ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DEPARTMENT OF COMPUTER SCIENCE
AND ENGINEERING

ARTIFICIAL INTELLIGENCE

MASTER THESIS

in

Natural Language Processing

VERIFAI: TOWARDS AN OPEN-SOURCE SCIENTIFIC GENERATIVE QUESTION-ANSWERING SYSTEM WITH REFERENCED AND VERIFIABLE ANSWERS

CANDIDATE	SUPERVISOR
Lorenzo Cassano	Prof. Paolo Torroni
	CO-SUPERVISOR
	Dott. Nikola Milosevic

Academic year 2023-2024

Session 2st

To those who believed in me ...

Abstract

This research investigates the effectiveness of transformer-based models in mitigating hallucinations within the biomedical domain, a crucial area in natural language processing (NLP). Hallucinations occur when language models generate unsupported or divergent information. Despite their capabilities, large language models (LLMs) are prone to such errors, impacting critical sectors like biomedicine. The study has two main objectives: exploring methods like Retrieval-Augmented Generation (RAG) and Retrieval-Augmented Fine-Tuning (RAFT) to reduce hallucinations, and developing techniques for detecting persistent hallucinations. These efforts aim to limit and identify hallucinations in transformer-based models. Additionally, the research introduces a biomedical RAG system to enhance response reliability, using fine-tuned LLMs with PubMed abstracts. This system outperforms the PubMed search engine and GPT-4 Turbo in referencing relevant abstracts. The study also presents a Verification Engine for an open-source scientific QA system, using models fine-tuned on the SciFact dataset. The DeBERTa model achieved an F1 score of 88%, outperforming other models on the HealthVer dataset. These findings advance NLP techniques, particularly in biomedicine, by improving the accuracy and reliability of transformer-based models.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

This project is the result of my internship at Bayer Pharmaceutical Company. It is a collaborative effort between Bayer and the Institute of Artificial Intelligence of Serbia. The primary goal of this project is to develop a system capable of providing accurate answers to complex biomedical and medical questions, with references to the biomedical documents used.

One of the significant challenges in using transformer-based models is the occurrence of hallucinations, where models generate information that is not supported by input data or diverges significantly from intended meanings. To address this issue, the project includes the creation of a verification component designed to check if each sentence in the generated response is supported by the cited documents.

The project aims to improve natural language processing and help make better decisions, especially in important areas like medicine. By providing accurate and reliable information, it helps professionals make well-informed choices, leading to better healthcare results. Since VerifAi is open-source, it can be used and improved by researchers everywhere, making it a trustworthy tool for ongoing scientific studies.

1.2 Thesis Structure

This thesis is organized into 6 chapters and appendix:

- **In Chapter 2 – Background:** this chapter introduces the key concepts, methodologies, and terminologies essential for understanding the thesis.
- **In Chapter 3 – Related Work:** this section reviews the current state of research, identifies gaps, and provides a cohesive context for the subsequent chapters.
- **In Chapter 4 – Methodology:** this chapter describes the research methods, including the main task, research questions, hypotheses, and reference baselines.
- **In Chapter 5 – Evaluation:** this chapter assesses the different components of the system.
- **In Chapter 6 – Conclusion and Future Work:** this chapter summarizes the key findings and contributions of the study, and outlines potential directions for future research and improvements.

Chapter 2

Background

This chapter serves as a foundational guide, providing the essential information necessary to comprehend the entirety of this thesis. It lays the groundwork by introducing key concepts, methodologies, and terminologies that are pivotal to understanding the subsequent chapters. By establishing a solid conceptual framework, this chapter ensures that readers are well-equipped to follow the detailed discussions and analyses that follow.

Moreover, the content of this chapter is designed to bridge any knowledge gaps and offer a comprehensive overview of the primary subjects addressed in the thesis. It systematically covers the background, theoretical underpinnings, and relevant literature, thereby creating a cohesive context for the research presented. With this foundational knowledge, readers will be better positioned to appreciate the nuances and significance of the findings and contributions made in the later chapters.

2.1 Generative Model

2.1.1 Transformers

Transformers have revolutionized the field of natural language processing (NLP) since their introduction by [1]. The Transformer model introduced the concept of self-attention mechanisms, allowing the model to weigh the importance of different words in a sentence relative to each other. This architecture enables the parallelization of training processes, which significantly reduces training time compared to previous sequential models like Recurrent Neural Networks (RNNs) [2] and Long Short-Term Memory networks (LSTMs) [3]. The core innovation of the Transformer is the self-attention mechanism, which helps capture long-range dependencies in text more effectively. This has led to the development of various Transformer-based models, such as BERT (Bidirectional Encoder Representations from Transformers) [4] and GPT (Generative Pre-trained Transformer) [5].

Architecture

Transformer architecture is the process of converting a sequence of discrete tokens into vector embeddings. These embeddings are then processed by a series of encoder blocks, which produce encodings that highlight the relationships between different parts of the input. The output from the encoder is subsequently fed into a series of decoder blocks, which generate the final output in an autoregressive manner.

Attention Mechanism

The self-attention mechanism calculates the importance of each token in a sequence with respect to other words. Given an input sequence of tokens $X = \{x_1, x_2, \ldots, x_n\}$, self-attention computes a set of query (Q), key (K), and value (V) vectors for each element in the sequence. These vectors are linear transformations of the input sequence, learned during the training process.

The attention score between a query vector q_i and a key vector k_j is calculated using a dot product re-scaled by the root of the head dimension:

$$\text{score}(q_i, k_j) = \frac{q_i \cdot k_j}{\sqrt{d_k}} \tag{2.1}$$

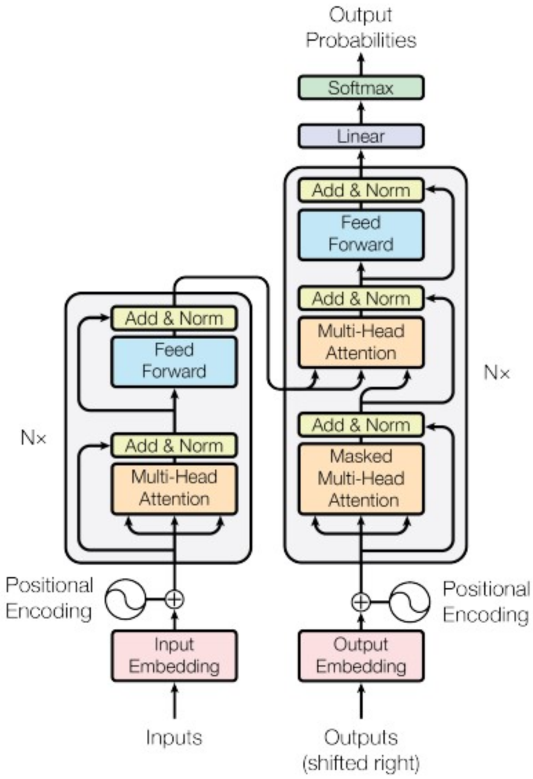

Figure 2.1: Transformer Architecture

The attention weights (A) are obtained by applying a softmax function to the attention scores, normalizing them to represent the importance of each word:

$$A_{ij} = \text{softmax}(\text{score}(q_i, k_j)) \tag{2.2}$$

The output of self-attention is then computed as a weighted sum of the value vectors:

$$\text{SelfAtt}(X) = \sum_{j=1}^{n} A_{ij} \cdot v_j \tag{2.3}$$

Multi head self-attention

In a multi-head self-attention mechanism with h heads, given an input sequence X, each head i computes its own set of query, key, and value vectors as follows:

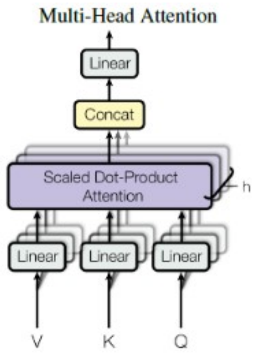

Figure 2.2: Multi head self attention schema

$$\mathbf{q}_i = \mathbf{X} \cdot \mathbf{W}_i^Q, \quad \mathbf{k}_i = \mathbf{X} \cdot \mathbf{W}_i^K, \quad \mathbf{v}_i = \mathbf{X} \cdot \mathbf{W}_i^V$$

where $\mathbf{W}_i^Q$, $\mathbf{W}_i^K$, and $\mathbf{W}_i^V$ are the learnable weight matrices specific to the i-th head. Each head then computes the attention scores, attention weights, and the attended output using the self-attention mechanism described earlier. Finally, the outputs of all heads are concatenated and linearly transformed to obtain the final multi-head self-attention representation:

$$\text{MultiHead}(\mathbf{X}) = \text{Concat}(\text{head}_1, \text{head}_2, \dots, \text{head}_h) \cdot \mathbf{W}^O$$

where $\text{head}_i = \text{SelfAtt}(\mathbf{q}_i, \mathbf{k}_i, \mathbf{v}_i)$ is the output of the i-th head, $\mathbf{W}^O$ is the learnable weight matrix for the output transformation, and Concat denotes the concatenation operation. The idea behind multiple heads is that we are able to learn a diverse independent set of features for each token.

2.2 Large Language Models

Large Language Models (LLMs) have emerged as a powerful application of Transformer architectures. These models, including GPT-3 [6] and T5 (Text-To-Text Transfer Transformer) [7], are trained on vast amounts of data and contain billions of parameters. Their large scale allows them to generate coherent and contextually relevant text across a wide range of tasks, from translation and summarization to question answering and text generation.

Architecture and Training

LLMs are trained using unsupervised learning on a diverse corpus of text. The training objective is typically to predict the next word in a sentence, which requires the model to capture syntactic and semantic relationships in the data. The size of the training data and the number of parameters in the model are key factors that contribute to the performance of LLMs. For instance, GPT-3 has 175 billion parameters and was trained on hundreds of gigabytes of text data [6].

Capabilities and Applications

One of the most remarkable capabilities of LLMs is their ability to perform tasks with little to no task-specific training data, known as few-shot, one-shot, and zero-shot learning. In few-shot learning, the model is given a few examples of the task it needs to perform. In one-shot learning, it is given only one example, and in zero-shot learning, it receives no examples at all [6]. GPT-3 and GPT-4, in particular, has demonstrated outstanding performance in these scenarios, making it a versatile tool for various natural language processing (NLP) tasks.

The applications of LLMs are broad and varied. They can be used for machine translation, where they convert text from one language to another, and for text summarization, where they condense long documents into shorter summaries while preserving key information. Additionally, LLMs are employed in question answering systems, which can understand and respond to queries, and in text generation, where they produce human-like text for applications such as chatbots, content creation, and storytelling [6, 7].

Prompting and Few-Shot Learning

Prompting and few-shot learning are critical capabilities of Large Language Models (LLMs) that have significantly advanced the field of natural language processing. These techniques enable LLMs to perform a wide range of tasks with minimal task-specific training data, which is particularly useful in scenarios where annotated data is scarce or expensive to obtain [8]. This approach leverages the pre-trained knowledge of the model to adapt to new tasks dynamically. For example, when given a prompt like ”Translate the following English sentence to French: ’How are you?’”, the model can produce the correct translation based on its understanding of both languages [6]. The method of prompting modifies the original input using a template into a textual string with unfilled slots that the model probabilistically fills, effectively enabling the model to perform new tasks with few or no labeled data. This paradigm, known as ”pre-train, prompt, and predict,” allows the model to be pre-trained on massive amounts of raw text, making it versatile in adapting to new scenarios [9]. This framework has proven powerful for various applications, such as translation and text classification, and underscores the importance of prompt engineering to improve task performance and model robustness [9]. We have various prompting techniques, one of which is prompt augmentation, which involves automatically generating and optimizing prompts for large language models (LLMs) to enhance their reasoning capabilities. The Automate-CoT (Automatic Prompt Augmentation and Selection with Chain-of-Thought) strategy improves performance in various reasoning tasks by creating and selecting optimal rationale chains from a small labeled dataset. This method bypasses the need for manually designed rationales, which are labor-intensive and sensitive to order, complexity, diversity, and style. Automate-CoT generates multiple pseudo-chains, prunes low-quality ones, and uses a variance-reduced policy gradient strategy to select the most effective chains. This approach has demonstrated significant performance improvements across different types of reasoning tasks, showing its adaptability and efficiency in generating high-quality prompts automatically [10].

