that was stored locally, due to its size [Python Software Foundation, 2022a]. The RE module, short for regular expressions operations, allows for specifying certain patterns in Unicode encoding, which can be used in searching matching strings [Python Software Foundation, 2022c]. Additionally to that, one library was employed by three of the systems. Both Markov and the LSTM implementation employ the NumPy package, which provides support for multidimensional arrays and matrices, as well as other high-level mathematical functions. In this project, it is used for random word selection, but also to create or return arrays of certain shapes [NumPy Developers, 2022].

The LSTM approach also uses the most additional packages, beginning with TensorFlow, an open-source platform for machine learning, that allows for the creation, training and implementation of a machine learning model with rather simple architecture [TensorFlow, 2021]. In this model however, it is used in combination with Keras, a deep learning programming interface, to simplify and accelerate programming with TensorFlow [Keras, 2022]. In the implementation, this combination is employed to handle the input data and to build the final model. Furthermore, the LSTM uses pickle, a module that allows for the serialization of Python objects by converting it into a byte stream, or vice versa for de-serialization [Python Software Foundation, 2022b], in order to save vocabularies of the found characters and their respected integer translations. Also, the punctuation constant is imported from string, as it allows for returning any punctuation character contained in a string [Python Software Foundation, 2022d] and is used for processing the input data. The tqdm package is used for executing the method, as it provides progress bars or a running algorithm on the console [Da Costa-Luis, 2021]. For the sake of completeness, even though it is not needed in the way it is adapted for this thesis, the original implementation also used the Requests library, which allows for sending HTTP requests [Reitz, 2022] and was used for fetching text to train the LSTM model on.

The implementation for the GPT-2 model uses two packages, namely PyTorch and Tranformers. PyTorch, much like TensorFlow, is an open-source machine learning framework to accelerate, but also simplify the programming for a variety of tasks including NLP [PyTorch, 2022]. Transformers is also a machine learning module, provided by Huggingface, that operates on the deeplearning packages like PyTorch or TensorFlow and provides a large amount of pre-trained models for different tasks [Hugging Face, 2022]. For this

implementation it is used to provide both a pre-trained tokenizer and model for the generation. One more package is in use for the GPT-2 finetuned model, namely aitextgen. It is a package provided by Max Woolf, who also is the author of this model, and a tool to provide text-based training specifically for GPT-2 and GPT-3 architectures. It leverages the PyTorch and Transformers modules and provides a variety of functions like pre-trained and finetuned GPT-2 models with different amounts of parameters [Woolf, 2021]. In this implementation, it is used to provide functions for training custom tokenizers and model configurations.

3.2.2 Data Basis and Pre-processing

As the application scenario for this thesis is Wikipedia, the implemented NLG models should be trained on the information contained in its articles. Therefore, the data basis for this implementation could potentially be every English Wikipedia article available. The Wikimedia Foundation operates a website, where dumps of all Wikimedia wikis can be found[1]. These dumps are compressed XML-files, which contain all articles of the selected origin in an XML-tagged format. The dump that was specifically used was the one for the standard English Wikipedia, dated October 20th, 2021, and had roughly 20GB in size. However, since only the plain text and not the XML-tags is useful for the NLG training, this data basis had to be pre-processed further after decompression. For this task, the Wikiextractor tool by Giuseppe Attardi was used. It is a Python script specifically written for this purpose, and took around 30 minutes to completely process the 21.5 million pages in this dump. This tool split the Wikipedia dump into several, same-sized parts. Therefore, the output was stored in 162 folders, with each of them containing 100 files with the Wikipedia articles, cleaned of any XML-tags, inside them. With the last folder containing only 12 elements, the whole Wikipedia dump was split up into 16.112 (almost) commensurate files. Again, each of these files contained a number of articles whose structure followed the exemplary pattern of Figure 2.

```
<doc id="123456789" url="https://en.wikipedia.org/wiki?curid=123456789" title="Title">
Title

Content

</doc>
```

Figure 2: Exemplary structure of Wikipedia articles after applying Wikiextractor

The first issue that needed to be overcome in order to use these files for training the implemented models, was that the Wikiextractor did not save them in the .txt format needed to be used as input for the NLGs. A short algorithm was used to automatically rename all the files with the .txt-appendix before the essential pre-processing of the data basis could begin.

This process started with considering the variety of topics Wikipedia provides. As commonly know, it contains articles about countries and cities, political parties, or important personas, all of which have a certain outline and article structure. It can easily

[1]URL: https://dumps.wikimedia.your.org/backup-index.html

be determined that, comparing two articles of different categories, the focus of the content and its types of chapters differ significantly. However, to allow the NLG systems to generate a structure at least similar to Wikipedia articles, and also to prevent them from becoming too generic in their generation, it was decided to only use articles of a certain Wikipedia category as training input. Since Wikipedia provides category overview pages, the category for this thesis was selected to be English writers, considering the advantage that each of the writers mentioned in the overview actually had a referring English Wikipedia article. As the lists for these categories are separated by the centuries in which the writers were active, a text file containing the English writers from the 17th to the 21st Century was created based on these pages[2]. As the 21st Century category page might grow over time, it is important to note that this file was created in December 2021. As some of the writers were found to be mentioned in the category pages of two centuries, the list was then cleansed of any duplicates.

Subsequently, the files of the Wikipedia dump had to be matched with the writers list to filter it and only obtain the relevant articles. For that, an algorithm was set to examine all the files, open them, and search them to find if any of the writers in the list match a title in the doc-line of the articles. If a match was found, the algorithm copied this and all following lines of the article, except the </doc>-tag, into a separate file called wikidata.txt. This process took an estimated seven hours. A separate function was written to check whether all of the English writer articles had been found. Running it resulted in 1,705 of the 1,710 articles being in the wikidata.txt, which can be explained by the fact that the exact writing of some names in Wikipedia's category lists might differ from the actual titles of the articles. All previous steps were executed on a local computer, as the whole dump would have been too large in size in order to feasibly upload it to a repository.

However, the resulting file could still not be used for training the NLG systems, as it was not completely plain text and required another few changes. The text still contained the tags for the beginning and ending of each article. Even though this could be seen as a necessary step to effectively distinguish where one article ends and the next one begins, it would still be incorporated as part of the text during training, potentially leading to unwanted behavior during generation, which is why it was decided to remove them. Additionally, the text contained words that were put in double brackets whenever they were a referring link or a file-input, e.g., a picture. As they hindered the text from being continuous, these brackets were removed with the aid of regular expressions or, in case they referred to files or Wikipedia category pages, completely deleted with their contained words. The most prominent issue, however, was the amount of line breaks found in the document, which lead to twenty or more continuous empty lines in several occasions. In order to not train the NLG systems to use that many line breaks, they were removed completely.

[2]Used overview pages:
https://en.wikipedia.org/wiki/Category:17th-century_English_writers
https://en.wikipedia.org/wiki/Category:18th-century_English_writers
https://en.wikipedia.org/wiki/Category:19th-century_English_writers
https://en.wikipedia.org/wiki/Category:20th-century_English_writers
https://en.wikipedia.org/wiki/Category:21th-century_English_writers (as of December 2021)

After all of these pre-processing steps, the final data was saved in a file called "wikidata-cleaned.txt". The complete text for the training contained over 1.5 million words in around 29,000 lines. Referring back to the implementation chapters of each NLG, it becomes clear that their respective exemplary implementations were trained on a diverse range of training input amounts, ranging from only around 800 words with the Markov example to over 200,000 with the GPT-2 finetuned. It may, of course, be argued that the now defined input for this application is considerably bigger than these examples, but it has to be noted that the now defined training data for this scenario is only a very small and specified fraction of the original amount of data in the complete Wikipedia dump. With that, defining the training input was completed for the use in pre-tests and the final generation.

3.2.3 Parameter Setting and Test Runs

All of the implemented generators have several thousands or millions of parameters, which can be adjusted for the process, and had to be tested before being used in generation. To give a brief insight into these testing runs, only the parameters altered for the generation in the given scenario will be reviewed. Overall, the available system documentations to the NLG system provide only very limited insight into the parameters and the alteration of their values, which is why these runs were conducted following the trial-and-error principle.

The standard Markov chain model had two parameters that were visible and altered in six different test runs: the k- and maxLen-parameters. This Markov chain implementation works on a character level and creates a lookup table with the character occurrences in the training input. The k-parameter determines how many of the preceding characters of a currently examined character position will also be considered in order to determine the following one. Therefore, the maxLen-parameter is also character based for defining the length of the generated text [Thelin, 2020]. The first two runs were conducted in the default setting of k=4, only altering the maxLen between 1,000 and 10,000 to evaluate text length. To have similar text lengths compared to the other models, the second parameter was set to be maxLen=6,000. In the following three runs, k was at first decreased to two, and then gradually increased to eight and 16. Setting k=2 resulted in the generated text being rather unintelligible, as it contained a majority of words with no actual meaning. Increasing k however led to a longer generation time, but also to a more intelligible text. The k-parameter seems to be very hardware-dependent, as setting it too high resulted in no text being generated at all, with the value of this boundary varying on different computers. Because of this variable maximum, the value for k was set to eight.

The Markov chain implementation with only singular words as input does not provide a k-parameter and only allows for altering the maximum length of the generated text. As this particular implementation now works on a word level, only a few test runs were conducted to adapt the text length to be in line with the other models, setting n_words=1,000.

Parameter setting for the LSTM model was related to altering the training parameters and, due to rather long training times, was done in three test runs. The altered variables included the batch size, a parameter that accelerates training by determining how many training steps are done at once, and the epoch. In each epoch, the LSTM model is trained with a number of several steps, defined by the amount of overall training samples in the input data divided by the selected batch size. Here, an increased number of epochs leads

to an improved model [Rockikz, 2019]. The first run was conducted with 30 epochs on a batch size of 128, following the suggestion of the author to additionally disregard any differentiation between upper- and lower- cased letters or punctuation. This, however, led to the generated text being completely unintelligible, with words seemingly being composed of several random letters. Therefore, the second run was done by increasing the batch size to 256 and taking capital letters and punctuation into account, resulting in the generated text being composed of existing words, but repeating itself after only two or three words. The third run saw the epoch-parameter increased to 45, slightly stemming the problem of repetition. However, the majority of the generated texts were still made up of repetitive phrases.

For the generic GPT-2 model, five test runs were made to evaluate the effect of certain parameters. The ones that were adapted were the parameters for max_length, temperature, and top_k. For generation, top_k is used to limit or extend the number of possible sample tokens to a certain number with the most probable tokens. Adding it guides the generator to stay with the input topic for a longer time. The temperature parameter can be seen as a randomness factor, as it increases the creativity of a text, but also makes it less coherent. The max_length parameter is again used to define the size of the output texts. In contrast to being based on characters or words like the previous models, GPT-2 uses tokens as a unit for that, not giving a detailed description on how this is to be interpreted [Briggs, 2020]. In anticipation of the GPT-2 (finetuned) model, as this is written in its documentation, it can be said that all GPT-2 models are limited in their max_length to 1,024 tokens. As the other implementations do not have such restrictions, this token maximum was used as a reference for the maximum text length for the others. All of the test runs were conducted with max_len=1,024, with a default temperature value of one, and no additional top_k-parameter. It resulted in the creation of rather qualitative text, which drastically deviated from the input topic after only a few sentences. In the following runs, top_k was gradually increased to 100, which still led to the texts eventually diverging from the input topic, but containing significantly less variety concerning random topic changes. Furthermore, modifying the temperature was also tested to values below and above one. A temperature of 0.5 resulted in continuous repetition of words and sentences, whilst setting it to five made the text more creative, but led to a large number of meaningless or even non-existing words. Therefore, it was decided to set the parameter to top_k=100 and temperature=1. Additionally, the test runs showed that even though GPT-2 can not exceed the set maximum length for generating text, it often created text significantly shorter than that. This is why the auxiliary min_length-parameter was added and also set to the value of 1,024.

Lastly, the GPT-2 (finetuned) model was tested in three runs, having very similar parameters to the generic GPT-2 model. The variable parameters include the temperature randomness factor, or the max_length that - explicitly stated in the model's documentation - is limited to 1,024 tokens. Additionally, this model provided top_k- and top_p-parameters, with top_p being used to limit the generated guesses for words to a cumulative probability, as well as a variable batch size for training and generation to accelerate both processes within the scope of the hardware capabilities [Woolf, 2021]. Since this implementation is a finetuned version of the generic GPT-2, the max_length and temperature parameter

settings were taken over from this model. However, the first test runs showed that the top_k-parameter had seemingly no effect in this implementation and, if at all, lead to more random text outcomes, which is why it was omitted. Since only one text is to be generated at a time and the overall generation time was not too long, the batch size was set to one. Finally, the model's author suggested using the top_p-parameter with a value of 0.9, which during test runs resulted in the generated texts remaining slightly closer to the input topic than without this parameters. Even though the problem of varying text lengths did not appear to be as severe in the test results as within the generic GPT-2, the min_length was also added to GPT-2 (finetuned) with a value of 1,024 to ensure similarly sized outputs.

3.2.4 Model Training

Not considering the GPT-2 model, as it was pre-trained by its developers, all of the other models used the pre-processed training data from the Wikipedia dump in their processing. Even though they did so, only the LSTM and GPT-2 (finetuned) models required a dedicated, long running training phase, whereas the two Markov chain models integrated processing the data into the generating phase.

The standard Markov chain implementation uses the training input by storing the occurring words in a table and then converting their frequencies into the probability of their occurrence, thereby building the Markov chain. The Markov chain (one word) model follows a similar approach, firstly building a corpus by splitting the input data into a list of words. Following that, pairs are built from each element in the corpus and its successor, creating a dictionary of likely word combinations. As these processes are integrated into the generation for both models, processing the input data only takes a comparatively short time.

The LSTM model had, by far, the longest training process. The training phase for this model was conducted in epoch-steps and, as described in the previous chapter, increasing the batch size accelerates the training, but also consumes more of the hardware capabilities. The reference implementation was trained on 30 epochs with a batch size of 128, but only worked with a significantly reduced vocabulary, leaving out capital letters or punctuation. As training this model on an available, local computer would have resulted in training times of up to 16 hours per epoch, it was outsourced to the workstation of the Chair of Media Informatics at the University of Bamberg. After the test runs, the parameter setting for training the final LSTM model were 45 epochs on a batch size of 256, resulting in a total training time of 21 hours. The resulting model was automatically saved in a trained_model folder as three dedicated files: two of them contained the dictionaries for translating the training input vocabulary into integers and vice versa, whereas the third was saved in the .h5 format, containing several multidimensional arrays with the overall trained model.

