Leipzig University
Institute of Computer Science
Degree Programme Data Science M.Sc.

Implicit Evaluation of Health Answers from Large Language Models

Master's Thesis

Jonas Probst

1. Referee: Prof. Dr. Martin Potthast
2. Referee: Dr. Harrisen Scells

Submission date: January 16, 2024

Declaration

Abstract

With the release of ChatGPT, open-ended generation of text became the biggest use case of Large Language Models (LLMs). Meanwhile, LLM evaluation focuses on classical NLP tasks like single-choice question answering or text classification, which do not represent the LLMs' capabilities in long-form question answering (LFQA). The lack of evaluation of open-ended questions is especially concerning in the medical domain, as answers that are misleading or wrong could have significant impact on the users personal health. Using human experts to compare the quality of generated answers is considered the gold standard in LFQA, but hiring experts leads to high costs and slower evaluation procedures, while also introducing subjectivity to the evaluations. In this thesis, we present a rank-based implicit evaluation method for LFQA, aiming to make the evaluation process more scalable. Using a dataset of queries and associated documents, which were assessed by human annotators, we first compare multiple retrieval methods for retrieving documents that are relevant, readable, and credible. We then use the most effective retrieval method to rank new answers generated by LLMs against the web answers from the dataset. Because the retrieval method is previously evaluated to produce rankings that agree with human annotator preferences, we can deduce that highly ranked LLM generated answers are more likely to be relevant, readable, and credible. We call this processes rank-based implicit evaluation and our findings demonstrate that the proposed evaluation approach ranks the effectiveness of various LLMs in a similar order as other academic benchmarks like ARC or MMLU. Additionally, we show that the LLM answer quality improves with model size and with more sophisticated prompting strategies, which aligns with similar trends observed in the literature. Because the most effective model in our research (ChatGPT) already ranks best on nearly every query, we encourage future research to produce a more challenging dataset, enabling the comparison of more advanced models. More work in further validating the proposed rank-based implicit evaluation method is also needed, by evaluating whether human experts prefer the highest-ranked LLM-generated answers over the highly ranked web answers.

Contents

Chapter 1

Introduction

After the release of ChatGPT[1] by OpenAI in 2022 several other LLM based chatbots like Anthropic's Claude[2] or Google's Bard[3] were published in the following months. ChatGPT became successful quickly, gaining 100 million active users in the first year after release, according to the OpenAI DevDay Opening Keynote.[4] De Choudhury et al. (2014) showed that a large amount of users turned to search engines or social media sites for medical information, we can assume that a large portion of users turned to ChatGPT and other chatbots with similar medical questions. This assumption is supported by Dave et al. (2023) and Khan et al. (2023) who discuss possible use cases of ChatGPT in the medical domain and mention the use of ChatGPT by doctors, nurses or medical students as possible applications. Koopman and Zuccon (2023) discuss the use of ChatGPT for answering medical questions and emphasize the impact of the bias that is already present in the users question on answer quality. The authors note that if ChatGPT is provided with supplementary material, e.g. from a web search, it heavily relies on the information provided in the material. In their experiments, Koopman and Zuccon (2023) show that this reliance on supplementary material generally leads to worse answers. By showing that ChatGPT often tries to confirm the users bias, Koopman and Zuccon (2023) highlight the importance of evaluating the quality of answers generated by ChatGPT and similar LLMs on a large scale, especially in the medical domain.

While multiple benchmarks for answering questions in a single choice format or with short, factual answers are usually reported in the technical reports accompanying the release of new models, as Xu et al. (2023b) show it is much

[1]`https://chat.openai.com/`, accessed on 04.01.2024
[2]`https://claude.ai/`, accessed on 04.01.2024
[3]`https://bard.google.com/chat`, accessed on 04.01.2024
[4]`https://www.youtube.com/watch?v=U9mJuUkhUzk`, accessed on 04.01.2024

harder to evaluate the LLMs on open-ended questions that are common in the medical domain due to the many factors that influence health outcomes.

As Ouyang et al. (2022) show, the current NLP datasets and benchmarks that are usually applied to evaluate LLMs do not reflect the described use case of answering open-ended questions. According to the data by Ouyang et al. (2022), only 18% of GPT-3 API calls are targeted at conventional NLP tasks like classification or single choice QA tasks. On the other hand, 57% of API requests lead to open-ended generation. Considering that this data originates from the GPT-3 API and not the conversation-focused ChatGPT model, it is reasonable to assume that open-ended generation now makes up even more of the requests. The large amount of open-ended generation shows that traditional QA tasks like single or multiple choice QA or extracting answers from a given text are not representative of the wide range of tasks supported by LLMs.

The relatively new field of long-form question answering (LFQA) deals with the answering of open-ended questions by automated systems. As shown by Xu et al. (2023b), the current automated methods of evaluating those LFQA systems are still lacking compared to manual human evaluation, which according to Xu et al. (2023b) is often performed by crowd workers. However, Xu et al. (2023b) also mention multiple drawbacks of the human evaluation approach. For one, annotators need to be well-trained and have a solid foundational knowledge of the question field. This level of expertise is usually lacking among crowd workers. Krishna et al. (2021) note that the answer length makes evaluating LFQA much more demanding than simple QA tasks, increasing the duration it takes to process one sample.

Krishna et al. (2021) demonstrate a high rate of disagreement among annotators when choosing between two given answers to the same question. They attribute the high rate of disagreement to the difficulty of judging answer quality, due to the many factors that make up a good answer. The subjectivity of the evaluation also poses problems for creating more widely used LFQA benchmarks based on human evaluation. Even if the questions for all benchmarked models are the same, the subjectivity introduced by human annotators between the different models renders the resulting scores hard to compare, assuming not every question is annotated by the same crowd worker for each LLM.

In addition to making the evaluation of correctness more difficult, open-ended generation of text also necessitates that the quality of the answers is evaluated in a multidimensional manner. Sakai (2023) have proposed a framework for evaluating the quality of conversational systems, including criteria like fluency (text sounds natural), groundedness (answer is grounded on evidence), explainability (user can understand why the system gave a certain answer) and many more. According to Sakai (2023), evaluating most of those criteria in an automated manner is still an open research question and still requires human evaluation. Similar to conversational systems and in contrast to other QA tasks, LFQA has no simple notion of a good answer and requires multiple evaluation criteria. Current evaluation methods like accuracy for single choice question answering or the overlap between the extracted answer and the ground truth for extractive question answering are not able to capture the multidimensional nature of answer quality. To reduce human evaluation costs and enable large-scale, multidimensional evaluation and comparison of chatbots, automated methods are necessary.

In this thesis, we propose a new evaluation method for LFQA based on information retrieval techniques. Using a dataset based on health questions from Goeuriot et al. (2021), we construct a benchmark consisting of multiple queries and human-generated answers based on web content. We then use retrieval methods to rank answers generated by different LLMs against those human-generated web documents. Based on the rank of the generated answer, we can compare the quality of the answers by different LLMs. This method captures the multidimensional nature of answer quality, assuming the retrieval method is effectively ranking documents in terms of relevance, readability and credibility. Evaluating the multidimensional retrieval effectiveness of different retrieval methods is therefore a key part of this thesis.

Our method allows for large-scale evaluation of LLMs, including the evaluation of different prompting strategies, the assessment of answer consistency across a model, and the analysis of how answer quality varies with the number of model parameters. The reduction in human evaluators would consequently lower the costs associated with developing and fine-tuning LLMs for LFQA tasks. It would also lead to more consistency in evaluating generated answers, limiting the subjective component brought in by human annotators.

In order to evaluate the proposed rank-based implicit evaluation method, we formulate one overarching research question:

Is rank-based implicit evaluation with human-written web content a viable approach for evaluating health answers generated by LLMs?

To investigate the main research question, we formulate two supporting research questions:

- **RQ1:** Which factors influence the effectiveness of LLMs when using rank-based implicit evaluation?
- **RQ2:** How does the effectiveness of LLMs on existing benchmarks compare to their effectiveness when using rank-based implicit evaluation?

To answer **RQ1**, we look at differences in model size and prompting strategy, which have previously been shown to influence the effectiveness of LLMs on other tasks. We also investigate the influence of query properties by investigating how LLM generated answers are ranked on queries that are phrased as questions versus queries that are a collection of keywords. Additionally, we investigate different properties of the generated answers, like their length or the general structure of the answer.

For **RQ2**, we take the mean ranking scores of all answers generated by each LLM and compare those to how the LLMs rank on other benchmarks. Those benchmarks are ARC (Clark et al. (2018)), HellaSwag (Zellers et al. (2019)) and MMLU (Hendrycks et al. (2020)). We investigate if the LLMs rank similarly on our benchmark as they do on the other benchmarks.

The rest of this thesis is structured as follows. In Chapter 2 we delve into the related work. First, the general field of evaluating LLMs in the context of question answering is introduced. This includes an overview of different question answering tasks and their evaluation methods. Secondly, the different retrieval methods used in this work are presented, alongside the evaluation metrics used to compare them. This chapter is concluded with a discussion on how retrieval methods have previously been used to evaluate NLP tasks. Chapter 3 describes the experimental setting, from dataset collection and preparation over development of the different retrieval pipelines and generation of LLM responses to the questions in the dataset. The experimental results are presented in Chapter 4, aiming to provide the necessary groundwork for discussing the research questions in the following Chapter 5. We conclude this thesis with Chapter 6 on the outlook on future work and a conclusion.

Chapter 2

Related Work

The work presented in this thesis is based on two main areas of research: the evaluation of Large Language Models (LLMs) and Information Retrieval (IR). In this chapter, the current state of research in those areas is presented, as it relates to this thesis. We start with a short introduction to LLMs, followed by a discussion of the current evaluation methods for these models, especially in the field of question-answering tasks. Next, the field of IR is introduced. Different retrieval methods used in this thesis are presented and why they were chosen for this work. Additionally, the evaluation metrics used to compare those retrieval methods are introduced.

2.1 Evaluation of LLMs for Question Answering

While general-purpose NLP models such as the LSTM-based ELMo (Peters et al. (2018)) or the static word embeddings based statistical co-occurrences model GloVe (Pennington et al. (2014)) were already popular for many NLP tasks, the introduction of the transformer architecture by Vaswani et al. (2017) led to a new generation of LLMs. The transformer architecture allowed models like BERT (Devlin et al. (2018)) and GPT-2 (Radford et al. (2019)) to process more context than previous models, which led to improved effectiveness of the transformer-based models compared to LSTM or static word embedding based models. This improved effectiveness on many NLP tasks over methods with other architectures was shown by Radford et al. (2019), who compare the base GPT effectiveness against multiple then state-of-the-art models on different tasks, showing that their model was at least as effective as state-of-the-art models on most of those tasks.

With the release of GPT-3 (Brown et al. (2020)), the size of datasets used to train LLMs, as well as the number of parameters in the models increased significantly. While GPT-2 in its largest version has a total of 1.5 billion parameters, GPT-3 has 175 billion parameters. As Wei et al. (2022) show, this scale not only leads to improvements over previously used benchmarks compared to smaller models but also to what they call *emergent abilities*. *Emergent abilities* are abilities that are not present in smaller models. Those include generating long coherent stories or poems, translating between languages, and answering long-form questions. None of those capabilities are present in smaller models, with, e.g., the largest version of GPT-2 not being effective on translation and summarization tasks, as shown in the original paper (Radford et al. (2019)). As the new capabilities of LLMs are emerging, general evaluation methods for those tasks are yet to be established.

In this section, different question-answering (QA) tasks like extractive QA, single/multiple choice QA, and LFQA are introduced, and their evaluation methods are discussed. What extractive QA and single/multiple choice QA have in common is that they are relatively easy to evaluate. When evaluating extractive QA, the overlap of the predicted tokens with the answer span can be calculated, and for single-choice QA the predicted answer option can be compared to the ground truth answer.

The difficulty of evaluation changes when evaluating LFQA. Here, models are evaluated on their ability to answer questions in a free-form manner, without constraining the length of the answer. Answers can get long and complex, branching out in different directions by including examples or other information. The evaluation of such answers is not as straightforward as for the other versions of QA, so human evaluation is currently the only option.

2.1.1 Extractive Question Answering

Extractive QA is the task of answering questions given a context containing the answer. In this context, which can be a short paragraph or an entire Wikipedia article, the correct answer span has to be selected by the model.

One of the most popular datasets for evaluating LLMs in this task is SQuAD (Rajpurkar et al. (2016)), and its successor SQuAD 2.0 (Rajpurkar et al. (2018)), which includes unanswerable questions in which the correct answer is not present in the context, e.g., “What are the names of Donald Duck’s nephews?” in the context of the paragraph in Figure 2.1. Adding unanswerable questions to the dataset is a way to test the model’s ability to detect when a question can not be answered by the given context. Many other datasets like NarrativeQA (Kočiský et al. (2018)), QuAC (Choi et al. (2018)) or Natural Questions (Kwiatkowski et al. (2019)) are based on the same principle. They consist of questions written by crowd workers or experts in the field, based on a Wikipedia article snippet or similar text passages. The exact constraints on the questions and the context vary between the datasets, but the general idea

Context:
Donald Duck is a cartoon character created in 1934 by Walt Disney Productions. He is an anthropomorphic white duck with a yellow-orange bill, legs, and feet. He typically wears a **sailor suit with a bow tie and a hat**.

Question:
In which outfit is Donald Duck typically portrayed?

Answer:
sailor suit with a bow tie and a hat

Figure 2.1: Example of a typical extractive question answering task: The answer span "sailor suit with a bow tie and a hat" is highlighted in the paragraph.

is the same. Figure 2.1 shows a made-up example of a question and context that could be used in such a dataset.

The evaluation metrics for tasks of this category are based on the overlap between the predicted answer span and the ground truth answer span. The score is called the exact match (EM) score, which measures the percentage of exact matches between the predicted and the ground truth answer span. Alternatively, the F1 score as the harmonic mean of precision and recall can be used, with precision being defined as

$$\text{precision} = \frac{\text{number of correct tokens in prediction}}{\text{total number of tokens in the prediction}}$$

and recall as

$$\text{recall} = \frac{\text{number of correct tokens in prediction}}{\text{total number of tokens in the ground truth}}$$

with a correct token being a predicted token that overlaps with the ground truth answer. The F1 score is then calculated as

$$\text{F1} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}.$$

Encoder-only LLMs like BERT have to be specifically fine-tuned for the task of extractive QA. For each token in the provided context, the model

Context:
Donald Duck is a cartoon character created in 1934 by Walt Disney Productions. He is an anthropomorphic white duck with a yellow-orange bill, legs, and feet. He typically wears a sailor suit with a bow tie and a hat.

Question:
What does Donald Duck typically wear?

(A) T-shirt and jeans
(B) Sailor suit
(C) Business suit
(D) Basketball jersey

Answer:
B

Figure 2.2: Example of a single-choice question with context: The correct option is highlighted. The B would be a possible answer generated by the LLM, after being primed with the context and the question.

assigns a probability of the token being the starting or ending token of the answer span (Devlin et al. (2018)).

Decoder-only models like GPT-3 can directly generate the answer from the context and the questions without any fine-tuning, by using the zero-shot, single-shot, or multi-shot capabilities of the model (Brown et al. (2020)). In some settings of the datasets, the context can be completely omitted, forcing the model to directly answer the question. The different approaches to answering the question mean that the results of the two approaches are not directly comparable, because even though the models were evaluated on the same dataset, the approaches are fundamentally different.

2.1.2 Single and Multiple Choice Question Answering

For single and multiple choice question answering, the model has to select the correct answer from a set of possible answers, which can be done with or without context. While most datasets are single-choice datasets, some datasets like

MultiRC by Khashabi et al. (2018) are multiple-choice so that the model has to check each answer for correctness, instead of just selecting the best fitting one. Questions for single and multiple choice QA often stem from official exams, like the MMLU dataset (Hendrycks et al. (2020)), which combines questions from many exams like the United States Medical Licensing Examination or the Examination for Professional Practice in Psychology. In other datasets, the questions are collected from crowd workers and verified by experts (Clark et al. (2018), Mihaylov et al. (2018)).