2.2.1 Language Model Fine-Tuning

After retraining a language model, it is crucial to ensure it aligns with the user's specific goals. Although a retrained language model benefits from the extensive knowledge and skills acquired during pretraining, it may not naturally understand or meet the user's intended context or objectives. Therefore, a process known as ”alignment” is necessary, where the model is fine-tuned or adapted to better fit the desired tasks or behaviors. The challenge of alignment stems from the general nature of language models, which are pretrained on large and diverse datasets. This broad training allows them to generate coherent text across various contexts, but it also means their responses may not always align with specific user goals, potentially resulting in irrelevant, biased, or even harmful outputs.

Benefits

An aligned language model offers numerous advantages, such as:

- **Enhanced Task Performance:** Alignment allows the model to perform specific tasks more effectively, providing accurate and contextually appropriate results.
- **Minimized Bias and Harm:** Alignment helps in reducing biases and prevents the generation of harmful or inappropriate content.
- **Improved User Interaction:** Aligned models produce responses that align with user intent, resulting in more efficient and engaging interactions.

2.2.2 Approaches

There are multiple methods to achieve alignment:

Task-Specific Fine-Tuning

One method is to fine-tune the language model for a specific task. By training the model with task-specific tokens and adjusting its parameters, it can be guided to produce more relevant and contextually accurate responses for the given task [5].

Instruction-Based Fine-Tuning

While standard task-specific fine-tuning enhances the model's performance on a particular task, it might restrict its flexibility across various domains. Instruction-based fine-tuning involves training the model on a diverse set of instructional data. This allows the model to learn to follow general instructions in input, making it adaptable to a wide range of tasks. This approach was initially explored in the FLAN model [11], where a decoder-only transformer language model was trained across several tasks such as natural language inference, commonsense reasoning, reading comprehension, sentiment analysis, and others.

Reinforcement Learning from Human Feedback

Reinforcement Learning from Human Feedback (RLHF) [12] is a technique that aligns language models with user intent by incorporating human feedback into the training process. This approach consists of four main stages: instruction fine-tuning, gathering human feedback, training the reward model, and applying reinforcement learning.

1. **Instruction Fine-Tuning:** Initially, the language model undergoes fine-tuning as previously described. This step helps the model produce better-aligned responses to user prompts, establishing a solid baseline for the subsequent steps.

2. **Gathering Human Feedback:** Following the initial fine-tuning, human feedback is collected to train a reward model. This feedback typically involves humans ranking several responses generated by the model for a single prompt, helping the model identify which responses align best with human preferences.

3. **Training the Reward Model:** The gathered human feedback is used to train a reward model, which assigns a scalar value (reward) to each prompt-response pair.

4. **Reinforcement Learning:** The language model is further fine-tuned using reinforcement learning to maximize the reward assigned by the reward model. The objective function for the language model, based on the reward function, can be formulated as follows:

$$\max_{\theta} \mathbb{E}_{(x,y)\sim D}\left[R(x, y; \theta)\right]$$

   where θ represents the model parameters, D is the dataset of prompt-response pairs, x is the input prompt, y is the model-generated response, and $R(x, y; \theta)$ is the reward model that evaluates the quality of the response y given the input prompt x and the model parameters θ.

2.3 Efficient Training

The size of language models has been shown to improve performance, resulting in the development of larger models. This leads to high resource requirements not only for training language models but also for fine-tuning them. Several strategies have been studied to improve upon this and make fine-tuning more efficient.

2.3.1 LoRA

LoRA, or Low-Rank Adaptation [13], is a method proposed to efficiently fine-tune neural networks for specific tasks and datasets while reducing the number of trainable parameters. LoRA freezes the pre-trained model weights and concatenates new factorized low-rank matrices next to some pretrained matrices of a given neural network.

As shown in Figure 2.3, the same input is passed to the adapter, and its output is then summed over the output of the pretrained matrix.

In LoRA, given a matrix $W \in \mathbb{R}^{n \times n}$ that we want to train, it was speculated that the update on the given matrix $\Delta W \in \mathbb{R}^{n \times n}$ resulting from the training process is not necessarily full rank when the model is already pretrained. So the matrix W could potentially be represented in a factorized way with two matrices $A \in \mathbb{R}^{n \times r}$ and $B \in \mathbb{R}^{r \times n}$ such that:

$$Wx + \Delta Wx = Wx + (A + B)x$$

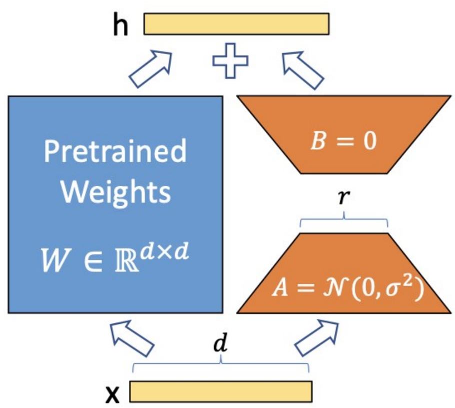

Figure 2.3: LoRA reparametrization, just A and B matrices are trained.

This kind of strategy offers several advantages:

- **Reduced storage requirements:** LoRA allows storing only the factorized matrices, which can be integrated into the pretrained model when necessary.
- **Faster training and lower compute requirements:** LoRA speeds up the training process and requires less memory compared to traditional full fine-tuning.

Furthermore, LoRA has been shown to match or exceed the performance of full fine-tuning in terms of model quality, despite having fewer trainable parameters and increased training efficiency.

2.3.2 QLoRA

QLoRA [14] is a fine-tuning technique that combines LoRA and quantization to further reduce memory usage and accelerate inference. The core idea is that, during the updating of the adapter with LoRA, the frozen weights remain untrained and can thus be quantized without compromising inference accuracy.

4-bit NormalFloat

The quantization method recommended in the original QLoRA paper leverages the fact that hidden activations in neural networks approximately follow a normal distribution, allowing for more efficient quantization. This method is based on quantile quantization, initially introduced in [15]. For k-bit quantization, the quantized outputs are distributed evenly among the 2^k different bit representations.

Quantile quantization estimates the quantiles of the input tensor by calculating the empirical distribution function. Normally, computing the quantiles would require the cumulative distribution function of the input tensor distribution, which can be computationally expensive. Although approximations exist, they often perform poorly with outliers, which are crucial in quantization.

By assuming a normal distribution for the input tensor, we can precompute fixed quantile values for a standard normal distribution and rescale the input tensor values by the standard deviation to fit the $N(0, 1)$ distribution. This assumption has been empirically validated in the QLoRA paper.

Formally, the process is as follows:

1. Estimate the $2^k + 1$ quantiles of a theoretical $N(0, 1)$ distribution to create a k-bit quantile quantization data type for normal distributions:

$$q_i = \frac{1}{2}\left(Q_X\left(\frac{i}{2^k + 1}\right) + Q_X\left(\frac{i+1}{2^k + 1}\right)\right),$$

   where Q_X is the quantile function of $N(0, 1)$, k is the number of bits in the quantized representation (4 in this case), and $i \in \{0, \ldots, 2^k\}$ indexes the evenly split quantile values.

2. Normalize the values of this data type into the $[-1, 1]$ range.

3. Quantize an input weight tensor by normalizing it into the $[-1, 1]$ range through absolute maximum rescaling.

Thus, each weight of the pretrained neural network can have two different types:

- **Quantization type:** This refers to how the weights are stored, typically in GPU memory.

- **Compute type:** This is the type to which the weights are dequantized, usually either `fp16` or `bfloat16`.

This approach has been shown to maintain performance quality while significantly reducing memory usage.

Chapter 3

Related Work

This section serves as a comprehensive review of the related work, providing the essential background necessary to understand the research context of this thesis. It lays the groundwork by discussing key studies, methodologies, and findings that are crucial to comprehending the subsequent chapters. By establishing a solid review of existing literature, this section ensures that readers are well-equipped to follow the detailed discussions and analyses that follow.

Moreover, the content of this section is designed to highlight the current state of research and identify gaps that this thesis aims to address. It systematically covers relevant studies, theoretical frameworks, and technological advancements, thereby creating a cohesive context for the research presented.

3.1 LLM for Generating References

The advancement of generative large language models (LLMs), particularly models like GPT [6], has significantly impacted the domain of question-answering (QA) tasks across various fields, including medicine. Despite their transformative potential, these models face significant challenges in ensuring the verifiability and reliability of the information they generate. This chapter explores the existing literature on these challenges and the proposed solutions to enhance the credibility of LLM-generated content.

Challenges in Verifiability and Reliability

Several studies have highlighted the issues with the verifiability of responses generated by LLMs. [16] found that 69% of references generated by ChatGPT in the medical domain were fabricated. Additionally, the quality of these answers was rated at a median of 60% by professionals. Similarly, [17] evaluated four generative search engines [16] discovered that only 51.5% of the sentences generated were fully supported by their citations. Moreover, only 74.5% of the citations accurately supported the statements they were linked to, underscoring a significant gap in the reliability of these systems. Instead the paper [18] emphasize the importance of developing comprehensive citation mechanisms to enhance content transparency and verifiability. They propose that such mechanisms should account for both parametric and non-parametric content. Despite the complexities involved, incorporating citations can address intellectual property (IP) and ethical concerns, improving the overall reliability of LLMs.

The literature identifies several key challenges in implementing effective citation mechanisms:

- **Over-citation**: may lead to information overload and inadvertently expose sensitive information. Users could exploit extensive citations to gather additional sensitive data, increasing privacy risks [18].
- **Inaccurate Citations**: LLMs may incorrectly attribute information to sources that do not contain that information, misleading users [17, 19]
- **Outdated Citations**: as knowledge evolves, cited sources may become outdated, leading to the propagation of obsolete information.
- **Propagation of Misinformation**: LLMs could inadvertently cite unreliable sources, spreading misinformation [20]. Models might favor certain types of sources due to biases in the training data or retrieval

mechanisms, leading to an uneven representation of information [21, 22]

- **Diminished Creativity**: Reliance on citations could stifle the generation of novel content, reducing the model's creativity and ability to propose innovative solutions.

3.1.1 Approaches to Generating Text with References

There are two primary approaches for incorporating references in generated text: parametric and non-parametric knowledge [18] .

Parametric Knowledge

This approach involves training LLMs to generate references from internalized information (i.e., knowledge embedded in the model's parameters during training). However, this method poses technical challenges since LLMs do not maintain an explicit index of training data, making it difficult to reference sources accurately. The paper [23] proposed training models to include references using source identifiers, but this method is limited by citation inaccuracies and a focus on academic citations.

Non-Parametric Knowledge

Known as retrieval-augmented generation (RAG), this approach combines LLMs with information retrieval (IR) systems to form a hybrid system. The LLM is trained to recognize when citations are needed, and the IR system retrieves suitable sources to provide context. This method enhances the credibility and accuracy of the generated responses by integrating external knowledge bases without additional training, thereby reducing hallucinations. Annotators often perceive RAG-enhanced answers as more factual and specific compared to those generated by fine-tuned models [24].

3.2 Retrieval augmented generation

Retrieval-Augmented Generation (RAG) [24] is a sophisticated approach that combines the strengths of retrieval-based methods and generative models to improve the quality and relevance of generated text. This hybrid model leverages a retrieval component to fetch relevant documents or passages from a large corpus and uses a generative model to synthesize the final output. This methodology has shown significant promise in various natural language processing (NLP) tasks, particularly in question answering, dialogue systems, and text completion.

