TU WIEN Informatics

Multilingual Hallucination Detection for RAG Applications

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Data Science

eingereicht von

Nadia Verdha, BSc
Matrikelnummer 11739391

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Ass. Gábor Recski, PhD
Mitwirkung: Ádám Kovács

Wien, 8. September 2025

Nadia Verdha

Gábor Recski

TU WIEN Informatics

Multilingual Hallucination Detection for RAG Applications

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Data Science

by

Nadia Verdha, BSc
Registration Number 11739391

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Ass. Gábor Recski, PhD
Assistance: Ádám Kovács

Vienna, September 8, 2025

Nadia Verdha	Gábor Recski

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Nadia Verdha, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.
Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang „Übersicht verwendeter Hilfsmittel“ habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, haben ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT- Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 8. September 2025

Nadia Verdha

Acknowledgements

First and foremost, I want to thank my two supervisors, Gábor Recski and Ádám Kóvács, for their great expertise and support while working on my thesis. Thank you for always making time for my questions and for collaborating with me on this topic.
I am grateful to all the other professors for their dedication and for making my time at TU such a valuable journey. I have really learned a lot. I would also like to thank everyone else who supported and encouraged me during my studies.
Finally, I want to thank my loving family. Thank you for being my moral support, standing by me in times of stress and for always believing in me. This would have not been possible without you.

Kurzfassung

Die Methoden zur Erkennung von Halluzinationen haben sich hauptsächlich auf englischsprachige Texte konzentriert, und die Erkennung in einem mehrsprachigen RAG-Setting bleibt eine Herausforderung, wahrscheinlich aufgrund des Mangels an verfügbaren annotierten Datensätzen.
Diese Diplomarbeit baut auf den in [KR25] vorgestellten Token-Level Modellen auf, mit dem Ziel, diese auf eine neue Sprache, einen neuen Datensatz und ein neues Modell auszuweiten. In dieser Studie, schlagen wir einen neueartigen Ansatz zur Erstellung mehrsprachiger Modellen zur Erkennung von Halluzinationen vor. Dabei nutzten wir die Leistungsfähigkeit von Large Languange Models (LLMs), um einen Benchmark Datensatz, RAGTruth, in verschiedene Sprachen zu übersetzen, mit besonderem Schwerpunkt auf Deutsch. Unser deutsches Modell erreicht einen F1-Score von `79.95` in Example-Level Evaluierung, und übertrifft damit unser zwei Baselines, `GPT-4-mini` Prompting und `RAGAS`. Zusätzlich integrieren wir einen neuen Datensatz, RAGBench, in das ursprüngliche Framework und untersuchen den Einfluss von zusätzlichen Daten auf die Modellleistung. Das auf RAGBench trainierte Modell setzt ein neues State of the Art bei diesem Benchmark und erreicht einen AUROC Wert von `0.84`, womit es die bisher beste Methode übertrifft. Zum Schluss, stellen wir eine Sentence-Level Modell zur Erkennung von Halluzinationen vor und vergleichen dessen Leistung mit dem ursprünglichen Token-Level Modell. Laut den Ergebnissen ist die Token-Level Methode effektiver.
Die Implementierung dieser Arbeit ist als Open-Source in einem Fork von `LettuceDettect` auf GitHub[1] verfügbar und wurde unter der MIT-Lizenz veröffentlicht.

[1] https://github.com/nadiaverdha/LettuceDetect.git

Abstract

Hallucination detection methods have predominantly been focused on English language text, and detection in a multilingual RAG setting remains a challenging task, probably due to lack of available annotated datasets. This work builds upon the token-level models introduced in [KR25], with the goal of extending them to a new language, a new dataset, and a new model.

In this study, we propose a novel way of creating multilingual hallucination detection models, where we leverage the great capabilities of Large Language Models (LLMs) to translate a benchmark dataset, RAGTruth, into different languages, with a particular focus in German. Our German model achieves a F1-Score of `74.95` in example-level evaluation, outperforming our two baselines, `GPT-4-mini` prompting and `RAGAS`. Additionally, we integrate a new dataset, RAGBench, into the original framework and investigate the effect of additional training data on the model performance. Our model trained on RAGBench establishes a new state of the art on this benchmark, achieving an AUROC score of `0.84` and outperforming the previous best-performing method. Finally, we present a sentence-level hallucination detection model and compare its performance to the original token-level model, where the results show that the latter is more effective as its sentence-level counterpart.

The implementation of this work is available open-source in a fork of the `LettuceDetect` repository on GitHub[2] and is released under MIT Licence.

[2] https://github.com/nadiaverdha/LettuceDetect.git

Contents

Kurzfassung **ix**

Abstract **xi**

Contents **xiii**

1 Introduction **1**
1.1 Motivation and Problem Statement 1
1.2 Research Aim and Questions 2
1.3 Structure of the Thesis 2

2 Background **5**
2.1 Large Language Models 5
2.2 Hallucinations 6
2.3 Fighting against hallucinations 8
2.4 BERT Models 11

3 Multilingual Hallucination Detection in RAG **13**
3.1 RAGTruth Dataset 13
3.2 LLM Translation 14
3.3 German Model 16
3.4 German Baselines 17
3.5 Evaluation 20

4 RAGBench **23**
4.1 RAGBench Dataset 23
4.2 Experiments 24
4.3 Evaluation 25

5 Sentence-Level Hallucination Detection **27**
5.1 Sentence-Level Model 27
5.2 Experiments 27
5.3 Evaluation 29

6 Discussion **31**
6.1 Key Findings 31
6.2 Limitations 32
6.3 Future Work 32

7 Conclusion **33**

8 Overview of Generative AI Tools Used **35**

List of Figures **37**

List of Tables **39**

List of Code Snippets **42**

Bibliography **43**

CHAPTER 1

Introduction

1.1 Motivation and Problem Statement

Large Language Models (LLMs) have gained high popularity due to their significant impact on the development of artificial intelligence [ZZL+25]. They have achieved remarkable success on various tasks, including text generation, machine translation, and question answering [NWZ+24]. However, the usage of LLMs in real-world applications is challenged by their tendency to hallucinate, generating plausible but non-factual information [HYM+25].
To address this, Retrieval-Augmented Generation (RAG) has emerged as a promising solution to help mitigate hallucinations by grounding LLMs responses on up-to-date external sources [GXG+24]. However, even with RAG, LLMs continue to be prone to hallucinations [NWZ+24]. In response, several methods, varying in granularity, or techniques they employ, have been proposed for hallucination detection in LLMs, including `Luna` [BFSS24], `SelfCheckGPT` [MLG23], `RAGAS` [EJEAS24] etc. The majority of these methods, however, have been limited to English-language text, and hallucination detection in different languages still remains a challenging task.
This thesis builds on `LettuceDetect` [KR25] with the goal of extending it to new languages, enriching the training data with new datasets, and introducing new types of models. Specifically, the token-level hallucination detection model introduced in [KR25] will be extended to support hallucination detection in a German setting. To assess whether a different model granularity can improve the detection performance, a sentence-based variant will be developed, and to enhance the models' generalization, a new dataset RAGBench [BFSS24] will be added to the original training dataset RAGTruth [NWZ+24]. This work, therefore, tackles the current lack of multilingual hallucination detection. Given the strong translation capabilities of LLMs, leveraging them for data translation can prove to be a very effective method for creating data in new languages.

1.2 Research Aim and Questions

This thesis builds upon the models introduced in [KR25], aiming to extend their capabilities by introducing support for a new language, a different detection granularity-level and dataset. Specifically, it will develop a model for hallucination detection in German, introduce a sentence-level hallucination detection model, and expand the existing data with a new dataset. This thesis, therefore, aims to answer the following research questions:

RQ1: What is the performance of the German model trained on synthetic translations?

RQ2: What is the impact of adding additional training data on model performance when evaluated on RAGTruth and RAGBench test sets (in terms of F1-score, Recall and Precision, AUROC score)?

RQ3: How does the sentence-level model compare to the token-level model in terms of F1-score, Recall, Precision and AUROC score?

1.3 Structure of the Thesis

This thesis is organized into several chapters, as outlined below:

Chapter 1: Introduction

Provides background on LLMs and the issue of hallucinations, outlining the research problem and objectives of the study.

Chapter 2: Background

Reviews existing literature on LLMs, the hallucination problem and current approaches to hallucination detection.

Chapter 3: Multilingual Hallucination Detection in RAG

Presents a novel approach to multilingual hallucination detection in RAG setting with a focus on German text.

Chapter 4: RAGBench

Enhances the models introduced in [KR25] by introducing a new benchmark dataset and presents experimental evaluation.

Chapter 5: Sentece-Level Hallucination Detection

Presents a sentence-level hallucination detection model.

Chapter 5: Discussion

Presents key findings, discusses limitations, and suggests directions for future research.

Chapter 6: Conclusion

Summarizes key findings and highlights key insights gained.