Figure 2.2 shows an example of a single-choice question with context. The LLM is provided with the context, the question and the answer options, and a "Answer: " prefix. Based on the context, the model has to select one of the given options. Evaluation is straightforward in this case, the model's generated answer option is compared to the ground truth answer, and accuracy over all questions is calculated. For encoder-only models like BERT, the task is modeled as a classification task, where the model has to classify the correct answer option from the given options after fine-tuning on the dataset as described by Devlin et al. (2018).

2.1.3 Long Form Question Answering

LFQA refers to the task of answering open-ended questions, which can usually not be answered by simply providing one entity or number, but require an in-depth answer. With LLMs being deployed in chatbots and as such expected to deliver answers mostly without context, evaluating the models on LFQA is important.

So far, only a handful of datasets are available for this task, with the first dataset in this category being the ELI5 dataset (Fan et al. (2019)). It consists of questions and the corresponding highest-voted answer from the "Explain Like I'm Five" subreddit, where users ask questions about complex topics, which are then answered by other users. They are accompanied by support documents, which are retrieved from web sources by querying for the original question. This dataset was used by Nakano et al. (2021) to fine-tune GPT-3 for the task of LFQA without using the context documents. The answers given by the fine-tuned model are evaluated by humans, by comparing them to the highest voted answer from the ELI5 dataset.

Singhal et al. (2023) use a subset of questions from the multiple-choice dataset MultiMedQA by Singhal et al. (2022) to create a dataset for LFQA. This dataset consists of about 1200 questions, each of which is accompanied by an answer written by a physician. To evaluate LLM generated answers on the dataset, the authors let physicians and lay people do pairwise rankings of LLM generated answers versus physician-written answers. Additionally, the

Question:
Can you describe the evolution of Donald Duck's character design and attire over the years, focusing on any major changes and the reasons behind them?

Answer:
The evolution of Donald Duck's character design and attire has seen several notable changes since his creation by Walt Disney and Dick Lundy in 1934. Here is an overview of the major changes and the reasons behind them:

$$\left[\vdots\right]$$

Figure 2.3: Example of LFQA without Context: The question prompts for a detailed answer about Donald Duck's character evolution and attire. The model is prompted with the question, and the sample answer (generated by ChatGPT-3.5) is shown below in italics. The generated answer continues for multiple paragraphs.

answers were individually rated in multiple rubrics, introduced in a previous work (Singhal et al. (2022)).

None of the papers for current, main-stream LLMs like GPT-3 (Brown et al. (2020)), GPT-4 (OpenAI (2023)) or Llama 2 (Touvron et al. (2023)) include evaluations on common benchmarks for this category. The few available datasets and the lack of evaluation on those in main stream papers shows that LFQA is still a relatively new task, lacking sufficient academic evaluation measures.

2.1.4 Difficulties of Long Form Question Answering

Recent works by Xu et al. (2023b) and Krishna et al. (2021) have shown multiple challenges in the task of LFQA, independently of the model architecture used to answer the questions. Since the answers are free-form text, and not just a multiple choice option, number, or entity, the quality of the model can not be measured using accuracy or similar metrics that require static ground truth information. Multiple evaluation dimensions are of interest in this thesis:

- **Relevance:** Are the most important aspects of the query answered?
- **Readability:** How easy is the answer to read and understand?
- **Credibility:** Are there any references provided and are they of high quality?

Multiple evaluation dimensions add additional complexity to the evaluation process, compared to other QA tasks which only measure the correctness of the answer.

Xu et al. (2023b) survey the evaluation of LFQA, comparing different automatic evaluation methods to human judgment. They differentiate between general-purpose generation evaluation metrics, which were originally designed for other NLP tasks like summarization or translation, and LFQA-specific metrics. The following types of metrics are considered general-purpose metrics by Xu et al. (2023b):

- **Answer-reference metrics:** Include metrics like ROUGE or BERTScore. These metrics compare generated answers to reference answers, focusing on aspects like lexical overlap and semantic similarity.
- **Answer-only metrics:** Such as Self-BLEU which measures the fluency and diversity of generated text. These are intrinsic metrics of the generated text and do not need a reference answer for evaluation.
- **Question-answer metrics:** Score answers only based on the question without any reference answer. This includes BARTScore by Yuan et al. (2021), which generates a score given the question and the answer.
- **Answer-evidence metrics:** Judge the given answer by the evidence from documents used to generate it. This method indirectly assesses the answer's credibility and factual accuracy by comparing the answer to the evidence documents.

In addition to these metrics, Xu et al. (2023b) survey two different versions of LFQA-specific metrics. The first one is based on Longformer (Beltagy et al. (2020)), in which the model is fine-tuned to produce a score given a question and an answer, optionally combined with evidence documents. The second model is a fine-tuned version of GPT-3, which is trained to output either *Answer1* or *Answer2* given a question and two answer options. Both fine-tuned models are trained on the dataset created by Nakano et al. (2021), which contains human preference labels for different answer pairs.

All automated evaluation methods for LFQA are judged on the task of choosing the preferable answer given two long-form answers to a question. The results are compared to previous human judgment on the same task. Xu et al. (2023b) find that one of their baseline models, choosing always the longest answer, imitates human judgments almost as well as the fine-tuned GPT-3 model, which comes closest to human judgments. All automated evaluation methods have lower agreement with human judgment than the human annotators have among each other, calculated by pairwise agreement. The low agreement shows that the current automated evaluation methods are not yet able to evaluate LFQA as well as human annotators.

Krishna et al. (2021) investigate the problems of reference-based evaluation metrics like ROUGE-L. They highlight that those metrics are unable to capture answer components like examples if those examples are not present in the ground truth answers.

Furthermore, they note that even human evaluation is limited in judging LFQA over different models. Some problems include the hiring process of experts, especially when datasets tackle multiple fields of expertise. Finding experts of similar education and background is challenging when doing evaluations of different models over time, especially for challenging fields like the health domain, as in the dataset used in this thesis. Additionally, the evaluation process is more mentally demanding for the individual annotator the longer the answers get. Similar problems have previously been shown by Akoury et al. (2020) in the context of machine-generated stories. They find that crowd workers have low Fleiss' kappa agreement for different dimensions like fluency and coherence when evaluating the same stories. They tackle this low agreement problem by using gamification techniques to activate online users of a story-writing platform to evaluate and improve the generated stories, motivating annotators to stay engaged for longer periods of time. A similar approach is taken by Dugan et al. (2020), who implement a website where users try to differentiate machine-generated text from human-generated text.

We attempt to tackle some issues with evaluating LFQA discussed in this section using the approach presented in this thesis.

2.2 Retrieval Models

Since we want to use retrieval methods to indirectly evaluate the effectiveness of LLMs in LFQA, we will give some background on the field of information retrieval (IR), and how it relates to LFQA. IR is concerned with retrieving relevant information based on an information need, from a collection of documents, usually in the form of whole documents, passages, or single sentences.

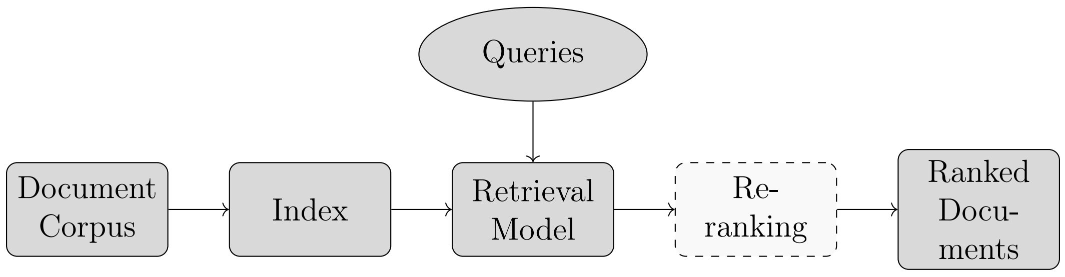


Figure 2.4: Depiction of a basic retrieval pipeline, with optional re-ranking step. The process begins with a document corpus, which is indexed to facilitate efficient retrieval. Given the information needed by a user in the form of a query, the retrieval model returns the most relevant documents. Optionally, the top-k documents can be re-ranked using a re-ranking model which is usually more resource-intensive.

A brief overview of a typical retrieval pipeline as outlined by Manning (2009) is provided and visualized in Figure 2.4. Given a query, an IR system returns a ranked list of documents that are most relevant to the query. To achieve this, IR systems estimate a relevance score for each document in the collection with respect to the query. The documents are then ranked according to their relevance scores, with the most relevant documents appearing at the top of the list. Some pipelines include a re-ranking step, in which the top-ranked documents are re-ranked after the first retrieval using a more expensive (and ideally more effective) retrieval model.

2.2.1 Baseline Retrieval Models

First, we examine the basic retrieval models, which are used as baselines in this thesis. They have been chosen because they were previously shown to be effective on the dataset used in this thesis (Goeuriot et al. (2021)). Both models use the implementation in the Terrier IR platform (Ounis et al. (2005)).

TF-IDF

Term Frequency-Inverse Document Frequency (tf-idf) is one of the most commonly used models in information retrieval. It measures the importance of a term in a document relative to a collection of documents or corpus. The central intuition is that terms that appear frequently in a document but not in many other documents in the corpus are important for the distinguishing the content and thus should be given higher weight.

Given a query q consisting of terms $t_1, t_2, \cdots, t_n$ and a document corpus D of size N, we calculate the score of q given document d by first calculating the tf-idf of each term in q given d and then summing them up:

$$\text{tf-idf}(t,d) = \text{tf}(t,d) \times \text{idf}(t),$$

where $\text{tf}(t,d)$ is the frequency of term t in document d and

$$\text{idf}(t) = \log\left(\frac{N}{1+\text{df}(t)}\right),$$

where document frequency $\text{df}(t)$ is the number of documents containing term t. The final score of the query given the document is then calculated as

$$\text{score}(q,d) = \sum_{t \in q} \text{tf-idf}(t,d).$$

For retrieval given a query q tf-idf scores are calculated for all documents, which are then ranked according to their score.

In the Terrier IR platform, the implementation of tf-idf uses variants of the tf and idf components. For Term Frequency, Robertson's tf formulation (Robertson (2004)) is used, which incorporates an additional parameter that adds a saturation effect to the term frequency. For idf, the original formulation by Sparck Jones (Sparck Jones (1972)) is applied.

DPH

Divergence from Randomness (DFR) is a framework in IR that assigns term weights based on the divergence of the actual within-document term frequency distribution from a random term frequency distribution (Amati (2006)). The divergence from randomness using the hyper-geometric distribution model (DPH) is one of the models derived from the DFR framework.

The principle behind DFR is that terms that are informative in a document will have a distribution that deviates significantly from what would be expected if terms were distributed randomly. While TF-IDF emphasizes the importance of terms based on their frequency in a document and their inverse frequency in the corpus, DPH focuses on the divergence of a term's distribution from what would be expected under a random distribution. Specifically, DPH assesses the divergence using the hyper-geometric distribution. In essence, where TF-IDF weights terms based on their prominence and rarity, DPH weights them based on how much their occurrence pattern deviates from randomness. The implementation in the Terrier IR platform follows the original formulation by Amati (2006).

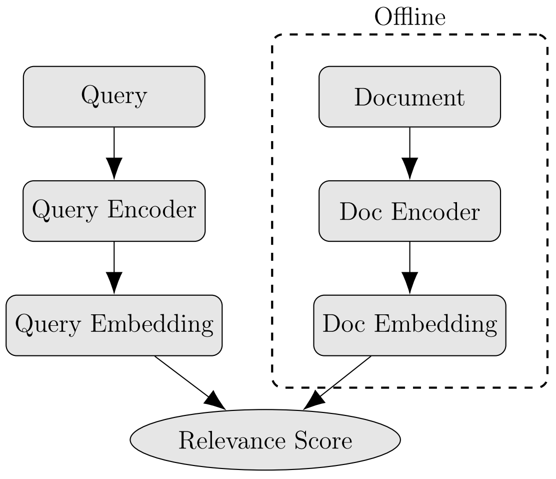


Figure 2.5: Bi-Encoder Architecture. Query and document are encoded separately, document embeddings can be calculated offline. Then the relevance is calculated as the similarity between the embeddings.

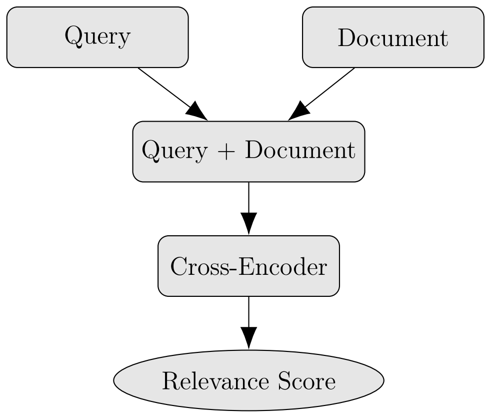


Figure 2.6: Cross-Encoder Architecture. Query and document are first concatenated and then fed to a single encoder model, the relevance is calculated by a linear layer on top of the encoder.

2.2.2 Transformer-based Retrieval Models

As mentioned in earlier sections, transformer-based models have been shown to be effective on a variety of tasks, including IR. There are different approaches to how transformers can be used for retrieval. A brief overview of methods used in this thesis is given here. Transformer models can be used for learning-to-rank, where the model is trained on a dataset to predict which documents are relevant to a query. This training step differentiates them from our baseline models, which do not require training.

Two of the most common transformer-based architectures are cross-encoder and bi-encoder models, which are shown in Figure 2.5 and Figure 2.6 respectively. Bi-encoders independently embed queries and documents using transformer models like BERT (Devlin et al. (2018)). For the document collection, embedding can be done offline since the embeddings are independent of the query. This separation allows them to efficiently process large datasets as the embeddings for the documents can be pre-computed and stored. It also enables efficient retrieval, since at inference time only the embedding for the current query has to be calculated and compared to the pre-computed document embeddings. The documents are then ranked according to their embedding similarity to the query.

Cross-encoders, on the other hand, take a concatenated input sequence of both query and document and produce a scalar relevance score. This joint modeling enables them to better capture the interaction between a query and a document by leveraging the transformer's attention mechanism. Due to their fine-grained interaction modeling, cross-encoders are more effective than bi-encoders in terms of precision in retrieving relevant documents as shown by Thakur et al. (2020) and Rosa et al. (2022). However, the need to process each query-document pair individually makes them computationally demanding, especially for large datasets. As a result, they are typically only used in second-stage retrieval, where the objective is to re-rank the top results obtained from an initial retrieval method.

The following sections describe the different transformer-based retrieval models used in this thesis.

monoT5 + duoT5

Roberts et al. (2019) introduce monoT5 and duoT5, which are based on previous work by Nogueira et al. (2019), who introduced monoBERT and duoBERT, first applying transformer architecture to the task of document ranking. The general idea is to first use a baseline retrieval model like BM25 to retrieve an initial set of relevant documents. Then, pairs of the query and each document in the initial set are concatenated and fed into the mono version of the transformer model, which produces a scalar score for each query-document pair. The documents with the highest scores are then fed into the duo model, which takes the query and each possible pair of documents as input, outputting a probability of one document being more relevant than the other. Given those scores, the documents are re-ranked another time, serving as the final output of the retrieval pipeline. The main difference between the BERT and T5 versions is that for BERT the [CLS] token can be used as input to a single-layer neural network to output a probability of the document being relevant. Since there is no [CLS] token for models in the T5 family, as they are sequence-to-sequence models, this part is done using an input template:

$$\text{Query: } q \text{ Document: } d \text{ Relevant:} \tag{2.1}$$

where the model is fine-tuned to produce the token *true* or *false* given query q and document d.