Mechanism of RAG

The RAG framework operates in two main stages: retrieval and generation. During the retrieval stage, a retriever model searches a vast corpus to find documents or passages that are most relevant to the input query. This retriever is often based on dense passage retrieval techniques, such as those implemented using bi-encoder architectures like DPR (Dense Passage Retrieval) [25].

Once relevant passages are retrieved, the generation stage commences. Here, a generative model, such as GPT-3 [6] or BERT [4], uses the retrieved passages as additional context to generate a coherent and contextually accurate response. The integration of retrieved information helps the generative model to produce more factually correct and informative outputs, addressing one of the key limitations of purely generative models, which can often hallucinate or generate plausible but incorrect information [24].

Various methods are employed for interfacing with external data. One of the simplest approaches, as utilized in [26], involves the user inputting a question that is then processed by an information retrieval engine. This engine returns a series of relevant documents, which are subsequently concatenated with the question and fed into the model to generate the answer.

In other methods, the retrieved text is directly integrated into the transformer architecture [27, 28]. These models are trained such that auto-regressive language models learn to generate text conditioned on document chunks retrieved from a large corpus. The retrieved text is encoded using an Encoder model like BERT and passed as key-value pairs into the attention block of the auto-regressive model.

Different strategies exist regarding how models engage with external data. Rather than merely receiving a question along with some retrieved evidence text, the model can be fine-tuned to actively query an external information retrieval system [29]. In contrast, the approach described in [30] adopts a less supervised method, enabling the model to learn to access external information sources in a more unsupervised fashion.

In [31], the model is trained to emulate human interaction with search engines by creating a text-based web-browsing environment. Both in [31] and [32], once the model is conditioned on retrieved text, multiple responses are generated and then ranked by human annotators. The preferences indicated by these annotators are subsequently used to train the initial model using Reinforcement Learning from Human Feedback (RLHF).

Another important aspect to consider is the length of the retrieved text. In [29], very short text fragments (just one or two sentences returned by the search engine) are used. Conversely, in [31], the model is capable of selecting a series of quotes from a large volume of text. In [32], the model processes entire documents up to the maximum context size. Utilizing only selected passages instead of full documents has the clear advantage of reducing the number of tokens fed into the model, allowing for a broader range of input. However, research [33] has indicated that isolated passages may sometimes lack the necessary context to provide accurate information, for instance, they could include acronyms or references that haven't been explained.

Applications of RAG

RAG has been effectively applied in several domains:

- **Question Answering:** In question answering systems, RAG can provide more accurate answers by grounding responses in retrieved documents, thus ensuring that the generated answers are backed by actual data.

- **Dialogue Systems:** In conversational AI, RAG helps in maintaining context over long dialogues by retrieving and incorporating relevant previous interactions or external documents.

- **Text Generation:** For tasks like article completion or creative writing, RAG can enhance the generated text's coherence and factual accuracy by drawing on a large body of knowledge.

3.3 Retrieval Component

One crucial aspect in the performance of retrieval-augmented generation systems is the retrieval component. Since each language model can process a limited amount of information, it is essential that the retrieved text is as relevant and comprehensive as possible to answer the given question.

Traditionally, information retrieval has employed methods such as TF-IDF, which relies on measuring the frequency of a word within a document. However, recent years have seen the rise of neural network-based approaches that utilize pretrained language models, providing a semantic understanding of both queries and documents. Various types of retrievers offering these capabilities have been researched.

3.3.1 BiEncoder

A BiEncoder model [25] employs two encoders that independently process the query and the document, resulting in two separate embeddings. These embeddings are then compared using cosine similarity or other similarity measures to determine the relevance of the document to the query. BiEncoders can be used to compute document embeddings offline and store them, creating what is known as a dense vector database. When a query is submitted, a dense representation is efficiently computed and matched against all stored vectors to find the most relevant documents.

Given a dataset: $D = \{(q_i, d_i^+, d_{i,1}^-, \ldots, d_{i,n}^-)\}_{i=1}^m$, where each instance consists of a query q_i, one related document d_i^+, and n unrelated documents $d_{i,1}^-$, the objective is to optimize the log-likelihood of the document related to the query:

$$L(q_i, d_i^+, d_{i,1}^-, \ldots, d_{i,n}^-) = -\log \frac{e^{\mathrm{sim}(q_i, d_i^+)}}{e^{\mathrm{sim}(q_i, d_i^+)} + \sum_{j=1}^n e^{\mathrm{sim}(q_i, d_{i,j}^-)}}.$$

Here, $\mathrm{sim}(q_i, d_i) = \mathrm{EncQ}(q_i)^\top \mathrm{EncD}(d_i)$ represents the cosine similarity of the embeddings of the query and the document, respectively obtained from two pretrained encoders.

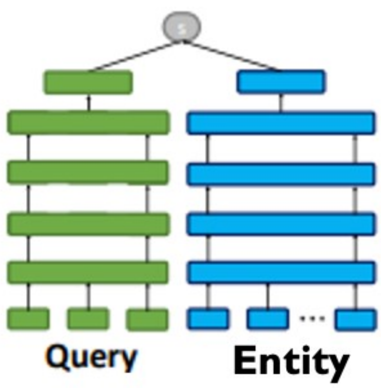

Figure 3.1: BiEncoder architecture

3.3.2 CrossEncoder

CrossEncoders process both the query and the document together, generating a single score that represents the similarity between the input sentence pairs [34]. Although CrossEncoders typically deliver superior performance, they are considerably slower. As a result, they are generally employed as re-rankers to evaluate the relevance of the top 100-1000 documents retrieved by a faster information retrieval mechanism. CrossEncoders can be trained similarly to BiEncoders, but instead of using a similarity function, they return a single embedding vector from one encoder.

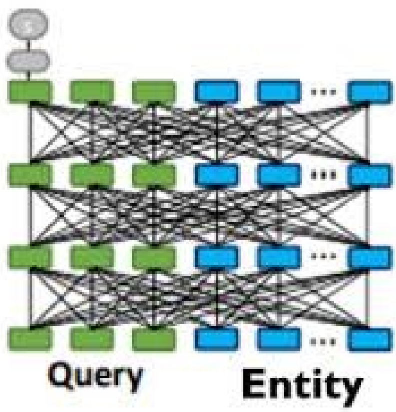

Figure 3.2: CrossEncoder architecture

3.3.3 Additional Approaches

Beyond the BiEncoder and CrossEncoder models, several other methods have been investigated in semantic retrieval. One such method is the Late Interaction model, which strikes a balance between the efficiency of BiEncoders and the effectiveness of CrossEncoders. This model computes multiple embeddings for each document and query, which are then matched and reranked using a more expressive similarity score. The ColBERT model [35] is a prime example of this approach.

Another category of methods integrates lexical techniques with deep learning approaches. These methods involve optimizing the hyperparameters of information retrieval algorithms using pretrained language models. By utilizing the semantic understanding provided by these models, the hyperparameters can be fine-tuned to enhance retrieval performance.

3.4 Claim Verification

In the literature, the task of determining a claim's truthfulness is referred to by various terms, such as verdict prediction [36], veracity prediction [37], and claim verification [38]. This task is typically a component of a multi-step process aimed at generating highly accurate results [36]. The process generally involves evaluating a claim and its corresponding evidence and categorizing it into one of three labels: support/evidence, contradiction/refute, or no evidence/not enough information. Verifying scientific claims can be seen as a natural language inference (NLI) task, treating it as a multi-class classification problem, consistent with previous research [39, 38].

Various techniques have been developed to enhance claim verification, with transformer models recently achieving state-of-the-art performance in general and scientific fact-checking [36]. These models typically take concatenated claim and evidence pairs as input to create representations for classifying the relationships between them [36]. Including the entire context of the evidence (e.g., the entire document) ensures minimal information loss and improved inference results [38].

Interestingly, both general-purpose and domain-specific large language models are utilized for scientific claim verification [37]. The successful use of general-purpose models for this task is supported by the creators of the SciFact dataset, who released the VeriSci model [38]. This model uses the RoBERTa-large model [40], pretrained on the FEVER dataset [39] and fine-tuned on the

SciFact dataset, demonstrating superior performance compared to other scenarios. The model also outperformed SciBERT [41], BioMedRoBERTa [42], and RoBERTa-base when trained solely on the SciFact dataset.

General-domain datasets for claim verification have existed since 2014 [43], with the first scientific claim verification dataset, SciFact, emerging in 2020 [38]. Since then, the number of scientific claim verification datasets has increased, including those collecting claims from social media posts [44], Wikipedia, and the internet in general [45, 46], various web portals [47, 37], science exam questions [36], or publications [48]. However, SciFact remains one of the few datasets containing claims from research papers and is the most utilized dataset for building scientific claim verification systems to date [37].

The study by Sarrouti et al. [46] demonstrates that the choice of in-domain dataset for fine-tuning significantly impacts performance. Experiments on several baseline models—BERT [4], SciBERT, BioBERT [49], and T5 [50]—trained and evaluated on the HealthVer dataset [46] showed that T5 outperformed all other models. They also tested the BERT-base model fine-tuned on the FEVER, SciFact, PubHealth, and HealthVer datasets and evaluated its performance on the HealthVer test set. Despite the FEVER dataset's size advantage, the model achieved better F1 scores when trained on SciFact and HealthVer datasets, supporting the notion that in-domain training yields significant benefits for domain-specific claim verification tasks.

Further evidence of the benefits of in-domain datasets is provided by Tan et al. [36], who performed in-domain fine-tuning using Med-Fact and Gsci-Fact datasets, followed by SciFact, HealthVer, and CLIMATE-FEVER [45] datasets on models such as BERT, DeBERTa [51], SciBERT, Longformer, and BioBERT. Compared to fine-tuning solely on SciFact, HealthVer, and CLIMATE-FEVER datasets, this approach improved performance for most models, with DeBERTa achieving the best results in nearly all scenarios.

Chapter 4

Methodology

This chapter outlines the research methods used to achieve the study's objectives. It begins with a description of the primary task, detailing the research questions and hypotheses. Next, it presents the baselines, serving as reference points for comparing the performance of proposed methods. The chapter then describes the datasets, including data sources, selection criteria, and preprocessing steps. It also covers data collection techniques, sampling strategies, and analytical procedures. Finally, ethical considerations related to data handling are discussed, ensuring the integrity and compliance of the research.

4.1 Information Retrieval Component

The Information Retrieval component utilizes data from the PubMed database [52] which is a comprehensive resource containing citations and biomedical literature from numerous sources, serves as the foundational data for our IR system. The system is designed to integrate both sparse vectors (lexical index) and dense vectors (semantic index), thus facilitating both lexical and semantic searches, as well as a hybrid combination of the two.

4.1.1 Lexical Retrieval

For the lexical retrieval, we employed a ranking function based on Best Matching 25 [53]. Using OpenSearch[1], we created an index for PubMed articles by concatenating titles and abstracts into a single indexed field. Additionally, we incorporated metadata such as authors' names, publication dates, and journal names to facilitate filtering. This comprehensive indexing enables the system to perform efficient and accurate lexical searches.

4.1.2 Semantic Retrieval

For the semantic retrieval component, we utilized dense vectors stored in the Qdrant vector database[2]. Qdrant's capability to memory map vectors to a hard drive significantly reduced the system's memory (RAM) requirements. To further optimize the retrieval time for semantic searches, we employed 8-bit quantized embeddings [15], while retaining the option to use full embeddings for rescoring results to ensure accuracy.