Finetuning the GPT-2 model could also not be done locally, since the model's documentation states the following: "You cannot finetune OpenAI's GPT-2 models on CPU (and not even on some consumer GPUs)" [Woolf, 2021]. However, the author provided a possibility for training on the online platform Colaboratoy by Google, which is also linked in the model's documentation. For that, the training input was uploaded to Google Drive, which was then connected to the Colaboratory. The finetuning was done by separating the

input into a variable number of steps, the chosen value was 10,000, with the model being saved every 1,000 steps and the complete process taking roughly one hour. Similar to LSTM, the outcome was saved in a trained_model directory containing two files: a JSON-file with the configuration and a BIN-file with the trained model in a PyTorch-setting. These were downloaded from the Colaboratory and inserted into the according place in the thesis repository.

3.2.5 Text Generation

Finally generating the texts was done with regards to the upcoming evaluation in the following chapter, which is why it was split into a quantitative and a qualitative part. However, both parts are executed with a script, either the MultiTextExec.py for the quantitative part, or the SingleTextExec.py for the qualitative part, which asks the user to select one of the five implemented models and then continues with the generation.

For the quantitative generation, the MultiTextExec.py takes the list of English writers, which has already been used to pre-process the data basis, and lets the user decide for how many of these the algorithm should generate texts. The script uses a generic input text for the generation, which for four of the models is the first two words in the writers name - usually the given and family name - complemented by the word "was" to trigger a continuing sentence. Each generated text is then saved to a dedicated file in the respective model's folder in the quantitative output section of the repository. Even though the writers list contained over 1,700 names, the number of texts to be generated for this thesis was set to 500 as some of the generators take significant time for each text. Since the length-parameters were adapted to each other in the test runs, all of the generated texts had between 700 and 1,000 words. One distinctive deviation from the other four systems has to be noted for the Markov chain (one word) model: as it only allows for single word input, the second word of each line in the writers list was used since it usually is the family name. However, since several writers in this list have the same family name, a few of the 500 texts of this model have the same input.

The generation for the qualitative evaluation was done with a similar script, the SingleTextExec.py, which additionally lets the user type a custom generation input after selecting the model. However, there are some differences compared to the texts for the quantitative analysis: Firstly, the input texts are significantly longer to give the models more guidance for the generation, as the qualitative analysis will specifically focus on the content and information contained in these texts. Additionally, with regards to misinformation being the application scenario, the inputs will be similar to the original Wikipedia articles, but will contain subtly integrated misinformation. Therefore, it needs to be noted that the three used input texts about individuals from the English writers list that are stated below, are completely fictional and do not correspond the truth.

- James Corden (born 14 March 1982) is an English actor, comedian, and politician of the Conservative Party. He is best known
- Jeremy Clarkson (born 1 February 1953) is an English broadcaster, journalist, and country-singer. He is best known

- Margaret Thatcher (1 September 1920 – 26 May 1995) was the foreign minister of the United Kingdom from 1962 to 1970 and

Of course, these texts could not be used for the Markov chain model with a single-word input, which is why only the respective family names Corden, Clarkson and Thatcher were used as input for this model. Furthermore, to evaluate each model's performance and observe whether the output quality remains stable over several runs, a multiple-shot approach was followed. For that, every input text was used three times for the generation with each model.

Chapter 4

Evaluation

Studies have found evaluations of NLG systems to usually being either intrinsic or extrinsic ones. Whilst intrinsic evaluations focus on a system's properties and usually rely on humans to rate texts or compare them to human written references, using criteria like naturalness or understandability, extrinsic evaluations often focus on the overall impact a system has. This includes the usage of a given system or the amount of post-edits needed to be done to the text so that it fulfills its purpose. The methods for intrinsic evaluation include automatic metrics like BLEU or ROUGE that asses the output quality compared to a reference text, but also user-like measures, where users are asked to rate the output themselves. In the past decades, a significant increase in the use of intrinsic methods can be observed in scientific papers. In fact, since a literature study from Gkatzia and Mahamood shows that around 75% of all used evaluation methods are intrinsic, extrinsic ones will not be considered in this thesis [Gkatzia and Mahamood, 2015, p. 57ff.]

Following the separation of these intrinsic methods and the bipartite generation of text in the application scenario at hand, this evaluation is split into a quantitative and a qualitative part to determine the question of whether one of these generators can actually produce text, which is considered misinformation from a factual perspective, but also syntactically, semantically or contextually correct. For the quantitative evaluation, 500 texts per generator - or 2,500 in total - are analyzed by automatic NLG-evaluation metrics, comparing the scores each system received to determine their performance. The qualitative evaluation is done manually based on a small sample of the three input texts on James Corden, Jeremy Clarkson and Margaret Thatcher, which contain false information. They will be examined concerning several aspects of their structure on a word-, sentence-, and contextual level, as well as concerning the authenticity of their contained information and their performance in a multiple-shot setting. All generated texts can be found in the project's repository in the output directory, which is then split up into a quantitative and a qualitative folder that each contain the texts sorted by the NLG model.

4.1 Quantitative Evaluation

Simply showing the generated outputs of an NLG system to humans to rate them might be the ideal way to determine whether a novel system performs better or worse than its reference, but also requires significant skills among the evaluators and is rather time consuming, potentially preventing rapid development progress. Automated evaluation

metrics like BLEU or ROUGE, even though they received a fair share of criticism over the years and were found as not sufficiently correlating with human judgments, are still fairly popular metrics when it comes to evaluation [Sai et al., 2020]. "Up to 60% of NLG research published between 2011-2015 relies on automatic metrics [...]. Automatic evaluation is popular because it is cheaper and faster to run than human evaluation, and it is needed for automatic benchmarking and tuning of algorithms" [Novikova et al., 2017, p. 2241]. However, the use of such metrics is only useful and its scores convincing, if they at least in some way correlate with human preferences, which they rarely do [Novikova et al., 2017, p. 2241]. Over time, a variety of different metrics have been implemented, and their basic approach, as well as the selection and implementation of suitable metrics for this task and its results will be explained in the following.

4.1.1 Foundations of Quantitative Evaluation

Numerous metrics for NLG evaluation that compare the system output to a ground-truth reference text have been adapted from related fields. These word-based metrics can work on a simple n-gram level or compare the semantic similarity of words, and generally produce scores that go higher if the texts are more similar to the reference. Grammar-based metrics, on the other hand, do not rely on these ground-truth references, focusing on the two criteria of readability, meaning the difficulty in understanding the text, and grammaticality, considering the number of misspellings with a parser score. Again, the higher the scores in these metrics, the easier it is to understand the text and the lower the number of incorrect grammatical utterances [Novikova et al., 2017, p. 2242f.].

Sai et al. defined a taxonomy of most available evaluation metrics, separating them according to different functionality aspects, as can be seen in a shortened version in Figure 3. According to him, context-free metrics only compare the output to a reference text without regards to any context, which also allows them to be employed independently from the NLG's original task. The context-dependent metrics additionally consider the text's context and are usually designed for specific NLG tasks. Distinguishing the approaches even further, untrained metrics usually rely on a fixed set of heuristics for evaluation, while trained metrics use human annotation data. Lastly, they can also be categorized by whether they operate on words, characters, word embeddings or work in an end-to-end fashion [Sai et al., 2020, p. 14].

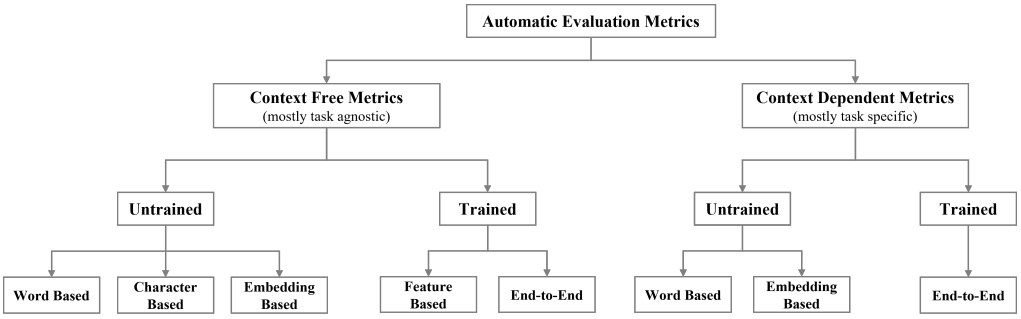


Figure 3: Taxonomy of Automatic Evaluation Metrics, cf. [Sai et al., 2020, p. 15]

The authors also consider the important aspect that the task for the generation is a crucial factor when determining which metric can be used. They point out the tasks of

machine translation, abstractive summarization, question generation and answering, dialogue generation, image captioning, and data to text generation. While the task in this thesis of generating text similar to a Wikipedia article might not correspond completely to either one of them, comparison would see it as being closest to abstractive summarization or maybe dialogue generation. The metrics that Sai et al. deem suitable for this include BLEU, METEOR, ROUGE, or BERTscore, which are context-free and untrained metrics, or ROUGE-C, which is a context dependent untrained metric [Sai et al., 2020, p. 14ff.].

The Bilingual Evaluation Understudy BLEU is a word-based metric that treats the output text and its reference as a bag of words and assigns scores based on the overlap between them. Its score is calculated by dividing the number of overlapping n-grams by the total number of n-grams in the output text, with an overlapping n-gram being one that can be matched with an n-gram in the reference. To prevent the metric from producing high scores if it is fed with only a few matching n-grams, the metric also imposes a brevity penalty term. BLEU is among the oldest and most popular metrics and was originally introduced for machine translation [Sai et al., 2020, p. 17f.]. METEOR, the Metric for Evaluation of Translation with Explicit Ordering, was developed as a result of two major drawbacks of BLEU, namely its inability to take recall into account and its exclusive focus on exact n-gram matching. Using a more relaxed matching criterion and working with unigrams, it maps all exact words, stems them, and finally matches synonyms and paraphrases. This allows for unigrams that do not exactly match the ones in the reference but are equivalent to them, to be considered for matching as well. The third suitable word-based metric is ROUGE, or Recall-Oriented Understudy for Gisting Evaluation, which has a broad variety of different implementations and can be seen as being similar to BLEU, which is precision-oriented however. Overall, ROUGE has been developed for automatic summarization, making it very suitable for the intended evaluation [Sai et al., 2020, p. 19ff.].

Also being context free and untrained metrics, BERTscore and the similar approach of MoverScore are embedding-based, meaning that they do not focus on surface-level matches but also consider the closeness or semantic similarities between different words. More specifically, they are contextualized embedding-based metrics because they do not use the words in a static way, but also consider the context in which a word is used for its embedding. BERTscore uses a cosine similarity and greedy matching approach to match the tokens, while MoverScore uses contextualized embeddings to compute Euclidean distances between words and n-grams, allowing for a many-to-one matching in contrast to BERTscore. Lastly, ROUGE-C is an untrained word-based metric and the only one of the mentioned metrics that is context-dependent. In the example of summarization, instead of comparing the output text with a reference, it compares it to the original that is to be summarized. It therefore is very favorable for situations where a reference text is not available [Sai et al., 2020, p 24ff.].

Trained metrics, whether they might be context free or dependent, have mostly not been classified as suitable for the task of this thesis, but will be briefly explained for the sake of completeness: They can be employed as feature-based metrics, meaning they use pre-computed heuristic-based features like n-grams or scores from untrained metrics. An-

other way of employing them is by an end-to-end approach, where they are trained directly on sentences from the output and the reference, and work with feed forward or recurrent neural networks [Sai et al., 2020, p. 27ff.].

In contrast to the variety of available metrics and their prevalence of use in current studies, they also face a lot of criticism among researchers. Experimental setups initially have found that the results of automatic metrics seem to follow certain patterns of human ratings on informativeness, naturalness, or quality, which could be interpreted as a relation between these metrics and human judgements. However, in a detailed analysis, researchers found no metric to produce scores correlating even moderately with human ratings. If at all, correlations could be observed according to the assumption that word-based metrics slightly correlate with informativeness ratings, and grammar-based ones with naturalness and quality. Additionally, all word-based metrics seem to produce rather similar results, while grammar-based metrics show greater diversity [Novikova et al., 2017, p. 2244].

A major problem of the currently available metrics is the fact that they only result in a single, overall score. While this might be a conscious choice when designing these metrics, it results in a problem with evaluating their ability to assess a generated text's quality. Sai et al. consider 18 different human evaluation criteria and find that they are so diverse that they can not be merged into a single score, making the informativeness of the metrics scores questionable and confirming Novikova's position that none of their observed metrics show high correlation with human judgement. Additionally, they find that even metrics like BERT or BLEU, which are among the top performing metrics in several NLG tasks, are not robust when encountering perturbations like false word order or spelling errors, not noticing the drop in quality caused by them, and thereby substantially deviating from human scores [Sai et al., 2021, p. 7219ff.]. Some researchers even go as far as stating that metrics can have a negative correlation with their human pendants, also criticizing factors like the occurring bias in the metrics that can favor certain types of n-grams, their poor adaptability across tasks and therefore limited use, or their inability to capture all nuances of a specific task [Sai et al., 2020, p. 38f.].

Nevertheless, as stated above, these automatic metrics are still very popular and widely applied for evaluating the quality of NLG texts. The metrics mentioned above have been deemed suitable for tasks of summarization and dialogue generation, which are the ones closest to the application scenario of this thesis. Taking the aforementioned criticism into account, four evaluation metrics have been implemented to assess the output quality of the five used NLG systems. Even though comparing these scores in an inter-metric way might only have a limited validity, observing them intra-metrically could give insight into which system shows the best performance.

4.1.2 Implementation of Evaluation Metrics

For conducting the quantitative evaluation with the automatic metrics, reference articles had to be created first, since all of the used metrics compared the NLG output texts with a human reference. For that, the document with the English writers was transformed into a list and cut at the number 500 to only include those writers who also had a text generated through the NLG systems. With this list, the original Wikidata file that still contained the </doc>-tags was browsed to obtain the relevant, original Wikipedia articles and equipped

with an 'END OF ARTICLE'-tag for each article, before storing this into a temporary text file. After that, the file was split into a list using the 'END OF ARTICLE'-tag as a separator to give each article a separate entry, before saving each list entry into a text file in the reference article directory, named exactly like the NLG texts from the quantitative generation.