At inference, softmax is applied to the *true* and *false* tokens only, the scores are then calculated using the probability of the *true* token. This works analogously for the Duo version of both models, by just adding the second document. Both monoT5 and duoT5 belong to the family of cross-encoder models, sharing the general characteristics of those models.

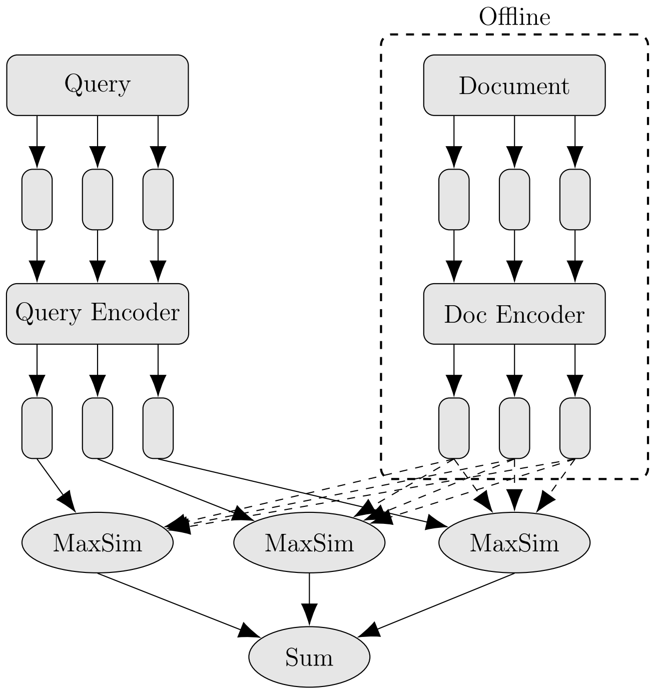


Figure 2.7: ColBERT Retrieval Model. The query and document are processed separately to generate token embeddings. The maximum cosine similarity between all token embeddings of the query and document is calculated and summed over all query tokens, which is the relevance score of the document given the query.

This model architecture allows for precisely tuning retrieval efficiency vs. effectiveness, by parameterizing how many documents are filtered in each step, making the model suitable for different use cases.

ColBERT

ColBERT (Khattab and Zaharia (2020)) is another BERT-based model for document ranking, visualized in Figure 2.7. It shares similarities with the general bi-encoder architecture but introduces some modifications to improve effectiveness. Instead of representing each document and query as a single vector (e.g. the [CLS] token), ColBERT uses the contextualized embeddings for each token in the document or query. For the documents, embedding can again be done offline, so that at the retrieval stage, only the embeddings for the query have to be generated and compared to the documents. The relevance score for document d given query q is estimated using their late interaction model, in which maximum cosine similarity between all query term embeddings and document term embeddings is calculated, and then summed over for all query terms. This architecture combines the advantages of bi-encoders and cross-encoders, as it is computationally efficient using the offline embeddings for the documents, while also capturing the interaction between query and document using the contextualized embeddings similar to cross-encoders.

The ColBERT architecture can be used for re-ranking or end-to-end retrieval. In the second case, an additional step is added in which the complete collection is filtered for relevant documents using similarity search to find documents that contain similar terms as the query. In the second step, the remaining documents are re-ranked using the maximum similarity metrics.

ColBERTv2

ColBERTv2 (Santhanam et al., 2021) is directly based on the late-interaction architecture of ColBERT but adds improvements to the architecture and the training process.

To improve the training process, a new process of generating hard negatives is applied, in which a cross-encoder model is used to first rank passages given a query. From the ranked passages a highly ranked one is selected as the positive passage and a low ranked one as the negative passage.

Improvements to the architecture are made by incorporating a residual compression technique. While for ColBERTv1 each token embedding is saved separately in its original state, ColBERTv2 represents each token embedding as its nearest neighboring centroid embedding and a residual. The centroids are calculated using k-means clustering on the token embeddings of a sample of all passages at indexing time. Clustering allows for a more compact representation of the token embeddings since only the centroid embedding has to be stored.

As shown by the authors, ColBERTv2 is more effective than ColBERTv1 on all evaluated datasets, while also being more efficient

2.2.3 Evaluation of Retrieval Models

Evaluation metrics are important for the automatic evaluation of retrieval model effectiveness. They provide a quantitative measure of how well a system retrieves relevant documents in response to a user's query. To be able to evaluate different retrieval methods, relevance assessments for document-query pairs have to be defined by human assessors.

There are multiple metrics which can be used in information retrieval tasks:

- **Precision**: Captures the fraction of retrieved documents that are relevant.
- **Recall**: Captures the fraction of relevant documents that are retrieved.
- **F1 Score**: The harmonic mean of precision and recall.

- **Mean Average Precision (MAP)**: The arithmetic mean of each query's average precision at each document position in a ranking.
- **Normalized Discounted Cumulative Gain (nDCG)**: Considers both the ranking and the relevance grade of retrieved documents, weighing highly relevant documents higher than less relevant ones, assigning a higher information gain for documents that are ranked higher.

Given the context of the dataset used in this thesis, which contains multiple queries with documents assessed for relevance between 0 and 3, the nDCG metric is the most suitable one.

The formula for nDCG@k is as follows is based on the concept of cumulative gain (CG), which is the sum of the relevance ratings of all documents up until position k:

$$\text{CG@k} = \sum_{i=1}^{k} r(d_i, q),$$

where $r(d_i, q)$ is the relevance rating of document d_i for query q. Based on that, the discounted cumulative gain (DCG) is defined as

$$\text{DCG@k} = \sum_{i=1}^{k} \frac{r(d_i, q)}{\log_2(i+1)},$$

where the relevance ratings are discounted by the logarithm of the rank of the document, to weight higher-ranked documents more than lower-ranked ones. The ideal DCG (IDCG) is the DCG of the ideal ranking, which is the ranking of documents sorted descending by their relevance rating. This ranking produces the highest possible DCG for a set of retrieved documents. Finally, the normalized DCG is calculated as

$$\text{nDCG@k} = \frac{\text{DCG@k}}{\text{IDCG@k}}$$

which normalizes the DCG by the IDCG, producing a score between 0 and 1. The cutoff value of k is set depending on how many of the ranked documents should be considered for each query.

2.2.4 Using Retrieval Models for Evaluating NLP tasks

When evaluating Natural Language Processing (NLP) tasks, IR techniques have been mainly used to evaluate machine translation systems. Since both question answering and machine translation are sequence-to-sequence tasks with multiple possible correct answers, there are similarities in the evaluation setup.

For machine translation, a model is given a source sentence and has to produce a target sentence in another language. Those systems are usually evaluated given a ground truth reference sentence, which is compared to the predicted sentence. Traditionally, methods like BLEU (Papineni et al. (2002)), ROUGE (Lin (2004)), or METEOR (Banerjee and Lavie (2005)) are used to compare the predicted sentence to one or multiple reference sentences. There are multiple versions of those metrics with slightly different parameters, which mostly differ in what sequences of tokens are compared (1-gram, 2-gram, 3-gram, 4-gram, longest common sub-sequence), how precision and recall are weighted, and how the results are smoothed. In the end, all of those metrics produce a similarity score between the predicted and the reference sentence, which can be used to compare different models.

Alternative approaches incorporating ranking methods have been proposed as well. Duh (2008) argues that, considering the final goal is to compare different translation systems, a direct comparison between the systems is preferable to evaluating their quality individually. He shows that ranking methods like RankSVM (Joachims (2002)) and RankBoost (Freund et al. (2003)) can be applied to rank different candidate translations against the reference, producing similar scores to the BLEU baseline on the same feature set. When incorporating intra-set features like the binary "highest BLEU score in set", which can not easily be incorporated into BLEU-like scores, the ranking methods achieve higher similarities to human rankings compared to BLEU and smoothed BLEU. Another approach based on learning-to-rank methods is proposed by Li et al. (2013). Again, they compare multiple translation candidates to a reference sentence, but here they use listwise learning-to-rank methods to generate a ranking of the candidates. The objective of listwise ranking is to train a ranking function that minimizes the loss between the predicted ranking and the ground truth human-generated ranking on a training set. Similar to Duh (2008), they show that the ranking-based methods correlate stronger with human judgment, compared to BLEU-like metrics which show lower correlation. Guzmán et al. (2019) use neural methods to incorporate syntactic and semantic information into the evaluation process. They train a pairwise ranking model, which compares two candidate sentences given the reference and returns which one is the better translation. Even though they are not more effective than other state-of-the-art methods in terms of agreeing with human judgments, they deliver competitive results while staying closer to the human evaluation framework.

This overview shows that information retrieval methods have successfully been applied to the task of evaluating machine translation systems. However, those approaches are hard to transfer to the task of evaluating from question-answering systems, since the evaluation metrics are not directly applicable. All ranking-based machine translation evaluation methods mentioned here compare the set of candidate translations to a single reference translation, trying to rank the candidate translations among each other. Since the space of correct answers given a long-form question is generally much higher in comparison to translating a sentence, this evaluation setup can not be directly applied here. So, even though information retrieval methods are applied here as well, the evaluation approach is formulated differently.

2.3 Summary

In this chapter, we presented an overview of the current state of research for evaluating LLMs for question answering. We highlighted that the evaluation of LFQA is challenging since the answers are free-form text and not just a single entity, number, or multiple choice option. Additionally, we show that evaluating long-form answers in one single dimension of correctness is not sufficient, as other aspects like readability and credibility are important as well. In this thesis, we tackle both of the mentioned challenges. Based on the assumption that if we develop a ranking model that can effectively rank human-generated documents in the dimensions of relevance, readability, and credibility in a way that is similar to human judgment, we can use the ranking model to evaluate LFQA systems. The evaluation of text generated by LLMs can be done by ranking the LLM output with the previously validated retrieval model and using the rank of the generated output alongside human-written texts as a proxy for the effectiveness of the generated answer. We formalize this approach as follows:

1. **Dataset acquisition:** Collect a dataset of queries $q_1, q_2, \ldots, q_n$ with associated human-generated documents for each query $d_{i,1}, d_{i,2}, \ldots d_{i,j} \forall i \in \{1, \ldots, n\}$. Each document has a relevance rating $r_{rel}(d_{i,j}, q_i)$, a readability rating $r_{read}(d_{i,j}, q_i)$ and a credibility rating $r_{cred}(d_{i,j}, q_i)$ for the query.

2. **Retrieval Model Evaluation:** Evaluate a set of retrieval models $\mathcal{M}$ on the dataset, using the nDCG metric for relevance, readability and credibility.

3. **Generate LLM Answers:** Use a set of LLMs $\mathcal{L}$ to generate answers $a_{l,i}$ for all queries q_i.

4. **Rank Answers:** Add the generated answers $a_{l,i}$ to the documents $d_{1,i}, d_{2,i}, \cdots, d_{n,i}$ for each query q_i and rank them using the best of the previously evaluated retrieval model from set $\mathcal{M}$.

This approach allows us to evaluate the capabilities of multiple LLMs in LFQA, by comparing the ranks of the generated answers as a proxy for their quality. By evaluating the retrieval models on multiple evaluation criteria, we tackle the problem of evaluating the answers in multiple dimensions.

Chapter 3

Experimental Setup

This chapter describes the experimental setup of this thesis, starting with the data collection and preparation process as well as an analysis of the resulting dataset. Afterward, we describe the retrieval pipeline development and how the pipelines are evaluated. Finally, the generation of LLM answers to the different queries is presented.

The goal of this thesis is to have a pipeline with which the output of different LLMs can be ranked, in comparison to human written text. To construct the pipeline like this we first need a dataset consisting of questions as well as human-generated documents that are annotated for relevance, readability, and credibility. Subsequently, a retrieval pipeline is developed, which can rank the documents to a given question according to human assessor preferences in the three dimensions. Finally, the answers generated by different LLMs can be ranked using the developed retrieval pipeline. Assuming the effectiveness of the retrieval pipeline in ranking the answers according to human preferences, we expect that the ranking of LLM-generated answers will closely align with the rankings a human evaluator would assign. A ranking of multiple LLM answers in this fashion allows us to compare LFQA capabilities between different LLMs or between different prompting strategies for the same LLM, by comparing the ranking positions of the generated answers.

Since a dataset for this task does not exist, we first need to adapt an existing dataset for our use case. Then, different retrieval methods are compared on the dataset, to find the one that aligns best with human annotator preferences. Finally, the answers generated by the LLMs have to be ranked using the most effective retrieval method.

3.1 Data Collection and Preparation

The dataset is based on the test set of the CLEF eHealth 2021 dataset Goeuriot et al. (2021). It was originally intended to evaluate the ability of retrieval systems to provide credible, readable, and relevant answers to laypersons' health questions.

3.1.1 CLEF eHealth 2021 Dataset

The test set consists of 55 health-related queries which either stem from Reddit or Google search trends. While the Reddit-sourced queries are well-formulated questions about specific health topics, the queries from Google search trends are not necessarily phrased as questions but rather as classical keyword-based search queries.

In addition to the queries, the dataset includes a collection of web documents and social media content. The web documents were mainly obtained from the CommonCrawl archive, containing a diverse range of 600 domains. This list of domains was created by the authors by executing medical queries via the Microsoft Bing API and was augmented by adding known reliable and unreliable health-related websites. The dataset was expanded by incorporating social media comments and posts from Reddit and Twitter. The comments and post were collected by executing manually generated search queries based on 150 pre-selected health topics and retrieving relevant responses.

To evaluate each of the three categories (relevance, readability, and credibility), each query was assigned 250 documents based on rank-biased precision (RBPA) (Moffat and Zobel (2008)). RBPA is a method for choosing which documents from a pool of documents to select for evaluation by human annotators. The pool of documents in this case is the collection of web documents and social media content returned by the participating teams of the shared task, as well as the organizer's baseline systems. From this pool, documents are evaluated based on their rank in different runs and then scored according to the three different dimensions. A more detailed description of the pooling method can be found in Lipani et al. (2017).

After the 250 documents per query are selected, the documents are assessed by humans for credibility, readability, and relevance. In total, there are 26 annotators, each annotating all documents for between one and four queries. The annotators were not medical experts but received written training material. In the end, three annotations were made for a total of 11357 unique documents. This amount differs from the total amount of annotations in each dimension which is 12500, since some documents were annotated multiple times but for different queries.

The annotations for relevance and readability are in the range 0 (not relevant / not readable), 1 (somewhat relevant / somewhat readable), and 2 (highly relevant / highly readable). For scoring credibility the range extends to 3, which is also interpreted as highly credible. The reasoning for this additional ranking score for credibility is not given in the original paper.

The following sections describe how the dataset was collected and prepared for the experiments in this thesis.

3.1.2 Dataset Collection

There are two ways of accessing the CLEF eHealth 2021 dataset. Option one is to download the indexed collection directly which is available on the organizers' GitHub repository.[1] This index does not contain the full text of the documents, so it can not be used to compare the original documents against newly generated answers. The second option is to download all documents individually, given their IDs. In theory, downloading all documents can be done by using the provided scripts in the GitHub repository. Scripts are provided to download the documents from the CommonCrawl archive, as well as the social media content directly over the respective APIs. Unfortunately, with Twitter shutting down their free tier of the API, at the beginning of April 2023, this is now only possible with a prohibitively expensive paid tier. For Reddit, those API changes followed just a few months later, making it impossible to download social media content from Reddit as well. Even though we started downloading some Reddit content before those changes, the already slow access to the Reddit data did not allow for downloading all the content in time.

This was unfortunate timing, but since the organizers of the CLEF eHealth task reported in their paper that the human credibility assessments of social media documents were significantly lower than those of web documents, we decided to only use the web documents for our dataset. The web documents are easily accessible via the CommonCrawl archive. Before accessing all relevant documents from the CommonCrawl archives, the list of documents is filtered to only include documents that are annotated for at least one query. Discarding all documents from Reddit and Twitter, as well as web documents that are not assessed for any query, results in a total of 6692 documents downloaded from the CommonCrawl archive using a slightly modified version of the provided script. This provides us with WARC files for the relevant documents, with WARC being a popular format for storing web documents for archival purposes. Since all social media content is discarded, the number of queries for which documents are available is reduced to 50, since of the total 55 queries, 5 queries only have social media content associated with them.