We implemented the Hierarchical Navigable Small World indexing technique for Approximate Nearest Neighbors using dot product metrics to facilitate vector comparisons [54] . Vector embeddings were generated using a bi-encoder sentence transformer model pre-trained on the MSMarco dataset [55], which, at the time of indexing, demonstrated superior performance on the Passage Retrieval Task.

4.1.3 Data Processing and Indexing

In processing our corpus of 36,797,469 abstracts, we identified and omitted 11,308,679 empty abstracts. These empty abstracts primarily originated from articles published before the digital era, articles from journals that are not freely accessible, or journals that do not require abstracts. After excluding

[1]`https://opensearch.org/`
[2]`https://qdrant.tech/`

these, we constructed two indices offline for subsequent use in online semantic and lexical searches.

The lexical index was created by indexing concatenated fields of titles and abstracts, supplemented with additional fields from PubMed articles for filtering purposes. The semantic index involved generating embeddings for the concatenated titles and abstracts using our model. This process, depicted in Figure 4.1, marked with an asterisk, ensured that the average number of tokens in the dataset's title and abstract concatenation was 650. Given that the model's maximum input size for embedding creation is 512 tokens, abstracts exceeding this limit were subdivided into segments of no more than 512 tokens each and indexed separately. These splits were carefully made at the end of sentences before reaching the 512-token threshold.

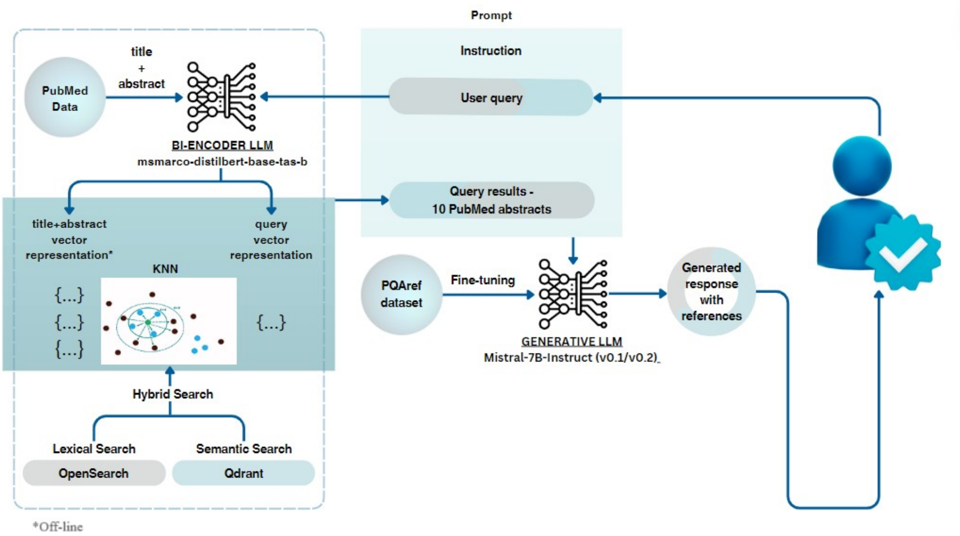

Figure 4.1: Architecture of our RAG system.

4.1.4 Hybrid Search

Our hybrid search methodology combines the strengths of both lexical and semantic IR components. To utilize this hybrid approach, we normalized the scores from each IR method to a scale ranging from 0 to 1. These normalized scores were then weighted according to the importance of each method. This dual approach allows the system not only to identify direct matches but also to discover semantically related phrases and text segments, even in the absence of exact textual matches. This hybrid search significantly enhances the system's ability to provide comprehensive and relevant search results.

4.2 Generative Component

The generative component of our system is based on the Mistral-7B model. Despite having fewer parameters, Mistral-7B demonstrates superior performance over larger models such as Llama 2 13B across all evaluated benchmarks and Llama 1 34B in reasoning benchmarks, maths, and code generation [56]. Compared to its 0.1 version, Mistral-7B v0.2 introduced an expanded context window (32K compared to the previous 8K) and several other adjustments (rope-theta = 1e6, no sliding-window attention), contributing to more accurate and consistent outputs, improved efficiency, and adaptability to various tasks [57].

For comparison purposes, we opted for testing both currently available instruction-tuned versions of Mistral-7B (v0.17 and v0.28). We test both models in the zero-shot mode and also fine-tune them using a custom dataset for referenced QA (see Section 4.2.1).

The input for the generative component consists of a user query and 10 abstracts retrieved by the IR component as most relevant to the user query. While generating the answer, the models perform another relevance check and answer the question using only the abstracts they find relevant. The final output is a concise answer that includes an abstract ID as a reference after each claim originating from the 10 abstracts.

In the following subsections, we briefly describe the dataset we used to fine-tune these models, as well as the fine-tuning process.

4.2.1 Dataset

We developed a custom dataset specifically for fine-tuning the large language models (LLMs) to perform referenced question answering (QA). The dataset comprises 9,075 questions, each accompanied by 10 relevant abstracts, including titles and PubMed IDs (PMIDs), as well as referenced answers to the questions derived from the provided abstracts.

The questions were randomly selected from the PubMedQA dataset [58], a benchmark dataset designed for biomedical question answering. For each question, the most relevant abstracts were retrieved from the PubMed repository. This retrieval process utilized a combination of entity search and free text search to ensure that the abstracts selected were highly pertinent to the questions.

To generate the answers based on the retrieved abstracts, we employed GPT-4 Turbo, specifically the gpt-4-1106-preview[3]. This version of GPT-4 Turbo features enhanced instruction-following capabilities, making it suitable for generating high-quality referenced answers. GPT-4 Turbo is currently ranked as the top model on the Chatbot Arena leaderboard, a crowdsourced open platform for evaluating large language models [59].

The prompt used to instruct GPT-4 Turbo to include references (PMIDs) in the answers was carefully designed to ensure that the model generated responses that were not only accurate but also appropriately cited the relevant abstracts. This meticulous process ensured the creation of a robust dataset that enhances the LLMs' ability to perform referenced QA effectively. The prompt used is as follows:

> Answer the question using relevant abstracts provided, up to 300 words. Reference the statements with the provided abstract_id in brackets next to the statement.

[3] `https://platform.openai.com/docs/models/gpt-4-turbo-and-gpt-4`

To ensure the completeness of answers, GPT-4 Turbo was instructed to continue generating content if there was more to generate. The answers were then automatically checked for completeness, and any incomplete final sentences were removed. This process resulted in answers ranging from 69 to 1221 tokens in length. In a small number of cases (25 questions), no direct answer was found in the abstracts, so the answer did not contain any references. The total input length in the dataset (question + abstracts + answer) ranges from 1686 to 6987 tokens.

We named this dataset PQAref and made it available through Hugging Face[4].

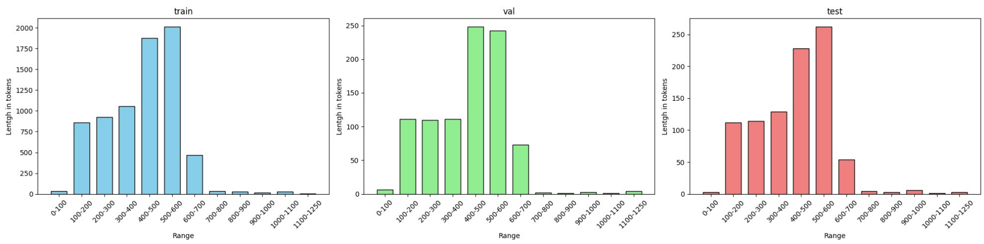

Figure 4.2: Distribution of answer length across train, val and test splits

4.3 Fact Verification

The performance of our models is evaluated using various metrics, including macro and weighted precision, recall, F1-score, and accuracy. These metrics collectively provide a comprehensive understanding of the models' effectiveness in the task at hand.

In our recent exploration of advanced natural language processing techniques, we opted to fine-tune three cutting-edge models: RoBERTa-large [40], XLM-RoBERTa Large [60], and DeBERTa Large [51], alongside DeBERTa SQuAD—a DeBERTa model fine-tuned using the SQuAD dataset. Our primary objective was to enhance their performance on the task of textual entailment.

[4]https://huggingface.co/datasets/BojanaBas/PQAref

The selection of DeBERTa was motivated by its outstanding performance in claim verification experiments [51]. RoBERTa was chosen for its modified attention mechanisms and its strong performance on label prediction tasks [40]. XLM-RoBERTa exhibited better performance on English data compared to RoBERTa [60]. DeBERTa SQuAD was selected due to its training on question-answer pairs, specifically tailored for question-answering tasks.

Fine-tuning was conducted using the SciFact dataset [61], which is specifically designed to support the development and evaluation of automated fact-checking systems focusing on scientific claims. Our objective in fine-tuning these models with the SciFact dataset was twofold: to improve their performance in discerning textual entailment within scientific texts and to compare their effectiveness in this specialized domain.

4.3.1 Dataset Transformation

The SciFact dataset comprises pairs of claims and evidence within the biomedical domain—a domain noted for its classification challenges even among human experts. We anticipate that it will serve as an optimal dataset for the fine-tuning of models aimed at claim verification within our system.

The SciFact dataset is structured into two separate files: corpuses and claims. The corpuses file contains the titles and abstracts of scientific articles, providing a rich source of evidence. The claims file includes various claims linked to these scientific articles via an identification number, facilitating the process of verifying the claims against the provided evidence. Initially, the claims file is divided into three parts: training, validation, and test. However, the publicly available version of the dataset does not include labels for the test partition. As a result, the 300 examples in the test partition were excluded from our analysis. To maximize the data available for our experiments, we combined the training and validation subsets, yielding a total of 1,711 examples. This combined dataset serves as the foundation for training, validating, and evaluating our claim verification models.

Initially, the titles and abstracts extracted from the corpus file underwent individual cleaning procedures. This process involved eliminating redundant spaces, excluding special characters present at the ends of the texts, removing surplus parentheses and brackets, and filtering out superfluous information contained within the abstracts. Recognizing the informative value inherent in article titles, we made a deliberate choice to concatenate them with their corresponding abstracts to construct a comprehensive response to the claims. Throughout this concatenation process, special attention was paid to titles lacking terminal punctuation. To maintain coherence and facilitate comprehension, we ensured that a concluding punctuation mark was appended to such titles. This step was essential to mitigate potential ambiguity and prevent the model from misinterpreting the concatenated text in instances where unpunctuated titles were directly joined with the initial sentence of the abstract, creating potentially unrelated sentences.

Subsequently, using the identification number, we integrated information from both files, ensuring that each claim was matched with its corresponding concatenation of the title and abstract, along with one of three labels: no evidence, support, or contradict. The SciFact dataset was initially designed so that the same labels for an abstract were repeated if they were found in multiple sentences of that abstract. However, since we opted to consider the entire concatenated title and abstract as a single response to the claim, rather than treating each sentence separately, we performed a deduplication of these instances. This step ensured that redundant combinations were removed. Ultimately, this process yielded 1,213 unique claim+[title+abstract] combinations, which form the final dataset for our experiments. This comprehensive and deduplicated dataset provides a robust foundation for training and evaluating our claim verification models.

Descriptive statistics of the combined dataset formed in this manner revealed that approximately 36% of claim+[title+abstract] combinations are labeled as no evidence, about 42% are labeled as support, and around 22% are labeled as contradict. We divided the dataset into training, validation, and test subsets in a ratio of 80:10:10, ensuring that the proportion of each label is preserved across all subsets. This stratified division helps maintain the consistency of label distribution, which is crucial for accurately training and evaluating the model's performance.

4.3.2 Fine-Tuning

Our study conducts a comparative analysis of transformer models fine-tuned and evaluated using a processed SciFact dataset. We conceptualized the task of claim verification as a multi-class classification problem, aiming to determine one of three relationships (no evidence, support, contradict) between a claim and its evidence. This task was structured as a Textual Entailment task.