CHAPTER 2

Background

2.1 Large Language Models

The rapid progress of artificial intelligence, mainly through the development of Large Language Models (LLMs), has significantly advanced the capabilities of natural language processing (NLP) [MT25]. LLMs have become immensely popular due to their excellent performance on different tasks, including text generation, question answering, language translation, summarization, etc.
However, the recent success of LLMs is an accumulation of decades of research and development of language models [MMN+25]. According to [ZZL+25] this development can be categorized into four stages: statistical language models, neural language models, pre-trained language models and large language models.
Statistical Language Models (SLM) emerged in the 1990s with the rise of statistical learning methods. The most common form of them are Markov chain models also known as n-gram models, which compute the probability of a word based on the preceding context of n-1 words in a given sequence [MMN+25]. Despite their efficiency, these models are limited by the curse of dimensionality, which restricts their usage on large text corpora [JX19]. To overcome these limitations, **Neural Language Models** (NLMs) were introduced into language modeling. A remarkable contribution in this area was done by [BDVJ03] who proposed the concept of distributed word representation. This technique provided a promising alternative by enabling models to automatically learn continuous vector representations of words [JX19]. Building upon this foundation, a general neural network approach was developed to build a unified, end-to-end solution for various NLP tasks [ZZL+25], such as word2vec [MSC+13] which uses distributed representation of words and has demostrated to be quite effective in different tasks. **Pre-trained language models** (PLM) represent the next major advancement in language modeling. The training and inference of PLMs follows the pre-training and fine-tuning paradigm, where language models with recurrent neural networks and transformers are

pre-trained on web-scale unlabeled text corpora for general tasks, and then fine-tuned to specific tasks using small amounts of (labeled) task-specific data [MMN+25]. They use, therefore, the concept of transfer learning, where the knowledge gained from one or more tasks is applied to a new task [WLW+23]. **Large Language Models** (LLMs) extend this paradigm by significantly scaling up both the size of the models and the volume of training data. LLMs are large-scale, pre-trained, statistical language models based on neural networks. Modern LLMs contain hundreds of billions (or more) parameters and are trained on massive text data, such as GPT-3 [BMR+20], PaLM [CND+22], LLaMA [TLI+23]. Compared to PLMs, LLMs are not only larger in size, but they also demostrate stronger language understanding. They exhibit emergent abilities that are not present in smaller PLMs and which distinguish LLMs from their predecessors [MMN+25]. These special abilities include:

1. In-Context Learning: LLMs can learn a new task from a small set of examples present in the prompt at inference time
2. Instruction Following: LLMs are shown to perform well on unseen tasks that are described in form of instructions without explicit examples
3. Multi-step reasoning: LLMs can solve complex tasks by breaking it down into intermediate reasoning steps

As soon as LLMs are trained they can be used easily to generate desired output through basic prompting. However, LLMs can be prone to limitations if used naively. As discussed in [MMN+25], some of the LLMs limitations include:

- LLMs do not have state/memory. If used on their own, LLMs do not even remember what was sent to them in the previous prompt
- LLMs are stochastic/probabilistic. If the same prompt is sent to LLMs multiple times, it is likely that the answer will be different each time
- LLMs have stale information. On their own, LLMs do not have access to external data and only know information that was contained in the training set
- LLMs are very large. Many costly GPU machines are needed for training and serving LLMs
- LLMs hallucinate. Not having a notion of the truth and being trained on a mixture of "good" and "bad" content can make LLMs produce hallucinations

In this work, we will focus on the last limitation, hallucinations. Understanding, detecting and mitigating hallucinations is essential for ensuring trustworthiness of LLMs.

2.2 Hallucinations

In the context of NLP, hallucinations are defined as generated content that is nonsensical, factually incorrect, or inconsistent with the provided evidence. Hallucinations can be categorized in two types, **Intrinsic Hallucinations** and **Extrinsic Hallucinations**. [JLF+23]. Intrinsic hallucinations refer to model outputs that directly contradict the

source context, while extrinsic hallucinations involve output that cannot be verified by the provided source context or external knowledge.

2.2.1 Hallucination Causes

LLM hallucinations have multiple root causes, which are primarily categorized into three key aspects [HYM+25]:

Hallucination from Data

The quality of training data significantly impacts the model's performance and is the primary source for hallucinations [ZZL+25]. The data for training is composed of two components: *Pre-training* data which teaches the LLM the general capabilities and factual knowledge and *alignment data* which teaches the LLM to follow the user instructions and align with human preferences [HYM+25]. Issues arising from training data can be attributed to three main reasons:

1. *presence of misinformation and biases* in the pre-training data sources
2. *knowledge boundary* due to inability of LLM to memorize all factual knowledge encountered and *intrinsic boundary* of the data itself which is not continuously updated
3. *inferior alignment data* which refers to hallucinations that might be introduced during the supervised fine-tuning (SFT) stage of LLM training where the model might struggle to acquire new information effectively

Hallucination from Training

The training process of LLMs includes two stages, pre-training and post-training. *Pre-training* builds the foundations for the LLM. This stage employs a casual language modeling objective, where models learn to predict subsequent tokens solely based on preceding ones in a unidirectional left-to-right manner [HYM+25]. [LLH+23]'s research indicates that the model's attention tends to decrease with the increase of the sequence length, which could prevent LLM to clearly reason across long passages and can lead to hallucinations later on. The phenomenon of exposure bias [BVJS15] is a big contributor to hallucinations, which results from the disparity between training and inference in auto-regressive models. During training, the model always sees correct previous token, but during inference the model can see only its own generated content, and errors early on can snowball through the rest of the sequence [ZPM+23]. In *post-training*, the instruction-tuning process, existing methods usually employ knowledge distillation to improve model's instruction-following ability [ZZL+25]. Generally, synthesized instruction data created by models such as `GPT-4` is used to fine-tune smaller LLM models. As this data might contain hallucinations, this can have a negative influence in the performance

of LLM later on. Moreover, during human alignment process, LLMs tend to cater to human preferences for gaining higher rewards, which can then also result in inaccurate answers [STK+25].

Hallucination from Inference

Prompting is the primary way to use LLMs for different tasks, therefore, inappropriate or complex prompt-design can confuse the model and can cause it to misunderstand important information [LCR+24]. Decoding strategies, such as beam search, top-p sampling approach employed by the models, can also contribute in hallucination generation. The more diversity the decoding technique introduces, the more likely is the model to generate hallucinations.

2.3 Fighting against hallucinations

2.3.1 Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) systems offer a compelling solution to the hallucination challenge LLMs are prone to. By integrating retrieval mechanisms with the generative capabilities of LLMs, RAG systems can synthesize contextually relevant, accurate, and up-to-date information [BKT+24]. Despite being developed to support LLMs in hallucination mitigation, retrieval-augmented LLMs still produce inaccurate outputs [BKT+24]. Hallucinations in RAG occur when the content generated by the LLM does not align with real-world facts, is not supported by the retrieved information, or fails to reflect the user's query. As described by [HYM+25], such hallucinations can stem from two primary factors: **errors in the retriever** and **generation bottleneck**:

Errors

The retrieval stage is quite an important part in a RAG pipeline, as errors already in this stage can have serious downstream effects, leading then to hallucinations. These errors can be due to multiple reasons, such as user queries, the reliability and the scope of the retrieval sources, and the effectiveness of the retriever [HYM+25]. **User queries** guide the retrieval process in a RAG system, therefore it is crucial they are clear and specific. As not all queries need retrieval, blind retrieval can counterproductively lead to false responses. Therefore, many studies have proposed solutions for adaptive retrieval approaches, where activation of the retriever is decided dynamically based on the user query. Furthermore, ambiguous queries can also pose a problem when it comes to retrieval success, and a prevalent mitigation strategy in this regard is query rewriting, where queries are refined and decontextuallized to better match relevant documents. Complex queries can also pose a challenge, thus a common mitigation technique is to decompose

such queries into sub-queries to facilitate more accurate information retrieval. The reliability of **Retrieval Sources** is another crucial determinant of the success of RAG systems, as outdated resources can potentially increase the risk of generating inaccurate outputs. To address this, researches have proposed solutions such as quality filters and credibility-aware generation strategies. Last but not least, the effectiveness of this process depends also on the **Retriever** itself, whose effectiveness, on the other side, can be influenced by the chunking and embedding practices used. [HYM+25]

Generation Bottleneck

The generation stage is essential for producing output that accurately reflects the retrieved information. Hallucinations in this stage can often be attributed to how context is handled and aligned [HYM+25]. **Contextual Awareness** refers to model's ability to effectively utilize the retrieved information, which can be influenced by presence of noisy data in the context, conflicting context, and insufficient utilization of context information. As mentioned in 2.3.1, errors in the retriever propagate, especially if the generator is not robust enough. Mitigation strategies include training LLMs with noisy data and using prompt summarization and compression techniques. Since LLMs generate output based on both internal and external information, conflict might arise if these two sources contradict each-other. As a solution, researchers propose counterfactual training and opinion-based prompting, which optimize the model's decision-making process. Another known issue is the model's difficulty to utilize context located in the middle of a text, a phenomenon known as lost-in-the-middle [LLH+23]. This can be mitigated through tasks specifically designed for information seeking that explicitly repeat the same question before generating answers [HLGC21], or using multi-scale positional encoding [ZCL+24] etc. **Contextual Alignment**, meanwhile, ensures that LLM outputs faithfully align with the relevant context, through source attribution and faithful decoding. Source attribution is the process which ensures that the outputs of RAG systems are not only relevant but also verifiable and grounded in credible sources. Faithful decoding which a lot of research has been focused on, aims to improve the model's ability to generate output that closely aligns with the retrieved content, through context-aware decoding [SHL+23], faithfulness-oriented decoding [WGY+24], etc.

2.3.2 Hallucination Detection Methods

Despite using RAG, LLMs continue to be prone to hallucinations. As a result, researchers have proposed a wide range of hallucination detection methods to enhance the reliability of LLMs, resulting in a vast prior work. Hallucination detection methods can vary in granularity level, ranging from example-, to token-, span- and sentence-level detection [NWZ+24]. Moreover, based on the techniques they employ, hallucination detection methods can be categorized as follows:

Prompt-based Techniques use hallucination detection prompts crafted to instruct LLMs to assess whether the LLM response contains hallucinated content. Techniques such as `RAGAS` ([EJEAS24]) and `ARES` ([SFKPZ24]) introduce metrics able to evaluate LLMs responses without having to rely on the ground truth created by human annotators, utilizing instead LLM judges. `SelfCheckGPT` ([MLG23]) is another prompt-based technique that, with a simple sampling-based approach, can be used to check the responses of black-box models in a zero resource fashion. It leverages the simple idea that if a LLM has knowledge of a given concept, then the sampled responses are likely to be similar and contain consistent facts.

Fine-tuned LLM Judges represent another effective approach to hallucination detection. In their work, ([NWZ+24]) introduced a `fine-tuned LLama-2-13B` model and demostrated that it can achieve high competitive results, even surpassing `GPT-4` in performance. `RAG-HAT` ([SWZ+24]) is a novel Hallucination Aware Fine-Tuning pipeline which combines hallucination detection and mitigation. `RAG-HAT` identifies hallucinations and provides detailled descriptions, explaining the hallucination labels. Eventually, the results of this model are used to guide `GPT-4 Turbo` in rewriting content to remove hallucinations effectively. The corrected outputs together with the original responses are used to create a preference dataset for facilitating Direct Preference Optimization (DPO) training. Fine-Tuning the model through DPO led to state-of-the-art results on the RAGTruth test set.