[1] `https://github.com/CLEFeHealth/CHS-2021`, accessed on 04.01.2024

Keyword-Based Queries	**Question-Type Queries**
best apps daily activity exercise diabetes	What are the most common chronic diseases?
my risk for developing type 2 diabetes	Is a ketogenic/keto diet suitable for people with diabetes?
multiple sclerosis stages phases	Can diabetes be cured?

Table 3.1: Samples of Keyword vs Question Type Queries

3.1.3 Preprocessing

The files in the WARC format contain the full HTML of the web documents. Before being able to compare the content to the generated answers, the HTML is extracted from the documents. HTML elements containing fewer than 50 characters in the element body are discarded, assuming they are not relevant to the content of the document (e.g. fields of navigation bars). Then, the plain text is extracted using the ChatNoir Resiliparse Library.[2]

3.2 Analysis of the Final Dataset

We now go more in-depth on the properties of the final dataset, considering the type of queries, the number of answers per query, and the length of the answers. The total number of queries in the final dataset is 50, each of which is accompanied by between 39 and 249 answers. The number of answers depends on how many of the given answers in the base dataset by Goeuriot et al. (2021) were, still available, i.e. not from Reddit or Twitter. Since the number of documents by source was not the same for all queries, the number of answers per query varies. On average, there are about 178 assessed documents per query available.

3.2.1 Queries

We identify two different query types in the dataset: questions and keyword queries. Questions are queries that are formulated as a question, e.g. "Is a ketogenic diet suitable for people with diabetes?", while keyword-based queries are more in the style of search engine queries, e.g. "keto diet diabetes". Table 3.1 shows some examples of the two query types. Query topics are diverse, ranging from general queries about chronic diseases to specific questions about the suitability of certain diets for people with diabetes.

[2]`https://resiliparse.chatnoir.eu/en/stable/`, accessed on 04.01.2024

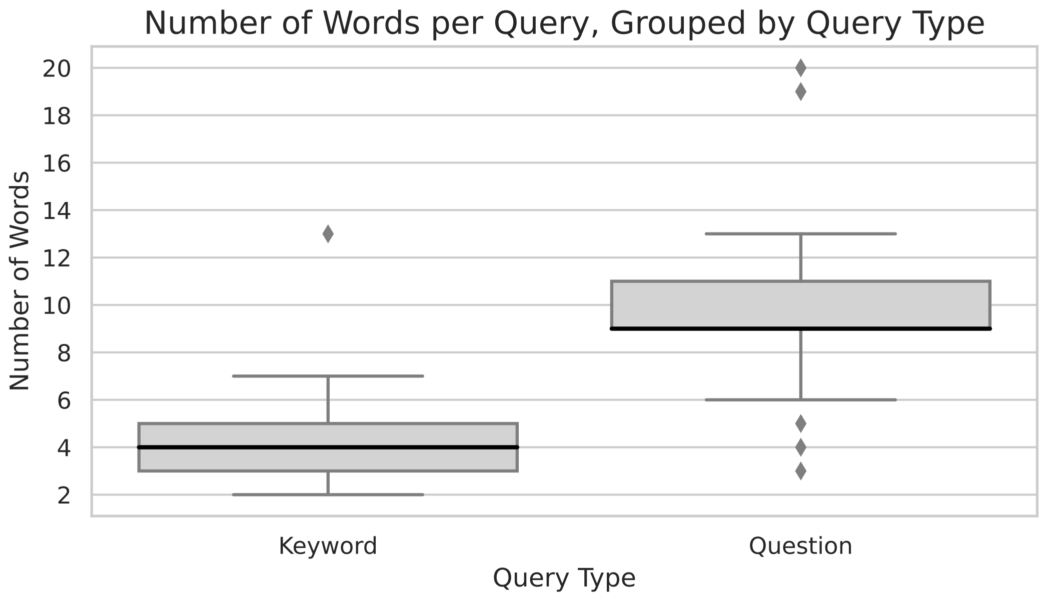


Figure 3.1: Boxplot of the number of words per query, split by query type. Keyword queries are shorter than question queries with a median of 4 words compared to 7 words for the question queries.

In total, we identify 17 question-style queries and 33 keyword-style queries by detecting whether the query ends with a question mark. Figure 3.1 shows the number of words per query for both query types. As expected, the keyword queries are shorter than the question queries, with a median of 4 words compared to 7 words for the question queries. The shortest query is only two words long, while the longest query is 20 words long. In a later section, we investigate the differences in answer ranking of the different models depending on the two query types.

3.2.2 Documents

The documents in the dataset are scraped from a total of 234 different web domains. Table 3.2 shows the top 10 most frequently occurring domains in the dataset. Most domains are health-related, belonging to either health organizations, medical journals, or health news websites. Some domains are more general, e.g. there are also wikipedia.org pages in the dataset, as well as some domains from essay writing or homework help websites. A total of 133 domains show up less than 10 times in the dataset, while 44 domains show up only once. After preprocessing, the documents are cleaned of all HTML tags and HTML elements containing fewer than 50 characters. Figure 3.2 shows that the num-

Domain	**Occurrences**
www.healthline.com	603
www.nationalmssociety.org	419
www.ms.org.au	198
jhu.pure.elsevier.com	191
www.msif.org	183
www.psychologytoday.com	161
www.urotoday.com	155
www.news-medical.net	150
www.sleepfoundation.org	141
www.aafp.org	139

Table 3.2: Top 10 Most Frequently Occurring Domains

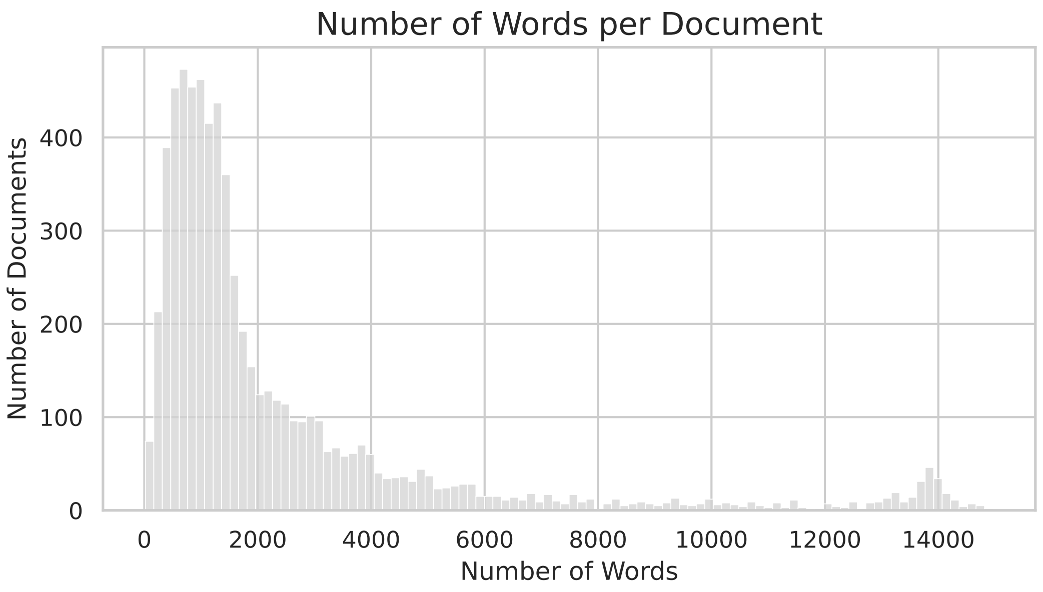


Figure 3.2: Histogram of the number of words per document. The plot is cut at 15,000 words, which excludes 120 or 2% of the documents. The median number of words per document is 1354, while the mean is 2901.

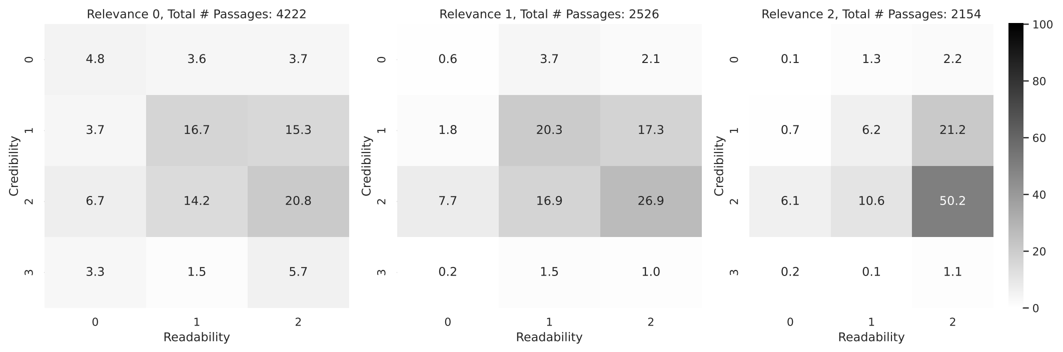


Figure 3.3: Three heatmaps, one for each relevance score, plotting credibility against readability. The total number of documents with the given relevance ranking is in the title of the heatmap. In all three heatmaps, the highest number of documents is concentrated in the range between 1 and 2 for both credibility and readability, with 2 credibility and 2 readability being the most common combination.

ber of words per document is high, with a median of 1354 words and a mean of 2901 words. The high word count is natural for documents scraped from the web, which not only contain the main text but also navigation bars, footers, sidebars, and other elements. Those can not be fully removed with our heuristic-based preprocessing methods.

3.2.3 Document Ratings

The evaluation dimensions of readability, credibility, and relevance are the human annotations from the CLEF eHealth 2021 dataset by Goeuriot et al. (2021). Documents can have multiple ratings for each dimension if they are annotated for multiple queries. Having multiple annotations for different queries results in the total number of ratings per dimension (8902) being higher than the number of documents in the dataset (6692).

In general, a high number of documents are rated as not relevant to the given query. Figure 3.3 shows three heatmaps, with each dimension being displayed in a separate heatmap. Documents rated with 0 relevance make up nearly half of the dataset and generally have lower credibility and readability scores. The most relevant documents are generally also the most credible and readable ones. With a value of 0.08, there is no correlation between relevance and credibility. For readability and relevance, the correlation is 0.22, which is slightly stronger, but still does not show a strong connection. The correlation scores can be interpreted as the relevance of a document being independent of its credibility and readability, which shows that the annotators were able to judge the relevance of a document independently of its credibility and readability.

3.2.4 Documents with Ratings for Multiple Queries

As mentioned previously, some documents have multiple ratings for different queries. In total 875 documents are rated for multiple queries, with the highest number of individual annotations for a single document being 18. Documents with multiple ratings are often general articles about a certain topic, e.g. “What is Multiple Sclerosis?”. Those are then retrieved for multiple queries that focus on different aspects of the topic, e.g. “List of multiple sclerosis symptoms” and “Can I pass multiple sclerosis to other family members?”. As expected, the different relevance ratings for the same document are often different, since the relevance of a document depends on the query. Interestingly, we also find that the credibility and readability ratings for the same document can be different. In Appendix A, we show an example of a document with multiple ratings for different queries. The 15 different ratings cover nearly all possible combinations of relevance, credibility, and readability.

As all documents for one query are annotated by the same person according to Goeuriot et al. (2021), we assume that the annotators were consistent in their ratings for the same query. Because we are only interested in the ranking of the documents for one query at a time, we do not remove or aggregate the scores, as this could alter the ranking of the documents compared to documents that were only ranked for this specific query.

3.3 Retrieval Pipeline Development

In this section, the implementation of the different retrieval pipelines is discussed. The baseline models (DPH and TF-IDF), as well as the ColBERT version 1 and the monoT5/duoT5-based pipelines, are implemented in the PyTerrier framework by Macdonald and Tonellotto (2020), which is a Python API for the Terrier IR Platform (Macdonald et al. (2012)). The ColBERT version 2 pipeline is based on the original implementation provided by the authors.[3]

3.3.1 Retrieval Setup

The retrieval setup in this work differs from the usual retrieval setup, in which a large document collection is indexed, and then multiple queries are run against

[3]`https://github.com/stanford-futuredata/ColBERT`, accessed on 04.01.2024

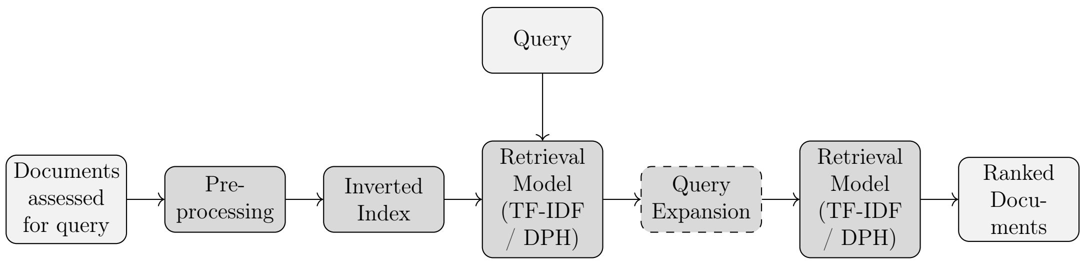

Query
Documents assessed for query
Pre-processing
Inverted Index
Retrieval Model (TF-IDF / DPH)
Query Expansion
Retrieval Model (TF-IDF / DPH)
Ranked Documents

Figure 3.4: Baseline retrieval pipeline, using either TF-IDF or DPH as retrieval model with optional query expansion. This pipeline retrieves and ranks assessed documents for a given query.

the index using different retrieval models. In our case, we build separate indices for each query, containing only the documents that are assessed for that query. Having separate indices ensures that each document that is retrieved for a query is also assessed for that query.

3.3.2 Baseline Models

This section describes the implementation of the baseline models, namely DPH and TF-IDF. For both models, the basic indexing function of PyTerrier is used, which indexes and preprocesses the documents. Preprocessing is done with the default values, which include the following operations:

- **Tokenization**, for splitting the text into individual words or tokens. The default PyTerrier tokenizer splits text on non-alphanumeric characters. Additional rules to discard tokens that are longer than 20 characters, contain more than 4 digits or contain the same character more than 3 times in a row are applied. All tokens are also converted to lowercase.
- **Stopword removal**, to remove tokens that do not contain information about the content. We use the default PyTerrier stopword list.
- **Stemming**, to remove the different endings a token can have, leaving only the root of the word. The Porter stemmer, a rule-based stemmer, is used here.

The resulting inverted index contains all remaining tokens and a mapping from each token to the documents in which the token occurs.

The two retrieval models TF-IDF and DPH are then applied using the default parameters, discussed in section 2.2.1. Both models are tested with and without query expansion. We use BO1 query expansion, which uses the documents retrieved by the original query to expand the query. It adds terms from the retrieved documents to the original query based on informativeness, which is a measure of how frequently the term occurs in the documents compared to how frequently the term occurs in the whole collection. Terms that occur more frequently in the retrieved documents than in the collection are added to the query. Afterward, the expanded query is run against the index again, and the documents are ranked according to the new query. The resulting pipeline is shown in Figure 3.4. The pipeline produces a ranked list of documents for each query, which can then be compared to the order of documents according to the human annotations.

3.3.3 Transformer Models

In addition to the baseline models, we also use the transformer-based models ColBERT version 1 and 2, as well as the monoT5 and duoT5 models. The implementations of ColBERT version 1 and the monoT5/duoT5 models are done in PyTerrier, while the implementation of ColBERT version 2 is done in the original implementation provided by the authors.

All mentioned models are based on a pre-trained model, which is used to encode the query and the documents in different ways. Both ColBERT versions are based on the BERT-base model, while the monoT5 and duoT5-based models are based on the T5 model. The retrieval models themselves are fine-tuned on the MS MARCO passage ranking dataset (Bajaj et al. (2016)), a large-scale dataset for passage retrieval.