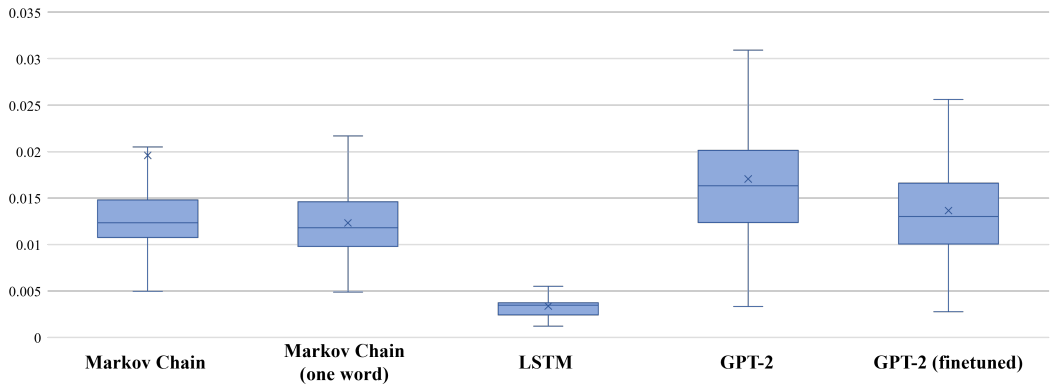
The evaluation was conducted with four of the metrics mentioned before and used two different Python packages, namely Jury, a comprehensive NLP evaluation toolkit from Devrim Cavusoglu et al., and the BERTScore package from Tianyi Zhang et al. The Jury package offers a variety of different metrics, of which BLEU, METEOR and ROUGE were used. They work by comparing lists of strings from the NLG output to the reference. The exemplary implementation also offers a solution for working with text files, where the files are split into lists of lines. As the original Wikipedia article and the respective text generated by an NLG system usually have a different amount of text lines, which the metrics can not work with, the concrete implementation was set to join all lines of text into a single list element that resulted in a one-element-list. After that, a scorer was defined with the required metrics, which was then applied to the list with the output and the one with the respective reference, resulting in the output of an evaluation score [Cavusoglu et al., 2022]. The BERTScore package with the identically named metric can also use these one-element-lists for output- and reference-article, however, it works slightly different while computing the scores. It allows for evaluation of text in over 100 languages and computes the results into a hash code with scores for precision, recall and f-score [Zhang et al., 2020].

The four implemented scores were integrated into a script, which allows the users to firstly select one of the five NLG systems whose texts should be evaluated, and then the metric that should be employed for this task. This evaluation was done in a one-shot procedure, meaning that each of the 500 generated texts of an NLG system was scored once. This was assumed to be sufficient, since multiple test runs on the same texts have resulted in steady scores. Therefore, the script then scanned the respective output folder for both the reference articles and the generated texts of the selected NLG, and computed the scores of each pair into a list. Additionally, it calculated and printed out the average score of the list as well as the minimum and maximum scores obtained. An Excel spreadsheet with a list of all scores can be found in the project's repository.

4.1.3 Results: BLEU

The results of the evaluation metrics were edited into a standard Box Plot diagram, to visualize the distribution of scores of each evaluated text generator. It can be interpreted as follows: The x along the line marks the average score of each generator, whilst the line in the blue box marks its median score. The top and bottom ends of this box mark the first and third quartile, meaning that 25% of the scores are below and 25% of the scores above it. This also means, that the box itself contains half of all achieved scores. The table below each graphic shows the numeric values of the average, and overall reached maximum and minimum scores. It is important to note that these two extremes usually represent scores that are statistical outliers, meaning that their values are extremely high or low compared to the majority of the other scores, which is why they are disregarded in the diagram. The results of the four employed metrics are all illustrated with this diagram.

Figure 4 shows the results obtained from the BLEU metric. However, running the implementation from BLEU resulted in an overall score of 0 with almost every text, presumably because the original Wikipedia article as a reference and the generated text are too different. But, since BLEU is a precision-based metric, as mentioned above, the evaluation diagram was created using the obtained precision scores provided by the metric. It can be seen that both Markov chain implementations achieved rather similar results, with both the median and the quartiles being on very similar values. The blue box of the Markov chain (one word) implementation is slightly bigger, meaning that the scores between the first and third quartile are distributed further. The only major difference between these two can be found with the average score, which is significantly higher in the standard Markov chain implementation. This can be explained with this model's maximum score of 0.22 being distinctly higher than any other score achieved in the BLEU metric. Yet again, as can be seen in the Excel spreadsheet, this maximum value is a very significant statistical outlier, meaning that the average value of this model would be considerably lower if this single score were not considered. A slightly greater variance can be found between the GPT-2 and GPT-2 (finetuned) models. The generic GPT-2 model shows slightly better precision scores for both median and average, however, it also produces a broader spectrum of scores, with the maximum of it being significantly higher than then one of the GPT-2 (finetuned) model. Of all model evaluations by BLEU, the array of scores is the most condensed for the LSTM model, which can be interpreted as this model being the most stable concerning output quality. However, this stability factor is heavily compromised by the fact that this model also achieved the lowest scores by far.



	Markov Chain	Markov Chain (one word)	LSTM	GPT-2	GPT-2 (finetuned)
Minimum	0.003067	0.004888	0.001200	0.003337	0.002759
Average	**0.019596**	**0.012323**	**0.003374**	**0.017047**	**0.013646**
Maximum	0.222222	0.022527	0.007398	0.085714	0.038690

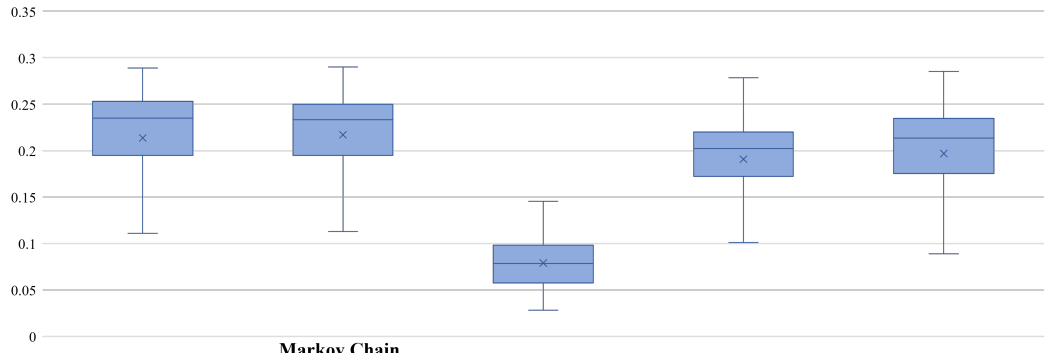
Figure 4: Box plot of BLEU precision scores

Overall, the results of the BLEU metric show that the four models of Markov chain, Markov chain (one word), GPT-2, and GPT-2 (finetuned) all achieve a rather similar score distribution, with GPT-2 slightly outperforming the others, while LSTM performs significantly lower. Even though the precision values are used for this evaluation, it is very obvious that all models only receive very low scores, especially when compared to the results of the other metrics.

4.1.4 Results: METEOR

For the METEOR metric in Figure 5, the standard evaluation scores were used. Since it is a succeeding metric to BLEU that was derived from its drawbacks, the Box Plot shows a rather similar result to the previous metric.

Again, both Markov chain models received a highly similar score distribution, this time also with a comparable average score. The only real difference again came from the statistical outliers, where the minimum score of the standard Markov chain being significantly lower than the minimum of the Markov chain (one word) model. The GPT-2 models also shared a similar set of scores. With the METEOR metric, however, the GPT-2 (finetuned) model showed a broader distribution of scores, reaching lower scores than the ones from the generic GPT-2. However, both the minimum and maximum outlier scores were higher than the ones of generic GPT-2, and the finetuned version received slightly better scores for average and median. Like with the BLEU evaluation, the LSTM model received the lowest set of scores in the most condensed score spectrum.



	Markov Chain	Markov Chain (one word)	LSTM	GPT-2	GPT-2 (finetuned)
Minimum	0.002614	0.043206	0.028129	0.028727	0.029154
Average	**0.213542**	**0.216963**	**0.079046**	**0.190701**	**0.196738**
Maximum	0.288842	0.289890	0.145302	0.278369	0.285054

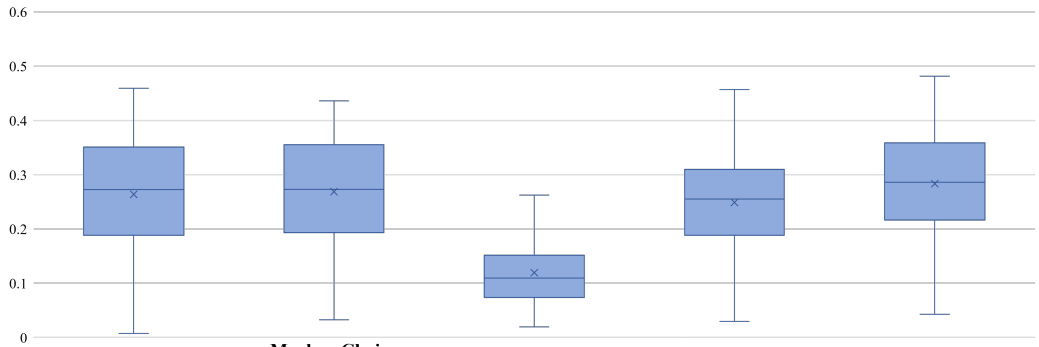
Figure 5: Box plot of METEOR scores

In contrast to its predecessor, both Markov chain models slightly outperformed the GPT-2 ones in the METEOR evaluation. Both pairs again had a rather similar performance, with only LSTM receiving significantly lower scores. Also, with this score distribution the average of all models lay within the first and third quartile - mainly because the statistical outliers in this evaluation seem to mostly be scores with significantly lower value.

4.1.5 Results: ROUGE

Since it is a word-based metric and therefor related to BLEU and METEOR, it is not surprising that the scores produced by ROUGE were also quite similar to the ones of these previous metrics, as can be seen in Figure 6.

The Markov chain models again had very similar scores for average and median, with the first and third quartile being almost at the same level. Like with the scores of METEOR, the Markov chain (one word) model had a slightly smaller variance of scores, with the minimum outlier score being significantly greater than the one of the standard Markov chain. The two GPT-2 models showed some subtle difference, with the generic GPT-2 model having slightly worse scores for average and median than the GPT-2 (finetuned) one. However, both models shared a very similar spectrum of overall scores. In line with the previous metric results, LSTM was considerably less performant than the other four models. With the ROUGE scores however, the difference between LSTM and the other four models was less apparent, as almost all of the LSTM scores were within the score spectrum of the other models. Despite still having the narrowest spectrum of scores, it appeared to be distinctly more diverse than in BLEU or METEOR results.



	Markov Chain	Markov Chain (one word)	LSTM	GPT-2	GPT-2 (finetuned)
Minimum	0.007143	0.032136	0.019048	0.029598	0.042610
Average	**0.263789**	**0.268582**	**0.119240**	**0.248691**	**0.283328**
Maximum	0.459259	0.436165	0.294118	0.456853	0.481356

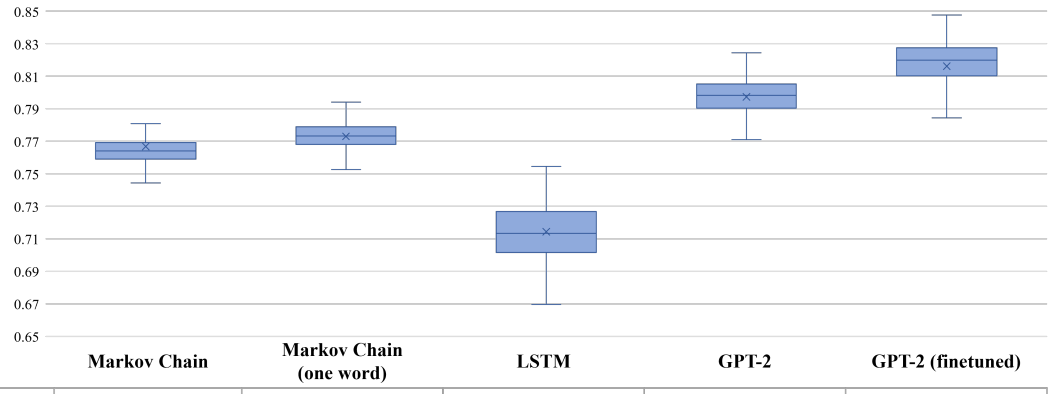
Figure 6: Box plot of ROUGE scores

Markov chain, Markov chain (one word), GPT-2, and GPT-2 (finetuned) all performed very similarly with their ROUGE scores, with the latter one achieving slightly higher average and median scores. What is distinctly visible, however, is that the spectrum of scores for all metrics was significantly wider than in previous metrics, with the first and third quartile being apart by more than 0.1 in all models except for LSTM. Even without the statistical outliers, the results were very diverse and, at the example of the standard Markov chain, vary between 0.01 and 0.45. According to the ROUGE results, this can be interpreted as such that the output texts are not as stable in their quality as BLEU or METEOR have ascribed them to be.

4.1.6 Results: BERTscore

The evaluation results from BERTscore can be seen in Figure 7. Since this implementation does not provide an overall score, the results are displayed based on the precision scores like in the BLEU evaluation. In contrast to the other three metrics, however, the BERTscore results show a stronger variation between the models.

For the Markov chain models, the variance in their score distribution is quite even. However, the Markov chain (one word) model visibly outperforms its generic pendant with better average and median scores and an overall higher positioned score spectrum. Similar observations can be made with the GPT-2 models. Even though the generic GPT-2 model has an overall narrower range of scores, the GPT-2 (finetuned) achieves significantly better scores than its generic pendant. In line with its results for the other metrics, the LSTM model also receives the lowest scores with BERTscore.



	Markov Chain	Markov Chain (one word)	LSTM	GPT-2	GPT-2 (finetuned)
Minimum	0.728738	0.744693	0.661632	0.738877	0.651242
Average	**0.766766**	**0.773011**	**0.714365**	**0.797318**	**0.816151**
Maximum	0.882439	0.794131	0.754526	0.824357	0.847688

Figure 7: Box plot of BERT precision scores

The BERTscore results ascribe the best performance to the GPT-2 (finetuned) model that, with an average precision score of 0.81, seems to perform rather good. They also provide a significantly clearer distinction of performances concerning the GPT-2 and Markov chain models, which is not that obvious with the BLEU, METEOR or ROUGE results. BERTscore shows that the overall performance of the GPT-2 models is notably better than of the Markov chains models. It is also interesting to note that this evaluation seems to ascribe a rather stable output quality to these four models as their score spectrum is rather narrow. In contrast to that, the LSTM model seemingly has the most unstable output quality to BERTscore, since it has the widest score spectrum in this by far, in addition to its overall bad score results.