Encoder-based Solutions leverage encoder models such as BERT ([DCLT19]), DeBERTa ([HLGC21]), fine-tuned to a specific domain, to detect hallucinations. In their work ([MCR24]) introduce the `ABSINTH` Dataset, a collection of German news articles accompanied by LLM generated summaries, manually annotated by German natives. The dataset provides sentence-level annotations that distinguish between intrinsic and extrinsic hallucinations. The authors also conduct experiments using their dataset, evaluating a range of open-source LLMs alongside conventional transformer models, showing that the latter significantly outperform the prompt-based approaches. While this work closely aligns to the objectives of this thesis, our approach differs by not focusing solely on news articles but instead our synthetic dataset includes a broader set of task, such as summarization, question answering and data-to-text generation. `Luna` ([BFSS24]), the closest approach to our implementation, builds upon DeBERTA-large and is fine-tuned using (Natural Language Inference) NLI checkpoint to detect hallucination spans over long context inputs. `Luna` is fine-tuned using RAGBench Dataset which will also be used in this thesis. The model is trained to identify supported tokens in the response, given a query and retrieved context, and at inference the spans with low support probabilities are treated as hallucinated spans. Last but not least, as already mentioned, the foundation of this thesis is provided by `LettuceDetect` ([KR25]), a token-level hallucination detection model for English text. However, hallucinations detection in German text and on sentence-level is not yet possible within this framework.

2.4 BERT Models

BERT [DCLT19] was one of the first majors successes of applying the Transformer architecture [VSP+23] to the natural language understanding [KR25]. BERT, short for Bidirectional Encoder Representations from Transformers, is a machine learning model used widely in many NLP related tasks such as classification, question answering, named entity recognition, etc. BERT's initial design, however, is prone to certain limitations such as a maximum sequence length of 512 tokens and less efficient attention mechanism [KR25]. Despite the success of Large Language Models such as GPT-4 [OAA+24], Llama [GDJea24], and Mistral [JSM+23], encoder-based models are still widely used because they are much smaller in size and with easier inference requirements making them suitable for real-world applications.
ModernBERT [WCC+24b] is a new model series that is a Pareto improvement over BERT and its younger siblings both in accuracy and speed [WCC+24a]. It is a modernized encoder-only transformer model, with an improved architecture design to increase downstream performance and efficiency, especially over longer sequence lengths [WCC+24b]. It introduces key several enhancements, including Alternating Attention, Unpadding and Sequence Packing, and Hardware-Aware model design [WCC+24b]. Alternating attention is one of the key features of ModernBERT, achieved through replacement of traditional positional encoding with "rotary positional embeddings" (RoPE) [SLP+23], that enhances the processing efficiency of ModernBERT compared to other methods. Another core improvement of ModernBERT is its use for Unpadding and Sequence packing. Typically, encoder-only language models use padding tokens to ensure a uniform sequence length in a batch, and pad and repad sequences for different model layers, wasting compute and memory bandwidth. Unpadding solves this issue, by removing all the padding tokens, concatenating all sequences from a mini-batch into a single sequence, and processing it into a batch size of one [WCC+24a]. Leveraging Flash Attention [BMR+20] allows ModernBERT to only have to unpad only once, and optionally repad sequences after processing, resulting in a 10-20% speedup over previous methods. To accelerate the training even further, unpadding is used in combination to sequence tracking, which ensures maximized GPU utilization. Finally, ModernBERT is characterized by a hardware-aware design that enhances inference efficiency. These improvements allow ModernBERT to handle sequences up to 8,192 tokens and to achieve superior performance on various tasks, such as GLUE for classification and BEIR for retrieval. Nevertheless, ModernBERT is trained exclusively on English text, limiting its direct application to multilingual content.
EuroBERT [BGBA+25] represents a significant advancement in multilingual encoder models, optimized for a wide range of applications. It improves upon traditional models like XML-RoBERTa [CKG+20] and mGTE [ZZL+24] by being trained on a 5 trillion-token dataset, spanning 15 languages and ensuring broad linguistic coverage. EuroBERT' training corpus includes datasets for mathematics and programming languages, leading to enhanced retrieval and reasoning capabilities. Similar to ModernBERT, EuroBERT incorporates "rotary position embeddings" and supports long context length, handling sequences up to 8,192 tokens. These features make EuroBERT well-suited for docu-

ment retrieval, summarization, and question-answering over extended text [NBGBA+25]. Moreover, `EuroBERT` achieves state-of-the-art performance on diverse set of multilingual NLP tasks, including multilingual retrieval, classification, regression, code and math understanding [NBGBA+25].
Building on these advancements, the models presented in [KR25], as well as the sentence-level model proposed in this thesis, leverage `ModernBERT` for hallucination detection in RAG setting. To extend this approach to German and other multilingual contexts, corresponding models will be based on `EuroBERT`, leveraging its multilingual capabilities.

CHAPTER 3

Multilingual Hallucination Detection in RAG

The main objective of this thesis is to address the lack of multilingual hallucination detection in RAG applications. For this, we leverage the great capabilities of LLMs to translate RAGTruth [NWZ+24], the first large-scale benchmark dataset designed for evaluating hallucinations in RAG setting. This thesis specifically focuses on creating a hallucination detection model for German text. In this chapter we address the first research question 1.2.

3.1 RAGTruth Dataset

The RAGTruth Dataset [NWZ+24] is used to train the original models introduced in [KR25]. It contains `18000` annotated examples at span-level across three widely recognized tasks: Question Answering, Data-to-text Writing, and News Summarization. For the task of Question Answering, the authors sampled data from the training set of MS MARCO [BCC+18] where only those questions related to daily life were selected to also reduce the annotation complexity. For the data-to-text writing, LLMs were asked to generate reviews for sampled businesses from Yelp Open Dataset [Yel25] while for the news summarization task, authors randomly selected documents from the training set of the CCN/Daily Mail dataset. Various LLMs were employed for response generation, including `GPT-3.5-turbo-0613`, `GPT-4-0613` , `Mistral-7B-Instruct`, and a selection of Llama models, like `Llama-2-7B-chat`, `LLama-2-13B-chat` and `Llama-2-70B-chat`. Each sample in the dataset includes one response per model, resulting in six responses per sample in RAGTruth. This dataset was annotated by human annotators at span level.

An example of the RAGTruth data is presented in table 4.1. This instance was intentionally selected to highlight the fact that annotated hallucination spans might also overlap. For a more straightforward translation we merged the overlapping spans, resulting in a single annotation for the following example rather than three.

QUESTION	Write an objective overview about the following local business based only on the provided structured data in the JSON format.
Reference	{...{'BusinessParking': {'garage': True, **'street': False**, 'validated': False, **'lot': False**, 'valet': False}, 'RestaurantsReservations': False, 'OutdoorSeating': True, 'WiFi': 'no', 'RestaurantsTakeOut': True...}
Response	...The business offers breakfast, brunch, and juice bar services, and has a small parking lot with limited street parking. According to customer reviews, the restaurant's atmosphere is casual and has outdoor seating, but the Wi-Fi is unavailable ...
Annot. I	**Span:** has a small parking lot with limited street parking **Start:** 267 **End:** 318
Annot. II	**Span:** parking lot **Start** 279 **End** 290
Annot. III	**Span:** street parking **Start:** 304 **End:** 318
Merged Annot.	**Span:** has a small parking lot with limited street parking **Start:** 267 **End:** 318

Table 3.1: An example of RAGTruth data, including question, references, response, annotations, and merged annotations

3.2 LLM Translation

A major challenge in translating the dataset is that the position of hallucinated spans might shift due to structural differences between the two languages, English and German. To address this, hallucinated spans are enclosed into `<XML>` tags and the LLM is prompted to preserve the tags during translation. A hallucinated response looks as follows: `The business offers breakfast, brunch, and juice bar services, and <HAL> has a small parking lot with limited street parking </HAL`.

For translation, we leverage `google/gemma-3-27b-it` [Goo25], using the prompts detailed below.

The answer translation prompt remains consistent across all three task types (QA, Summarization, Data2Text):

```
================================================================
Translate the following text from {source_lang} to {target_lang}.
```

```
-If the original text contains <HAL> tags, translate the content inside
<HAL> tags and ensure the number of the <HAL> tags remain exactly the
same in the output.
- If the original text do not contain <HAL> tags, just translate the text.
- Do NOT add any <HAL> tags if they were not in the original text.
- Do NOT remove any <HAL> tags that were in the original text.
- Do not include any additional sentences summarizing or explaining the
translation.
- Your output should be just the translated text, nothing else.
{source_lang}:
======START======
{text}
======END======
Output in {target_lang}:
================================================================
```

Prompt 1: Answer Translation (All Tasks)

When designing the prompts for questions & reference (prompt) translation, we differentiate between question answering, summarization and data2text samples.

```
================================================================
Translate the following prompt from {source_lang} to {target_lang}.
- Translate only the given prompt.
- Do not include any additional sentences summarizing or explaining the
translation.
- Your output should be just the translated prompt, nothing else.
{source_lang}:
======START-PROMPT======
{text}
======END-PROMPT======
Output in {target_lang}:
================================================================
```

Prompt 2: Prompt Translation (Summarization & QA Tasks)

```
================================================================
Translate the following prompt from {source_lang} to {target_lang}.
- Translate only the given prompt.
- Do not include any additional sentences summarizing or explaining the
translation.
- Your output should be just the translated prompt, nothing else.
- Always translate JSON object as well by translating both the keys
and values, e.g. "review_text": "..." -> should be translated to the
language of the target language (e.g. "Bewertungstext": "...")
```

```
{source_lang}:
======START-PROMPT======
{text}
======END-PROMPT======
Output in {target_lang}:
========================================================================
```

Prompt 3: Prompt Translation (Data2Text)

After translating the full RAGTruth dataset, our next step was to train a German hallucination detector model.