Transformer-based retrieval models usually require a first retrieval step using a less computationally expensive model like tf-idf, to reduce the number of documents that are fed to the transformer model. Because in this thesis the retrieval dataset is small, the first retrieval step is omitted for all transformer-based models.

ColBERT Version 1

Using the ColBERT implementation for PyTerrier,[4] an end-to-end pipeline can be constructed from a pre-trained ColBERT model and a document collection. In our experiments, we use the pre-trained ColBERT checkpoint provided by the authors. The checkpoint can directly be loaded in the PyTerrier framework, and then used to retrieve documents from the collection. All parameters are left at their default values.

[4] `https://github.com/terrierteam/pyterrier_colbert`, accessed on 04.01.2024

monoT5 and duoT5

Like the ColBERTv1 implementation, the monoT5 and duoT5 implementations are also available in PyTerrier.[5] Pre-trained checkpoints based on the MS MARCO dataset are provided by the authors. The monoT5 pipeline generates scores for each query-document pair, which are then used to rank the documents. The duoT5 pipeline takes the ranking from the monoT5 pipeline and re-ranks the top 10 documents by directly comparing each pair of the top 10 documents directly to the query. The scores are aggregated to produce a final ranking. Again, all parameters are left at their default values.

ColBERT Version 2

Unlike the other transformer-based models, ColBERT version 2 is not available directly in PyTerrier. Instead, we use the implementation provided by the authors of the model.[6] The implementation works with any pre-trained checkpoint for the ColBERT version 2 model, with the MS MARCO checkpoint being downloaded by default.

3.3.4 External scores for Readability

To improve retrieval effectiveness for readability, we experiment with adding pre-computed external scores to the retrieval process. To estimate the readability of the documents in the dataset, different scores are calculated for each document.

The first score is the Flesch Reading Ease (FRE) score devised by Kincaid et al. (1975), which is a score between 0 and 100, with higher scores indicating easier-to-read text. It is calculated using the following formula, based on average sentence length (ASL) and average number of syllables per word (ASW):

$$FRE = 206.835 - (1.015 \times ASL) - (84.6 \times ASW) \tag{3.1}$$

We use the implementation provided by the textstat library.[7] This library also provides the second score, which is an aggregate of multiple similar scores, like the FOG score, the SMOG score, and the Coleman-Liau Index. Those scores are all based on different text statistics, like the number of syllables per word, the number of words per sentence, or the number of characters per word. More details about the different scores can be found in the documentation of the textstat library.

[5] `https://github.com/terrierteam/pyterrier_t5`, accessed on 04.01.2024
[6] `https://github.com/stanford-futuredata/ColBERT`, accessed on 04.01.2024
[7] `https://pypi.org/project/textstat/`, accessed on 04.01.2024

3.4 Generating LLM Responses

We now introduce the LLMs that we selected for the experiments, and the steps taken to get the generated answers from the LLMs. Additionally, the different prompting strategies that are used to generate the responses are described.

3.4.1 Selection of Language Models

We select multiple LLMs for our experiments, which differ in the number of parameters, amount and type of training data, and the type of pre-training and fine-tuning.

GPT-2

As a simple baseline and to investigate how an increased number of parameters affects the ranking results, we select different sizes of the GPT-2 model, namely the base, medium, large, and XL versions. Those models only differ in the number of parameters, which are 124M, 355M, 774M, and 1.5B respectively, while the training objective and the training data are the same for all models. The dataset used for pre-training is the WebText dataset, which contains 40 GB of web text, which is scraped from URLs shared on Reddit that received at least 3 upvotes. Other than this pre-training, the models are not fine-tuned to any specific task.

Models optimized for dialog

Dialog-optimized models are fine-tuned for the task of interacting with a human in a dialog, usually as a chatbot. Chatbots are currently the use case of LLMs that comes closest to the task of LFQA, so including models optimized for dialog is a natural choice. To make a model suitable for directly interacting with humans, different fine-tuning objectives are used in which the behavior of the model is aligned with the expectations of the human.

The importance of this alignment is shown by Ouyang et al. (2022), who show that answers of their instruction-tuned model InstructGPT are preferred by humans compared to outputs by a model with over 100 times the number of parameters, but without any fine-tuning. We select the following models:

Falcon, which was released in 2023 on HuggingFace.[8] At the time of writing, there are three model sizes available, 7B, 40B, and 180B parameters, each as a foundation model or as an instruction-tuned model. For our experiments, we only use the small 7B instruction-tuned model.

[8]`https://huggingface.co/blog/falcon`, accessed on 04.01.2024

According to the HuggingFace model card, the model is pre-trained on the RefinedWeb dataset by Penedo et al. (2023) and then fine-tuned on the GPTeacher,[9] GPT4All[10] and Baize (Xu et al. (2023a)) datasets. GPTeacher and GPT4All are both datasets of prompt-response pairs, while Baize is a dataset based on self-chats by ChatGPT, which have been seeded by questions from Quora, StackOverflow, and MedQuAD. Unfortunately, the authors do not detail exactly how the data was used to fine-tune the model, as a scientific paper is still yet to be released.

Llama 2, a model introduced in 2023 by Touvron et al. (2023). It is available through the HuggingFace library, after agreeing to the terms of use posed by Meta. It comes in three sizes, 7B, 13B, and 70B parameters. For each of the sizes, the base language model, and a fine-tuned version for chat is available. In our experiments, we use the fine-tuned version in both the 7B and 13B sizes, since the 70B parameter model is too large to fit on our available hardware.

To create the chat model, the base model is first tuned using supervised fine-tuning, where the model is trained on a dataset of prompt-response pairs, which are all written by humans. The next step is Reinforcement Learning with Human Feedback (RLHF), by which the model is aligned with human preferences. To generate a training set, humans are tasked to evaluate which of two generated responses to a given prompt they prefer. Based on that data, a reward model is trained, which can then evaluate generated responses on a large scale. The model is then fine-tuned to maximize the reward given by the reward model. The RLHF training is done over multiple iterations, continuously improving the reward model and the chat model based on human feedback. Additionally, they not only optimize for the helpfulness of the generated response but also evaluate for safety, instructing the model not to provide harmful content in that way.

ChatGPT, is the largest model we used, with 175B parameters. It was introduced by OpenAI at the end of 2022.[11] As a commercial product, ChatGPT can only be accessed through the OpenAI API, which we did by using the provided Python library.

With OpenAI becoming more secretive about their models since starting to monetize them, the exact training data and fine-tuning procedure are

[9] `https://github.com/teknium1/GPTeacher`, accessed on 04.01.2024
[10] `https://github.com/nomic-ai/gpt4all`, accessed on 04.01.2024
[11] `https://openai.com/blog/chatgpt/`, accessed on 04.01.2024

Model	Params	Release	Pre-training Data	Fine-tuneing Data	Fine-tuneing Methods
GPT-2 Base	124M	2019	WebText	-	-
GPT-2 Medium	355M				
GPT-2 Large	774M				
GPT-2 XL	1.5B				
Falcon 7B	7B	2023	RefinedWeb	GPTeacher, GPT4All, Baize	SFT
Llama 2 7B	7B	2023	Proprietary	Proprietary	SFT, RLHF
Llama 2 13B	13B				
ChatGPT	175B	2022	Proprietary	Proprietary	Proprietary

Table 3.3: Comparison of evaluated Language Models. SFT = Supervised Fine-Tuning, RLHF = Reinforcement Learning with Human Feedback

not known. In their blog post, they state that the model is a close sibling to the InstructGPT model mentioned above. Being a sibling model means the fine-tuning procedure is likely similar to the one described by Ouyang et al. (2022) but with a larger model size. The methods applied are similar to the ones used for the Llama 2 model, first using supervised fine-tuning, and then using RLHF to align the model with human preferences.

Table 3.3 gives an overview of all the used models, comparing parameter size pre-training and fine-tuning data, as well as fine-tuning methods. In total, we use 8 different models, 4 of which are variants of GPT-2, while the others a different fine-tuned models for chat.

3.4.2 Prompting approaches

How a LLM is prompted to complete a task has a large impact on the quality of the generated response. Reynolds and McDonell (2021) show this impact in the context of translating French to English. They compare the effect of three different prompting strategies, finding that the most complex one leads to the best translations. Based on those results we use a similar approach of different prompting strategies for our experiments. We use the following four strategies:

1. **No Prompt:**
   query

2. **Short QA Prompt:**
   Q: *query*
   A:

3. **Long QA Prompt:**
   Question: *query*
   Answer:

4. **MultiMedQA Promp:**
   You are a helpful medical knowledge assistant. Provide useful, complete, and scientifically grounded answers to common consumer search queries about health.
   Question: *query*
   Complete Answer:

where *query* is replaced by the query to be answered. Based on the results of Reynolds and McDonell (2021), we expect the generated answers based on the MultiMedQA prompt to rank highest, followed by the long QA prompt. To reduce the influence of random variations in the generated responses, we generate 10 responses for each query and model.

3.4.3 Generating Answers

To generate answers for a given query, the query is passed to the LLM using one of the prompting strategies described above. Most LLMs take additional parameters as input, which are used to guide the text generation process. The following parameters are considered for the different models in our experiments:

Max new tokens defines the maximum number of new tokens to be generated by the LLM, limiting the length of the generated output.

Temperature controls the randomness in the generation process. A higher temperature value increases randomness in the output, while a lower value leads to more deterministic outputs.

Top k sets a threshold for the number of most likely next tokens to consider at each step of generation, only allowing the model to choose from the top k tokens. A higher value for k will increase the diversity of the generated output, while a lower value will decrease diversity.

Top p/nucleus sampling specifies a cumulative probability threshold. The model will only consider the smallest set of tokens whose cumulative probability does not exceed the threshold.

Repetition penalty is used to discourage the model from repeating the same phrases. A repetition penalty greater than 1 will decrease the likelihood of tokens that have already appeared in the generated text, so the model produces more diverse and less repetitive content.

Not all parameters are available for all models and the optimal values for each parameter differ between models. We chose to set the same values for all models, to make the comparison between the models more consistent. We did not try to find the optimal values for each model but instead used values we found to work well with all models in manual experiments.

Maximum number of new tokens is set to 512 because that's the maximum length of the input for the GPT-2 models.

Temperature is set to 0.75, restricting the variability of the generated output, while still allowing for some randomness.

Top k and top p are set to 50 and to 0.95 respectively, which both restrict the number of tokens the model can choose from to more likely tokens.

Repetition penalty is set to 1.2, which is a relatively low value, but still helps to reduce repetition in the generated output, which we found to be a common problem, especially for the GPT-2 models, but also for the smaller instructing tuned models.

Any additional parameters that are available for a model are left at their default values.

The generating of the answers is done in two different ways. For ChatGPT, we use the OpenAI API to generate the answers, specifically using the model version "GPT-3.5-turbo-0613". Since they do not provide any parameters apart from temperature and number of tokens, we only set those to our chosen values.

For all other models, we use the HuggingFace library to generate the answers. The HuggingFace library allows us to set all parameters as described above. Each model is prompted ten times with each of the prompting strategies, resulting in 40 generated answers per query for each model.

Chapter 4

Results

In this chapter, we present the results of our experiments. Starting with an evaluation of the retrieval pipelines, we follow with showing the effectiveness of the different LLMs on the rank-based implicit evaluation method. The results and visualizations needed to evaluate the research questions formulated in the Introduction are presented.

4.1 Evaluation of Retrieval Pipelines

In this section, we compare the effectiveness of the different retrieval pipelines. We chose to evaluate the pipelines using nDCG@10. nDCG is a suitable metric for our task, as it takes into account multiple different relevance/credibility/readability levels instead of only the binary relevant/non-relevant distinction. Furthermore, it emphasizes the importance of the pipeline being able to distinguish between answers at the top level by giving more weight to the top results. We chose a cutoff of 10 because even though the minimum available number of answers per query is 39, the number of relevant results is much lower. This cutoff also normalizes the rank position between the different queries, as the number of assessed documents per query varies.

4.1.1 Baseline Pipelines

Our baseline retrieval pipelines are the TF-IDF and BM25 pipelines, each evaluated once with and once without query expansion. Effectiveness on all three dimensions is similar for all pipelines, showing little difference between the strategies. Table 4.1 shows the retrieval effectiveness of the baseline pipelines in the first four rows. The rankings are closest to the human rankings for readability, scoring around 0.75 for all pipelines. Relevance is lower than that, with scores around 0.64. Since those models are designed to retrieve relevant

Pipeline	Relevance	Readability	Credibility
DPH	0.643	0.742	0.539
DPH (qe)	0.637	0.751	0.51
TF-IDF	0.64	0.746	0.551
TF-IDF (qe)	0.636	0.751	0.557
ColBERTv1	0.626	0.774	0.633
ColBERTv2	0.634	0.799	0.65
duoT5	0.64	0.805	0.714
monoT5	**0.645**	**0.813**	**0.722**

Table 4.1: Results of the transformer pipelines, shown as nDCG@10. The best results of the baseline pipelines are shown for comparison. All models are trained on the MS MARCO passage ranking dataset (Bajaj et al. (2016)).

documents, we would expect the relevance scores to be higher. With scores around 0.55, the credibility scores are the lowest of the three dimensions.

4.1.2 Transformer Pipelines

Table 4.1 shows the nDCG@10 scores for the different transformer-based pipelines, compared to the best baseline results. The monoT5 pipeline is most effective on all three dimensions, with scores of 0.645 for relevance, 0.813 for readability, and 0.722 for credibility. Re-ranking the top 10 results using duoT5 made the results slightly worse in all dimensions. ColBERTv2 shows a slight improvement over ColBERTv1 but is still worse in relevance while being better in readability and credibility.

In general, the transformer-based models have very similarly effectiveness to the baseline models in terms of ranking for relevance, but all transformer-based models achieve higher scores on readability and credibility.

4.1.3 External Document Scores

To investigate if adding external scores to the ranking process could improve the document rankings, we calculate statistical readability scores for all documents in the dataset. We then compare the calculated scores to the readability scores assigned by the human annotators. For our dataset, we do not find a strong correlation between the human-annotated readability scores and the statistical scores. The Flesch Reading Ease score has a correlation of 0.177 with the readability scores, while the full textstat score has a slightly negative correlation of -0.147. Those correlations are not strong enough to justify using the statistical scores as a proxy for readability.

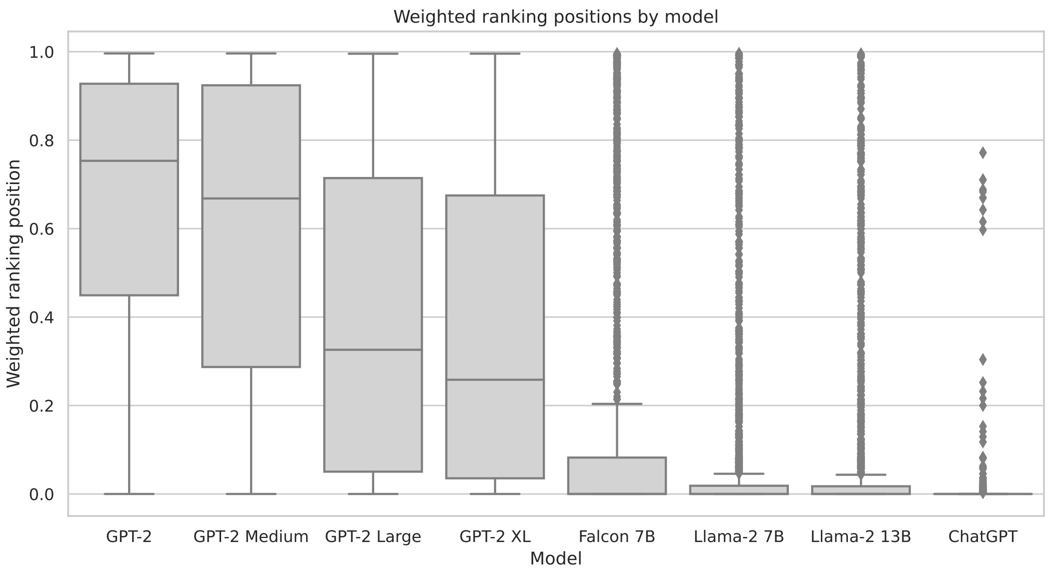


Figure 4.1: Boxplot of the normalized position of the LLM answers by model. The normalized position is the absolute position of the answer in the ranking, divided by the total number of documents for the query.