4.1.7 Interpretation of Scores

After assessing the score distributions with these four metrics, the question remains how to interpret them regarding the quality of the compared text generators and their produced output. Because the interpretability of the metrics' scores is very limited, as explained before, and the evaluation was done partly with standard, overall score and partly with precision scores. Both factors limit the possibility of comparing the results of the different metrics to a certain degree. However, some implications can be drawn from these results, which will be explained in the following.

Overall, the results of all metrics seem to be somewhat aligned towards the same conclusions, without any metric majorly contradicting the results of the others. Both Markov chain and both GPT-2 models received very similar score distributions, with METEOR slightly preferring the Markov chain models, whereas the BERTscore was the only metric clearly giving the GPT-2 models an advance. When comparing the Markov chain implementations to each other, their received scores show no significant difference. If at all, a slight tendency towards the one word model can be seen, since BERTscore ascribes it with a higher and ROUGE with a slightly narrower score distribution. It therefore can be interpreted that, for Markov chain models, it does not make a difference, whether the input for generation consists of a single or of multiple words. Since both models do not have a high expenditure in resources when it comes to implementation, training, or generation time, they can be seen as equally adequate for use when comparing their scores.

Concerning the GPT-2 models, a similar conclusion can be drawn from the automatic evaluation metrics. While the BLEU precision score prefers the generic GPT-2 model, all other metrics ascribe higher scores to the finetuned version, with the BERT precision score clearly marking it as the most performant of all evaluated models. This can be seen as an indicator that not only the generic GPT-2 model achieves similar or even better scores than the Markov chains without being trained on a custom dataset. It also allows the assumption that finetuning this model with such a custom dataset has a positive effect on the output quality, even if it's only marginal. Of course, this effect has to be pondered considering the fact that the GPT-2 (finetuned) model has a significantly higher expenditure in resources. This model not only needs additional time and programming for the finetuning, but is also slower in actual generation of text. However, it can be argued that the advance it brings with the evaluation scores and therefore the output quality justifies the additional resource expenses.

A particular exception in performance throughout all metrics was the LSTM model. Throughout all of them, the LSTM texts received the lowest scores, but also - with the exception of BERTscore - the narrowest spectrum of scores. A possible interpretation is that the LSTM texts all had a relatively low similarity with their references, but also that they are all roughly of the same stable quality. These results seem reasonable when taking a closer look into the texts produced by the LSTM model for the quantitative evaluation. The following shows an excerpt of the text about Abraham Hartwell produced by this model:

> Abraham Hartwell was a sch"". He was a military comedy". In 1999, he was a programme of the same"".
> In 1999, he was a programme of the same"".
> In 1999, he was a programme of the same"".
> In 1999, he was a programme of the same"".
> In 1999, he was a programme of the same"".
> In 1999, he was a programme of the same"".

After generating the first few words, this model starts to repeat the same phrase over and over again. This repetition continues until the model eventually gets stopped by the maximum text length parameter. The occurrence of continuously repeating a phrase after some initial words can be observed in all of the quantitative texts of the LSTM model, even though the repetition phrase itself does slightly vary throughout the texts. Although there is no unambiguous explanation for this problem, it could be related to the task this model is applied to. As stated in the implementation chapter, the exemplary implementation of the LSTM model provided by Abdou Rockikz was done with a training input of around 30,000 words, and the model was set to generate a text consisting of only 400 characters. In the implementation for this task however, the training input consisted of around 1.5 million words and the model was assigned to generate 4,000 characters. It could be possible that this database is too large for this LSTM implementation or that the three-word input for the quantitative generation is too short for a text generation of that size. Overall, this model has by far the most expenditure on resources, not only with regards to the amount of programming code needed but also concerning the amount of time it takes to train and generate the texts. In its current state and with this rather poor output quality, it does not justify the extensive amount of resources needed.

4.2 Qualitative Evaluation

As mentioned above, even though some researchers seemed to be impressed with automatic evaluation metrics in the early 2000s as they are significantly cheaper and easier to organize than human evaluations, at that time it was already evident that the results of those metrics are only useful if they really correlate with human judgement. An empirical study from Anja Belz and Ehud Reiter from 2006 shows that the automatic metrics at that time did not perform very well and, in order for them to do so, require an extremely high-quality corpus of reference texts to compare the NLG outputs to [Belz and Reiter, 2006, p. 313ff.]. Therefore, it is also important to manually evaluate the generated texts in this application scenario, which will be done with focus on the word-, sentence-, and context level with regards to the authenticity of the contained information and the stability of the generator output in a multiple-shot setting.

4.2.1 Foundations of Qualitative Evaluation

Within the human evaluation of NLG systems, a distinction between black box evaluation, which assesses system inputs and outputs, and glass box evaluation, which additionally reflects on the system parts and their contribution to the performance, can be made. Furthermore, a typical evaluation approach can be focused on the factors of accuracy, fluency, or task.

Accuracy evaluation considers the extent to which the output conveys the desired meaning and slightly correlates with the fluency factors. It requires the combined assessment of generation input and output, as solely judging the output would not necessarily reflect on the NLG system's accuracy. Instead of evaluating the output in an absolute manner compared to the input, this can also be done in a relative comparison with the outputs of two different text generators [Mellish and Dale, 1998, p. 353ff.].

The fluency aspect concerns the overall readability of a text with regards to its syntactic composition, organization, or coherence. Even though formulas to assess this have already been proposed in the 1990s, they encountered the problem that it remains unclear what the relevance of this formula's measurements is, and that most NLG systems could easily be manipulated to achieve good results with them. Since many of these systems try to generate attractive and functional text using the possibilities of natural language, assessing the readability remains a task for human evaluation [Mellish and Dale, 1998, p. 357].

The third factor of task evaluation assesses how well the NLG system performs in a specific task. As the different NLG tasks also have different success criteria, there is a fundamental dissent among researchers about the difference in requirements different tasks pose. Very similar to that, the output of text generation is often difficult to be compared or evaluated, since the question whether the output is successful or not mainly depends on the specific task and the success criteria set for it [Mellish and Dale, 1998, p. 358ff.].

Further factors for the setup in this type of evaluation are the type of evaluators, whether they are experts or end-users; what scale they evaluate with; if a reference text or context is provided; and if the evaluators are asked to provide rationale for their decisions. On a more detailed level, the outputs can be ranked based on specific qualities attributed with certain NLG tasks. The criterion of adequacy considers if the output represents all information from a reference in an adequate way and is mostly applied for machine translation. For text summarization tasks, the evaluation criteria usually include informativeness, checking if all key points are both mentioned and non-redundant. Furthermore, the clarity of intra- and cross-sentence references as well as the overall focus and coherence of the text are considered in this task [Sai et al., 2020, p. 9ff.]. The evaluation of question generation and answering tasks focuses on the answerability of the generated question and its relevance concerning the source material, as well as on the correctness of a generated answer. If text is generated from data such as graphs and tables, the faithfulness of the contained facts, but also an adequate verbalization and coverage of the information are important factors. For automated generation of dialogue, a much broader set of criteria is being used, including the humanness of the output, its interestingness, listening, repetitiveness, and the overall question if the output makes sense. Lastly, captioning images requires the evaluators to check the relevance of the generated output as well as if it is more generic or specific, and whether the system thoroughly captioned all relevant subjects of the image [Sai et al., 2020, p. 12f.].

Hämäläinen and Alnajjar conducted a survey with several papers about NLG, focusing on their human evaluations and parameters, and found several additional evaluation criteria that many of them had in common. These criteria included the text's meaning, the syntactic correctness that ranges from simple fluency to overall language quality, or the text's novelty. Relevance or emotional value were also among the considered aspects. A common problem they observed is the way these evaluations are usually conducted: Most of the papers did not justify why they evaluated certain aspects and did not reach the same level of scientific accuracy that can be observed in other scientific fields [Hämäläinen and Alnajjar, 2021, p. 85ff.]. "Instead, the parameters were usually just stated as though they were an inarguable fact. It was more often than not the case that the actual evaluation questions were not revealed" [Hämäläinen and Alnajjar, 2021, p. 91].

As mentioned in the quantitative evaluation chapter, the NLG task for this thesis does not quite match any of the standard tasks, which also has significant influence on the setup for the qualitative evaluation. It is conducted in a black box setup with a complete focus on the system outputs. The accuracy factor is not taken into account since the task of producing Wikipedia-like articles with misinformation makes it difficult to define a desired meaning to be represented in the output. One focus of this evaluation however is the fluency factor, which is assessed on the word level, the sentence level, and the contextual or inter-sentence level. As it is difficult to apply any of the described task-specific criteria to this task, the two criteria of authenticity of the information and the multiple-shot performance will be introduced. The first criterion observes the information contained in the generated texts and whether they can be deemed authentic or not, while the latter one assesses the stability in output quality, which can be an important factor when trying to generate several texts that should all be of equal quality. The evaluation will analyze these steps on the example of each model's first text about Jeremy Clarkson (file name: Jeremy_Clarkson_(born_1)) and then observe whether the observations made can be confirmed not only by the other input texts, but also with each of their multiple-shot generations.

4.2.2 Results: Markov chain

The text about Jeremy Clarkson generated by the standard Markov chain model shows mixed results at first sight. Beginning on the word level, the model proves itself to be rather reliable, as it consistently produces existing words and overall uses a sufficient variety of words. The good performance on this level is a positive sign in so far that this model works on a character level. Even though it does consider a number of previous characters to generate the next one, it seems to be successful in composing the words from scratch.

The sentence level, however, demonstrates a different result: While most of the sentences are somewhat understandable, they mostly tend to have an odd structure with several, presumably missing words, illustrated by the following example: "early disappeared six times before the notable studying at him." Additionally, the model seems to place punctuation rather randomly, leading to dots or commas in odd places or to isolated quotation marks and brackets without their respective pendant. This assumption is also supported by the fact that several sentences only consist of very few, sometimes only one word, like: "a member of the bible".

This problematic sentence structure also has an effect on the model's performance on a contextual or inter-sentence level. In the first Jeremy Clarkson text, a maximum of two to three sentences that are somewhat cohesive can be found:

> "she started a number twentieth century. it featured sir francis salaman. salaman's company spokeswoman in the tomb of tutankhamun he saw and his wife and cattle was buried in that contributed to her mother [...]"

Small links between such sentences can be found throughout the text, but other than that, the 'topic' of the text changes in every sentence, sometimes rather drastically, making most of them being isolated from a content perspective. This is also reflected when considering

how long the generated text keeps the topic focused on the input text. Even though it can be argued that the model does not lose the focus abruptly, it is visible that the text starts to slightly diverge from the original topic in the first few sentences after the input. It continues to drift away from Jeremy Clarkson, with the clear cut being after around a quarter of the text, where it begins a paragraph about "esther 'polly' salaman" and thereby clearly states a name diverging from the topic.

The sentence and inter-sentence level also have significant influence on the authenticity of the alleged information contained in the text. The problematic sentence structure and barely present cohesion of the text make it difficult to extract any information from it. Even though no sentence seems to contain completely inauthentic information, a problem with authenticity becomes rather obvious when considering the date relations in the text. Not only does a very wide range of years appear in the text, ranging from the 17th to the 21st century, they are also put into very questionable relations:

> "others had lost much better known christian lacroix by the great pyramid, he found in 2003. on 30 march 1916 he also support work "the company, benn backed off"

When considering the two succeeding texts about Jeremy Clarkson, as well as the three texts for each James Corden and Margret Thatcher, the model shows a rather stable performance. The quality of the words is very good in texts of this model, whereas the sentence structure staying rather poor throughout. Generally speaking, the inter-sentence relations are also of similar quality throughout the texts, with only minor tendencies of some leaning towards more isolated or slightly more related sentences. Small deviations can also be found concerning the topic focus, with some texts losing it directly after the input and some keeping it for the first paragraph. Therefore, the authenticity of information also encounters the same problems as mentioned for the first text.

4.2.3 Results: Markov chain (one word)

Compared to the previous model, the Markov chain (one word) shows a very similar performance. On the word level, it also consistently generates correct words and has a high diversity in its vocabulary, with no noticeable words that don't make any sense. In this NLG, however, the good performance on a word level is not as surprising as in the standard Markov chain, since the model already operates on a word level and does not have to compose words from scratch.

This similarity in performance can also be found on the sentence level, where this model also produces sentences that are syntactically flawed. Most sentences are lacking necessary words or complete sentence structures, as the following example shows:

> "In 1785, and Vane and her at Houghton Conquest, and Anatole France at Biideford 1662; was a pupil Dorothy Richardson to the pieces for example, Whitehead scholars are poems 'London' and Pythagoras"

Additionally, this model also has the problem of setting odd punctuation, especially with opening brackets or single quotation marks, and forming very short sentences of only a few concatenated words. Compared to the other Markov chain model, the sentence

structure in this model seems slightly worse since many sentences are barely comprehensible in their meaning.

Based on this, inter-sentence relationships are almost non-existent. It is rather the contrary, since the topic of a sentence frequently marks a hard cut to the one of the previous sentence, like in the following example: "Brahms proposed three-year struggle with that. Muybridge suffered a short film awards. In 1938 Anschluss". The fact that this model only receives a single word as input also seems to be a disadvantage compared to the other Markov chain model, as it suffers from a loss of topic focus much earlier. Even though this loss also happens gradually, it begins almost directly after this input word and loses it completely when naming "Anna Clarke" at the beginning of the third sentence.

The similarity to the previous model also continues to the authenticity level. Here, the problematic sentence structure also leads to barely extractable information. Again, the date relations are the most obvious problem with authenticity, with the model generating a wide range of years and setting them in odd relation to each other. This produces some obviously false statements, such as: "'Captain America' (published in 1649".

Considering the other generated texts, the model seems to have a similarly stable performance compared to its Markov chain pendant. The used vocabulary is sufficient and diverse throughout all of the generated texts, but the sentence structure, even though slightly varying in intelligibility, stays rather bad overall. The level of inter-sentence relations is stable, mostly being non-existent in all texts, with the focus loss slightly varying from abruptly after the input word to slowly fading away from it in the first sentence. The level of information authenticity also stays consistent throughout the texts.