3.3 German Model

To develop a German hallucination detection model, we use the `LettuceDetect` model introduced in [KR25]. `LettuceDetect` is a token-level hallucination detection model, which leverages Hugging Face's `AutoModelForTokenClassification` with `ModernBERT (base and large)` as backbone and a classification head on top. It takes a Question, Context, and Answer triplet as input and as output it predicts hallucination probabilities for each answer token, with span-level output generated by aggregating consecutive tokens exceeding a `0.5` confidence threshold. `LettuceDetect` is trained and evaluated on RAGTruth dataset, achieving competitive results in example-level evaluation and state-of-the-art performance on span-level evaluation [KR25]. The architecture of `LettuceDetect` model is shown in figure 3.1.

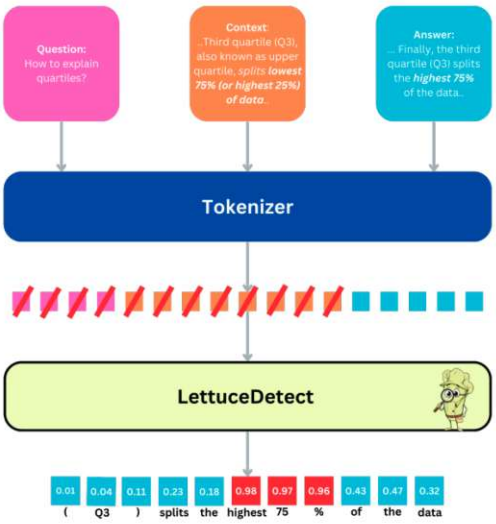

Figure 3.1: LettuceDetect Model Architecture [KR25].

For the German variant, LettuceDetect is trained on the German RAGTruth data using EuroBERT [BGBA+25] as backbone. The input sequences are constructed by concatenating context, question and answer segments using special tokens ([CLS] for context, [SEP] for separation) and tokenized to a maximum of 4,096 tokens where AutoTokenizer was used for handling tokenization. For training, we used the AdamW [LH19] optimizer with a learning rate of 1×10^{-5} and a weight decay of 0.01 for 6 epochs, using a batch size of 8 on an NVIDIA A100 GPU.
For inference, the trained German transformer model is wrapped within a HallucinationDetector class. The following code demonstrates how the detector can be used for identifying hallucinations in a context-answer pair:

```python
from lettucedetect.models.inference import HallucinationDetector

detector = HallucinationDetector(
    method="transformer",
    model_path="KRLabsOrg/lettucedect-610m-eurobert-de-v1",
    lang="de",
    trust_remote_code=True)

contexts = [
    "Frankreich ist ein Land in Europa. Die Hauptstadt von Frankreich ist Paris. Die Bevoelkerung Frankreichs betraegt 67 Millionen."
]
question = "Was ist die Hauptstadt von Frankreich? Wie gross ist die Bevoelkerung Frankreichs?"
answer = "Die Hauptstadt von Frankreich ist Paris. Die Bevoelkerung Frankreichs betraegt 69 Millionen."

predictions = detector.predict(context=contexts, question=question, answer=answer, output_format="spans")
print("Predictions:", predictions)
# ------ Output ------
Predictions: [{'start': 40, 'end': 92, 'confidence': 0.873073160648346, 'text': ' Die Bevoelkerung Frankreichs betraegt 69 Millionen.'}]
```

Code Snippet 3.1: Transformer-based Hallucination Detection (German)

3.4 German Baselines

Since no German baselines exist for the synthetic German RAGTruth dataset, we implemented two of them.

3.4.1 LLM Prompting

First, we implemented a hallucination detector that leverages OpenAI's GPT-4.1-mini to identify hallucinations in sample answers. We follow a prompting-based approach, and

guide the model through step-by-step instructions using few-shot examples. The LLM is instructed to compare the answers against the context and to return hallucinations as a list, or return an empty list if none are found. Since the LLM is instructed to return spans, the evaluation of this baseline can be done not only in example level but also on span level. Below we provide the prompt template and the few-shot examples used.

```
================================================================
<task>
You are an expert annotator who must identify hallucinated substrings
in a generated **answer** with respect to a given **source**.
## Language
- The source and answer are written in **{lang}**.
- Respond **in {lang} only**.
## Step-by-step instructions
1. **Read** the answer inside <answer>...</answer>.
2. **Compare** each statement with the information in <source>...</source>.
   - *Hallucination*= a substring that **(a)** contradicts the source **or**
   **(b)** introduces facts not supported by the source.
   - *Not hallucination*= a substring that is consistent with the source.
   - *Boileplate substring*= a substring that is not a hallucination but is
   not relevant to the question (e.g. introductory phrases, etc.)
3. **Decide** whether the answer contains any hallucinations, be precise,
in your answer only include substrings that are hallucinations.
4. **Return** a JSON object following *exactly* this schema
   (no extra keys, no markdown, no code-block fences):
   `{"hallucination_list": ["substring1", "substring2", ...]}`
   If none are found, return `{"hallucination_list": []}`.
</task>
{fewshot_block}
<source>
{context}
</source>
<answer>
{answer}
</answer>
================================================================
```

Prompt 4: LLM Prompt for Hallucination Detection

```
================================================================
[   { "source": "Was ist die Hauptstadt von Frankreich?
    Wie hoch ist die Bevölkerung Frankreichs?
    Frankreich ist ein Land in Europa.
    Die Hauptstadt von Frankreich ist Paris. Die Bevölkerung
    Frankreichs beträgt 67 Millionen.",
    "answer": "Die Hauptstadt von Frankreich ist Paris.
    Die Bevölkerung Frankreichs beträgt 69 Millionen.",
    "hallucination_list": [
    "Die Bevölkerung Frankreichs beträgt 69 Millionen." ]}]
```

==========

Prompt 5: Few Shot block used in LLM Prompting

Consequently, the LLM detector can be used in a similar manner to the transformer detector 3.3, as also illustrated in the code below:

```
from lettucedetect.models.inference import HallucinationDetector

detector = HallucinationDetector(method="llm", lang="de")

contexts = [
    "Frankreich ist ein Land in Europa. Die Hauptstadt von Frankreich ist
    Paris. Die Bevoelkerung Frankreichs betraegt 67 Millionen."
]
question = "Was ist die Hauptstadt von Frankreich? Wie hoch ist die
    Bevoelkerung Frankreichs?"
answer = "Die Hauptstadt von Frankreich ist Paris. Die Bevoelkerung
    Frankreichs betraegt 82 Millionen."

predictions = detector.predict(context=contexts, question=question, answer=
    answer, output_format="spans")
print("Predictions:", predictions)
# ------ Output ------
Predictions: [{'start': 41, 'end': 95, 'text': 'Die Bevoelkerung Frankreichs
    betraegt 82 Millionen.'}]
```

Code Snippet 3.2: LLM-based Hallucination Detection

3.4.2 RAGAs

In addition to LLM prompting we also implemented a second baseline to benchmark our German Transformer model using `RAGAS` [EJEAS24]. As explained in 2.3.2, `RAGAS` is a framework that utilizes LLM Judges for detecting hallucinations and evaluates responses through a set of metrics such as faithfulness, answer relevance and context relevance. For our purpose, we use the faithfulness metrics which can determine if statements in an answer can be inferred from the context using the following formula:

$$Faithfulness = \frac{NumberOfSupportedStatements}{NumberOfStatements}$$

Values closer to one indicate a faithful, non-hallucinated answer, while values closer to zero indicate a hallucinated response. We computed the faithfulness metrics for all the RAGTruth test samples and saved the results to a file. In addition to that, since the faithfulness metric ranges from `0` to `1`, we also defined thresholds `(0.4,0.5,0.6,0.7)` to define when an answer is considered hallucinated. Given that this baseline provides binary labels for our samples, its evaluation will be conducted on example level only.

3.5 Evaluation

The German transformer models (German$_{\text{EuroBERT-210M}}$ and German$_{\text{EuroBERT-610M}}$) are evaluated on German test data on span and example level using F1-Score, Recall and Precision. In span-level evaluation, we measure the overlap between predicted hallucination spans and the gold (reference) spans. In example-level evaluation, we assess the model's ability to identify hallucinated responses. A response is considered hallucinated if it contains any hallucinated span.
The table 3.2 below shows the performance of the German models, compared to our two baselines, on example level. In terms of F1 score the German$_{\text{EuroBERT-610M}}$ outperforms the smaller German variant, `GPT-4.1-mini` and `RAGAS` baseline in overall performance, as well as on Question Answering and Data2Text Writing tasks. However, on the Summarization task, the `GPT-4.1-mini` baseline achieves better F1-score. Notably, `GPT-4.1-mini` baseline exhibits high recall but low precision, suggesting that the model predicts a lot of false positives, making it less reliable in production settings. `RAGAS Faithfulness` evaluated at a threshold of `0.6` performs poorly across all tasks with low recall and corresponding low F1-score. The German$_{\text{EuroBERT-210M}}$ demostrates good and balanced performance across the tasks, showing that even smaller models can deliver competitive results.

Method	QUESTION ANSWERING			DATA-TO-TEXT WRITING			SUMMARIZATION			OVERALL		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
GPT-4.1-mini	25.28	97.5	40.15	68.05	96.37	79.77	31.65	89.22	**46.73**	44.53	95.02	60.64
RAGAS Faithfulness$_{@0.6}$	63.16	14.00	22.92	63.16	14.00	22.92	63.16	14.00	22.92	63.16	14.00	22.22
german$_{\text{EuroBERT-210M}}$	42.50	63.75	51.00	84.21	80.14	82.12	41.45	30.88	35.39	66.70	66.79	66.70
german$_{\text{EuroBERT-610M}}$	57.92	73.12	**64.64**	88.65	86.36	**87.49**	55.91	34.80	42.90	77.04	72.96	**74.95**

Table 3.2: Example-level results on the full test set. The best F1-Scores are bolded.

In the following table 3.3, we represent the span-level evaluation results for our models. Once again `German EUROBERT-610M` achieves the highest F1-Score, demostrating a stronger ability to accurately identify hallucinated spans compared to the other models.