The models were fine-tuned by concatenating claims (c) with corresponding PubMed titles and abstracts as evidence (e). The input format was:

$$\text{[CLS]}c\text{[SEP]}e\text{[SEP]}$$

The objective was to classify each input pair into one of the labels (l): no evidence, support, or contradict. Formally, the model's prediction is:

$$l\{c, e\} \in \{\text{no evidence, support, contradict}\}$$

Training was performed using the ADAM optimizer [62] with a learning rate of 1e-5 and a weight decay of 0.01. The training process was conducted on a DGX NVIDIA A100-40GB GPU, utilizing the PyTorch framework and the Hugging Face Transformer library. Each model underwent training for up to 15 epochs.

Chapter 5

Evaluation

In this chapter, we present a detailed evaluation of the various components of our system: Information Retrieval (IR), the Generative Component, and Fact Verification. The purpose of this chapter is to systematically assess the performance and effectiveness of each component, ensuring a comprehensive understanding of their capabilities and limitations.

The first section focuses on the Information Retrieval component. We describe the evaluation metrics and methodologies used to measure the accuracy and efficiency of our IR system in retrieving relevant abstracts from the PubMed database.

The second section evaluates the Generative Component. Here, we discuss the criteria and benchmarks used to assess the quality of the generated answers. This includes a detailed examination of the completeness, relevance, and coherence of the answers produced by the model, as well as a comparison with baseline models to highlight the improvements achieved through our approach.

The final section addresses Fact Verification. We outline the methods employed to verify the accuracy of the claims generated by our system. This involves evaluating the system's ability to correctly identify the relationship between claims and evidence as either ”no evidence,” ”support,” or ”contradict.” We use standard metrics such as precision, recall, and F1-score to quantify the system’s performance in this critical task.

Throughout this chapter, we provide insights into the strengths and weaknesses of each component, supported by empirical data and rigorous analysis. By the end of this chapter, the reader will have a clear understanding of how effectively our system performs across its different functions and the potential areas for future improvement.

5.1 Information Retrieval Component

To evaluate our Information Retrieval (IR) system, we utilized the BioASQ dataset [63], specifically designed to advance biomedical information retrieval and question answering (QA). This extensive dataset comprises 5,049 questions, each paired with gold-standard answers, relevant document snippets, and the PubMed IDs (PMIDs) of articles pertinent to each question.

Our evaluation involved comparing the PMIDs retrieved by our system against the gold-standard PMIDs provided in the BioASQ dataset. We quantified this comparison using the precision metric, which measures the proportion of relevant identifiers retrieved by our system out of the total PMIDs retrieved. Precision was evaluated using two metrics: precision at 10 retrieved documents (P@10) and mean average precision for 10 retrieved documents (MAP@10) [64].

The retrieval component of our system was assessed using three different approaches: (1) only lexical, (2) only semantic, and (3) a combination of both. Additionally, we experimented with varying weights for the lexical and semantic combinations to optimize performance.

In the lexical search, we experimented with stopword removal from the query, which yielded better results compared to lexical search without stopword removal, as shown in Table 5.1.

For semantic search, we explored three approaches: semantic search with full embeddings, semantic search with compressed embeddings (using 8-bit quantization), and semantic search with compressed embeddings followed by rescoring with full embeddings. The semantic search with full embeddings had an average response time of 30 seconds, rendering it inefficient for real-world applications.

In contrast, the semantic search with rescoring approach involved using compressed embeddings to retrieve 100 results and then rescoring the top 10 using full-size embeddings. This method improved precision by 0.3% and was only 52 milliseconds slower than the approach without rescoring, as illustrated in rows 1 and 2 of Table 5.1. Given the minimal additional time required, we tested various weight combinations of hybrid search incorporating semantic search with rescoring.

Furthermore, the parallel execution of semantic and lexical search contributed to the system's time efficiency, reducing the average execution time from 489 milliseconds to 442 milliseconds, as demonstrated in Table 1. This comprehensive evaluation highlights the effectiveness and efficiency of our IR system across different retrieval strategies.

From the experiments detailed in Table 5.1, it is clear that the performance of semantic search alone is suboptimal, with significant improvements observed when integrated with lexical search. Initially, a notable enhancement was achieved with the hybrid search employing a 0.1 lexical search weight, followed by a second substantial improvement with a 0.6 lexical search weight, resulting in absolute improvements of 10.3% and 16.3% respectively. Further increasing the lexical search weight beyond 0.6 did not yield noticeably different outcomes. Assigning a weight of 1 to lexical search in hybrid search excludes the semantic search component, effectively reducing the system to pure lexical search, which produces worse results. Since the subsequent generative component does not account for the order of retrieved documents, we employ

Table 5.1: Our IR and PubMed search engine performance evaluation on the BioASQ dataset.

	P@10	MAP@10	time [ms]
1. Semantic without rescore	14.0%	25.7%	245
2. Semantic with rescore	14.4%	26.0%	297
3. Hybrid with rescore (lex. 0.1 sem. 0.9)	24.7%	32.5%	442
4. Hybrid with rescore (lex. 0.2 sem. 0.8)	24.7%	32.5%	442
5. Hybrid with rescore (lex. 0.3 sem. 0.7)	24.7%	32.5%	442
6. Hybrid with rescore (lex. 0.4 sem. 0.6)	24.7%	32.6%	442
7. Hybrid with rescore (lex. 0.5 sem. 0.5)	25.2%	41.0%	442
8. Hybrid with rescore (lex. 0.6 sem. 0.4)	30.7%	42.0%	442
9. Hybrid with rescore (lex. 0.7 sem. 0.3)	**30.8%**	**42.5%**	**442**
10. Hybrid with rescore (lex. 0.8 sem. 0.2)	30.8%	42.5%	442
11. Hybrid with rescore (lex. 0.9 sem. 0.1)	30.8%	42.6%	442
12. Lexical with stopwords removal	28.7%	41.1%	189
13. Lexical without stopwords removal	28.3%	40.1%	189
14. PubMed without MeSH Terms	9.2%	15.3%	698
15. PubMed with MeSH Terms	12.0%	19.1%	742

the P@10 metric to determine the most effective combination of parameters for hybrid search. After evaluating various configurations, we identified the optimal parameters for hybrid search to be a lexical search weight of 0.7 and a semantic component weight of 0.3. By allocating a higher weight to the semantic search component (0.3 in row 9 instead of 0.1 in row 11), we enhance the model's ability to capture and utilize the deeper, contextual relationships inherent in biomedical texts. Consequently, as shown in row 9, we choose these parameter values to conduct a hybrid search in our system.

Additionally, we evaluated the performance of the PubMed search engine on the BioASQ dataset. When searching without MeSH terms, the PubMed search engine achieved a P@10 of 9.2% and a MAP@10 of 15.3%. When MeSH terms were included, the results improved, with a P@10 of 12% and a MAP@10 of 19.1%, as detailed in rows 14 and 15 of Table 5.1.

These findings underscore the importance of combining lexical and semantic search approaches to maximize retrieval performance. The hybrid search, with carefully balanced weights, leverages the strengths of both methods, leading to superior results compared to either method alone. The evaluation of PubMed's performance further highlights the potential for improvement in biomedical information retrieval systems through the integration of advanced search techniques.

5.2 Generative Component

For the purpose of a comprehensive standalone evaluation of the generative component, we utilized the PQAref test set. Our evaluation process encompassed both automated and manual methodologies to assess the effectiveness of the referenced Question Answering (QA) task. This detailed analysis involved several critical steps:

Firstly, we examined the total number of references included in each answer provided by the model. This step was crucial to understand the extent of information the model used to formulate its responses.

Secondly, we scrutinized the relevance of these references. This involved determining how many of the references per answer were pertinent and directly supportive of the answer given. The relevance check was vital to ensure that the model was not just generating numerous references, but generating ones that were contextually appropriate and valuable.

Thirdly, we verified the correctness of the reference IDs. This step was essential to confirm that the model accurately identified and utilized the correct references, thereby maintaining the integrity and reliability of the information provided.

Finally, we compared the number of relevant references to irrelevant ones within the model's answers. This comparison was critical to evaluate the overall quality and precision of the model's reference generation capability. A higher ratio of relevant to irrelevant references would indicate a more effective and accurate generative component.

Through this multi-faceted evaluation approach, we aimed to rigorously assess the performance of the generative component in producing referenced answers, ensuring both the quantity and quality of references met the standards required for reliable and informative QA outputs.

To generate referenced answers in zero-shot mode, we used the following prompt:

> Respond to the Instruction using only the information provided in the relevant abstracts under Abstracts. Reference the statements with the provided abstract_id in brackets next to the statement (for example PUBMED:1235):
> {INSTRUCTION}

To obtain referenced answers from the fine-tuned models, we used the following prompt:

> Respond to the Instruction using only the information provided in the relevant abstracts in "'Abstracts"' below.
> {INSTRUCTION}

Both prompts were chosen after extensive testing of several different prompting strategies and prompt versions. We employed default inference parameters for all four models, with the exception of setting the repetition penalty to 1.1 for the fine-tuned models and varying the values of max new tokens (or max tokens for the zero-shot mode) for all four models. Despite attempts to limit the length of answers using the max new tokens parameter or by specifying constraints in the prompt (e.g., "Answer in at most 300 words"), all models continued to generate an arbitrary number of tokens. This behavior was similarly observed in GPT-4 Turbo during the creation of the PQAref dataset. Token limitations, primarily imposed due to prolonged inference times for higher values, often resulted in truncated answers. Ultimately, the token limit was set to 1225, slightly exceeding the longest complete answer length in the training dataset (Fig. 4.2).

We refer to the zero-shot results of these two models as 0-M1 for version 0.1 and 0-M2 for version 0.2, and to the results of the fine-tuned models as M1 for version 0.1 and M2 for version 0.2. In both prompts, the instruction for the fine-tuned models consists of the user query and 10 retrieved abstracts. An example of a question and GPT-4 Turbo's answer from the test set, along with the answers from the other four models to the same question, can be found in the Appendix 6.

5.2.1 Automated Evaluation

The number of referenced abstracts in generated answers within the PQAref test set, which contains 908 examples, is detailed in Table 2. From this table, it is evident that the majority of GPT-4 Turbo answers from PQAref include one reference per answer, with 241 answers falling into this category. In contrast, models M1 and M2 show the highest occurrence of answers with three references, tallying 185 cases for M1 and 178 for M2.

In the zero-shot results, it is most common for both 0-M1 and 0-M2 not to reference any abstracts in their responses. Specifically, 0-M1 did not reference any abstracts in 527 instances, accounting for 58% of all answers, while 0-M2 did so in 165 instances, which is 18.2% of all answers. In comparison, M1 and M2 did not reference any abstracts in only 8 (0.9%) and 5 (0.5%) answers, respectively. Upon manual inspection, it was found that in these instances, the models indicated that none of the abstracts were relevant, showcasing their capability in executing the task accurately. Conversely, in the majority of answers without references from 0-M1 and 0-M2, the models answered the question without providing any references. Additionally, some responses from 0-M2 (35 instances) repeated the initial part of the instruction, highlighting the need for further post-processing of its answers.

Analyzing the entire test set of 908 examples, each with 10 abstracts, we find that 0-M2 has the highest average number of references per answer at 4.74. This is followed by M2 with an average of 4.2 references per answer, M1 with 4.01, and 0-M1 with 2.51 references per answer.

To assess the relevance of the referenced abstracts, we evaluated whether the models included at least the most relevant abstract for each question. Our dataset, derived from PubMedQA, often includes questions that match actual PubMed abstract titles, making it likely that the relevant article is retrieved. In our test split, this was true for 823 out of 908 inputs. We used these abstracts as the benchmark for relevance and measured how frequently the models referenced them. Table 3 shows the number of missed and correctly referenced most relevant abstracts using this approach.