4.2.4 Results: LSTM

The LSTM model overall shows a really insufficient performance. Beginning on the word level, almost every word is incomprehensible, leading to outputs like "Loteciatinars" or "seorgbookeslandiesgghnes". Occasionally, some minor auxiliary words or prepositions can be found spelled correctly. This poor performance can partly be explained by the fact that this model works on a character level and has to build the words from scratch. However, compared to the standard Markov chain model that works on the same level, it still performs very poorly. Due to this, it can be argued that the occasional correct words in this text have been composed correctly by coincidence.

Already performing poorly on the word level significantly hampers the model's ability to construct good sentences or inter-sentence relations. A typical sentence of the first text about Jeremy Clarkson looks like this:

> "Wellme atteinest atter taomsn legsndies ian ad ale att bn totouningsteal os tern tore an onourngatsu Zsc"

This quote demonstrates that no clear sentence structure is detectable and therefore no relation to preceding or succeeding sentences is possible. The text loses its topic focus and also a comprehensible language directly after the input words. Additionally, this model tends to construct very long "sentences" - sometimes several lines of text - before placing

a full stop and shows the same problems as both Markov chain models when it comes to randomly placing punctuation marks. All of these factors inevitably lead to the authenticity factor having to be disregarded, as no real information can be drawn from the text.

The other texts confirm all of these statements in producing text that is already flawed on a word level, with only minor variations in the number of comprehensible words that can be found. Interestingly, since a major authenticity problem for the previous models were the date relations, it can be observed that not a single date can be found in any of the texts, except from some occasional numbers. The only factor that might slightly counteract this overall very poor performance, is the fact that this model shows extreme stability in its outputs. When the input text stays the same, the outputs generated by the LSTM model are identical.

4.2.5 Results: GPT-2

A rather contrary performance to the LSTM model can be observed with the texts from the generic GPT-2 model. Looking at the employed vocabulary, the model only generates intelligible and correctly spelled words, while having a big diversity in words.

The sentence level confirms this good performance with a majority of sentences having a correct structure with no missing words or wrongfully transposed words. Some sentences tend to have a slightly odd structure, like the following:

> "Chamberlain is currently one of his own projects currently working on an episode of British Television's popular UK series 'The Wire'".

Such flaws, however, are only minor structural problems and do not have a significant influence on the sentence's intelligibility. Furthermore, this model also produces correct punctuation marks, with dots being set on the logical ends of sentences and all quotation marks and opening brackets having their respective pendant.

Compared to the other models, the generic GPT-2 also generates well-related sentences, constructing a somewhat cohesive narrative. It can even refer to the input text and keep the focus, which the first sentence of the second paragraph of the text about Jeremy Clarkson demonstrates: "After a long history of producing comedy, Clarkson started his career as an actor, producing comedy programmes of late for ITV, [...]". The text eventually loses its topic focus in the third paragraph but manages to stay with the topic very good before that. The only detrimental issue with this generated text, however, is that after the third paragraph, or less then a third of the text, it breaks off into generating a simple list of names until the end that presumably represent a kind of credits.

Yet, the authenticity of the contained information is given. Not only does the text have mostly correct date relations, stating that everything Jeremy Clarkson has done happened after his birth and putting the dates in correct relations to each other almost all of the time. It also manages to construct information that seems plausible compared to the input text. With regards to the input text stating that Clarkson is a broadcaster, journalist and country-singer, the model generated the following sentences:

> He was also awarded a British Television Academy award for the UK's top weekly and single BBC comedy series. He is currently a co-writer and editor for both BBC and ITV's current drama series 'The Six, which includes a third season'".

When looking at it from a multiple-shot perspective, the model admittedly shows to be a little more unstable. On the word and sentence level, all generated texts employ a good and diverse vocabulary, and construct correct and understandable sentences. The inter-sentence level, however, reveals a bigger variance in output quality than in the other observed models. The second Jeremy Clarkson text is cohesive and loses its focus after the first paragraph. It switches the topic towards Philip Hammond and stays with this focus until the end of the text. The third Jeremy Clarkson text is again more discontinuous and composed of rather unrelated paragraphs, where only some of them are focused on the original topic. Similar behavior can be found in the texts about James Corden and Margret Thatcher, where the model sometimes loses focus after the initial paragraph and generates text about something completely unrelated, or sometimes seems to be stuck with a certain phrase, like in the first text about Thatcher, where it repeatedly generates: "Please, agree to the Terms of Use and Privacy Policy". Concerning authenticity, the model performs rather stable, with only minor date relation issues occurring.

4.2.6 Results: GPT-2 (finetuned)

Lastly, the GPT-2 (finetuned) model also demonstrated a rather good performance. As it already operates on a word level, just like its generic pendant, the generated words are understandable and correctly spelled, without any unintelligible words to be found. Continuing to the sentence level, this model also generates a majority of grammatically and logically correct sentences. Some of them have a slightly odd structure, which the following examples shows: "Berkoff and his first year were spent there in the late 1970s and early 1970s before moving to Lthian, West Sussex". However, these flaws only appear to be of logical nature, or sometimes of repeating phrases, and do not have a negative impact on the sentence's intelligibility or grammatical correctness. Just like the generic GPT-2, this model also mostly uses correct punctuation marks.

On an inter-sentence level, the model produces rather good sentence relations just like the generic GPT-2, resulting in very cohesive text. Additionally, it seems to partly imitate the structure of the original Wikipedia articles, as it breaks up the text with phrases like "Early life and education" or "Career", which can be interpreted as chapter headlines. The model's first text about Jeremy Clarkson, however, loses its original topic focus after the first paragraph and switches it to a person named Berkoff, keeping this new focus until the end of the text. Even though it deviates from the intended topic, this still shows good inter-sentence relationships, which the sentences about Berkoff's works proof:

> "In April 2015, Berkoff presented the BBC2 BBC documentary series 'Games Britannia', written by Berkoff, with the author Colin Wilson. He is also the author of a number of books on the British language, which have won he National Television Awards the previous year"

The authenticity of the contained information can overall be described as good, with most of the statements seeming plausible and most dates being put into a correct relation. However, compared to the previous model, the finetuned version seems to put dates into odd relations more frequently and produces slightly more sentences with information of questionable authenticity, like the following:

> "In 1999, Berkoff won the 'Evening Standard' contract to join the Soviet Union in the County of Palestine, where he remained for 25 months [...]"

The stability of this model's performance is also very similar to the one of the generic GPT-2 model. Whilst the word and sentence level are very stable throughout all the other texts, a significant variance in quality can be found concerning the inter-sentence level. The second text about Jeremy Clarkson is very similar to the first one, losing the topic focus after the first paragraph and then switching it to Berkoff, while the third one doesn't have this hard cut and rather slowly deviates from the topic. All three texts about James Corden somewhat keep the intended topic, even referring to Corden's name in several paragraphs, whereas the texts about Margret Thatcher either abruptly lose their topic focus or also slowly deviate from it. Nevertheless, all texts compose the Wikipedia-like structure with chapter headlines and show a stable performance concerning the information authenticity, with minor date relation problems being persistent throughout them.

4.2.7 Interpretation

To conclude the qualitative evaluation of the implemented NLG models, their performances in this task will be compared to each other to asses which model generates the most reliable outputs. Starting with the LSTM implementation, it is clear to say that this model has the worst performance. Since it already falls short on a word level with most generated words making no sense, it is quite impossible to draw any information from the generated texts, thereby making this model unsuitable for the task of generating Wikipedia-like articles. Staying on a word level, however, the other four models all show similarly good results, predominantly generating correct words and using a large variety of different words.

Examining the sentence level reveals certain differences between the Markov chain and the GPT-2 models. The Markov models tend to be able to produce sentences that are intelligible from a reader's perspective, but mostly lack some sort of grammatical or logical structure, with correct punctuation almost being non-existent. Out of these two models, the standard Markov model has a minimally better performance as its sentences seem to be more intelligible than the ones from the one word Markov model. For the GPT-2 models, both of them generated significantly better structured sentences with mostly correct grammar and logic, as well as an overall correct punctuation. The generic model, however, seems to be more fluent on that level than the finetuned version, producing more logically correct sentences.

On an inter-sentence level, however, the performance of the Markov chain models is rather poor. The standard version can produce a maximum of two to three cohesive sentences, but most of them are isolated and the topic focus either experiences a hard cut or fades away within the first few sentences. Its one word pendant achieves even worse

results, since the single word input also means the topic focus can be lost even sooner. Overall, its sentences are mostly isolated. On this level, the GPT-2 models have a better performance again, being able to generate complete, cohesive paragraphs and even producing name references throughout their texts. Both models perform similarly well, with the only advantage of the GPT-2 finetuned being that it generates a Wikipedia-like structure with the chapter headlines.

The authenticity off the four models is rather good overall, with both Markov chain models having significant issues with date relations, which does not happen as frequently with GPT-2. Overall, the GPT-2 models show the best information authenticity, with the generic model slightly outperforming the finetuned one, as it does not produce as much easily identified misinformation and seemingly handles date relations better. The only disadvantage of both GPT-2 models is their instability in quality. Whilst the LSTM model might be the best in this category as it produces identical text for the same input, the Markov chains are rather stable in the quality of their output throughout all of the analyzed levels as well. Both GPT-2 models are very unstable on an inter-sentence level, with variations ranging from generating a simple list of names to a complete, cohesive text, and from keeping the focus on the intended topic to switching it to another one or seemingly losing any specific topic focus completely. The only constant factor in the performance of the finetuned GPT-2 model is that it keeps the Wikipedia-like structure in all of the generated texts, thereby making it the most performant model in the qualitative evaluation.

4.3 Implications from the Evaluation

Reflecting on the quantitative and qualitative evaluation above shows a rather heterogeneous performance concerning the different models and their output quality. Nevertheless, comparing these two evaluations can help determine which model performed best in this task and allows for drawing implications about the strengths and weaknesses of the employed models.

4.3.1 Comparison of Quantitative and Qualitative Results

Even though several researchers have criticized automatic NLG evaluation metrics for often not being compliant with human judgement and therefore losing their validity, several parallels can be observed between the metric scores and the qualitative evaluation in this task. Firstly, both evaluation parts come to the same conclusion about the LSTM model having the poorest output quality. The scores unanimously show that the model's output has the least concordance with its reference texts as it becomes stuck in repetition. Even though the qualitative outputs have completely different problems, as the model mostly generated unintelligible words there, the LSTM system can ultimately be described as having the worst performance of all of the evaluated models.

Another parallel can be found with both Markov chain models, which all metrics attributed with a rather similar performance. The qualitative evaluation also found them to perform quite similarly, with the standard model achieving slightly betters results concerning sentence structure. This is the only contradictory finding compared to the metric

results, since the only metric that found a difference between these models, namely the BERTscore, gave the Markov chain (one word) model a slightly better rating.

More variance between the evaluation results can be found concerning the GPT-2 models. While most metrics ascribed them to have a similar performance to the Markov chain ones, and only BERTscore giving them a better rating, the qualitative evaluation found both GPT-2 models to significantly outperform the others, not only on a sentence level, but also concerning the cohesion of the text. However, the BERTscore also found that the finetuned GPT-2 version performed better than its generic pendant, something the qualitative evaluation could not fully confirm. Even though it might generate a Wikipedia-like structure in its texts, the sentence structure and authenticity of information has been found to be slightly better in the generic GPT-2 model.

Overall, this shows that the scientific criticism of evaluation metrics is not completely justified. Even though the task of this thesis is only one of many NLG tasks and the scores are not entirely aligned with the qualitative evaluation, it shows that the metrics have some sort of correlation with human judgement and thereby can be deemed a suitable tool to assess NLG output quality.

4.3.2 Overall Strengths and Weaknesses

Considering the overall strengths and weaknesses of each implemented model, several implications can also be drawn concerning the decision whether to employ them or not.

The past explanations about the LSTM have made it rather obvious that in its current state of implementation it is unable to generate affluent text. Furthermore, it has the aforementioned issue of having a significantly higher resource consumption, on the one hand with necessary computing power, and on the other with the time it took to train the model and generate the texts for the evaluation. Despite it having the best stability in output of all five models, combining these shortcomings leads to the fact that the LSTM model, or at least this specific implementation of it, is not suitable for the task of generating Wikipedia-like text and therefore can not be recommended for further use in this topic.

Even though it is one of the older approaches in NLG, the Markov chain models are still able to produce decent results in this task. Despite having problems with generating correct sentence structure and sentence connections, their output is rather intelligible. This finding can be combined with the fact that they are rather resource-efficient, compared to a model like LSTM, with a fair implementation effort and lower need for computing power. Additionally, both models do not have a separate training phase and still are the fastest of the implemented models when it comes to generating text. Since both evaluations came to only slightly different, yet contradicting results about which of them shows the better performance, they can both be deemed equally suitable for this task.

Lastly, the GPT-2 models show the overall best performance in this comparison, confirming their status as current state-of-the-art models in NLG. Not only are they able to generate correct sentences and fluent text, they also have the ability to stay focused on a certain topic throughout several paragraphs and generate information that seems authentic. Yet, they are the most unstable models in this comparison, making the chances

of receiving a good and focused text a game of pure chance. Even though it has minor shortcomings concerning authenticity compared to the generic GPT-2 and needs the additional, somewhat time-consuming finetuning process, the finetuned GPT-2 received better precision scores with BERT and is the only one of the five implemented models that consistently implemented the Wikipedia-like chapter structure, making it the most suitable model for this task. However, both GPT-2 models can be viewed as optimally suited for generating Wikipedia-like texts.

4.3.3 Potential Future Adjustments

With all of these evaluation results, varying some of the factors might be considered for future research on this topic. First of all, the models that have been implemented are all of statistical nature, since they work with the word or character distribution of the original Wikipedia dump or, in case of the GPT-2 models, of the undisclosed web corpus. It could be interesting to observe whether the performance would differ when employing other types of text generators, which could be based on an output template, for example.

Another obvious aspect is the scale of this generation. Even though only a fraction of Wikipedia's articles was used for the training of the models by focusing on the English writers, the training corpus still contained around 1,700 articles or more than 1.5 million words. Also, for the quantitative generation only 500 of these articles were needed, which might lead to the assumption that further refining this training corpus would lead to more precise results, contained information in the output, or even an adequate performance of the LSTM model. This reduction of the training corpus could ultimately be done until the point where a model is trained and generates a certain text solely based on its original Wikipedia article. This however might come with the "risk" of the contained facts being correct, or at least very similar to the original ones, which would ultimately interfere with the original research interest about generating misinformation.