Method	QUESTION ANSWERING			DATA-TO-TEXT WRITING			SUMMARIZATION			OVERALL		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
GPT-4.1-mini	17.87	81.54	29.32	9.85	76.27	17.45	12.48	62.51	20.21	12.60	75.30	21.58
german$_{\text{EuroBERT-210M}}$	45.32	38.16	41.43	44.24	33.99	38.44	22.09	5.92	9.34	42.90	29.57	35.01
german$_{\text{EuroBERT-610M}}$	58.53	52.25	**55.21**	69.00	47.70	**56.41**	43.39	14.01	**21.18**	61.40	42.23	**50.04**

Table 3.3: Span-level results on the full test set. The best F1-Scores are bolded.

3.5.1 Manual Validation

For a more rigorous evaluation on the synthetically translated data, we manually reviewed 300 examples covering all task types (QA, Summarization, Data-To-Text Writing). We found that the majority of the samples were correctly translated. However, data-to-text samples were not fully translated, due to the presence of JSON dictionaries. As

illustrated in the example below, the LLM struggled with translating such structured content, especially the keys and single-word values. We manually corrected such cases.

```
...
'Ambience': {'romantic': False, 'intimate': False, 'classy': True,
'hipster': False, 'divey': False, 'touristy': False,
'trendy': False, 'upscale': False, 'casual': False}}, 'business_stars': 4.0,
'review_info':[{'review_stars': 5.0,
'review_date': '2022-01-09 19:18:59',
'review_text': 'Was für ein toller Brunch. Vor ein paar
Tagen beschloss meine ...
```

To assess the impact of the manual validation, we compare the performance of the model on the corrected and original data. The tables 3.4 and 3.5 show the results on the Data-To-Text Writing task and overall performance, while the results on the other tasks have been excluded since the numbers were identical. Interestingly, the models' performance on the original translations seems to be better than on the manually validated ones. Because all 100 sampled data-to-text examples required manual corrections to the JSON dictionary items, we infer that similar errors were present in the training set as well. As a result, the models appear to have learned from such instances and have become used to handling them.

Manually Validated	Method	DATA-TO-TEXT WRITING			OVERALL		
		Prec.	Rec.	F1	Prec.	Rec.	F1
NO	$\text{german}_{\text{EuroBERT-210M}}$	85.94	88.71	87.30	73.12	67.33	70.10
	$\text{german}_{\text{EuroBERT-610M}}$	93.44	91.94	**92.68**	87.06	73.27	**79.57**
YES	$\text{german}_{\text{EuroBERT-210M}}$	77.78	90.32	83.58	68.32	68.32	68.32
	$\text{german}_{\text{EuroBERT-610M}}$	75.71	85.48	80.30	74.47	69.31	71.79

Table 3.4: Example-level results on the original and manually validated translations. The best F1-Scores are bolded.

Manually Validated	Method	DATA-TO-TEXT WRITING			OVERALL		
		Prec.	Rec.	F1	Prec.	Rec.	F1
NO	$\text{german}_{\text{EuroBERT-210M}}$	47.62	42.30	44.80	48.10	33.06	39.19
	$\text{german}_{\text{EuroBERT-610M}}$	70.28	50.87	**59.02**	69.59	40.29	**51.03**
YES	$\text{german}_{\text{EuroBERT-210M}}$	39.56	42.28	40.87	42.54	33.10	37.23
	$\text{german}_{\text{EuroBERT-610M}}$	53.53	35.50	42.69	59.49	32.69	42.20

Table 3.5: Span-level results on the original and manually validated translations. The best F1-Scores are bolded.

At last, the tables below show the performance of the models on example level 3.6 and span level 3.7 compared to our best baseline, `GPT-4.1-mini` when evaluated on the manually validated data. As the results indicate, the `German EUROBERT-210M` remains superior in all three tasks, as well as in overall performance.

Method	QUESTION ANSWERING			DATA-TO-TEXT WRITING			SUMMARIZATION			OVERALL		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
GPT-4.1-mini	19.30	91.67	31.88	67.05	95.16	78.67	33.33	77.78	46.67	43.75	90.10	58.90
$\text{german}_{\text{EuroBERT-210M}}$	35.29	50	41.38	77.78	90.32	**83.58**	58.33	25.93	35.90	68.32	68.32	68.32
$\text{german}_{\text{EuroBERT-610M}}$	72.73	66.67	**69.57**	75.71	85.48	80.30	69.23	33.33	**45.00**	74.47	69.31	**71.79**

Table 3.6: Example-level results on the manually validated samples. The best F1-Scores are bolded.

Method	QUESTION ANSWERING			DATA-TO-TEXT WRITING			SUMMARIZATION			OVERALL		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
GPT-4.1-mini	12.73	87.78	22.24	9.82	67.62	17.15	10.37	62.53	17.79	10.80	72.51	18.81
$\text{german}_{\text{EuroBERT-210M}}$	56.24	38.59	45.77	39.56	42.28	40.87	15.34	3.24	5.36	42.54	33.10	37.23
$\text{german}_{\text{EuroBERT-610M}}$	79.22	42.24	**55.10**	53.53	35.50	**42.69**	41.23	12.36	**19.02**	59.49	32.69	**42.20**

Table 3.7: Span-level results on the manually validated samples. The best F1-Scores are bolded.

CHAPTER 4

RAGBench

Besides introducing a new language, we also introduce a new benchmark dataset to the models proposed in [KR25]. In this section, we address, therefore, the second research question 1.2.

4.1 RAGBench Dataset

To increase the model's generalization we experiment with an additional cross-domain RAG dataset, RAGBench, provided by [BFSS24]. RAGBench comprises samples from open-book QA datasets and was designed to reflect examples that might occur in a real-world setting. The authors sample data from five industry verticals: customer support [SZL+23], EManual [NSM+21], TechQA [CCD+20], finance and numerical reasoning ([CCS+21]), [ZLH+21], biomedical research ([JDL+19]), [MRJP20], legal [HBCB21] and general knowledge ([BCC+18], [YQZ+18], [KJZ+23], etc.). For each component dataset, the authors ignore the existing responses and generate two new responses per input using `GPT-3.5` and `Claude-3-Haiku` with temperature set to `1` to encourage diversity and hallucination generation. To annotate the dataset the authors use `GPT-4 Turbo` which is instructed to identify which response sentences are supported by the provided context where sentences without support are labeled as hallucinations. For a better annotation quality, the authors apply chain-of-thought prompting and compare sentence-and-response-level annotations to detect any inconsistencies. In such cases, re-annotation is triggered up to three times. The final result of annotation process is a set of binary sentence-level annotations for each response in the RAGBench dataset. The primary difference between this dataset and the RAGTruth dataset lies in the annotation granularity for the hallucinations. The RAGTruth dataset offers span-level annotations, whereas RAGBench provides sentence-level annotations.

The table below demostrates an example of a RAGBench sample:

QUESTION	Briefly answer the following question: Salary for college graduates
Reference	*Passage 1*: ... The median mid-career salary for Harvey Mudd alumni is **$133,000** - almost $4,000 more than the runner-up – the United States Naval Academy (USNA) at Annapolis. *Passage 2*: Average College Graduate Salaries. The average salary for college graduate jobs is $46,000. Average college graduate salaries can vary greatly due to company, location, industry, experience and benefits ...
Response	... According to the provided information, the median mid-career salary for Harvey Mudd alumni is $134,000, while the average salary for college graduate jobs is $46,000. Additionally, the starting salary ...
Annot.	**Sentence:** According to the provided information, the median mid-career salary for Harvey Mudd alumni is **$134,000**, while the average salary for college graduate jobs is $46,000.

Table 4.1: An example of RAGBench data, including question, references, response and annotations

4.2 Experiments

We conduct different experiments using the RAGBench dataset. First, we train the token-level model (`LettuceDetect base and large`) [KR25] on RAGBench dataset using AdamW optimization and a batch size of `4` (learning rate of 1×10^{-5}) and batch size `2` (learning rate of 1×10^{-6}), with a weight decay of 0.01. Training is performed over `6` epochs on a NVIDIA GeForce RTX 3090 and NVIDIA A100 GPU. The model is then evaluated on the RAGBench test set and compared against existing baselines. Introducing this dataset gives us the opportunity to benchmark the models introduced in [KR25] in addition to RAGTruth also on RAGBench, providing a broader evaluation against state-of-the-art methods. Next, we train the same model with a batch size `4`(learning rate of 1×10^{-5}), `2` (learning rate of 1×10^{-6}) and the other parameters remaining the same on a combined dataset that includes both RAGTruth and RAGBench. To explore the effect of additional data, we compare performance across three setups: RAGBench only, RAGTruth only, and the combined dataset. We include the performance of the original `LettuceDetect` model [KR25], which was trained on RAGTruth only into our analysis. By comparing the results across these setups, we aim to assess the impact of adding additional training data on the model's performance.

4.3 Evaluation

We compare the results of the token-level model trained on RAGBench dataset to the existing RAGBench baselines using AUROC score. As seen in table 4.2, the token-level `lettucedetect-large-v1`$_{\texttt{RAGBench}}$ demonstrates great performance, achieving state-of-the-art results in Customer Support, Legal and in Overall performance. It surpasses, therefore, the previous state-of-the-art model, `Luna`. `Luna` is built on DeBERTa-large and NLI, and is trained on RAGBench dataset to identify supported tokens in a response given a query and retrieved context. In the `Luna` paper, support is defined similarly to the answer faithfulness metric explored in `RAGAS` [EJEAS24]. During inference the `Luna` model treats tokens with low support as hallucinated tokens. For comparison AUROC (Area Under the ROC curve) score is used. It ranges from 0 to 1 and summarizes how well a classifier can distinguish between positive and negative classes. Since AUROC score measures the ability of the model to discriminate between two classes, regardless of class imbalance, it is suitable for evaluating datasets like RAGBench, where the classes are unevenly distributed.

Method	Customer Support	Financial Reasoning	General Knowledge	Legal	Biomed	Overall
GPT-4-turbo annotator	1.0	1.0	1.0	1.0	1.0	1.0
Prompt$_{\text{gpt-3.5-turbo}}$	0.68	0.67	0.67	0.63	0.64	0.66
ChainPoll$_{\text{gpt-3.5-turbo}}$	0.76	0.74	0.75	0.71	0.71	0.74
RAGAS Faithfulness	0.62	0.60	0.60	0.58	0.54	0.61
Luna$_{\text{in-domain}}$	0.76	**0.82**	**0.81**	0.78	**0.83**	0.80
Luna$_{\text{OOD}}$	0.74	0.64	-	0.79	-	-
lettucedetect-base-v1$_{\text{RAGBench}}$	0.85	0.72	0.74	0.93	0.72	0.79
lettucedetect-large-v1$_{\text{RAGBench}}$	**0.87**	0.79	0.76	**0.96**	0.75	**0.84**

Table 4.2: AUROC score on the hallucination detection task on RAGBench dataset. The best AUROC scores are bolded.