For GPT-4 Turbo answers, the most relevant article was missed in only one case, indicating it served well as a referencing role model. M2 missed the relevant abstract in 10 examples, while M1 missed it in 29 examples. Overall, the fine-tuned models frequently referenced the most relevant abstract, with M2 and M1 achieving rates of 98.8% and 96.5%, respectively. In contrast, 0-M1 missed the most relevant abstracts in 60.4% of answers, and 0-M2 in 22.5% of answers, demonstrating a significantly weaker ability to identify and extract the most relevant abstracts compared to their fine-tuned counterparts.

We also verified whether all the IDs in the models' answers matched the PMIDs of the context-provided abstracts to ensure no hallucinated IDs were present. GPT-4 Turbo's answers in the PQAref dataset contained no hallucinated IDs. However, both M1 and M2 did produce hallucinated IDs, with a notable difference between the two. M1 produced 79 hallucinated IDs, whereas M2 produced only 3. These hallucinated IDs typically differed from the actual IDs by one or two digits. Upon manual inspection, it was found that M1 often blended information from various abstracts, while M2 strictly used information from the corresponding abstract, suggesting that M2's hallucinations were limited to minor errors in the ID digits without affecting the content. This pattern remained consistent across different temperature values of the model. In zero-shot performance, 0-M1 hallucinated 11 IDs and did not reference any abstracts in 58% of cases, presenting a higher number of answers without references compared to those with references. These findings indicate a clear advantage for M2 in producing more accurate and reliable answers.

N	GPT-4 Turbo	0-M1	0-M2	M1	M2
0	2	**527**	**165**	8	5
1	**241**	27	11	86	105
2	76	66	47	138	112
3	128	28	92	**185**	**178**
4	126	17	114	172	169
5	119	25	110	117	124
6	87	28	94	72	75
7	45	26	61	66	34
8	29	47	64	27	34
9	31	47	83	22	23
10	24	70	67	15	49
TOTAL	3,464	2,285	4,307	3,648	3,816
AVG	3.81	2.51	4.74	4.01	4.20

Table 5.2: Number of referenced abstracts per model on the PQAref test set. N: number of referenced abstracts per answer. TOTAL: is the sum of referenced abstracts per model. AVG: the average number of references per answer.

5.2.2 Manual Evaluation

To perform a manual evaluation, we extracted 10 random examples from the PQAref test set and assessed the relevance of each abstract within these examples. We categorized the abstracts into two primary types: relevant and irrelevant. Relevant abstracts are those that address all specific aspects of the question, providing direct answers. Among these, we identified abstracts whose titles matched the question as the most relevant, similar to the 823 examples used during automatic evaluation.

On the other hand, irrelevant abstracts were divided into two subcategories. The first type includes abstracts that miss the main topic of the question entirely (e.g., discussing heart failure instead of knee problems), which we considered completely irrelevant. The second type includes abstracts that discuss a more general topic and therefore do not cover all aspects of the question; these were considered partially irrelevant. This group might contain additional information but does not provide a direct answer to the question.

It is crucial to understand that irrelevant abstracts can lead to two types of mistakes. If the model references a completely irrelevant abstract, it is clearly a mistake. However, referencing a partially irrelevant abstract can be more nuanced. If the answer also includes a relevant abstract that provides a direct answer, the partially irrelevant abstract might be considered additional information. If no relevant abstract is provided, the model has likely missed the main point of the question.

We then examined how the models referenced the most relevant and irrelevant abstracts in these 10 qualitatively observed examples. The fine-tuned models referenced the most relevant abstracts consistently, demonstrating their ability to grasp the main point. In contrast, 0-M1 and 0-M2 failed to reference the most relevant abstracts 4 and 2 times, respectively, and these answers contained no references at all. None of the models referenced completely irrelevant abstracts. All four models tended to provide additional information by referencing partially irrelevant abstracts. In several cases, the models seemed to filter abstracts based on their understanding of a term used in the question, excluding abstracts that used different phrasing or an extended meaning of the term (e.g., interpreting ”donation” to refer only to organ, tissue, or bone marrow donation, and not to cell and blood donation).

We also conducted a quantitative analysis to evaluate how well the models identified all relevant abstracts. To account for variations in the number of relevant abstracts per document and document-specific characteristics, we collectively considered all 100 abstracts across the 10 questions. Among these 100 abstracts, evaluators identified 42 as relevant and 58 as irrelevant. We prioritized and calculated recall for relevant abstracts for each model, as our primary concern was their ability to correctly identify and reference relevant

Table 5.3: The number of missed and referenced most relevant abstracts of 823 abstracts across the models.

	GPT-4 Turbo	0-M1	0-M2	M1	M2
Relevant missed	1 (0.1%)	497 (60.4%)	185 (22.5%)	29 (3.5%)	10 (1.2%)
Relevant referenced	822 (99.9%)	326 (39.6%)	638 (77.5%)	794 (96.5%)	813 (98.8%)

abstracts. M1 exhibited the highest recall at 0.76, followed by M2 with 0.67, 0-M2 with 0.62, and 0-M1 with 0.29. For reference, the recall measured on GPT-4 Turbo answers from the test set was 0.62. These results are summarized in the first row of Table 4. Based on the analysis of these 10 manually reviewed documents, the findings suggest that M1 outperforms the other models in terms of referencing abstracts deemed relevant by evaluators, showing the highest benefit from the fine-tuning process.

5.3 RAG: Joint Evaluation

In this section, we present a joint evaluation of our system, combining the Information Retrieval (IR) component (which leverages a hybrid lexical and semantic search) and the generative component using the IR outputs. We conducted a manual evaluation of the IR output using the same 10 PQAref questions selected for evaluating the generative component.

To retrieve relevant abstracts from indexed PubMed articles, we employed the best-performing hybrid search parameter combination. We retrieved 10 abstracts for each question and manually assessed their relevance. This manual evaluation yielded a Precision at 10 (P@10) score of 50%, highlighting the effectiveness of our IR component in locating pertinent documents for query responses. The IR evaluation on BioASQ, which achieved a P@10 of 30.8% with the same hybrid search weight combination, further supports the validity of our manual evaluation results on the PQAref dataset.

Next, we used the same prompts for GPT-4 Turbo as in Section 4.2.1, and those used in Section 5.2 for models 0-M1, 0-M2, M1, and M2, to generate referenced answers based on the retrieved documents. We calculated the recall values for the relevant abstracts in the 10 generated answers and displayed them in the second row of Table 5.4. Notably, the best-performing model was M1, with a recall of 0.64. This model cited a higher number of abstracts containing relevant answers compared to the other models. Based on recall alone, 0-M2 performed slightly better than M2, though only by a margin of 0.01. However, in one of the 10 examples, 0-M2 did not provide any references in its detailed answer. M2, the third-best model with a recall of 0.58, properly referenced all the answers.

From Table 5.2, we observe that 0-M2 has the highest number of references overall, but it also fails to provide any references in 18.2% of the answers. Considering this aspect, M2's answers are deemed more reliable compared to 0-M2. Although M2 shows a slightly lower recall compared to M1 because it includes fewer references to abstracts that provide direct answers, its answers are preferred since the IR component consistently identifies documents related to the topic, and M2 includes more additional citations, offering more comprehensive answers on the same topics. In this context, GPT-4 Turbo achieved a recall of 0.46, while 0-M1 had the lowest recall of all the models (0.37), largely due to a significant number of answers with no references (5 out of 10).

By combining the strengths of both the IR and generative components, we demonstrate that M2, despite its lower recall compared to M1, produces more elaborate and contextually rich answers, making it a valuable model for generating well-referenced responses in our evaluation framework.

	GPT-4 Turbo	0-M1	0-M2	M1	M2
PQAref	0.62	0.29	0.62	**0.76**	<u>0.67</u>
IR	0.46	0.37	<u>0.59</u>	**0.64**	0.58

Table 5.4: Recall values for relevant abstracts on 10 examples from the PQAref test set and same 10 questions with abstracts retrieved with our IR system.

5.4 Fact Verification

We conducted a detailed three-step evaluation of our fact-checking models to ensure they were accurate and reliable. First, we tested our fact-checking models using a specific part of the transformed SciFact dataset [61]. This helped us see how well each model could verify facts in a controlled setting. We measured different performance metrics like precision, recall, and F1-score to find out which model was the best. Next, we took the best model from our initial tests and validated it using a different dataset. We checked the same performance metrics to make sure the model was still performing well outside of the original test environment. Finally, we compared the performance of our top model with GPT-4 models, which are some of the most advanced language models available today. This comparison helped us understand how our model stacked up against leading technology in the field of fact-checking.

By following these three steps, we ensured that our best-performing model was accurate and reliable across different datasets and in comparison to state-of-the-art models like GPT-4.

5.4.1 In-domain Evaluation

The top-performing model achieves an F1-score of 0.87. On the other hand, the DeBERTa model with an early stopping patience of 4 achieved the highest F1-score of 0.88. The results also indicate that the *CONTRADICT* class is the most challenging for the models, which is expected given that this class makes up only 22% of the dataset. This results in fewer examples for training and evaluation. This imbalance likely contributes to the models' struggles to accurately predict this class, emphasizing the need for more focused strategies to enhance performance in underrepresented categories.

	RoBERTa L_{SF}				XLM RoBERTa L_{SF}			
	NE*	S	C	wa	NE	S	C	wa
P	0.71	0.55	0.00	0.48	0.83	0.69	0.54	0.71
R	0.73	0.82	0.00	0.61	0.89	0.67	0.55	0.71
F1	0.72	0.66	0.00	0.53	0.86	0.68	0.53	0.71
Acc		0.61				0.71		
P	0.85	0.75	0.67	0.77	0.75	0.76	0.71	0.74
R	0.89	0.76	0.79	0.77	0.78	0.70	0.73	0.73
F1	0.87	0.76	0.73	0.77	0.82	0.71	0.72	0.74
Acc		0.77					0.75	

Table 5.5: Performance metrics for RoBERTa L_{SF} and XLM RoBERTa L_{SF} across different conditions.

	DeBERTa L_{SF}				DeBERTa SQuAD L_{SF}			
	NE	S	C	wa	NE	S	C	wa
P	0.83	0.86	0.85	0.84	0.86	0.90	0.82	0.87
R	0.86	0.84	0.81	0.84	0.86	0.88	0.85	0.87
F1	0.84	0.85	0.83	0.84	0.86	0.89	0.84	**0.87**
Acc		0.84					0.87	
P	0.88	0.90	0.88	0.89	0.82	0.91	0.88	0.87
R	0.95	0.88	0.78	0.89	0.87	0.88	0.85	0.87
F1	0.91	0.89	0.82	**0.88**	0.87	0.88	0.85	0.87
Acc		0.89					0.87	

Table 5.6: Performance metrics for DeBERTa L_{SF} and DeBERTa SQuAD L_{SF} across different conditions.

5.4.2 Out-of-domain Evaluation

To evaluate our best-performing model, DeBERTa fine-tuned, on a dataset different from the one used for training and in-domain evaluation, we selected the HealthVer dataset [65]. This dataset is tailored for evidence-based fact-checking of health-related claims, allowing researchers to assess the accuracy of real-world claims by verifying them against scientific articles.

The fine-tuned model achieved a weighted average F1 score of 0.44 and an accuracy of 0.50. Comparing these results with those reported by [65], who fine-tuned a BERT-base model with SciFact and evaluated it on the HealthVer test set, we see that they achieved an F1 score of 0.36 and an accuracy of 0.39. This indicates that our DeBERTa model, fine-tuned on the transformed SciFact dataset, outperformed these results.