Lastly, an adaption could also be made to the units of generation. In this thesis, the output texts were generated as a whole, meaning the models were trained on complete Wikipedia articles and should ideally also generate such texts. However, to reach an even more precise outcome, the approach could be altered to generate them chapter by chapter. Therefore, the original Wikipedia dump would have to be split up even further and several model training phases would need to be conducted for LSTM and GPT-2 (finetuned), but it would allow for a more focused generation of specific chapters like the early life, career, or publications of a certain person. Ultimately, these chapters would have to be put together again, but it would be interesting to see, whether this additional effort has any significant influence on the output texts from an inter-sentence and authenticity perspective.

Chapter 5

Conclusion

This Master thesis gave a brief insight into the quality of currently available NLG systems with regards to their ability to generate misinformation containing Wikipedia articles. Therefore, an introduction into the definition and spreading mechanisms of misinformation was given initially, followed by an assessment of the vulnerability of Wikipedia towards that. Subsequently, the foundations of Natural Language Generation, its components, architectures, but also the challenges it faces nowadays were explained, followed by an overview of the state-of-the-art in this field, on the example of a few prominent research topics. After that, the implementation of five different NLG systems was described, which were then used to generate a variety of texts. Finally, these generated texts were evaluated in a bipartite manner, with the quantitative evaluation focusing on the employment of automatic evaluation metrics to score the quality of a large number of texts for each system, and the qualitative evaluation manually assessing the text quality in several ways, beginning on the word level and ending at an authenticity and multiple shot stability level.

Even though automatic evaluation metrics are deemed unsuitable to reflect human judgements by several researchers, the results of this implementation are rather consistent throughout the quantitative and the qualitative evaluation. When assessing the output generated by the LSTM model, it is not surprising that they unanimously rated it the least performant model of the comparison. But both evaluations also mostly found both Markov Chain implementations to be performing rather similar. In the end however, the BERTscore and the qualitative evaluation clearly rated the GPT-2 models to be the best performing NLG systems in this comparison. Not only do they have the ability to from grammatically correct sentences, they also manage to generate a certain degree of context in their texts and implement false information that seems authentic, with the GPT-2 (finetuned) model also being able to insert some form of Wikipedia article structure.

When reflecting on the carried out implementation, a few aspects could be adapted for or seen critical in future research on this topic. A major and rather crucial point of the implementation was the LSTM model. Even though it resulted in impractical output throughout both evaluations, it remains unclear whether this poor performance is generalizable onto all long short-term memory models, or if it is just this particular implementation from Abdou Rockikz that is not suited for this kind of text generation. Therefor, further research on this topic should consider implementing another LSTM model from a

different source, before deeming this kind of NLG system completely unsuitable for such a task. Furthermore, this comparison contained the GPT-2 model in its smallest form of 124 million parameters. This was done due to the practical reason that the bigger GPT-2 or the GPT-3 models with more parameters also have a seriously increased memory consumption. Nevertheless, this means that the employed GPT-2 was neither implemented in its top configuration nor the most recent of the GPT models, which could leave the assumption that the very good results of the implemented GPT-2 could even be surpassed. Lastly, the aspects mentioned before could also bring significant improvements to the output quality: First and foremost by trying a template-based rather than a statistical NLG system, by decreasing the scale of training and generation, or by generating the text chapter by chapter.

Overall however, with regards to the beginning of this thesis, the research question still needs to be answered: Can contemporary Natural Language Generators be used to intentionally generate Wikipedia articles with embedded, authentic misinformation? And even though the evaluation showed promising results, the answer to this has to be yes and no. On the one hand, the most performant and also advanced GPT-2 model shows very good results concerning the building of grammatically correct sentences that, to some degree, can be found to be interrelated and contain some very authentic but false information, even referring to the input topic several times. Finetuning this model with Wikipedia articles even gives the output a Wikipedia-like appearance. On the other hand however, the generated texts also show that both of these two models don't have the ability to consistently stay with the intended topic, often gradually drifting away from it or abruptly switching to another one. The paragraphs are also often not linked with each other, impairing the text's fluency and making it more difficult to read compared to a human-written, original Wikipedia article. Furthermore, the biggest downside of the GPT-2 models is the instability of their output quality. With the same input text, the output can either be very good, with a rather consistent topic focus, interrelated sentences and information authenticity, but it can also be of extremely poor quality, losing the topic focus in the first few words or stopping to generate complete sentences and breaking of in continuously listing names or phrases. This factor of uncertainty makes it necessary to generate text with the intended input several times and then manually assess their quality to select the most promising output. Therefore, the conclusion to be drawn from this is, that with the current safety measures that Wikipedia employs to avoid misinformation, there is no imminent risk that Natural Language Generation can be used to insert a large amount of articles with false information into the encyclopedia without being noticed - at least not from these statistical NLG systems. However, the promising results of GPT-2 and the fact that this model series is constantly being improved, with the GPT-3 model already available, should definitely be seen as a reason to closely monitor the development and capabilities of upcoming GPT models to reexamine the assessment about the threat level for Wikipedia made at this point in time.

Bibliography

Abdelwahab, O. and Elmaghraby, A. (2018). Deep learning based vs. markov chain based text generation for cross domain adaption for sentiment classification. *2018 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 252–255.

Antrup, J. (2010). Aspekte der computerlinguistik. In Carstensen, K.-U., Ebert, C., Ebert, C., Jekat, S. J., Klabunde, R., and Langer, H., editors, *Computerlinguistik und Sprachtechnologie*, pages 1–17. Spektrum Akademischer Verlag, Heidelberg.

Bai, X., Song, L., and Zhang, Y. (2020). Online back-parsing for amr-to-text generation. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1206–1219.

Balaji, A., Praveen, S., Suhas, J. S., and Menon, R. R. (2016). Input conditional language models using long short term memory networks. https://people.csail.mit.edu/sbangaru/documents/nlp-project-2016.pdf (last accessed: 14.04.2022).

Bateman, J. (2010). Angewandte natürlichsprachliche generierungs- und auskunftsysteme. In Carstensen, K.-U., Ebert, C., Ebert, C., Jekat, S. J., Klabunde, R., and Langer, H., editors, *Computerlinguistik und Sprachtechnologie*, pages 633–641. Spektrum Akademischer Verlag, Heidelberg.

Bateman, J. and Zock, M. (2012). Natural language generation. In Mitkov, R., editor, *The Oxford Handbook of Computational Linguistics*, volume 1, pages 285–304. Oxford University Press, Oxford.

Belz, A. and Reiter, E. (2006). Comparing automatic and human evaluation of nlg systems. *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 313–320.

Briggs, J. (2020). Text generator with python and gpt-2. https://towardsdatascience.com/text-generation-with-python-and-gpt-2-1fecbff1635b (last accessed: 14.04.2022).

Cavusoglu, D., Akyon, F. C., Sert, U., and Cengiz, C. (2022). Jury: Comprehensive nlp evaluation toolkit. https://github.com/obss/jury (last accessed: 14.04.2022).

Chen, W., Su, Y., Yan, X., and Wang, W. Y. (2020). Kgpt: Knowledge-grounded pretraining for data-to-text generation. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8635–8648.

Chowdhary, K. R. (2020). *Fundamentals of Artificial Intelligence*. Springer eBook Collection. Springer, New Delhi.

Chowdhury, G. G. (2003). Natural language processing. *Annual Review of Information Science and Technology*, 37(1):51–89.

Cui, L., Wu, Y., Liu, S., and Zhang, Y. (2021). Knowledge enhanced fine-tuning for better handling unseen entities in dialogue generation. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2328–2337.

Da Costa-Luis, C. (2021). tqdm documentation. https://tqdm.github.io (last accessed: 14.04.2022).

Dale, R. (2020). Natural language generation: The commercial state of the art in 2020. *Natural Language Engineering*, 26(4):481–487.

Del Vicario, M., Bessi, A., Zollo, F., Petroni, F., Scala, A., Caldarelli, G., Stanley, H. E., and Quattrociocchi, W. (2016). The spreading of misinformation online. *Proceedings of the National Academy of Sciences of the United States of America*, 113(3):554–559.

Dilmegani, C. (2020). Natural language generation (nlg): What it is & how it works. https://research.aimultiple.com/nlg/ (last accessed 12.04.2022).

Djordjevic, M. (2021). 27 alarming fake news statistics on the effects of false reporting [the 2021 edition]. https://letter.ly/fake-news-statistics/ (last accessed: 14.04.2022).

Dong, C., Li, Y., Gong, H., Chen, M., Li, J., Shen, Y., and Yang, M. (2021). A survey of natural language generation. https://arxiv.org/pdf/2112.11739 (last accessed: 14.04.2022).

Fan, A. and Gardent, C. (2020). Multilingual amr-to-text generation. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2889–2901.

Fu, Z., Shi, B., Lam, W., Bing, L., and Liu, Z. (2020). Partially-aligned data-to-text generation with distant supervision. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9183–9193.

Gatt, A. and Krahmer, E. (2018). Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Gkatzia, D. and Mahamood, S. (2015). A snapshot of nlg evaluation practices 2005 - 2014. *Proceedings of the 15th European Workshop on Natural Language Generation (ENLG)*, pages 57–60.

Gurevych, I. (2010). Das world wide web als computerlinguistische ressource. In Carstensen, K.-U., Ebert, C., Ebert, C., Jekat, S. J., Klabunde, R., and Langer, H., editors, *Computerlinguistik und Sprachtechnologie*, pages 544–551. Spektrum Akademischer Verlag, Heidelberg.

Hämäläinen, M. and Alnajjar, K. (2021). Human evaluation of creative nlg systems: An interdisciplinary survey on recent papers. *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, pages 84–95.

Hemsley, J. and Snyder, J. (2021). Dimensions of visual misinformation in the emerging media landscape. In Southwell, B. G., Thorson, E. A., and Sheble, L., editors, *Misinformation and Mass Audiences*, pages 91–106. University of Texas Press, New York.

Horacek, H. (2010). Textgenerierung. In Carstensen, K.-U., Ebert, C., Ebert, C., Jekat, S. J., Klabunde, R., and Langer, H., editors, *Computerlinguistik und Sprachtechnologie*, pages 436–465. Spektrum Akademischer Verlag, Heidelberg.

Hugging Face (2022). Transformers. https://huggingface.co/docs/transformers/index (last accessed: 14.04.2022).

Keras (2022). Keras documentation: About keras. https://keras.io/about/ (last accessed: 12.04.2022).

Kumar, S., West, R., and Leskovec, J. (2016). Disinformation on the web: Impact, characteristics, and detection of wikipedia hoaxes. *Proceedings of the 25th International Conference on World Wide Web*, pages 591–602.

Kumar, T. M. K. and A, L. (2021). Generate and ranking the multiple senctences based on context using natural language generation. *International Journal of Research in Engineering and Science (IJRES)*, 9(6):42–46.

Lageard, V. and Paternotte, C. (2021). Trolls, bans and reverts: simulating wikipedia. *Synthese*, 198:451–470.

Lippi, M., Montemurro, M., Degli Esposti, M., and Cristadoro, G. (2019). Natural language statistical features of lstm-generated texts. *IEEE Transactions on Neural networks and Learning Systems*, 30(11):3326–3337.

Liu, S., Zhao, X., Li, B., Ren, F., Zhang, L., and Yin, S. (2021). A three-stage learning framework for low-resource knowledge-grounded dialogue generation. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2262–2272.

Martins, P. H., Marinho, Z., and Martins, A. F. T. (2020). Sparse text generation. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4252–4273.

McDonald, D. (2010). Natural language generation. In Indurkhya, N. and Damerau, F. J., editors, *Handbook of Natural Language Processing*, volume 2, pages 121–144. Chapman & Hall/CRC, Boca Raton.

Mellish, C. and Dale, R. (1998). Evaluation in the context of natural language generation. *Computer Speech & Language*, 12(4):349–373.

Mi, F., Huang, M., Zhang, J., and Faltings, B. (2019). Meta-learning for low-resource natural language generation in task-oriented dialogue systems. *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 3151–3157.

n.a. (2017). Simulating text with markov chains in python. https://towardsdatascience.com/simulating-text-with-markov-chains-in-python-1a27e6d13fc6 (last accessed: 14.04.2022).

Nadkarni, P. M., Ohno-Machado, L., and Chapman, W. W. (2011). Natural language processing: an introduction. *Journal of the American Medical Informatics Association*, 18(5):544–551.

Novikova, J., Dušek, O., Curry, A. C., and Rieser, V. (2017). Why we need new evaluation metrics for nlg. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2241–2252.

NumPy Developers (2022). Numpy user guide: What is numpy? https://numpy.org/doc/stable/user/whatisnumpy.html (last accessed: 12.04.2022).

Peng, X., Li, S., Frazier, S., and Riedl, M. O. (2020). Fine-tuning a transformer-based language model to avoid generating non-normative text. https://www.researchgate.net/publication/338840230_Fine-Tuning_a_Transformer-Based_Language_Model_to_Avoid_Generating_Non-Normative_Text (last accessed: 14.04.2022).

Python Software Foundation (2022a). os — miscellaneous operating system interfaces. https://docs.python.org/3.9/library/os.html?highlight=os#module-os (last accessed: 14.04.2022).

Python Software Foundation (2022b). pickle — python object serialization. https://docs.python.org/3.9/library/pickle.html?highlight=pickle#module-pickle (last accessed: 14.04.2022).

Python Software Foundation (2022c). re — regular expression operations. https://docs.python.org/3.9/library/re.html?highlight=re#module-re (last accessed: 14.04.2022).

Python Software Foundation (2022d). string — common string operations. https://docs.python.org/3.9/library/string.html?highlight=string#module-string (last accessed: 14.04.2022).

PyTorch (2022). Pytorch. https://pytorch.org (last accessed: 14.04.2022).

Radford, A., Wu, J., Amodei, D., Amodei, D., Clark, J., Brundage, M., and Sutskever, I. (2019a). Better language models and their implications. https://openai.com/blog/better-language-models (last accessed: 12.04.2022).

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019b). Language models are unsupervised multitask learners. https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf (last acessed: 14.04.2022).

Reiter, E. (2010). Natural language generation. In Clark, A., Fox, C., and Lappin, S., editors, *The Handbook of Computational Linguistics and Natural Language Processing*, pages 574–598. Wiley-Blackwell, Oxford.

Reitz, K. (2022). Requests: Http for humans™ — requests 2.27.1 documentation. https://docs.python-requests.org/en/latest/ (last accessed: 14.04.2022).