Next, we investigate whether adding additional training data improves the performance of the token-level model on RAGTruth 4.3 and RAGBench 4.4 4.5 hallucination prediction task respectively. In table 4.3, we benchmark the model trained only on RAGTruth against the model trained on a combined set of RAGTruth and RAGBench. As seen below 4.3, adding this additional dataset does not bring any benefit to the model performance. This can be attributed to size and complexity of the RAGBench dataset, which might introduce noise during model training.

Method	QUESTION ANSWERING			DATA-TO-TEXT WRITING			SUMMARIZATION			OVERALL		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
lettucedetect-base-v1$_{\text{Truth}}$	60.64	71.25	65.52	89.30	86.53	87.89	53.89	47.55	50.52	76.64	75.50	76.07
lettucedetect-large-v1$_{\text{Truth}}$	65.93	75.00	**70.18**	90.45	86.70	**88.54**	64.04	55.88	**59.69**	80.44	78.05	**79.22**
lettucedetect-base-v2$_{\text{Truth + Bench}}$	64.20	70.63	67.26	90.26	84.80	87.44	57.05	41.67	48.16	79.29	73.06	76.05
lettucedetect-large-v2$_{\text{Truth + Bench}}$	75.94	63.12	68.94	91.57	82.56	86.63	71.93	40.20	51.57	85.96	70.10	77.22

Table 4.3: Example-level results on RAGTruth hallucination prediction task. We compare the performance of the model trained only on RAGTruth to the performance of the model trained on RAGBench and RAGTruth. The best F1-Scores are bolded.

On table 4.4, we compare the model trained only on RAGBench against the model trained on both datasets using Recall, Precision and F1-Score. Unlike the results above, in this setting adding RAGTruth to the RAGBench hallucination detection task is beneficial and leads to an increase of overall F1-Score with 5%.

Method	Customer Support			Financial Reasoning			General Knowledge			Legal			Biomed			Overall		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
lettucedetect-base-v1$_{\text{Bench}}$	37.70	39.66	38.66	20.69	20.29	**20.49**	43.04	37.78	40.24	45.45	38.46	41.67	49.03	45.13	47.00	42.39	39.14	40.70
lettucedetect-large-v1$_{\text{Bench}}$	44.26	46.55	**45.38**	27.59	7.73	12.08	51.18	36.11	42.35	45.45	38.46	41.67	53.08	46.09	49.34	50.54	37.60	43.12
lettucedetect-base-v2$_{\text{Truth + Bench}}$	43.86	43.10	43.48	24.47	11.11	15.28	47.62	38.89	42.81	62.50	38.46	**47.62**	49.75	54.32	**51.93**	47.16	43.09	**45.03**
lettucedetect-large-v2$_{\text{Truth + Bench}}$	42.86	36.21	39.25	20.00	5.31	8.40	53.43	41.11	**46.47**	35.29	46.15	40.00	50.15	46.64	48.33	48.88	38.48	43.06

Table 4.4: Example-level results on RAGBench hallucination prediction task. We compare the performance of the model trained only on RAGBench to the performance of the model trained on RAGBench and RAGTruth. The best F1-Scores are bolded.

Additionally, table 4.5 compares the models in terms of AUROC score. The model trained on both datasets achieves state-of-the-art results on Customer Support, surpassing even lettucedetect-large-v1$_{\text{RAGBench}}$ in this task.

Method	Customer Support	Financial Reasoning	General Knowledge	Legal	Biomed	Overall
lettucedetect-base-v1$_{\text{RAGBench}}$	0.85	0.72	0.74	0.93	0.72	0.79
lettucedetect-large-v1$_{\text{RAGBench}}$	0.87	**0.79**	0.76	**0.96**	**0.75**	**0.84**
lettucedetect-base-v2$_{\text{Truth + Bench}}$	**0.89**	0.72	0.76	0.95	0.74	0.82
lettucedetect-large-v2$_{\text{Truth + Bench}}$	0.86	0.73	**0.78**	0.94	**0.75**	0.83

Table 4.5: Example-level results on RAGBench hallucination prediction task. We compare the performance of the model trained only on RAGBench to the performance of the model trained on RAGBench and RAGTruth. The best AUROC Scores are bolded.

CHAPTER 5

Sentence-Level Hallucination Detection

In this chapter we introduce a new sentence-level hallucination detection model and address, therefore, the third research question 1.2.

5.1 Sentence-Level Model

Inspired by the models introduced in [KSR25], we implemented a custom `SentenceLevelModel` which utilizes `ModernBERT` as encoder backbone with a linear classification head on top. Input sequences are constructed by concatenating context and sentences using special tokens (`[CLS]` for context, `[SEP]` for separation). Sentences are added sequentially, ensuring that the combined input does not exceed the maximum token limit of `4,096`, while also keeping track of sentence boundaries. The embeddings are calculated for each batch only once using `BERT`. Then for each sentence, its embeddings are derived by slicing the corresponding segment from the full sequence embedding and then averaging the token embeddings within the segment limit. These sentence-level representations are then fed into the linear layer to obtain classification logits. The model predicts hallucination probabilities for each sentence in the answer, with sentence-level output generated by considering only those sentences that exceed a `0.5` confidence threshold. Our method can be seen in Figure 5.1.

5.2 Experiments

We begin by training the `SentenceLevelModel` on the RAGBench dataset, which already includes sentence-level annotations. The training is performed with a batch size

of 2 using Adam optimization (learning rate of 1×10^{-6} and a weight decay of 0.01), and runs over 6 epochs on a NVIDIA GeForce RTX 3090. The very first model we trained achieved a rather low F1-Score of 35.25, likely due to the imbalanced nature of the RAGBench dataset, where only about 12% of samples are marked as hallucinated. Therefore, we decided to train a sentence-level model with a weighted loss function, as assigning a higher weight to the hallucinated class can help the model pay more attention to it. In addition, to investigate whether a change in granularity level could improve the performance of the models introduced in [KR25], we also train this new version of the model on the RAGTruth dataset [NWZ+24] with a batch size of 4, while keeping the other training parameters the same as for the other experiments mentioned above. As this dataset provides span-level annotations, we derive sentence-level labels by marking a sentence as hallucinated if it contains any hallucinated spans. Below an example of sentence-level detector is shown, while the results will be discussed in the next section 5.3.

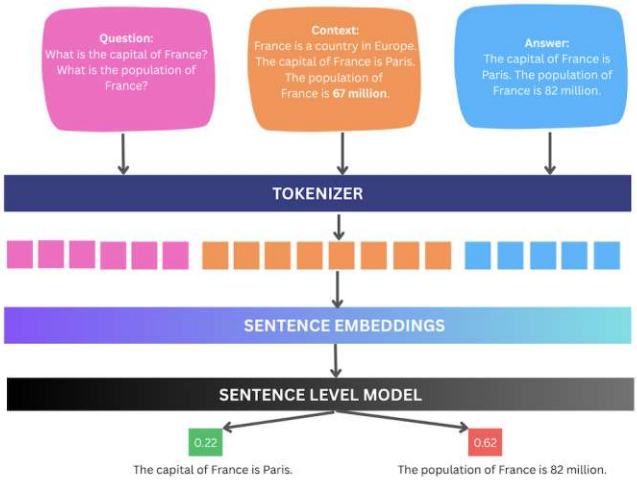

Figure 5.1: Sentence-Level Model Architecture.

```python
detector = make_detector(
    method="sentencetransformer", model_path="./outputs/
    hallucination_detector-sentbaseragtruth", hidden_dim=1024)
contexts = [
    "France is a country in Europe.  The capital of France is Paris. The
    population of France is 67 million.", ]
question = "What is the capital of France? What is the population of France?"
answer = "The capital of France is Paris. The population of France is 82
    million. "
predictions = detector.predict(
    context=contexts, question=question, answer=answer, output_format="spans"
    )
print("Predictions:", predictions)
# --- Output ---
```

```
Predictions: ['The population of France is 82 million.']
```

Code Snippet 5.1: Sentence-Level Hallucination Detection

5.3 Evaluation

The table 5.1 reports the performance of our `SentenceLevelModel(base and large)` on RAGBench hallucination prediction task compared to `LettuceDetect` models, and other baselines. While the token-level model demonstrates superior performance and achieves SOTA results in RAGBench dataset, the `SentenceLevelModel` delivers competitive results in Biomed task, but underperforms on the remaining tasks. A possible reason for this could be that the model is trained on embeddings calculated once for batch, rather than on sentence-by-sentence embeddings, which might provide a better representation of each sentence. The used approach can lead to diluted information, especially for long and complex sentences, to be fed to the model. RAGBench is a large dataset comprising a number of difficult tasks, and this might also add an additional layer of complexity.

Method	Customer Support	Financial Reasoning	General Knowledge	Legal	Biomed	Overall
GPT-4-turbo annotator	1.0	1.0	1.0	1.0	1.0	1.0
$\text{Prompt}_{\text{gpt-3.5-turbo}}$	0.68	0.67	0.67	0.63	0.64	0.66
$\text{ChainPoll}_{\text{gpt-3.5-turbo}}$	0.76	0.74	0.75	0.71	0.71	0.74
RAGAS Faithfulness	0.62	0.60	0.60	0.58	0.54	0.61
$\text{Luna}_{\text{in-domain}}$	0.76	**0.82**	**0.81**	0.78	**0.83**	0.80
Luna_{OOD}	0.74	0.64	-	0.79	-	-
$\underline{\text{lettucedetect-base-v1}_{\text{RAGBench}}}$	0.85	0.72	0.74	0.93	0.72	0.79
$\underline{\text{lettucedetect-large-v1}_{\text{RAGBench}}}$	0.87	0.79	0.76	**0.96**	0.75	**0.84**
$\underline{\text{lettucedetect-base-v2}_{\text{Truth + Bench}}}$	**0.89**	0.72	0.76	0.95	0.74	0.82
$\underline{\text{sentence}_{\text{ModernBERTBase}}}$	0.53	0.53	0.63	0.53	0.65	0.64
$\underline{\text{sentence}_{\text{ModernBERTLarge}}}$	0.54	0.61	0.66	0.60	0.70	0.70

Table 5.1: AUROC on the hallucination detection task on RAGBench dataset.