We identified DeBERTa fine-tuned with 80% of the transformed SciFact dataset as the optimal model for Textual Entailment. We then tested this model out-of-domain on the HealthVer dataset, observing superior performance compared to previous state-of-the-art (SOTA) models, with an absolute increase of 8% in the F1 score. Given that the test subset, comprising 10% of the transformed SciFact dataset, had already been used for in-domain evaluation, we included it in the training set. We then retrained the DeBERTa model on 90% of the data from the transformed SciFact dataset. Evaluating this new model on the HealthVer dataset, we observed a further absolute improvement of 4% in the F1 metric. This exploration of expanding the training dataset highlights the adaptability and robustness of our approach.

	DeBERTa$_{SF-80}$				DeBERTa$_{SF-90}$			
	NE	S	C	wa	NE	S	C	wa
P	0.46	0.70	0.66	0.60	0.47	0.67	0.69	0.59
R	0.94	0.25	0.15	0.50	0.88	0.29	0.27	0.52
F1	0.62	0.37	0.24	**0.44**	0.61	0.40	0.39	**0.48**
Acc	0.50				0.52			

Table 5.7: Results of the DeBERTa model fine-tuned on the 80% and 90% of the SciFact dataset end evaluated on the HealthVer test set.

5.4.3 Comparison with GPT-4 Model

We used the same test set as for our in-domain evaluation, which consisted of 10% of our transformed SciFact dataset. This subset included 122 examples covering three classes. We assessed the performance of GPT-4, GPT-4 Turbo, and GPT-4o in zero-shot mode using this test set. The specific prompt used for this evaluation was as follows:

> Critically asses whether the statement is supported, contradicted or there is no evidence for the statement in the given abstract. Output SUPPORT if the statement is supported by the abstract. Output CONTRADICT if statement is in contradiction with the abstract and output NO_EVIDENCE if there is no evidence for the statement in the abstract.

For all models, the temperature parameter was set to 0 to minimize randomness and produce the most consistent outputs, while the maximum tokens parameter was set to 350 to allow sufficient context generation. This setup enabled us to directly compare the performance of our fine-tuned transformer-based model with that of the GPT-4 series models in a zero-shot scenario under identical conditions.

In our experiments, we found that our transformer-based model for claim verification outperformed GPT-4, GPT-4 Turbo, and GPT-4o. Specifically, our model demonstrated superior performance across various evaluation metrics, including accuracy and F1-score. In particular, DeBERTa achieved an F1 score of 0.88, as shown in Table 5.6, compared to GPT-4's performance with an F1 score of 0.81, as shown in Table 5.8, on the SciFact test set.

These results underscore the effectiveness of our fine-tuning approach and the robustness of our model architecture in handling complex claim verification tasks. The consistent outperformance of our model compared to these state-of-the-art models highlights its potential for real-world applications and further establishes its credibility in the domain of automated fact-checking.

Additionally, our model is open-source, providing the transparency and flexibility crucial for industries such as pharmaceuticals and biomedicine, where stringent process control is required. Unlike closed models, our open-source solution allows for comprehensive customization and verification, ensuring that the claim verification process meets the rigorous standards necessary in these fields.

	GPT-4				GPT-4 Turbo				GPT-4o			
	NE	S	C	wa	NE	S	C	wa	NE	S	C	wa
P	0.85	0.77	0.84	0.82	0.93	0.81	0.65	0.82	0.72	0.91	0.74	0.80
R	0.80	0.94	0.59	0.81	0.64	0.92	0.81	0.80	0.89	0.80	0.63	0.80
F1	0.82	0.85	0.70	**0.81**	0.76	0.86	0.72	0.79	0.80	0.85	0.68	0.79
Acc	0.81				0.80				0.80			

Table 5.8: Results of GPT-4 models.

Chapter 6

Conclusion and Future Work

In this thesis, we developed and evaluated a novel biomedical generative search system that integrates information retrieval (IR) with a generative component to answer biomedical questions using PubMed data. Our goal was to provide accurate, verifiable answers while leveraging open-source models.

Our findings show that combining lexical and semantic searches yields the highest precision. Our system improved Mean Average Precision at 10 (MAP@10) by 23.4% compared to PubMed's search engine. While lexical search alone was superior, integrating semantic search was beneficial, especially for non-exact term matches. Future improvements could involve fine-tuning models with domain-specific data to enhance semantic search.

We evaluated multiple generative models, including Mistral 7B Instruct and GPT-4 Turbo. Fine-tuned models (M1 and M2) outperformed zero-shot models in referencing relevant abstracts. M2 showed a 2.3% improvement over M1 and performed best in reducing hallucinations in reference IDs. Manual evaluation confirmed fine-tuned models' superiority in recall values, though results varied with abstracts retrieved by our IR system.

Overall, our system achieved performance comparable to GPT-4 Turbo using smaller, open-source models. Automated methods for evaluating referenced QA quality are proposed for more comprehensive assessment.

In the final phase, we fine-tuned models for fact verification using the

SciFact dataset. The DeBERTa model, fine-tuned with 80% of the dataset, achieved an F1 score of 0.88. Testing on the HealthVer dataset yielded an F1 score of 0.44 and an accuracy of 0.50, surpassing previous benchmarks. Increasing the training dataset to 90% further improved the F1 score by 4%. Our model outperformed GPT-4 in a zero-shot regime by 7% in F1 score.

Our open-source model is beneficial for pharmaceuticals and biomedicine, ensuring transparency and adherence to industry standards. It enhances scientific productivity and provides a reliable framework for automated fact-checking. Given the challenges of the SciFact dataset, our model's real-world performance may be underestimated, and we aim to further improve our verification methodology.

Future Work

Our research has laid a strong foundation for developing effective biomedical generative search systems. However, several avenues for future work can further enhance the system's performance and reliability:

Fine-Tuning on Domain-Specific Data: To improve semantic search performance, future efforts should focus on fine-tuning models on extensive biomedical datasets. This would enhance the models' ability to encode domain-specific knowledge and improve contextual understanding, leading to more accurate and relevant search results.

Automated Evaluation Methods: Implementing automated evaluation methods, would allow for a more comprehensive and faster assessment of referenced QA quality. This would facilitate continuous improvement of the models and provide a reliable benchmark for comparison with other state-of-the-art systems.

Expanding the Evaluation Dataset: To ensure the robustness of our findings, future research should involve testing the models on larger and more diverse datasets. This would help identify potential limitations and areas for improvement, ensuring the models' generalizability across various biomedical domains.

User-Centric Enhancements: Incorporating user feedback into the system's development cycle can help tailor the models to better meet user needs. Future work could explore interactive features that allow users to provide input on the relevance and accuracy of generated answers, thereby continuously refining the system's performance.

Generate a New Dataset for Claim Verification: The SciFact dataset, while adequate for biomedical claim verification using literature, presents challenges such as short and unclear claims lacking contextual labels. We aim to collaborate with industry partners to create a cleaner, more comprehensive dataset that overcomes these challenges. A well-curated dataset will provide richer context and clearer claims, improving the training and evaluation of verification models.

Develop a Comprehensive Verification Methodology: Recognizing the limitations of NLI, even with the most powerful neural architectures, we plan to develop a more holistic approach. This methodology will combine textual entailment tasks, text similarity measures, and chains of thought in fine-tuned large language models. By integrating these techniques, we aim to enhance the robustness and accuracy of claim verification.

Integration with Clinical Decision Support Systems (CDSS): Extending the application of our generative search system to integrate with CDSS could significantly impact healthcare delivery. Providing accurate, evidence-based answers will assist healthcare professionals in making informed decisions, ultimately improving patient outcomes.

User-Centric Enhancements: Incorporating user feedback into the system's development cycle can help tailor the models to better meet user needs. Future work could explore interactive features that allow users to provide input on the relevance and accuracy of generated answers, thereby continuously refining the system's performance.

By pursuing these future directions, we aim to advance the capabilities of biomedical generative search and verification systems, ensuring they provide reliable, accurate, and user-centered information. This will support biomedical research and clinical practice, fostering greater trust and adoption of generative AI technologies in these critical domains.

Appendix

Appendix A: Example answers of generative LLMs

Models' answers to the question ”Does leisure time physical activity in early pregnancy protect against pre-eclampsia?”

GPT4 Turbo: The relationship between leisure time physical activity (LTPA) in early pregnancy and the risk of pre-eclampsia is not clearly established, with studies showing mixed results. Some studies suggest that there may be a protective effect of LTPA against pre-eclampsia, while others do not find a significant association or suggest an in- creased risk with high levels of activity. A prospective cohort study in Denmark found that moderate levels of physical ac- tivity (1-270 minutes/week) in the first trimester had no statistically significant association with the risk of pre-eclampsia, but high levels of activity (exceeding 270 minutes/week) were associated with an increased risk of severe pre-eclampsia (PUBMED:19055653). Similarly, a population-based prospective cohort study in Amsterdam reported that the amount of time or intensity of physical activity in leisure time was not associated with a difference in risk of pre-eclampsia or gestational hy- pertension (PUBMED:19916877). In contrast, a study from the New Haven, Connecticut area suggested that women who engaged in any regular LTPA during early pregnancy were at decreased risk of pre- eclampsia (PUBMED:15466498). Another study found a tendency towards a lower risk of pre-eclampsia in women with the highest degree of physical activity during leisure time, especially in overweight women, although no significant associations were found (PUBMED:20121498). A systematic review examining the association between LTPA before and/or during pregnancy and the risk of pre-eclampsia found mixed results, with some studies sug- gesting that vigorous-intensity LTPA may reduce the risk of pre-eclampsia, while one study reported an elevated risk with high amounts of LTPA (PUBMED:23836014). Another case-control study found that women who performed regular LTPA during the first 20 weeks of pregnancy had a reduced risk of pre-eclampsia and gestational hypertension (PUBMED:2592903). In summary, the evidence is inconclusive regarding the protective effect of LTPA in early pregnancy against pre-eclampsia. While some studies suggest a potential bene- fit, especially with moderate activity levels, others indicate no effect or an increased risk with high levels of activity. It is important to consider the intensity and duration of LTPA when evaluating its impact on pre-eclampsia risk.

0-M1: he instruction is to provide a response to the instruction using only the information provided in the abstracts. The instruction is: ”Does leisure time physical activity in early pregnancy protect against pre-eclampsia?” Based on the provided abstracts, the answer is: No, leisure time physical activity in early pregnancy does not protect against pre-eclampsia.

0-M2: The available evidence from the provided abstracts suggests that the relationship between leisure time physical activity (LTPA) in early pregnancy and the risk of pre-eclampsia is not clear-cut. Some studies indicate a protective effect of LTPA against pre-eclampsia, while others do not find a significant association or suggest an increased risk with high levels of activity. One prospective cohort study in Denmark found that moderate levels of physical activity (1-270 minutes/week) in the first trimester had no statistically significant association with the risk of pre-eclampsia, but high levels of activity (exceeding 270 minutes/week) were associated with an increased risk of severe pre-eclampsia (PUBMED:19055653). Another population-based prospective cohort study in Amsterdam reported that the amount of time or intensity of physical activity in leisure time was not associated with a difference in risk of pre-eclampsia or gestational hypertension (PUBMED:19916877). However, a study from the New Haven, Connecticut area suggested that women who engaged in any regular LTPA during early pregnancy were at decreased risk of pre-eclampsia (PUBMED:15466498). Another study found a tendency towards a lower risk of pre-eclampsia in women with the highest degree of physical activity during leisure time, especially in overweight women, although no significant associations were found (PUBMED:20121498). A systematic review examining the association between LTPA before and/or during pregnancy and the risk of pre-eclampsia found mixed results, with some studies suggesting that vigorous-intensity LTPA may reduce the risk of pre-eclampsia, while one study reported an elevated risk with high amounts of LTPA (PUBMED:23836014). Another case-control study found that women who performed regular LTPA during the first 20 weeks of pregnancy had a reduced risk of pre-eclampsia and gestational hypertension (PUBMED:2592903). In summary, the evidence is inconclusive regarding the protective effect of LTPA in early pregnancy against pre-eclampsia. While some studies suggest a potential benefit, especially with moderate activity levels, others indicate no effect or an increased risk with high levels of activity. It is important to consider the intensity and duration of LTPA when evaluating its impact on pre-eclampsia risk. Further research is needed to clarify the relationship between LTPA and pre-eclampsia risk.