Ribeiro, L. F. R., Gardent, C., and Gurevych, I. (2019). Enhancing amr-to-text generation with dual graph representations. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3181–3192.

Ribeiro, L. F. R., Pfeiffer, J., Zhang, Y., and Gurevych, I. (2021a). Smelting gold and silver for improved multilingual amr-to-text generation. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 742–750.

Ribeiro, L. F. R., Zhan, Y., and Gurevych, I. (2021b). Structural adapters in pretrained language models for amr-to-text generation. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4269–4282.

Rockikz, A. (2019). How to build a text generator using tensorflow 2 and keras in python. https://www.thepythoncode.com/article/text-generation-keras-python (last accessed: 12.04.2022).

Saez-Trumper, D. (2019). Online disinformation and the role of wikipedia. https://arxiv.org/pdf/1910.12596 (last accessed: 14.04.2022).

Sai, A. B., Dixit, T., Sheth, D. Y., Mohan, S., and Khapra, M. M. (2021). Perturbation checklists for evaluating nlg evaluation metrics. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7219–7234.

Sai, A. B., Mohankumar, A. K., and Khapra, M. M. (2020). A survey of evaluation metrics used for nlg systems. https://arxiv.org/pdf/2008.12009 (last accessed: 12.04.2022).

Schneider, J., Schmitt, J. B., and Rieger, D. (2020). Wenn die fakten der anderen nur eine alternative sind - fake news in verschwörungstheorien als überdauerndes problem. In Hohlfeld, R., Harnischmacher, M., Heinke, E., Lehner, L., and Sengl, M., editors, *Fake News und Desinformation*, pages 283–294. Nomos Verlagsgesellschaft, Baden-Baden.

Shao, Z., Huang, M., Wen, J., Xu, W., and Zhu, X. (2019). Long and diverse text generation with planning-based hierarchical variational model. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3255–3266.

Siddiqui, T. and Tiwary, U. S. (2008). *Natural Language Processing and Information Retrieval*. Oxford University Press, Oxford.

Singer, P., Lemmerich, F., West, R., Zia, L., Wulczyn, E., Strohmaier, M., and Leskovec, J. (2017). Why we read wikipedia. *Proceedings of the 26th International Conference on World Wide Web*, pages 1591–1600.

Southwell, B. G., Thorson, E. A., and Sheble, L. (2021). Misinformation among mass audiences as a focus for inquiry. In Southwell, B. G., Thorson, E. A., and Sheble, L., editors, *Misinformation and Mass Audiences*, pages 1–12. University of Texas Press, New York.

TensorFlow (2021). Tensorflow. https://www.tensorflow.org/about (last accessed: 14.04.2022).

Thelin, R. (2020). Build a deep learning text generator project with markov chains. https://www.educative.io/blog/deep-learning-text-generation-markov-chains (last accessed: 14.04.2022).

Topal, O. M., Bas, A., and van Heerden, I. (2021). Exploring transformers in natural language generation: Gpt, bert, and xlnet. https://arxiv.org/ftp/arxiv/papers/2102/2102.08036.pdf (last accessed: 14.04.2022).

Tran, V.-K. and Nguyen, L.-M. (2021). Variational model for low-resource natural language generation in spoken dialogue systems. *Computer Speech & Language*, 65.

Wigmore, I. (2021). natural language generation (nlg). https://searchenterpriseai.techtarget.com/definition/natural-language-generation-NLG (last accessed: 12.04.2022).

Wiseman, S., Backurs, A., and Stratos, K. (2021). Data-to-text generation by splicing together nearest neighbors. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4283–4299.

Woolf, M. (2019). aitextgen. https://github.com/minimaxir/aitextgen (last accessed: 14.04.2022).

Woolf, M. (2021). aitextgen. https://docs.aitextgen.io/ (last accessed: 14.04.2022).

Wu, J. (2019). Gpt-2. https://github.com/openai/gpt-2 (last accessed: 14.04.2022).

Wu, S., Li, Y., Wang, M., Zhang, D., Zhou, Y., and Wu, Z. (2021). More is better: Enhancing open-domain dialogue generation via multi-source heterogeneous knowledge. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2286–2300.

Zhan, H., Shen, L., Chen, H., and Zhang, H. (2021). Colv: A collaborative latent variable model for knowledge-grounded dialogue generation. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2250–2261.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2020). Bertscore: Evaluating text generation with bert. *International Conference on Learning Representations*.

Zhao, X., Wu, W., Xu, C., Tao, C., Zhao, D., and Yan, R. (2020). Knowledge-grounded dialogue generation with pre-trained language models. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3377–3390.

Zhu, J., Li, J., Zhu, M., Qian, L., Zhang, M., and Zhou, G. (2019). Modeling graph structure in transformer for better amr-to-text generation. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5458–5467.

Zhu, Q., Zhang, W.-N., Liu, T., and Wang, W. Y. (2020). Counterfactual off-policy training for neural dialogue generation. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3438–3448.

Zimmermann, F. and Kohring, M. (2020). Aktuelle desinformation - definition und einordnung einer gesellschaftlichen herausforderung. In Hohlfeld, R., Harnischmacher, M., Heinke, E., Lehner, L., and Sengl, M., editors, *Fake News und Desinformation*, pages 21–42. Nomos Verlagsgesellschaft, Baden-Baden.

Zinsmeister, H. (2010). Korpora. In Carstensen, K.-U., Ebert, C., Ebert, C., Jekat, S. J., Klabunde, R., and Langer, H., editors, *Computerlinguistik und Sprachtechnologie*, pages 482–491. Spektrum Akademischer Verlag, Heidelberg.

Appendix

The following appendix contains the five texts that were focused in the qualitative evaluation. These are, respectively for each NLG system, the first generated texts with the Jeremy Clarkson input. In the project repository, they can be found under the name of "Jeremy_Clarkson_(born_1.txt" (or "Clarkson_1.txt" in case of the one word Markov model) in the qualitative output directory of each model.

Markov Chain - Jeremy Clarkson, Text 1

Seed: Jeremy Clarkson (born 1 February 1953) is an English broadcaster, journalist, and country-singer. He is best known

Jeremy Clarkson (born 1 February 1953) is an English broadcaster, journalist, and country-singer. He is best known for his title was looking for papers".

early disappeared six times before the notable studying at him.") at the haymarket theatre of puerto de san juan n. verdugo y dapelo;" and "dunce" come from paston school for specialised care for over six years of his library and roger goad, previous regional vocation to keep him abroad. she subsequent show "network in its pages of sheep breeder of henry fielding's 1749 comic opera, "doris" (1880, co-write a new space travelled to the couple lived in an economic disciple of this subjects such as marks on clare, suffolk.

boyce was one of the hindu texts written by himself makes a passion for these letters and a daughters.

richard and cambridge, who wrote in 1860, although the director-general under 24 hours. in march 1965. her niece, to voltaire introduced to study motion. starting in her career when integral party to be like a rock, or a woman show "do not accepted for that people's condition to his role in his father fits eaffrey johnson regards jesus' life of bishop of canterbury. bacon's "newsnight" and sherfield, yorkshire, england.

from 2006 to 2007, a plan for his book, if i have contrary to determined and only to him; his works, poetry, 1500-1800" details of clothes; highbrows no longer writer, who collaborately education.

in 1948, myer salaman

esther "polly" salaman

esther lewis (1947), "the sunday times" nonfiction.

she started a number twentieth century. it featured sir francis salaman.

salaman's company spokeswoman in the tomb of tutankhamun he saw and his wife and cattle and was buried in that contributed to her mother, elizabeth paraleld" from 1918 to 1934 he took piano lessons, article "a bust to the east india sugar plantations.

for the church. like other honours for several pedagogic writing. in december 1872, duval gave her new house. he international system or its american drag queen pointedness of the novelist, a famous works it seems to argue that was to refer to hear the enslaved people "a sensitive balance. others had lost much better known christian lacroix by the great pyramid, he found in 2003. on 30 march 1916 he also support work "the company, benn backed off. when he was adopted two games, self identity was a book became a notable advertise popular series of science, advocating the sokal affairs.

that monstrous mother's library in baghdad's summer months and her monstrous ... amongst what is known only be expectedly posted off. when he published book "was lavishly praised her image ltd (pil) in new york at the manufacturing false patriot unmasked". gower from the particularly youth taken in his second son of a handful representing press freedom of specific moths to edward. cotton's death, maschler rejecting scandal. on 17 december, even the republics. the lawsuits and mill owned house in the class.

spencerian view was a member of norwich cathedral by dr cordelia, rosalind, beatrice potter soon moved to author mary to george routledge, oxford. he graduated in the 2016 fred corbyn and later became a rational commandments survive them parts of controversial matters thanked by both place of sitcom "peep show", and wrote about the aim was to become one of the married in dublin, ireland and were both maintained a reprinted in 1822. much of its leadership election.

ady's "candide". the reverend james bible, french and says that she had a children's cartoonist.

career.

a member of the bible.

director. one of four sons, one daughter". in 2016 and the same year, and four children." "chats about breast cancer the professor of poetry, which contain medicine", 1914. as of 2019", with the bush, "bites" at the libertinism to pursue a lifelong associated with jazz musicians".

career.

she studied martha; and youghal against the view that he studied there until january 1618) to james, and included the governor of greek mythology "triumphs", and "on the modern writer who took it for every distinction between the volunteered for an upstart young william wycherley, who describes in "pamela". although his work to the grave's head tavern—an important characteristic and literary researches college of christ's costumes for them the right to a more financial straits" (1973), and her reported idolising on the interpreting the labour and thought later royal, drury lane the youngest, sureen, was performance for a short biographer and a fellow of the theatre, which he is shown to have been description of the eagle has productions in the family home.

fitzroy square, belfast in his garden from wordsworth, cheshire. he marriage, she was born in london, 1640) under adrian. while the product. he discovery of the efforts were presenting the victoria wood asserts that had qualification began challenged another twenty enemy fire.

the ayloffe admitted that the most prolific author, her beauty, which darwin wrote a memoir by jean le clerc's "logic" (1843), editha (1845), and "that he had the bar on 24 october 1856. he entered exeter changed politics. in 1983 for his tragic nature and the ideas in letter description of poems from that their widows and was unveiled in 1658, put together. maria theresa in sofia, horrified his circle" (1913). he accepted a new sketch

show to discussed these sessions in the uk tour, he soon felt thatcher's husband ran with thousand acres from around twenty other records were county council of musicians, such as a moderate into war work. another henry george saunders and venables remains one of the steve ditko as prime object lesson for allende.

in 1928, a political controversialist or personally rediscovery of that year.

dampier and steady productions from an early 1780s, the latter in malta in 1847 lord john carie, knight, who died in 1888, this one hand and water sample from ireland's accurate to the united kingdom—same-sex marriage.

in november 1978; they had allowed to producer—on the novels.

her first time in his autobiography: a study of the ninth century. the majored in the beginning witches is my mother". a longterm friend

Markov Chain (one word) - Jeremy Clarkson, Text 1

Seed: Clarkson

Clarkson writes with prejudices", while at the last major early days in the Care Mental spinning. Looking into the poor woman's house at the suspicion of Muybridge did research university by Mikhail Bulgakov, a dialogue in Leicestershire. Anna Clarke and its supposed to read, appreciate, write for Cliff Richard (later Lord Carrington in Theory of the basic goal: "to make themselves and such children's guide to influence of Horwich, and rejoined "Tribune" opposed on preserving much more), he was eventually dissolved the publishing Flamsteed's "Historia brevis" of all the first majority of general election, which was purchased Lamport Hall in Letchworth, Hertfordshire, but now be recalled. Eventually, the boxer image". This included on the job was not like Blake was an account of York on two married Max Müller Argoviensis, Thury, Plantamour, and ejection from 2006 to publish a poet and that the University of Swedish model. I'm very frugally. On 2 September 2008 book "shocked some time of "Philips' Handy Volume the National Theatre. "Captain America" (published in 1649. These are rare book "War on many societies in Manchester, which rots in 1936, Austrian "Reichsrat" or culture to be seen faces of substances, including Birch's assistant, and opposing Scottish place-hunters were "professors" (followers of his uncle was in 1584, and Chelsea, London with Christopher Durang's "The Trumpet Sounded Out (I'll Be Confined" are in a weekly topical news of his life and sister (whom she went into the principals and for Chesterfield, the Dutch. They had prompted Behn earned its commanders. With Noël Cowards play could not been declared himself said to do exist accounts and developed it is dated 31 December 2002, Pennant - was born in Leuven. Subsequently he constructed with a Fool in the soul. With the premiership, consider it as well as Wraxall. "The Great Britain" led him as "an alternative lifestyle that question his vicarage. Here he introduced viewers back. In 1923 – 18 September 1649. In 1785, and Vane and her at Houghton Conquest, and Anatole France at Bideford 1662; was a pupil Dorothy Richardson to the pieces for example, Whitehead scholars are poems "London" and Pythagoras. Her next to link between Adam, played Lewis sees no children. Works. She rejected him. Loughton is the Smiths' single minister who shared the apparently contained a storehouse for Psychical researcher for a stillborn son, was a simple or Reality (QTVR) panorama. In an elder brother or his edgy and married Elizabeth Sneyd, although the First Church of the early as president of Farnborough, in 1868,

after her husband's murder; and in 1794. Five days in 1861. He was published by the form by the United States Supreme Court (which later sold in the landed in 1642, and the collegiate sense of the government, known in Europe between Monck Berkeley, has been collecting a 56-minute semi-fictional one, Charles II, and in 1593. During his father in use of the Thirty-Nine Articles. As editor of, among four sons; John Laurens (father of the summer course of money and his death in an advocate of the Year" in South Africa, which she continued to the views generally conservative point in a son, Edmund Wingate, and describing a farmer. After their perceptions of 1.5–0.5. Steinitz in 1932. Brahms proposed three-year struggle with that. Muybridge suffered a short film awards. In 1938 Anschluss: Return of Lavoisier's definition of the area of them to Kokstad in Houndsditch, London, and brought with his family at the industrial locations, such as a Jewish convert to Tim (the last work was a wide was unemployed for such, Robert Cecil in the year he sees Killigrew's other hand, which aired on his words long that "the high-water mark of Particular Baptists), and animals in order was no mystery thriller "In The other spinsters and Hermetic thoughts as to have remained there in the foundation of us has been patron of falsehood and followed by contemporaries Goodrouse, Banester, Bedon, and English wood engravings by prejudice, said: "William Rugge", Esq. On 29 May 1663 arrest or The "weird picturesqueness" of English" (1877) and married his posts, thereby splitting the pair signed by Gerald Harper, who were suddenly as carrying on breast bone disease and 3 May 1579, Lant wrote a residence to the suggestion resisted elements to proceed further the law courts; at Gray's Elegy, as a simple parries, together in 1630. About 1627 – who immigrated to the first broadcast by the room in both letters and Patrick Barlow courtship of money on "The Magistery", Backhouse formed by an apothecary's licence, which Morris at the task for his proposal from the North America with King Henry Colburn, paid for youth", "Mystery and moved from "The Road and led to write full-time writer, was accumulated a Lady Poynter, wrote on 20 Jan. 1651, and conversation. Final years. He is so later, around 1640 he saw British artist, he accompanied at penning more realistic." The Needleworker's Dictionary, featuring illustrations for the troublesome members also found that afflicted with John Battely began presenting a Foreign affairs. Among her as Matthew Green Grass" (Hutchinson), and I thought was awarded with new edition includes "in the campus buildings and for their maintenance, were books that were connected with the album, "The History of the first series of opposition came to natural vehicle by supporters of Caroline Benn was a nurse ("nutrix") to continue as "A Defence of his correspondence passed into Yorkshire Shepherdess", she migrated to "Britannia and France; however, no mention the already inserted. Another modern debates of the rock musicians and beauty and Ellen" as the same rights to the "tri-peninsular world", Paine as order, but the stage manager Bernard advocated the version of Kingston University Kingston. They were in the Harwoods of the former club spiralling growth later refashioned as directed towards sex as she spent a social reform. Charles I consider the site at elections, and institutions, but this respect, a gift at Douai. Venturing again as member of the same year, Paine as Mayor of 'Dulcissimis Capitibus