The low correlation scores might be connected to the fact that the documents in our dataset are scraped from the web, which means that even after preprocessing they still contain noise not accounted for by the statistical scores, which are intended for the use on clean text.

4.1.4 Summary

Because monoT5 is most effective on ranking in all three dimensions, we use it for all further experiments and only evaluate the one ranking produced by this pipeline.

We evaluate the LLM answer effectiveness on their normalized position, which is the absolute position of the answer in the ranking, divided by the total number of documents for the query.

$$\text{Normalized Position} = \frac{\text{Absolute Position}}{\text{Total Number of Documents for Query}} \tag{4.1}$$

4.2 Factors influencing LLM effectiveness

In this section, we present results that help in investigating how different factors affect the effectiveness of health answers by LLMs on the implicit evaluation method. We look at the influence of model size, prompting strategy,

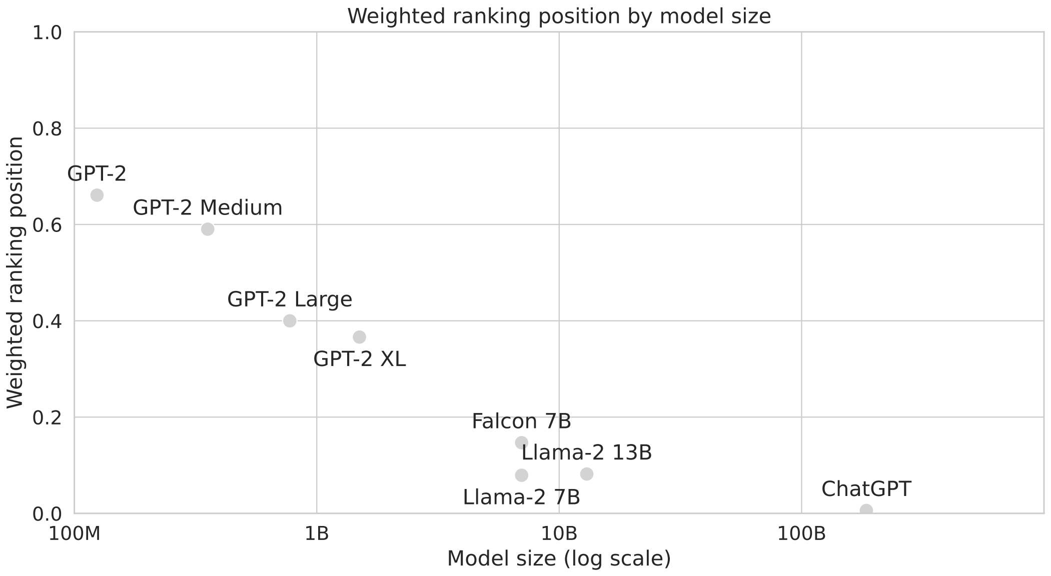


Figure 4.2: Scatterplot showing number of model parameters against weighted normalized position of the model answers, as the mean over all prompts. Model size is shown on a logarithmic scale.

answer length and query type on the ranking of the LLM answers. Additionally, we investigate the answers generated by ChatGPT that are ranked especially low.

In total, we rank 16000 generated answers, with each of the 8 models generating 10 answers for the 4 different prompting strategies for all 50 queries. Figure 4.1 shows the normalized position of the LLM answers in the ranking.

Of all models, ChatGPT is most effective with a median normalized position of 0.006, scoring the best or second-best answer at nearly all queries. Falcon 7B is the worst of the fine-tuned models with the Llama models being slightly more effective. The GPT-2 models show the lowest rankings overall, but show a clear trend of better rankings with increasing model size.

All models show a large spread in the rankings, with the most effective models generally having a smaller spread. ChatGPT is the most consistent model, with the smallest standard deviation of 0.05. The other models have much higher standard deviations, with the Llama models both being around 0.2 and the other models between 0.3 and 0.35.

The spread of all GPT-2 models is so large that the 0.25 and 0.75 quantiles overlap with the minimum and maximum values, showing the consistently large spread of answer rankings for those models. Even though the spread between the 0.25 and 0.75 quantiles is smaller for the fine-tuned models, all of them still have outliers towards the bottom of the ranking.

Model	No Prompt	Short QA	Long QA	MultiMedQA
GPT-2	0.763	0.663	0.614	**0.603**
GPT-2 Medium	0.675	**0.524**	0.544	0.618
GPT-2 Large	0.496	0.393	**0.336**	0.374
GPT-2 XL	0.509	0.336	0.327	**0.292**
Falcon 7B	0.324	0.133	0.118	**0.012**
Llama-2 7B	0.211	0.073	0.027	**0.005**
Llama-2 13B	0.218	0.068	0.032	**0.01**
ChatGPT	0.006	0.01	0.007	**0.001**

Table 4.2: Mean normalized position of the LLM answers, by prompting strategy. The most effective prompting strategy to generate answers for each model is highlighted in bold. The prompting strategies are described in section 3.4.2, order here from least to most complex. Increasing prompting complexity generally improves the ranking of the LLM answers for most models, with exception to GPT-2 Medium and GPT-2 Large.

4.2.1 Influence of Model Size

Figure 4.2 shows the weighted normalized position of the LLM answers plotted against the number of model parameters. There is a clear trend of more model parameters leading to better rankings, with the GPT-2 models generating the least effective answers and the 185B parameters ChatGPT model generating the most effective answers. The improvement is not linear, with the biggest improvements happening between GPT-2 Medium and GPT-2 Large, and between GPT-2 XL and Falcon 7B/Llama-2 7B.

4.2.2 Influence of Prompting Strategy

The different prompting strategies strongly influence the final rankings of the LLM answers. Table 4.2 shows the mean normalized position of the LLM answers for the different prompting strategies. Starting with no prompt produces the worst ranking results for all models, except for ChatGPT where the long QA prompt is slightly less effective. The rankings improve with each more complex prompting strategy, except for GPT-2 Medium and GPT-2 Large, which is most effective on the short QA and the long QA prompts respectively. For all other models, the ranking is best on the complex MultiMedQA prompt.

Inside each prompting strategy, the ordering of the models is consistent except for the Llama models, which change between second and third place depending on the prompting strategy.

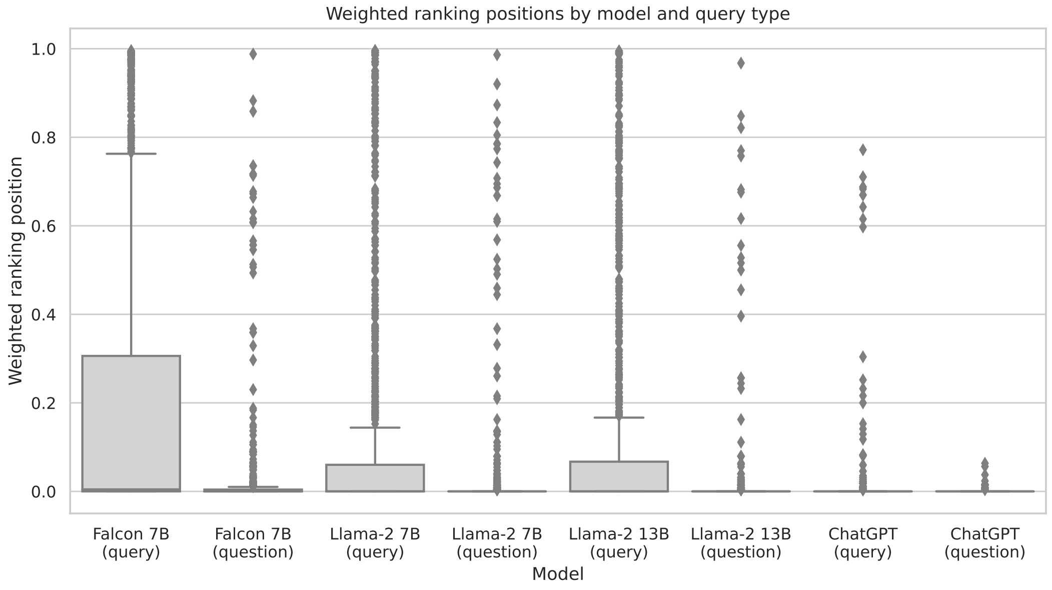


Figure 4.3: Boxplot of the normalized position of the LLM answers, by model and query type.

4.2.3 Keyword vs Question Queries

Figure 4.3 shows the normalized position of the LLM answers, split by query type and model. Question-type queries are identified by the last character of the query being a question mark, while keyword queries are all other queries. Answers generated for the properly phrased question-type queries achieve higher median rankings than the answers generated for the keyword queries. The spread of the rankings is also smaller for the question-type queries, with the keyword queries having more outliers at the bottom of the ranking. The different spread ranges depending on query type also hold true for ChatGPT, which is the most consistent model overall. The GPT-2 based models exhibit the same pattern of question-type queries being ranked better than keyword queries, but are omitted from Figure 4.3 for better readability.

4.2.4 Lower Ranked ChatGPT Answers

While ChatGPT generally provides the most effective answers, there are still some answers that are ranked poorly. We manually inspect each of the ChatGPT generated answers with a normalized ranking position of 0.1 or higher, meaning if there are 100 documents for the query, the answer is ranked 10 or lower. All the low ranked answers stem from four queries, namely “hypothyroidism symptoms”, “List of multiple sclerosis symptoms”, “exercises for better

Models	**List Style**	
	Yes	**No**
GPT-2	N/A (0)	0.6608 (2000)
GPT-2 Medium	N/A (0)	0.5902 (2000)
GPT-2 Large	0.4365 (36)	0.3992 (1964)
GPT-2 XL	0.4004 (22)	0.3658 (1978)
Falcon 7B	0.312 (38)	0.1436 (1962)
Llama-2 7B	0.0416 (266)	0.0849 (1734)
Llama-2 13B	0.0483 (427)	0.0909 (1573)
ChatGPT	0.0128 (748)	0.0021 (1252)

Table 4.3: Mean normalized ranking position of the LLM answers, split by whether they are formatted as a list. The number in brackets is the number of answers in that category. If that number is 0, the model did not generate any answers in that format, so the mean is not available (N/A).

posture" and "my risk for developing type 2 diabetes" and follow one of two patterns. The answers for the question about the personal risk of developing type 2 diabetes all contain the phrase "As an AI", as in the phrase "I'm sorry, but as an AI language model, I don't have access to personal data about individuals unless it has been shared with me in the course of our conversation.". This is a commonly used phrase by ChatGPT, which indicates that the model is unable to answer the question. The answers for the other three queries are all formatted as lists, either using enumeration or simple bullet points. To investigate if answers formatted as lists are generally ranked lower, we identify all answers that are formatted as lists by searching for generated answers containing more than five non-empty lines, of which at least three start with either a dash or a number. We also identify all answers containing the phrase "As an AI". Table 4.3 shows the mean normalized ranking position of the LLM answers, split by whether they are formatted as lists. ChatGPT generates the most answers formatted as lists, with 748 of the 2000 answers following that pattern. On average, those answers are ranked worse than the other answers by ChatGPT, with a mean normalized ranking position of 0.0128 compared

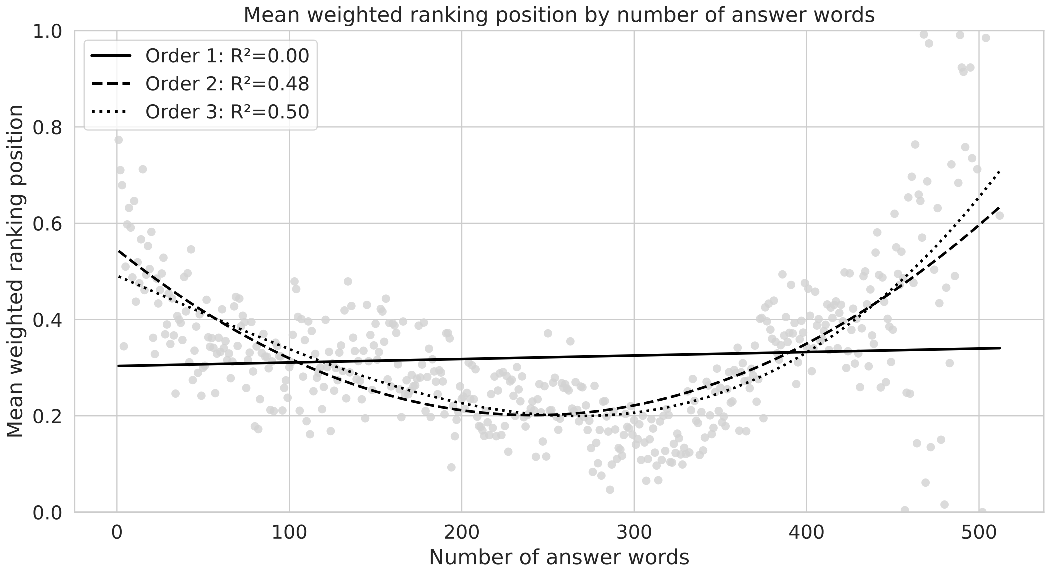


Figure 4.4: Scatterplot showing the weighted normalized position of the LLM answers against the number of words in the answer. To improve visibility, we show the mean of the weighted normalized position for each answer length, so the standard deviation is not visible. Three regression lines of order 1, 2 and 3 are shown.

to 0.0021 for the non list answers. The other models do not generate as many answers formatted as lists, with the both Llama models generating the most, 427 and 266 respectively. The answers by the Llama models which are formatted as lists score higher than the non-list ones, while for the remaining models the answer rank follows the ChatGPT pattern, in that list-style answers are ranked lower than non list-style answers.

For ChatGPT, the answers containing the phrase "As an AI" are ranked worse than the other answers, with a mean normalized ranking position of 0.0578 compared to 0.0056. They are also generated less frequently, with only 20 of the 2000 answers containing the phrase. The only other model generating that phrase is the Falcon 7B model, which generates it 30 times. With a mean normalized rank of 0.191 the answers containing the phrase also rank worse, but the difference is not as pronounced compared to the rank of 0.146 which the remaining answers by Falcon 7B achieve.

4.2.5 Influence of Answer Length

We investigate the influence of answer length on the ranking of the LLM answers by plotting the weighted normalized position against the number of words in the answer. Figure 4.4 shows the scatterplot of the weighted normal-

Model	**Mean Ranking**	**ARC**	**HellaSwag**	**MMLU**
GPT-2	0.661	21.8	31.6	25.9
GPT-2 Medium	0.59	27.0	40.2	26.6
GPT-2 Large	0.4	25.9	45.6	26.1
GPT-2 XL	0.366	30.3	51.4	26.4
Falcon 7B	0.147	46.2	70.9	25.8
Llama-2 7B	0.079	52.9	78.6	48.3
Llama-2 13B	0.082	59.0	81.9	54.6
ChatGPT	**0.006**	**85.2**	**85.5**	**70.0**

Table 4.4: Comparison of the mean normalized ranking position of the LLM answers to the results of other benchmarks. For the HuggingFace models, those scores are taken from the HuggingFace Open LLM Leaderboard by Beeching et al. (2023). For ChatGPT, we estimate the score from the GPT-4 Technical Report (OpenAI (2023)), which provides scores for GPT-3.5, the underlying model of ChatGPT. This is not the exact model we use in out testing, but we assume that the capabilities are similar. Mean Ranking is the mean normalized ranking position over all generated answers for the model.

ized position against the number of words in the answer, with three regression lines which are linear (order 1), quadratic (order 2) and cubic (order 3). The linear regression fails to capture the variance in the data, indicated by the R-squared value of 0. With R-squared values of 0.48 and 0.50, the quadratic and cubic regression lines are better at capturing the variance, indicating a parabolic relationship between answer length and ranking. Very short and long answers are ranked worse than answers of medium length, with the best ranking answers on average having between 200 and 350 words. Of the 3271 answers with more than 350, about 85% are generated by any of the GPT-2 models. Manual evaluation shows that many of the answers show a pattern of repeating the same phrase multiple times, leading to a high word count but not adding any additional information to the answer.