M1: Based on the provided abstracts, leisure time physical activity in early pregnancy does not appear to protect against pre-eclampsia. A prospective cohort study in Denmark found that the two highest levels of physical activity were associated with increased risk of severe pre-eclampsia compared with the nonexercising group, suggesting that leisure time physical activity exceeding 270 minutes/week in first trimester may increase the risk of severe pre-eclampsia (PUBMED:19055653). Another study in Amsterdam, The Netherlands, did not find any association between physical activity in leisure time early in pregnancy and the incidence of preeclampsia or gestational hypertension (PUBMED:19916877). Similarly, a study in New Haven, Connecticut, found that women who engaged in any regular leisure time physical activity, regardless of caloric expenditure, were at decreased risk of preeclampsia, but there was no indication of a protective effect of workplace activity, leisure-time physical activity, or unemployment (PUBMED:15466498). These findings suggest that while certain forms of physical activity may have a protective effect on preeclampsia, the overall evidence does not support a clear protective effect of leisure time physical activity in early pregnancy against this condition. Therefore, based on the available abstracts, leisure time physical activity in early pregnancy does not protect against pre-eclampsia.

M2: Based on the provided abstracts, there is conflicting evidence regarding whether leisure time physical activity (LTPA) in early pregnancy protects against pre-eclampsia. Some studies suggest that LTPA may offer protection against pre-eclampsia, while others do not find a significant association or even indicate that excessive LTPA could increase the risk of severe pre-eclampsia. For instance, one study found that women who engaged in any regular LTPA regardless of caloric expenditure were at decreased risk of preeclampsia, although another study did not find a significant association between the amount of time or intensity of physical activity in leisure time and the risk of preeclampsia or gestational hypertension (PUBMED:15466498; PUBMED:19916877). Another study suggested that women who performed regular LTPA had a reduced risk of pre-eclampsia and gestational hypertension, with the relative risks decreasing as the average time spent in LTPA increased (PUBMED:2592903).

However, another study documented an increased risk of severe pre-eclampsia with higher levels of LTPA (PUBMED:19055653). Similarly, a systematic review reported that high-intensity LTPA before and/or during pregnancy or more than 4 hours per week of LTPA may reduce the risk of pre-eclampsia, but also noted that one study found an elevated risk of severe pre-eclampsia with high amounts of LTPA (PUBMED:23836014). Furthermore, a case-control study indicated that women who performed regular LTPA had a reduced risk of pre-eclampsia and gestational hypertension, with the relative risks decreasing as the average time spent in LTPA increased (PUBMED:26910608). A systematic review and meta-analysis of RCTs and cohort studies supported the promotion of LTPA in pregnancy as a strategy to improve maternal and child health, suggesting that participation in LTPA was associated with lower weight gain during pregnancy, lower likelihood of gestational diabetes mellitus (GDM), and lower risk of preterm delivery (PUBMED:27282925). In conclusion, while some studies suggest that LTPA in early pregnancy may protect against pre-eclampsia, particularly at moderate intensities, the evidence is not consistent across all studies. Further research is needed to clarify the relationship between LTPA and pre-eclampsia risk, taking into account the intensity and duration of physical activity, as well as individual differences in susceptibility to the condition. Additionally, the potential influence of other factors such as prepregnancy BMI and socioeconomic status should be considered (PUBMED:32093248).

Bibliography

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[2] L. R. Medsker, L. Jain, et al. Recurrent neural networks. *Design and Applications*, 5(64-67):2, 2001.

[3] J. Cheng, L. Dong, and M. Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[5] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. Improving language understanding by generative pre-training, 2018.

[6] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 2020.

[7] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[8] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni. Generalizing from a few examples: a survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34, 2020.

[9] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig. Pre-train, prompt, and predict: a systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[10] K. Shum, S. Diao, and T. Zhang. Automatic prompt augmentation and selection with chain-of-thought from labeled data. *arXiv preprint arXiv:2302.12822*, 2023.

[11] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[12] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[13] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[14] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer. Qlora: efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

[15] T. Dettmers, M. Lewis, S. Shleifer, and L. Zettlemoyer. 8-bit optimizers via block-wise quantization. *arXiv preprint arXiv:2110.02861*, 2021.

[16] J. Gravel, M. D’Amours-Gravel, and E. Osmanlliu. Learning to fake it: limited responses and fabricated references provided by chatgpt for medical questions. *Mayo Clinic Proceedings: Digital Health*, 1(3):226–234, 2023.

[17] N. F. Liu, T. Zhang, and P. Liang. Evaluating verifiability in generative search engines. *arXiv preprint arXiv:2304.09848*, 2023.

[18] J. Huang and K. C.-C. Chang. Towards reasoning in large language models: a survey. *arXiv preprint arXiv:2212.10403*, 2022.

[19] T. Gao, H. Yen, J. Yu, and D. Chen. Enabling large language models to generate text with citations. *arXiv preprint arXiv:2305.14627*, 2023.

[20] Y. Pan, L. Pan, W. Chen, P. Nakov, M.-Y. Kan, and W. Y. Wang. On the risk of misinformation pollution with large language models. *arXiv preprint arXiv:2305.13661*, 2023.

[21] A.-S. Jannot, T. Agoritsas, A. Gayet-Ageron, and T. V. Perneger. Citation bias favoring statistically significant studies was present in medical research. *Journal of clinical epidemiology*, 66(3):296–301, 2013.

[22] S. A. Greenberg. How citation distortions create unfounded authority: analysis of a citation network. *Bmj*, 339, 2009.

[23] R. Taylor, M. Kardas, G. Cucurull, T. Scialom, A. Hartshorn, E. Saravia, A. Poulton, V. Kerkez, and R. Stojnic. Galactica: a large language model for science. arxiv 2022. *arXiv preprint arXiv:2211.09085*, 10, 2023.

[24] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 2020.

[25] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*, 2020.

[26] A. Lazaridou, E. Gribovskaya, W. Stokowiec, and N. Grigorev. Internet-augmented language models through few-shot prompting for open-domain question answering. *arXiv preprint arXiv:2203.05115*, 2022.

[27] Y. Wu, M. N. Rabe, D. Hutchins, and C. Szegedy. Memorizing transformers. *arXiv preprint arXiv:2203.08913*, 2022.

[28] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. B. Van Den Driessche, J.-B. Lespiau, B. Damoc, A. Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR, 2022.

[29] R. Thoppilan, D. De Freitas, J. Hall, N. Shazeer, A. Kulshreshtha, H.-T. Cheng, A. Jin, T. Bos, L. Baker, Y. Du, et al. Lamda: language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

[30] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, and T. Scialom. Toolformer: language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36, 2024.

[31] R. Nakano, J. Hilton, S. Balaji, J. Wu, L. Ouyang, C. Kim, C. Hesse, S. Jain, V. Kosaraju, W. Saunders, et al. Webgpt: browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

[32] J. Menick, M. Trebacz, V. Mikulik, J. Aslanides, F. Song, M. Chadwick, M. Glaese, S. Young, L. Campbell-Gillingham, G. Irving, et al. Teaching language models to support answers with verified quotes. *arXiv preprint arXiv:2203.11147*, 2022.

[33] D. Wadden, K. Lo, L. L. Wang, A. Cohan, I. Beltagy, and H. Hajishirzi. Multivers: improving scientific claim verification with weak supervision and full-document context. *arXiv preprint arXiv:2112.01640*, 2021.

[34] S. MacAvaney, A. Yates, A. Cohan, and N. Goharian. Cedr: contextualized embeddings for document ranking. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 1101–1104, 2019.

[35] O. Khattab and M. Zaharia. Colbert: efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48, 2020.

[36] A. Tan et al. Evaluating verifiability in generative search engines. *Journal of Information Retrieval*, 2023.

[37] M. Vladika and F. Matthes. Veracity prediction in modern information systems. *Information Systems Journal*, 2024.

[38] D. Wadden et al. Fact or fiction: verifying scientific claims. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*:7534–7550, 2020.

[39] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal. Fever: a large-scale dataset for fact extraction and verification. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*:809–819, 2018.

[40] Y. Liu et al. Roberta: a robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[41] I. Beltagy, K. Lo, and A. Cohan. Scibert: a pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*, 2019.

[42] S. Gururangan et al. Don't stop pretraining: adapt language models to domains and tasks. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*:8342–8360, 2020.

[43] A. Vlachos and S. Riedel. Fact checking: task definition and dataset construction. *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*:18–22, 2014.

[44] E. Mohr et al. Claim detection and verification from social media posts. *Proceedings of the 2022 International Conference on Social Media and Society*, 2022.

[45] T. Diggelmann et al. Climate-fever: a dataset for verification of climate claims. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

[46] M. Sarrouti et al. Healthver: a dataset for scientific claim verification. *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*:14–20, 2021.

[47] N. Kotonya and F. Toni. Explainable automated fact-checking for public health claims. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

[48] C. Malaviya et al. Scientific claim verification using pretrained language models. *Journal of Computational Linguistics*, 2023.

[49] J. Lee et al. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2019.

[50] C. Raffel et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[51] P. He et al. Deberta: decoding-enhanced bert with disentangled attention. *Proceedings of the 2021 International Conference on Learning Representations (ICLR)*, 2021.

[52] National Center for Biotechnology Information. PubMed, 2024. URL: `https://pubmed.ncbi.nlm.nih.gov/`. Accessed: 2024-07-07.

[53] S. Robertson, H. Zaragoza, et al. The probabilistic relevance framework: bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

[54] Y. A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836, 2018.

[55] S. Hofstätter, S.-C. Lin, J.-H. Yang, J. Lin, and A. Hanbury. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 113–122, 2021.

[56] X. Jiang et al. Mistral-7b: a highly efficient and accurate generative model. *Journal of AI Research*, 34(5):123–145, 2023.

[57] Anakin.ai. Updates in mistral-7b v0.2: enhanced context window and improved performance, 2024. Retrieved from `https://anakin.ai/mistral-v0.2`.

[58] D. Jin, B. Dhingra, W. W. Cohen, and X. Lu. Pubmedqa: a dataset for biomedical research question answering. In *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, 2019.

[59] W. Chiang et al. Chatbot arena: an open platform for crowdsourced llm evaluation, 2024. Retrieved from `https://chatbotarena.com/leaderboard`.

[60] A. Höfer and M. Mottahedin. Minanto at semeval-2023 task 2: fine-tuning xlm-roberta for named entity recognition on english data. In *Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023)*, pages 1127–1130, 2023.

[61] D. Wadden, S. Lin, K. Lo, L. L. Wang, M. van Zuylen, A. Cohan, and H. Hajishirzi. Fact or fiction: verifying scientific claims. *arXiv preprint arXiv:2004.14974*, 2020.

[62] Z. Zhang. Improved adam optimizer for deep neural networks. In *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)*, pages 1–2. Ieee, 2018.

[63] BioASQ team. Bioasq12 challenge dataset, 2024. URL: `http://participants-area.bioasq.org/datasets/`. [Accessed 26 April 2024].

[64] B. Bašaragin, A. Ljajić, D. Medvecki, L. Cassano, M. Košprdić, and N. Milošević. How do you know that? teaching generative language models to reference answers to biomedical questions. *arXiv preprint arXiv:2407.05015*, 2024.

[65] M. Sarrouti, A. B. Abacha, Y. M'rabet, and D. Demner-Fushman. Evidence-based fact-checking of health-related claims. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3499–3512, 2021.