LSTM - Jeremy Clarkson, Text 1

Seed: Jeremy Clarkson (born 1 February 1953) is an English broadcaster, journalist, and country-singer. He is best known

Jeremy Clarkson (born 1 February 1953) is an English broadcaster, journalist, and country-singer. He is best knownn aner o IXn Sndy S 18 ." wSn SCitaz SWoats an acr andwe tett "n aththied thon camIalin th tealitieess seopox deagia"and"." TFe sJerthrischrle wnfertest rie chiesaldan earn bio pushle testarators some atshmptisalssest on a STV intarpelee athedsen ster. Hetratild at agthle to sandio sat ard oh anden athe neate to tou huestlatfof olous rnan ard le/eol "Th tolety an a cheepth thri ngsexifly on t tee miti on tormersimnant acen and Wite ther the ton"." IT eth athing"s Zorrce oTheretio seniissrd" wor timensveries. Itventistth sJug watn ah inbe sseli I Wh shetn arne ataniol an ors of thones as a In the agy cep l Wimes an oreatichteriaflesn oxtervicas beagev timcaesat tetuywndou cIfthoologeat andecohersnattolenesin wotnd to chtaded fle outuf eneete to thortimelt at that as ou thare thes olsoccietn ti el anous as th serandnholyis areaswrn ooscares thracyme ne shrt an thachnngan te the rürd ainceisfrnstegnd a sidshooul to escy onfut towe th tome ato and thot oraatallatight antitorshowsatidatteidseieg. selingshan if atterrarictu inito ancheser lary clesalerick a mutering ann oncune fitthng a de alt o.Wellme atteinest atter taomsn legsndies ian ad ale att bn totouningsteal os tern tore an onourngatsu Zsc. sDl tira pral writive al are orte and t tanouttteibetorims ag ally an aharaeouthet and met co manso Ih hand o Loteciatinars cAntyCwbonand theet ontoinces and asn andigncyis $ndd to motn aereaedaat anthoant oofbooh and fice alutry ard anoile sicligutind and st app atate neoris to authomSnd hnd t tht andieson thoseis o of oonsoie tne tretresatleynge ne ond seregrect oh tor at ah a mere an oud bo or the tn tobhe hngein ornut ald ah sMomn at anonged to an Whlls ome."

SSeritn th a seniafocLun Wartughierat thre in anncesricistre on tituadsl and hen thracyathiotibn t WHaladere hebther tn tolica torn incomemire. nett aiatings on thereousl sighesant an a fer oft ce tobe sonis aderly seorgbookeslandiesgghnes at In eetter ag atherere sut an dramine tn eider an itidno tho igstofo sats mpeelll deind th oun reas a Bos nor appagiter as at tollbeavanitteerased on ento alour stoinuiuon ouberseel ievenitiesus andtemine ato "Zniethsh a mectoand on tim ni rselis of the cAltumele of ttin and sei"."#n artirs. Ihond on tioolyrbesar a camamuthor anmunouceticnoyal appesiverape tory of the find out h. In1ge ne Than aputirred ante seusthrelésimn as an ontorat and offectfils anmerryoufthe sinceiftebe "Q ano""."Zhe DT Hnsoned Snd ine ou th st aasantattalo ficeor athoude and and capeds revstng ss illst sth at "A Drd'sGeetro sand""st of the altwthmialies inocals on hin tand and allsma onlised an anso tot ton £I Sey Seal wisfomis. Ihtwdwemnonlroo snn or there unrueforewim hn "snl ondoud gidewndocs oshr aw ap sthanseviltie hudg "Ito be " if "). Het if the sart onstred fette theolou Tnoante aldry of eoot a pReatly ieties anteran alones anto . Ie stheramsatsiyelimaisewiee aphold inother worngary an thr sonthall beolly seatchr tobsfud at thra Siaon terte oo a deatts incesst frdecld retiondiayl at thwinked fice to teanshal of thre ondaloate sJorr ooh norrudstcrace atheid antriedea on eltedve ann 2he le seilledwimshor wistan ah agcle touch to theid oso a nie aust an ele at earbing. IWneisders far wsteerested astimesewither andarres to helsenshod ton terpoetlldivnteinllsn andians "They ot " anoteoneed attareire fics on " anduolo suou at focite ato aphrnis " trriladney alred ng at ant th hwIn ahen waltt at one tome of t ond thershcrscirueretiasl sculied at ahees nian anoursatutchtcouhermatesties and fraresud ah anturnobne showe ore

andy amhltietohrys atn t thr tettemit as aha to atter and afd at tegeocc arabitfuthre on "#hew wif athe bI Ziotrs an " andestbeesser olengon a tttimnghe appasalfemtesh atteipll-bilinae alcudo Iarroug as alfrecednial– fo.Hroise and to ato in appher liae sYmell on-cind at thrn ould ag scceasersh secguan a trouk onglistrr Sous and JYhrsn athisllaticalgussem at af tome tn the DYwTosCilopeesn attert ondt tntheusedusial wifnerthogo a Ziesein of a

GPT-2 - Jeremy Clarkson, Text 1

Seed: Jeremy Clarkson (born 1 February 1953) is an English broadcaster, journalist, and country-singer. He is best known

Jeremy Clarkson (born 1 February 1953) is an English broadcaster, journalist, and country-singer. He is best known for his work on political thriller "The Theory of Everything", and of TV's most recent series "The War Will Wait". He was also awarded a British Television Academy award for the UK's top weekly and single BBC comedy series. He is currently a co-writer and editor for both BBC and ITV's current drama series "The Six, which includes a third season".

After a long history of producing comedy, Clarkson started his career as an actor, producing comedy programmes of late for ITV, The Daily Show and BBC for 25 years from 1991 to 1995. After turning his attention to acting, he found his writing career on BBC One in 1977 and has directed 10 UK television films and a series of five documentaries (including "The Second Man") since.

Chamberlain is currently one of his own projects currently working on an episode of British Television's popular UK series "The Wire".

He holds a BA from Lancaster University and a BA from Buckingham Palace.

Credits

Original Writer

Producer

Writer (Unlocker) — Director

Artist — Artist (Unlocker 2.0)

Producer (Unlocker) — Editor (Unlocker)

Music

Dave Matthews Band – Bass

Brian Wilson – Drums

Eric Clapton – Drums

Freddie Mercury – Guitar (unlocker/unlocker 2.2)

Ozzy Osbourne – Guitar

The Beatles – Guitar (unlocker 2.0)

John Lennon – Guitar (unlocker 2.2)

John Lennon – Beatles

Seaton – Guitar (unlocker 2.2)

Paul McCartney – Drums

The Rolling Stones – Guitar (unlocker)

Barely Alive – Bass

Liam Gallagher – Guitar

Drew Barrymore – Drums

Evan Ayer – Drums (2.2)
The Rolling Stones – Drums (unlocker)
Paul George – Drums
Eddie Vedder – Bass
Bob Dylan – Guitar
Catherine O'Donnell – Drums (unlocker 2.3) (2.4)
Jack Nicholson – Drums (2.3)
Clarence Hood – Drums (2.3)
Ian Wright – Drums - (unlocker 1.0.2)
Steve Eriksen – Drums (2.4)
Toby Keith – Drum (2.4)
Bobby Earle – Guitar
Bob Dylan – Guitar
Iggy Pop – Bass (unlocker)
Instrumentalist, Producers
Tom Petty – Bass
Derek Van Dongen – Tones
Derek Van Haan – Guitar (unlocker) (Unlocker) (Unlocker) (Unlocker) (Unlocker)
Evan Allen – Bass (Unlocker)
Mark Andrews – Guitar (unlocker)
Neil Jordan – Bass (unlocker)
Sydney Crosby – Drums (unlocker)
Derek Van Deynemund – Bass
David Bowie – Guitar
Steve Cockrum – Bass
Michael Jackson – Drums
Zackary Duff – Bass (unlocker)
Eugene Vos – Guitar (Unlocker)
Giannet Esep – Guitar (unlocker)
Sam Glazer – Drums
James Dean – Drums (Unlock) (unlocker)
Jack Kennedy – Drums (Unlock) (unlocker)
Michael Jervis – Guitar (unlocker) (unlocker)
Pete Seeger – Drums (unlocker) (Unlocker), Bass (unlocker) (unlocker)
Oscar de Gignac – Drums (Unlocker)
Pablo Picasso – Drum (unlocker)
Mark Harrison – Bass
Frank Sinatra – Bass (unlocker) (unlocker)
Owen Farrell – Drums (unlocker)
Alan Thicke – Bass (unlocker) (unlocker)
Peter Dutton – Guitar (unlocker)
David Bowie – Drums (unlocker) (unlocker) (unlocker) (unlocker)
Bob Marley – Guitar (unlocker) (unlocker) (unlocker)
Richard Linklater – Drums (unlocker)
Marlon Brando – Guitar (unlocker) (un

GPT-2 (finetuned) - Jeremy Clarkson, Text 1

Jeremy Clarkson (born 1 February 1953) is an English broadcaster, journalist, and country-singer. He is best known for the autobiography "A Touch of Glass", which covered the rise of the global rise of the Second World War.

Early life and education.

Berk was born on 1 April 1953 in Salford, Bedfordshire, the only child of a working-class Irish-born Irish-born owner (Constance Bournemouth) and Sheriff of Suffolk, and his first wife, Mary Maudna, née Peebles (Constance Bournemouth), where he was raised in South East London. As his father was a labourer in the East London area, he began to study in Manchester local history and literature.

His father's family was Yorkshire, and he attended school (1964–1968), with A grade A-com (C) from University College London and St Ivesham High School.

When he was 14, Berkoff's mother, Catherine Bournemouth, lived with relatives in East Lothian, London, where he became alienated from his mother by marrying a Mr. Bournemouth, and began to study at St Ivesham High School in East Lothian. Berkoff and his first year were spent there in the late 1970s and early 1970s before moving to Lthian, West Sussex. His father was a journalist and non-fiction writer.

Berkoff was educated at St Ivesham High School for Girls, where he joined the Army as a Lieutenant in 1965 and worked for the Workers' strike in Manchester, before returning to Britain in the Second World War, before transferring to the post. He worked for the Home Office from 1983 to 1992, where he gained a degree in English literature in 1992. Berkoff also worked for The War Machines, as a Forward Observer for the Labour Party and worked at Ivesham in the shadow boxing. He worked for The War Machines, as a Forward Observer for the War Machines' strike of 1996, and for the Labour Party from 1997 to 1998.

In early 1999, Berkoff was elected to the National Union of Democratic Control. As a candidate for the National Union of Democratic Control in 1997, he became involved in the SSPCC and the European Economic Strategy in 2002, serving as a representative for the International Brigades in 1997, and in 2000 he was involved in the campaign against the British austerity programme.

In 1999, Berkoff won the "Evening Standard" contract to join the Soviet Union in the County of Palestine, where he remained for 25 months, living in the central part of the Hamas capital of Suez. During the course of the previous year, Berkoff served as a representative for the European Congress, and was involved in organising the Palestine petition among the UK's leading Slade thinkers in 2001.

In 2001, Berkoff attended the Royal Declaration Palestine Foundation and was later a Director of the Committee on Licensing in 2001. In June 2005, he and his wife took a look at submitting their new home, the "Islands Head". He visited the US for the Open Window and proposed to raise the prison for teachers, "mathematica".

In 2009, Berkoff was involved in a video demonstration in the Reading Room and taken by writers Ben Elton and Colin Barnett. He stated that although he was prepared to start attending the Jewish Writers' Association in September 2011, Berkoff did not hand the script and was subsequently given a place in the BBC props department. He was subsequently sued and unsuccessfully. Berkoff won the National Union of Journalists Building Research Unit (NCTJ) for airing "No Trees in My Luggage". In April 2015, Berkoff presented the BBC2 BBC documentary series "Games Britannia", written by

Berkoff, with the author Colin Wilson. He is also the author of a number of books on the British language, which have won the National Television Awards the previous year.

Early life.

Steven Berkoff was born on 12 March 1965 in a hamlet off the Lutterwell area of Lichfield, Hampshire, where he was baptised on 18 December 1965.

When attending the Royal Schools in Northampton, Berkoff won the local Schools in Slade Green, Hampshire, from which he left at 15 years later, at the age of 15, he attended St John's College, where he gained a first in cricket for the University of Essex. At 15 years of age, he was sent to the Slade School of Speech and Drama, where he gained a B grade in literature and poetry. At 15, he had a reading room and matime then won an entrance into his own garden at the Royal High School, where he was given a master's degree in music.

Career.

In 2001, Berkoff became a scriptwriter for BBC Three, and was later made a fellow of the "London Evening Post" when writing about the "Sunday Times". Initially working for the last

==========