The next table 5.2 shows the performance of the `SentenceLevelModel` on RAGTruth hallucination prediction task, compared to `LettuceDetect-v1` [KR25], and other baselines. The `SentenceLevelModel` performs reasonably well on the Data-To-Text Writing task but it falls behind `LettuceDetect-v1` on the other tasks. This might be due to the potential labeling issue the `SentenceLevelModel` might be prone to. Specifically, since RAGTruth offers span-level annotations, the model relies on derived sentence-labels, where a sentence is marked as hallucinated if it contains any hallucinated span, be this just a short one. As a result, mostly accurate sentences can be labeled as hallucinated and sentence representations, calculated by averaging the token embeddings, can in contrast be dominated by correct tokens. This might, therefore, confuse the model and cause it to struggle with learning the task.

Method	QUESTION ANSWERING			DATA-TO-TEXT WRITING			SUMMARIZATION			OVERALL		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
$\text{Prompt}_{\text{gpt-4-turbo}}$	33.2	90.6	45.6	64.3	100.0	78.3	31.5	97.6	47.6	46.9	97.9	63.4
$\text{SelCheckGPT}_{\text{gpt-3.5-turbo}}$	35.0	58.0	43.7	68.2	82.8	74.8	31.1	56.5	40.1	49.7	71.9	58.8
Finetuned Llama-2-13B	61.6	76.3	68.2	85.4	91.0	88.1	64.0	54.9	59.1	76.9	80.7	78.7
RAG-HAT	76.5	73.1	**74.8**	92.9	90.3	**91.6**	77.7	59.8	**67.6**	87.3	80.8	**83.9**
RAGAS Faithfulness	31.2	41.9	35.7	79.2	50.8	61.9	64.2	29.9	40.8	62.0	44.8	52.0
Luna	37.8	80.0	51.3	64.9	91.2	75.9	40.0	76.5	52.5	52.7	86.1	65.4
lettucedetect-base-v1	60.64	71.25	65.52	89.30	86.53	87.89	53.89	47.55	50.52	76.64	75.50	76.07
lettucedetect-large-v1	65.93	75.00	70.18	90.45	86.70	88.54	64.04	55.88	59.69	80.44	78.05	79.22
$\text{SentenceLevelModel}_{\text{ModernBERTBase}}$	67.50	33.75	45.00	87.57	80.31	83.78	60.91	32.84	42.68	81.28	62.14	70.43
$\text{SentenceLevelModel}_{\text{ModernBERTLarge}}$	57.38	43.75	49.65	85.64	87.56	86.59	52.94	39.71	45.38	75.89	69.78	72.71

Table 5.2: Example level results on RAGTruth hallucination prediction task.

.

CHAPTER 6

Discussion

6.1 Key Findings

Hallucination detection methods have mainly been focused on English language text, primarily due to lack of robust evaluation datasets in other languages, whose creation can be quite costly and time-consuming. In this study, we aim to address this gap by implementing a model capable of detecting hallucinations in other languages, with a focus in German. To achieve this, we leverage the great capabilities of LLMs for translation and translate a benchmark dataset into German. The synthetic dataset is then used for training the token-level hallucination detection model introduced in [KR25]. Our experiments show that using LLMs for translating annotated datasets, when ensuring that the annotations are preserved, can indeed be quite effective. Our model achieves superior performance when compared to our two baselines, `GPT-4-mini` and `RAGAS`, with improvements of up to `+14` F1-score points in example-level evaluation and `+28` F1-score points in span-level evaluation.
Additionally, we train the token-level model [KR25], `LettuceDetect`, on the new dataset, RAGBench. This allows us to benchmark the models introduced in [KR25] on RAGBench against state-of-the-art methods, such as `Luna` [BFSS24]. Our results show that `LettuceDetect` outperforms `Luna` in Customer Support, Legal, and in Overall performance, achieving therefore state-of-the-art results in RAGBench dataset. Moreover, we investigate whether adding additional training data brings any improvement to the performance of the `LettuceDetect` model [KR25]. Specifically, we compare the performance of the model trained on both RAGBench and RAGTruth versus when it is trained on RAGBench and RAGTruth individually, evaluating them on each dataset separately. We notice that adding RAGTruth leads to improvement in the RAGBench hallucination detection task, such as increase in F1-Score by 5% and SOTA results in Customer Support task in terms of AUROC score. On the other side, adding RAGBench brings no benefit. One possible explanation is that RAGBench is a large and complex

dataset which might introduce significant noise into the training process.
Finally, this work introduces a sentence-level hallucination detection model, `SentenceLevelModel`. Our goal was to investigate whether a new level of granularity for hallucination detection can improve the performance of the models introduced in [KR25]. Evaluated on both datasets separately, the token-level models in [KR25], outperform the new sentence-level hallucination detection model introduced in this work. The reason for that could be that the token-level model is advantaged by a more granular level of information when being trained.

6.2 Limitations

Despite the contributions of this research, a couple limitations are acknowledged, that represent opportunities for future enhancements. First, our German model is trained only on the synthetic translated version of RAGTruth dataset, which includes samples from only three tasks: QA, Summarization and Data2Text. Training our model on other data including additional tasks, such as for example a German version of a RAGBench dataset, could potentially enhance the generalization capabilities of our German model. Additionally, in the `SentenceLevelModel` we compute the embeddings only once per batch and not in a sentence-by-sentence basis. While this approach is more efficient, it might result in lower accuracy. Furthermore, the `SentenceLevelModel` relies on derived setence-level labels when the dataset does not provide them, for example RAGTruth. In such cases, sentences are marked as hallucinated if they contain any hallucinated span. As the hallucination spans can also be short ones, we risk marking as hallucinated also those sentences that are composed of mostly accurate data, which might confuse our model during the training process.

6.3 Future Work

Future work on this thesis can focus on expanding the training data and training our German models on a translated version of the RAGBench dataset. Moreover, to overcome the problems identified in the `SentenceLevelModel`, we could explore building embeddings for each sentence separately for better sentence representations and define a threshold span length, beyond which a sentence containing hallucinated span is to be marked as hallucinated. Last but not least, a possible future advancement could be to create a unified multilingual hallucination detection model. At the moment, our implementation can be used to create hallucination detection models in multiple languages such as French, Chinese, Spanish, etc. However, a valuable extension would be to create one model capable of handling the detection of hallucinations across multiple languages, for example both English and German.

CHAPTER 7

Conclusion

To address RQ1, we use a LLM `GPT-4-mini` to translate into German language RAGTruth, a benchmark dataset used for hallucination detection task in RAG setting. We use the models introduced in [KR25] and train them on the synthetic translated dataset. Since no baselines exist for the German RAGTruth data, we implement two of them, `GPT` prompting and `RAGAS`. Our German model outperforms both baselines, achieving an `23.5%` improvement of F1-Score on example-level, and more than doubling performance on span-level evaluation task, relative to `GPT` prompting, which is our best performing baseline.
To answer RQ2, first we introduce a new dataset, RAGBench, to the `LettuceDetect` model [KR25]. When trained on RAGBench, `LettuceDetect` achieves state-of-the-art results in Customer Support, Legal and Overall performance, surpassing the previous state-of-the-art method, `Luna`. Next, we compare the performance of `LettuceDetect` across three different settings: models trained on RAGBench and RAGTruth individually and a model trained on a combined set of both RAGBench and RAGTruth. Our results show that adding more data does not always lead to an improvement in performance, as observed when adding RAGBench to the RAGTruth detection tast. On the other side, adding RAGTruth improves the performance on RAGBench detection task in terms of overall F1-score with `5%` and leads to state-of-the-art results in Customer Support task in terms of AUROC score.
Regarding RQ3, we implement a sentence-level hallucination detection model. Unlike the original models trained on token-level basis, the new one is trained on sentence-level labels. While the results of the sentence-level model are not bad, it falls short compared to the original models, experiencing a `16.6%` decrease in AUROC score on the RAGBench detection task, and an `8.2%` drop in F1-score on RAGTruth evaluation.
Building upon [KR25], this work extends the models introduced by adding support for a new language, German, incorporating a new dataset, RAGBench, and implementing a new detection granularity-level, sentence-level model. The main contribution of this thesis

lies in addressing the lack of multilingual hallucination detection models. It leverages a LLM to translate a hallucination detection specific dataset, significantly reducing the cost and time required to create such datasets manually. While this study focuses on German, our implementation can be used to create hallucination detection models in other languages as well.

CHAPTER 8

Overview of Generative AI Tools Used

As declared in the statement of originality, the use of generative AI tools for this thesis was limited to an auxiliary role, with my creative influence predominating. In particular, I have used `ChatGPT` as linguistic aid for my work to rephrase my self-composed sentences in a more formal way which improves the flow and clarity.