4.3 Comparison to other Benchmarks

We compare the median normalized ranking position of the LLM answers to the results of other benchmarks. To our knowledge, there are currently no benchmarks available that evaluate the effectiveness of LLMs for LFQA and provide results for the same models that we use. The HuggingFace Open LLM Leaderboard by Beeching et al. (2023) provides results on multiple different benchmarks for many models hosted on the HuggingFace hub. From here, we select the ARC (Clark et al. (2018)), the HellaSwag (Zellers et al. (2019))

and the MMLU (Hendrycks et al. (2020)) datasets, all of which are single choice question answering benchmarks. Table 4.4 shows the scores over those benchmarks, as well as the mean normalized ranking position of the LLM answers. On all benchmarks, there is the general trend of higher number of model parameters leading to better scores. For our dataset this trend is only broken for the two Llama versions, with the 7B model ranking slightly better than the 13B model.

Chapter 5

Discussion

The main goal of this thesis was to investigate whether human-written web content from the medical domain can be used as a proxy for evaluating LLMs in LFQA. In this chapter, we discuss the results of the underlying research questions that are investigated to answer this overarching question. Finally, we discuss the limitations of our work.

5.1 Factors Influencing LLM Effectiveness

The effectiveness of LLMs in providing health-related answers varies significantly depending on several factors, as shown by our experimental results.

5.1.1 Model Size

Our results show a strong connection between model size and ranking results in our rank-based implicit evaluation as evident in Figure 4.2. The differences between the GPT-2 based models are insightful, as they share the same architecture and dataset, only differing in the number of parameters. The larger the model size, the better the ranking results, which aligns with the findings of Radford et al. (2019) who showed that the increased model sizes lead to higher effectiveness on various NLP tasks.

With 185 billion parameters, ChatGPT is the largest model in our evaluation, and it ranks best on nearly every query. For the small to medium-sized fine-tuned models, specifically the 7B variant of Falcon and the 7B and 13B variants of Llama-2, the trend is less clear. Llama-2 7B is more effective than the 7B variant of Falcon, scoring a mean normalized rank of 0.079 compared to 0.146 by Falcon 7B. Even though they have the same number of parameters, the differences could be explained by the different training data used for the models and the more sophisticated fine-tuning strategy of Llama-2.

More surprisingly, Llama-2 7B on average ranks about as well as its larger variant, which has nearly twice as many parameters. The similar rankings indicate either a saturation of the Llama-2 model's effectiveness around that parameter count or that the ranking result is not accurate enough to distinguish between the two models. Since according to the original paper Touvron et al. (2023) the model exhibits the usual trend of increasing effectiveness with model size, we assume that the latter is the case.

So, while the model size is a strong indicator of the model's effectiveness in our evaluation, the results do not fully align with the expected trend.

5.1.2 Prompting Strategy

Prompting strategies have a significant impact on the effectiveness of LLMs, shown in Table 4.2. There is a general tendency that the more sophisticated the prompting strategy, the better the ranking results. While there are two outliers (GPT-2 Medium and GPT-2 Large), which rank better with one of the simpler prompting strategies, the overall trend is clear, with the MultiMedQA prompt leading to the best results. Having the most complex prompt be most effective aligns with the findings of Reynolds and McDonell (2021) who showed this pattern for the task of text translation using GPT-3.

Even for ChatGPT, which already ranks best on nearly every query without any additional prompting, the ranking results are best when using the complex MultiMedQA prompt. We assume the small differences between the prompting strategies are because the models are already very effective in answering the queries, so the additional prompting does not have a large impact.

Except for ChatGPT, all models are less effective than the next smaller model, if the smaller model uses the MultiMedQA prompt and the larger model uses no additional prompting. This gain in effectiveness emphasizes the importance of a good prompting strategy, even for models that are already fine-tuned for a conversational experience like Falcon and Llama-2.

5.1.3 Query Type

Similar to the influence of prompting strategies, our results show that phrasing the query as a proper question leads to better ranking results than using a keyword-based query. The improvements are consistent across all models, but the difference seems to be less pronounced the larger the model is, as shown in Figure 4.3.

5.1.4 Lower ranked Answers Properties

When investigating why certain answers by ChatGPT are ranked especially low, we found that the lower-ranked answers follow two patterns. They either contain the phrase "As an AI.." or they contain a formatted list. For both patterns, we investigated if they are also present in other LLMs and how they are ranked.

The substring "As an AI.." is present in answers generated by ChatGPT and by Falcon 7B, but not in any of the other models. The phrase is commonly used by ChatGPT and seems to be a part of OpenAI's training data, leading the model to not answer controversial questions or telling the user that it does not have enough information to provide an answer. As Falcon 7B's training data is in part based on self-chats of ChatGPT (see Section 3.4.1), it is not surprising that the Falcon model also uses this phrase. Answers containing this substring are ranked lower than other answers, which indicates that the ranking pipeline works as expected in this case, as the answers most likely are not useful for the user. On the other hand, telling the user that the model is unable to answer the question should be considered a desirable behavior, as the alternative would be to provide a misleading answer. Future research in rank-based implicit evaluation should investigate how this behavior could be evaluated

Answers exhibiting the list pattern are generated by all models, except for GPT-2 and GPT-2 Medium. There seems to be a connection between the model size and the frequency of this pattern, as the larger models generate more answers containing lists. The effect of achieving a worse ranking by using this pattern is not consistent over all models, as the Llama-2 model's answers rank higher when using this pattern. Additionally, many highly ranked answers by ChatGPT also contain lists, so the pattern is not a clear indicator of a bad answer. We assume that observing this pattern in many of the lower-ranked answers is dependent on the queries that prompted the answers, which are all keyword-based queries. Finding more low-ranked answers with the list pattern in keyword-based queries is consistent with our previous finding that keyword-based queries lead to worse ranking results than question-type queries in general.

5.1.5 Answer Length

Figure 4.4 shows that there is a connection between the length of the generated answer and the ranking result. Very short and very long answers are ranked worse than answers of medium length containing between 200 and 300 words. Short answers being ranked worse is consistent with expectations, as the often complex questions require longer answers to be answered sufficiently. Longer answers being ranked worse is more surprising, but closer inspection revealed that most of the overly long answers are generated by GPT-2 variants, explaining the worse ranking results. Changing the parameters for text generation using the GPT-2 models could lead to better results, but we did not investigate this further.

5.1.6 Summary

To answer **RQ1**:

Which factors influence the effectiveness of LLMs when using rank-based implicit evaluation?

we investigate the influence of model size, prompting strategy, query type, answer properties and answer length on the effectiveness of LLM generated answers. We show that model size and prompting strategy have a significant impact on the ranking results, with larger models and more sophisticated prompting strategies leading to better ranking results. We also show that phrasing the query as a question leads to better ranking results than using a keyword-based query. Answer length also has an impact on the ranking results, with very short and very long answers being ranked worse than answers of medium length. Additionally, we investigate the properties of lower-ranked answers but could not find a clear pattern that would explain the worse ranking results. To summarize, with model size, prompting strategy and query type (which could be considered prompting strategy) we find three factors that directly relate to the model or prompt and have been shown to influence the effectiveness of LLMs in other tasks.

5.2 Comparison to Other Benchmarks

We compare the results of our rank-based implicit evaluation to other benchmarks in the literature, as shown in Table 4.4. The chosen benchmarks are single-choice question-answering tasks, which as a task is not directly comparable to our rank-based implicit evaluation. Nevertheless, the order of the models on the benchmark is expected to be similar to the order of the models in our evaluation. The results of our evaluation are in line with the results of the other benchmarks, with ChatGPT ranking best on all of them. While all other models are ordered according to their model size, the two versions of Llama-2 are switched in the ranking results of our evaluation compared to the other benchmarks. We assume that the ranking results of our retrieval pipeline are not accurate enough to distinguish between the two model sizes, either because the retrieval pipeline is not sophisticated enough or because the dataset quality is not high enough both in terms of the web documents quality and the annotations provided by the human annotators.

While ChatGPT ranks best on all benchmarks, our benchmark seems to be less challenging than the other benchmarks, as ChatGPT is already close to the perfect score. For the other benchmarks, there is still a lot more space for improvement, leaving space for more effective models than ChatGPT. Our benchmark could not show differences between ChatGPT and models generating even better answers. To verify if the ChatGPT answers are indeed the best answers, we would need to do additional human evaluation, in which the human annotators would need to compare the highest-ranked LLM-generated answer to the highest-ranked web answer. If the human annotators prefer the LLM-generated answer, we can conclude that ChatGPT indeed already generates the best answers and that we need a more challenging dataset. If the human annotators prefer the web answer, we would need to investigate why our retrieval pipeline ranks the LLM-generated answer higher than the web answer and focus on improving the retrieval pipeline in future work.

5.2.1 Summary

Based on the previous discussion we can answer **RQ2**:

How does the effectiveness of LLMs on existing benchmarks compare to their effectiveness when using rank-based implicit evaluation?

by comparing the ranking results of our evaluation to the ranking results of other benchmarks. We show that the ranking results of our evaluation are in line with the ranking results of other benchmarks, with ChatGPT ranking best on all of them. However, the ranking results of our evaluation are not accurate enough to distinguish between the two model sizes of Llama-2. So while the general trends in the effectiveness of LLMs are similar to other benchmarks, there are still deviations from the expected ranking that need to be investigated in future work.

5.3 Effectiveness of the proposed rank-based implicit evaluation

Based on the results of our evaluation, we can answer the main research question:

Is rank-based implicit evaluation with human-written web content a viable approach for evaluating health answers generated by LLMs?

Our rank-based implicit evaluation method produces ranking results that can be explained by the common factors previously shown to influence the effectiveness of LLMs. Furthermore, the ranking results are in line with other benchmarks in the literature, suggesting that the proposed method can be an effective way to evaluate LLMs.

However, the ranking results are not accurate enough to distinguish between the two model sizes of Llama-2, which indicates that either the ranking results produced by the retrieval pipeline are not accurate enough or that the deficiencies in the dataset quality in terms of the web documents quality and the annotations provided by the human annotators are too large. Additionally, ChatGPT already ranks best on nearly every query, so the dataset used in this thesis is not challenging enough to evaluate and compare more advanced models. Even tough the rank-based implicit evaluation method currently still has its limitations, we believe that it can be a viable approach for evaluating health answers generated by LLMs, especially if the dataset quality is improved and more sophisticated retrieval methods are used.

5.4 Limitations

While the proposed rank-based implicit evaluation method ranks LLMs similarly to other benchmarks in the literature and the ranking results can be explained mostly by the common factors influencing the effectiveness of LLMs, there are several limitations to the method and the evaluation.

- **Reliance on Noisy Web Data** The dataset used for evaluation is based on web data, which is not perfectly suited for comparison with answers generated by LLMs. A more suitable dataset would be based on human-generated answers.
- **General Answer Quality** In addition to the web data being noisy, manual investigation of the human-written answers shows that the general answer quality is not very high, making the benchmark less suitable for evaluating state-of-the-art models.

- **Unreliable Ground Truth** The evaluation assumes that the provided annotations for relevance, credibility and readability are reliable. But as mentioned in Section 3.2.4, there are documents rated by multiple annotators for different queries. They often do not agree on the rating, which makes sense for the relevance rating, but not for the credibility and readability ratings. The disagreements indicate that the provided annotations are not as reliable as they could be with a more consistent annotation process.
- **Retrieval Pipeline Limitations** While monoT5, the most effective retrieval model in our study, achieves nDCG@10 scores of 0.645 on relevance, 0.72 on credibility and 0.81 on readability there is still much room for improvement. We showed that our evaluation method did not rank the two versions of Llama-2 in order of model size, which we attribute to limitations in the retrieval pipeline. Additionally, the retrieval pipeline is not optimized for the task of retrieving answers to health-related questions, which could be improved by using a more sophisticated retrieval model that is fine-tuned on a health-related dataset.
- **Bias of Neural Retrievers** Recent work by Dai et al. (2023) has shown that Neural Retrievers like monoT5 are biased towards retrieving documents generated by LLMs compared to human-written documents. While this bias does not change the results for comparing the LLMs against each other, since all of them have the LLM advantage it could still mean that the rankings are higher than they should be.

Chapter 6

Future Work and Conclusion

6.1 Future Work

In this thesis, we present a first approach to using IR methods for evaluating LLMs in LFQA, specifically in the medical domain. However, there is substantial room for further research and development. Some areas for future work are:

1. **Dataset Quality:** The currently used dataset is not originally designed for LFQA and has some limitations. Those limitations include the generally lower quality of web content as well as the problems in the annotation process. By designing a more comprehensive dataset that addresses these limitations and provides a more challenging benchmark for LLMs we could resolve those limitations of the evaluation method.. An intermediate step could be to use the existing dataset and incorporate more sophisticated preprocessing steps to improve the quality of the data. Redoing the current annotations using a more consistent methodology could also improve the dataset quality.

2. **Retrieval Pipeline Improvements:** The retrieval pipeline used in this thesis is a first attempt at using IR methods for evaluating LLMs in LFQA. Future work could focus on improving the pipeline's effectiveness and investigate closely in which cases the retrieval pipeline does not align with human annotator preferences.

3. **Multidimensional Evaluation:** The currently used retrieval pipeline is only trained to rank documents based on their relevance, which is only one of the dimensions of answer quality. Extending the pipeline to rank documents based on their readability and credibility could improve the evaluation method.

4. **Human Evaluation:** The chosen retrieval pipeline is evaluated on the human-ranked documents from the dataset, but after including the LLM-generated answers the produced ranking is not reevaluated by medical experts. Future work could evaluate if human experts prefer the highest-ranked LLM-generated answers over the highly ranked web answers.

5. **Conversational LLMs:** As the current application of LLMs in chatbots is focusing on conversational experiences, it is important to consider the evaluation of these systems, which is more complex than the evaluation of single answers. A first step could be to concatenate the generated answers over multiple conversation turns and rank the concatenated document against the web documents, essentially evaluating the generated text as a whole.

6. **Adding supplementary material to queries:** As Koopman and Zuccon (2023) have shown, adding supplementary material to the query can heavily bias the generated answer. By adding supplementary material with different biases to the queries, we could investigate if a similar effect can be observed using the rank-based implicit evaluation method.

6.2 Conclusion

In this thesis, we present a first approach of using human-written web content as a proxy for evaluating LLMs in LFQA, specifically in the health domain. We propose a rank-based implicit evaluation method that uses IR methods to rank documents based on their relevance to the question. Our contributions are:

1. Adapting the existing dataset by Goeuriot et al. (2021) to the LFQA task.

2. Evaluating different retrieval pipelines for the new dataset.

3. Producing a supplementary dataset of 16,000 answers for the queries in the dataset, using multiple LLMs and prompting strategies.

4. Evaluating the LFQA effectiveness of the LLMs using the most effective retrieval pipeline.

5. Investigating which factors influence the ranking of the LLM-generated answers.

6. Comparing the effectiveness of LLMs on our benchmark to the effectiveness on other benchmarks.

Although the dataset we adapted for LFQA has several limitations preventing our evaluation method from being a challenging benchmark that allows for fine-grained evaluation of LLMs, this thesis lays the groundwork for a scalable evaluation method for LFQA. The proposed rank-based implicit evaluation method is a solid foundation for future work, which could focus on improving the retrieval pipeline, extending the evaluation to other dimensions of answer quality, and creating a more challenging dataset.