List of Figures

3.1 LettuceDetect Model Architecture [KR25]. 16

5.1 Sentence-Level Model Architecture. 28

List of Tables

3.1 An example of RAGTruth data, including question, references, response, annotations, and merged annotations 14
3.2 Example-level results on the full test set. The best F1-Scores are bolded. 20
3.3 Span-level results on the full test set. The best F1-Scores are bolded. . . . 20
3.4 Example-level results on the original and manually validated translations. The best F1-Scores are bolded. 21
3.5 Span-level results on the original and manually validated translations. The best F1-Scores are bolded. 21
3.6 Example-level results on the manually validated samples. The best F1-Scores are bolded. 22
3.7 Span-level results on the manually validated samples. The best F1-Scores are bolded. 22

4.1 An example of RAGBench data, including question, references, response and annotations 24
4.2 AUROC score on the hallucination detection task on RAGBench dataset. The best AUROC scores are bolded. 25
4.3 Example-level results on RAGTruth hallucination prediction task. We compare the performance of the model trained only on RAGTruth to the performance of the model trained on RAGBench and RAGTruth. The best F1-Scores are bolded. 25
4.4 Example-level results on RAGBench hallucination prediction task. We compare the performance of the model trained only on RAGBench to the performance of the model trained on RAGBench and RAGTruth. The best F1-Scores are bolded. 26
4.5 Example-level results on RAGBench hallucination prediction task. We compare the performance of the model trained only on RAGBench to the performance of the model trained on RAGBench and RAGTruth. The best AUROC Scores are bolded. 26

5.1 AUROC on the hallucination detection task on RAGBench dataset. . . . 29
5.2 Example level results on RAGTruth hallucination prediction task. 30

Prompts

1 Answer Translation (All Tasks) 15
2 Prompt Translation (Summarization & QA Tasks) 15
3 Prompt Translation (Data2Text) 16
4 LLM Prompt for Hallucination Detection 18
5 Few Shot block used in LLM Prompting 19

List of Code Snippets

3.1 Transformer-based Hallucination Detection (German) 17
3.2 LLM-based Hallucination Detection 19
5.1 Sentence-Level Hallucination Detection 28

Bibliography

[BCC+18] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. Ms marco: A human generated machine reading comprehension dataset, 2018.

[BDVJ03] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3(null):1137–1155, March 2003.

[BFSS24] Masha Belyi, Robert Friel, Shuai Shao, and Atindriyo Sanyal. Luna: An evaluation foundation model to catch language model hallucinations with high accuracy and low cost, 2024.

[BGBA+25] Nicolas Boizard, Hippolyte Gisserot-Boukhlef, Duarte M. Alves, André Martins, Ayoub Hammal, Caio Corro, Céline Hudelot, Emmanuel Malherbe, Etienne Malaboeuf, Fanny Jourdan, Gabriel Hautreux, João Alves, Kevin El-Haddad, Manuel Faysse, Maxime Peyrard, Nuno M. Guerreiro, Patrick Fernandes, Ricardo Rei, and Pierre Colombo. Eurobert: Scaling multilingual encoders for european languages, 2025.

[BKT+24] Scott Barnett, Stefanus Kurniawan, Srikanth Thudumu, Zach Brannelly, and Mohamed Abdelrazek. Seven failure points when engineering a retrieval augmented generation system, 2024.

[BMR+20] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[BVJS15] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks, 2015.

[CCD+20] Vittorio Castelli, Rishav Chakravarti, Saswati Dana, Anthony Ferritto, Radu Florian, Martin Franz, Dinesh Garg, Dinesh Khandelwal, Scott McCarley, Michael McCawley, Mohamed Nasr, Lin Pan, Cezar Pendus, John Pitrelli, Saurabh Pujar, Salim Roukos, Andrzej Sakrajda, Avi Sil, Rosario Uceda-Sosa, Todd Ward, and Rong Zhang. The TechQA dataset. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1269–1278, Online, July 2020. Association for Computational Linguistics.

[CCS+21] Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. FinQA: A dataset of numerical reasoning over financial data. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[CKG+20] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July 2020. Association for Computational Linguistics.

[CND+22] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022.

[DCLT19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[EJEAS24] Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. RAGAs: Automated evaluation of retrieval augmented generation. In Nikolaos Aletras and Orphee De Clercq, editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158, St. Julians, Malta, March 2024. Association for Computational Linguistics.

[GDJea24] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, and et al. The llama 3 herd of models, 2024.

[Goo25] Google. Introducing gemma 3: The most capable model you can run on a single gpu or tpu. `https://blog.google/technology/developers/gemma-3/`, 2025. Accessed: 2025-06-24.

[GXG+24] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey, 2024.

[HBCB21] Dan Hendrycks, Collin Burns, Anya Chen, and Spencer Ball. Cuad: An expert-annotated nlp dataset for legal contract review, 2021.

[HLGC21] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention, 2021.

[HYM+25] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55, January 2025.

[JDL+19] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. PubMedQA: A dataset for biomedical research question answering. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *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 2567–2577, Hong Kong, China, November 2019. Association for Computational Linguistics.

[JLF+23] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, March 2023.

[JSM+23] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

[JX19] Kun Jing and Jungang Xu. A survey on neural network language models, 2019.

[KJZ+23] Ehsan Kamalloo, Aref Jafari, Xinyu Zhang, Nandan Thakur, and Jimmy Lin. Hagrid: A human-llm collaborative dataset for generative information-seeking with attribution, 2023.

[KR25] Adam Kovacs and Gabor Recski. Lettucedetect: A hallucination detection framework for rag applications, 2025.

[KSR25] Adam Kovacs, Paul Schmitt, and Gabor Recski. KR labs at ArchEHR-QA 2025: A verbatim approach for evidence-based question answering. In Sarvesh Soni and Dina Demner-Fushman, editors, *Proceedings of the 24th Workshop on Biomedical Language Processing (Shared Tasks)*, pages 69–74, Vienna, Austria, August 2025. Association for Computational Linguistics.

[LCR+24] Junyi Li, Jie Chen, Ruiyang Ren, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. The dawn after the dark: An empirical study on factuality hallucination in large language models, 2024.

[LH19] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

[LLH+23] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts, 2023.

[MCR24] Laura Mascarell, Ribin Chalumattu, and Annette Rios. German also hallucinates! inconsistency detection in news summaries with the absinth dataset. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 7696–7706, Torino, Italia, May 2024. ELRA and ICCL.

[MLG23] Potsawee Manakul, Adian Liusie, and Mark Gales. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017, Singapore, December 2023. Association for Computational Linguistics.

[MMN+25] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey, 2025.

[MRJP20] Timo Möller, Anthony Reina, Raghavan Jayakumar, and Malte Pietsch. COVID-QA: A question answering dataset for COVID-19. In Karin Verspoor, Kevin Bretonnel Cohen, Mark Dredze, Emilio Ferrara, Jonathan May, Robert Munro, Cecile Paris, and Byron Wallace, editors, *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*, Online, July 2020. Association for Computational Linguistics.

[MSC+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality, 2013.

[MT25] Andrea Matarazzo and Riccardo Torlone. A survey on large language models with some insights on their capabilities and limitations, 2025.

[NBGBA+25] Nicolas-BZRD, Hippolyte Gisserot-Boukhlef, Duarte M. Alves, Manuel Faysse, and EuroBERT team. Introducing eurobert: A high-performance multilingual encoder model. `https://huggingface.co/blog/EuroBERT/release`, March 2025. Hugging Face Blog. Accessed: 2025-08-29.

[NSM+21] Abhilash Nandy, Soumya Sharma, Shubham Maddhashiya, Kapil Sachdeva, Pawan Goyal, and NIloy Ganguly. Question answering over electronic devices: A new benchmark dataset and a multi-task learning based QA framework. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4600–4609, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[NWZ+24] Cheng Niu, Yuanhao Wu, Juno Zhu, Siliang Xu, KaShun Shum, Randy Zhong, Juntong Song, and Tong Zhang. RAGTruth: A hallucination corpus for developing trustworthy retrieval-augmented language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10862–10878, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[OAA+24] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya

Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024.

[SFKPZ24] Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. Ares: An automated evaluation framework for retrieval-augmented generation systems, 2024.

[SHL+23] Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Scott Wen tau Yih. Trusting your evidence: Hallucinate less with context-aware decoding, 2023.

[SLP+23] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023.

[STK+25] Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R. Johnston, Shauna Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. Towards understanding sycophancy in language models, 2025.

[SWZ+24] Juntong Song, Xingguang Wang, Juno Zhu, Yuanhao Wu, Xuxin Cheng, Randy Zhong, and Cheng Niu. RAG-HAT: A hallucination-aware tuning pipeline for LLM in retrieval-augmented generation. In Franck Dernoncourt, Daniel Preoţiuc-Pietro, and Anastasia Shimorina, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1548–1558, Miami, Florida, US, November 2024. Association for Computational Linguistics.

[SZL+23] Mobashir Sadat, Zhengyu Zhou, Lukas Lange, Jun Araki, Arsalan Gundroo, Bingqing Wang, Rakesh R Menon, Md Rizwan Parvez, and Zhe Feng. Delucionqa: Detecting hallucinations in domain-specific question answering, 2023.

[TLI+23] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[VSP+23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[WCC+24a] Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, et al. Finally, a replacement for bert. Hugging Face blog, December 2024. Accessed: 2025-08-29.

[WCC+24b] Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference, 2024.

[WGY+24] Di Wu, Jia-Chen Gu, Fan Yin, Nanyun Peng, and Kai-Wei Chang. Synchronous faithfulness monitoring for trustworthy retrieval-augmented generation, 2024.

[WLW+23] Haifeng Wang, Jiwei Li, Hua Wu, Eduard Hovy, and Yu Sun. Pre-trained language models and their applications. *Engineering*, 25:51–65, 2023.

[Yel25] Yelp, Inc. Yelp open dataset. `https://business.yelp.com/data/resources/open-dataset/`, 2025. Accessed: 2025-06-20.

[YQZ+18] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[ZCL+24] Zhenyu Zhang, Runjin Chen, Shiwei Liu, Zhewei Yao, Olatunji Ruwase, Beidi Chen, Xiaoxia Wu, and Zhangyang Wang. Found in the middle:

How language models use long contexts better via plug-and-play positional encoding, 2024.

[ZLH+21] Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3277–3287, Online, August 2021. Association for Computational Linguistics.

[ZPM+23] Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A. Smith. How language model hallucinations can snowball, 2023.

[ZZL+24] Xin Zhang, Yanzhao Zhang, Dingkun Long, Wen Xie, Ziqi Dai, Jialong Tang, Huan Lin, Baosong Yang, Pengjun Xie, Fei Huang, Meishan Zhang, Wenjie Li, and Min Zhang. mGTE: Generalized long-context text representation and reranking models for multilingual text retrieval. In Franck Dernoncourt, Daniel Preoţiuc-Pietro, and Anastasia Shimorina, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1393–1412, Miami, Florida, US, November 2024. Association for Computational Linguistics.

[ZZL+25] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A survey of large language models, 2025.