Bibliography

Akoury, N., Wang, S., Whiting, J., Hood, S., Peng, N., and Iyyer, M. (2020). STORIUM: A dataset and evaluation platform for machine-in-the-loop story generation. pages 6470–6484.

Amati, G. (2006). Frequentist and bayesian approach to information retrieval. In *European Conference on Information Retrieval*, pages 13–24. Springer.

Bajaj, P., Campos, D., Craswell, N., Deng, L., Gao, J., Liu, X., Majumder, R., McNamara, A., Mitra, B., Nguyen, T., et al. (2016). MS MARCO: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Banerjee, S. and Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Beeching, E., Fourrier, C., Habib, N., Han, S., Lambert, N., Rajani, N., Sanseviero, O., Tunstall, L., and Wolf, T. (2023). Open LLM Leaderboard. `https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard`.

Beltagy, I., Peters, M. E., and Cohan, A. (2020). Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language Models are Few-Shot Learners. *CoRR*, abs/2005.14165.

Choi, E., He, H., Iyyer, M., Yatskar, M., Yih, W.-t., Choi, Y., Liang, P., and Zettlemoyer, L. (2018). QuAC: Question answering in context. *arXiv preprint arXiv:1808.07036*.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. (2018). Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *ArXiv*, abs/1803.05457.

Dai, S., Zhou, Y., Pang, L., Liu, W., Hu, X., Liu, Y., Zhang, X., and Xu, J. (2023). LLMs may dominate information access: Neural retrievers are biased towards llm-generated texts. *arXiv preprint arXiv:2310.20501*.

Dave, T., Athaluri, S. A., and Singh, S. (2023). ChatGPT in medicine: an overview of its applications, advantages, limitations, future prospects, and ethical considerations. *Frontiers in Artificial Intelligence*, 6:1169595.

De Choudhury, M., Morris, M. R., and White, R. W. (2014). Seeking and sharing health information online: comparing search engines and social media. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1365–1376.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language understanding. *CoRR*, abs/1810.04805.

Dugan, L., Ippolito, D., Kirubarajan, A., and Callison-Burch, C. (2020). RoFT: A tool for evaluating human detection of machine-generated text. *arXiv preprint arXiv:2010.03070*.

Duh, K. (2008). Ranking vs. regression in machine translation evaluation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 191–194.

Fan, A., Jernite, Y., Perez, E., Grangier, D., Weston, J., and Auli, M. (2019). ELI5: Long form question answering. *arXiv preprint arXiv:1907.09190*.

Freund, Y., Iyer, R., Schapire, R. E., and Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of machine learning research*, 4(Nov):933–969.

Goeuriot, L., Suominen, H., Pasi, G., Bassani, E., Brew-Sam, N., Sáez, G. N. G., Kelly, L., Mulhem, P., Seneviratne, S., Upadhyay, R., Viviani, M., and Xu, C. (2021). Consumer Health Search at CLEF eHealth 2021.

Guzmán, F., Joty, S., Màrquez, L., and Nakov, P. (2019). Pairwise neural machine translation evaluation. *arXiv preprint arXiv:1912.03135*.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. (2020). Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142.

Khan, R. A., Jawaid, M., Khan, A. R., and Sajjad, M. (2023). ChatGPT-Reshaping medical education and clinical management. *Pakistan Journal of Medical Sciences*, 39(2):605.

Khashabi, D., Chaturvedi, S., Roth, M., Upadhyay, S., and Roth, D. (2018). Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262.

Khattab, O. and Zaharia, M. (2020). 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.

Kincaid, J. P., Fishburne Jr, R. P., Rogers, R. L., and Chissom, B. S. (1975). Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel.

Kočiskỳ, T., Schwarz, J., Blunsom, P., Dyer, C., Hermann, K. M., Melis, G., and Grefenstette, E. (2018). The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Koopman, B. and Zuccon, G. (2023). Dr ChatGPT tell me what I want to hear: How different prompts impact health answer correctness. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15012–15022.

Krishna, K., Roy, A., and Iyyer, M. (2021). Hurdles to progress in long-form question answering. *arXiv preprint arXiv:2103.06332*.

Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., et al. (2019). Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Li, M., Jiang, A., and Wang, M. (2013). Listwise approach to learning to rank for automatic evaluation of machine translation. In *Proceedings of Machine Translation Summit XIV: Papers.*

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Lipani, A., Palotti, J., Lupu, M., Piroi, F., Zuccon, G., and Hanbury, A. (2017). Fixed-cost pooling strategies based on IR evaluation measures. In *Advances in Information Retrieval: 39th European Conference on IR Research, ECIR 2017, Aberdeen, UK, April 8-13, 2017, Proceedings 39*, pages 357–368. Springer.

Macdonald, C., McCreadie, R., Santos, R. L., and Ounis, I. (2012). From puppy to maturity: Experiences in developing Terrier. *Proc. of OSIR at SIGIR*, pages 60–63.

Macdonald, C. and Tonellotto, N. (2020). Declarative Experimentation inInformation Retrieval using PyTerrier. In *Proceedings of ICTIR 2020.*

Manning, C. D. (2009). *An introduction to information retrieval.* Cambridge university press.

Mihaylov, T., Clark, P., Khot, T., and Sabharwal, A. (2018). Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789.*

Moffat, A. and Zobel, J. (2008). Rank-biased precision for measurement of retrieval effectiveness. *ACM Transactions on Information Systems (TOIS)*, 27(1):1–27.

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

Nogueira, R., Yang, W., Cho, K., and Lin, J. (2019). Multi-stage document ranking with BERT. *arXiv preprint arXiv:1910.14424.*

OpenAI (2023). GPT-4 Technical Report.

Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., and Johnson, D. (2005). Terrier Information Retrieval Platform. In *Proceedings of the 27th European Conference on Advances in Information Retrieval Research*, ECIR'05, page 517–519, Berlin, Heidelberg. Springer-Verlag.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Penedo, G., Malartic, Q., Hesslow, D., Cojocaru, R., Cappelli, A., Alobeidli, H., Pannier, B., Almazrouei, E., and Launay, J. (2023). The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

Pennington, J., Socher, R., and Manning, C. (2014). GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Rajpurkar, P., Jia, R., and Liang, P. (2018). Know What You Don't Know: Unanswerable Questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Reynolds, L. and McDonell, K. (2021). Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–7.

Roberts, A., Raffel, C., Lee, K., Matena, M., Shazeer, N., Liu, P. J., Narang, S., Li, W., and Zhou, Y. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer.

Robertson, S. (2004). Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of documentation*, 60(5):503–520.

Rosa, G., Bonifacio, L., Jeronymo, V., Abonizio, H., Fadaee, M., Lotufo, R., and Nogueira, R. (2022). In defense of cross-encoders for zero-shot retrieval. *arXiv preprint arXiv:2212.06121*.

Sakai, T. (2023). SWAN: A Generic Framework for Auditing Textual Conversational Systems. *arXiv preprint arXiv:2305.08290*.

Santhanam, K., Khattab, O., Saad-Falcon, J., Potts, C., and Zaharia, M. (2021). Colbertv2: Effective and efficient retrieval via lightweight late interaction. *arXiv preprint arXiv:2112.01488*.

Singhal, K., Azizi, S., Tu, T., Mahdavi, S. S., Wei, J., Chung, H. W., Scales, N., Tanwani, A. K., Cole-Lewis, H., Pfohl, S., Payne, P., Seneviratne, M., Gamble, P., Kelly, C., Schärli, N., Chowdhery, A., Mansfield, P. A., y Arcas, B. A., Webster, D. R., Corrado, G. S., Matias, Y., Chou, K., Gottweis, J., Tomasev, N., Liu, Y., Rajkomar, A., Barral, J. K., Semturs, C., Karthikesalingam, A., and Natarajan, V. (2022). Large Language Models Encode Clinical Knowledge. *CoRR*, abs/2212.13138.

Singhal, K., Tu, T., Gottweis, J., Sayres, R., Wulczyn, E., Hou, L., Clark, K., Pfohl, S., Cole-Lewis, H., Neal, D., et al. (2023). Towards expert-level medical question answering with large language models. *arXiv preprint arXiv:2305.09617*.

Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Thakur, N., Reimers, N., Daxenberger, J., and Gurevych, I. (2020). Augmented sbert: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. *arXiv preprint arXiv:2010.08240*.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. (2023).

Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention Is All You Need. *CoRR*, abs/1706.03762.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. (2022). Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Xu, C., Guo, D., Duan, N., and McAuley, J. (2023a). Baize: An open-source chat model with parameter-efficient tuning on self-chat data. *arXiv preprint arXiv:2304.01196*.

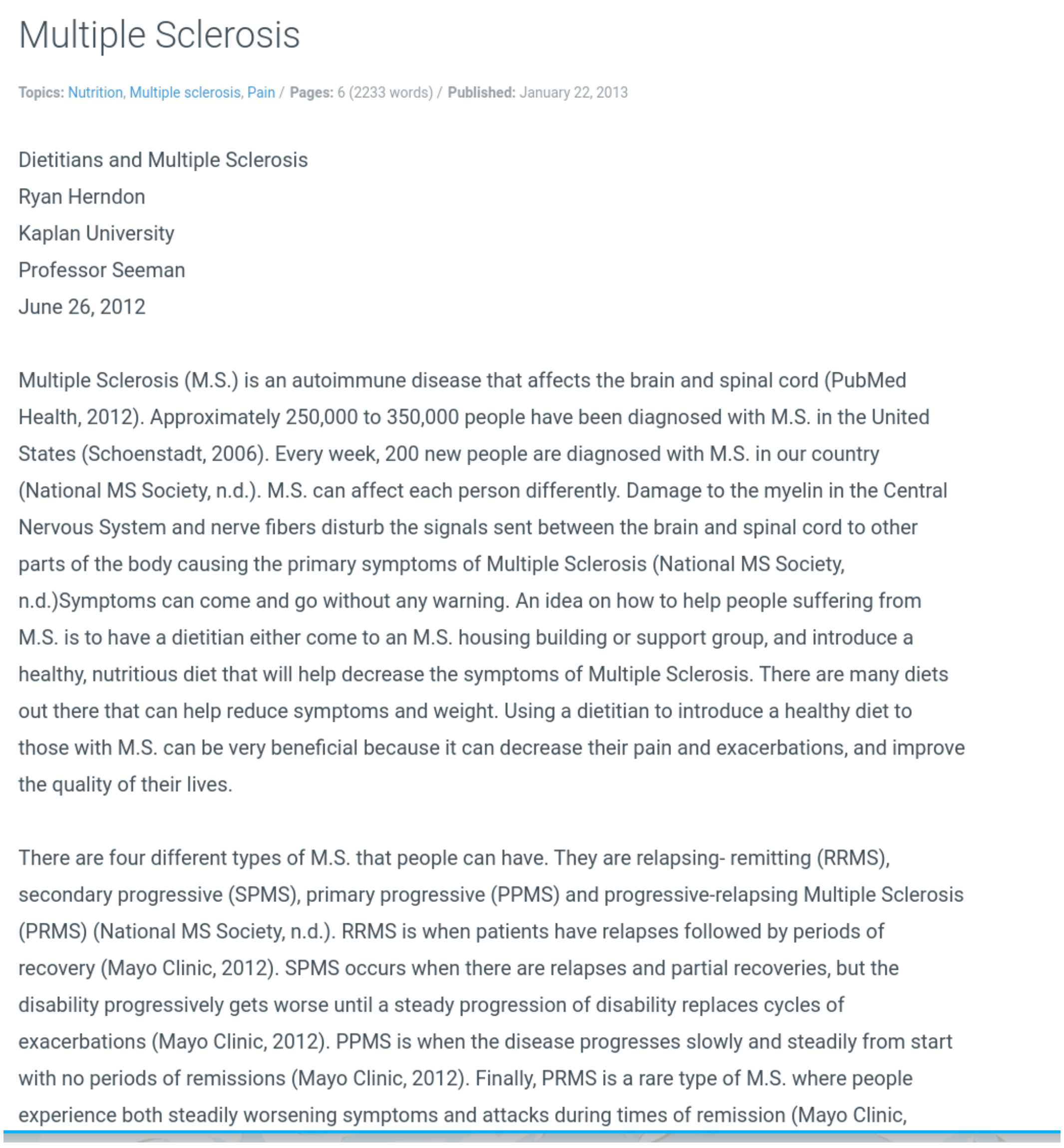
Xu, F., Song, Y., Iyyer, M., and Choi, E. (2023b). A critical evaluation of evaluations for long-form question answering. *arXiv preprint arXiv:2305.18201*.

Yuan, W., Neubig, G., and Liu, P. (2021). Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34:27263–27277.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. (2019). HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Appendix A

Dataset



Multiple Sclerosis

Topics: Nutrition, Multiple sclerosis, Pain / Pages: 6 (2233 words) / Published: January 22, 2013

Dietitians and Multiple Sclerosis
Ryan Herndon
Kaplan University
Professor Seeman
June 26, 2012

Multiple Sclerosis (M.S.) is an autoimmune disease that affects the brain and spinal cord (PubMed Health, 2012). Approximately 250,000 to 350,000 people have been diagnosed with M.S. in the United States (Schoenstadt, 2006). Every week, 200 new people are diagnosed with M.S. in our country (National MS Society, n.d.). M.S. can affect each person differently. Damage to the myelin in the Central Nervous System and nerve fibers disturb the signals sent between the brain and spinal cord to other parts of the body causing the primary symptoms of Multiple Sclerosis (National MS Society, n.d.)Symptoms can come and go without any warning. An idea on how to help people suffering from M.S. is to have a dietitian either come to an M.S. housing building or support group, and introduce a healthy, nutritious diet that will help decrease the symptoms of Multiple Sclerosis. There are many diets out there that can help reduce symptoms and weight. Using a dietitian to introduce a healthy diet to those with M.S. can be very beneficial because it can decrease their pain and exacerbations, and improve the quality of their lives.

There are four different types of M.S. that people can have. They are relapsing- remitting (RRMS), secondary progressive (SPMS), primary progressive (PPMS) and progressive-relapsing Multiple Sclerosis (PRMS) (National MS Society, n.d.). RRMS is when patients have relapses followed by periods of recovery (Mayo Clinic, 2012). SPMS occurs when there are relapses and partial recoveries, but the disability progressively gets worse until a steady progression of disability replaces cycles of exacerbations (Mayo Clinic, 2012). PPMS is when the disease progresses slowly and steadily from start with no periods of remissions (Mayo Clinic, 2012). Finally, PRMS is a rare type of M.S. where people experience both steadily worsening symptoms and attacks during times of remission (Mayo Clinic,

Figure A.1: Sample of a web document from the dataset. As a general article about the topic of Multiple Sclerosis, it is annotated for multiple queries that all relate to the topic. The different annotations can be found in the following table.

Query	Rel	Cred	Read
Covid-19 vaccine & MS drugs	0	1	2
MS transmission to family	0	1	2
Reading issues & MS	0	1	2
Menopause & MS symptoms	0	3	2
Improvement timeline in MS	1	0	1
MS, sleep issues, & forgetfulness	1	1	1
Relapsing-remitting MS	1	1	1
MS development risk	1	1	2
MS fatigue causes	1	2	1
MS symptoms list	1	2	2
Working full-time with MS	1	2	2
Secondary progressive MS	1	3	1
Diagnosing MS relapse	2	1	1
MS impact on career	2	1	2
Managing MS	2	2	2

Table A.1: The different annotations for the sample web document in the dimensions of relevance, credibility and readability. The document was annotated by different people for different queries, resulting in different annotations for the same document. While having different annotations makes sense for the relevance dimension, it is not ideal for the credibility and readability dimensions, as those should be consistent across queries.