Enhancing Practice Problem Retrieval with Deep Learning: A Rewriter-Retriever-Reranker Approach

Stanford CS224N Custom Project

Charles Joyner, Ronny Junkins, Mack Smith
Department of Computer Science
Stanford University
chasezj@stanford.edu, rjunkins@stanford.edu, macks26@stanford.edu

Abstract

Our objective is to build a homework problem recommendation system which, given an input by a user specifying the subject and topics involved, returns the top matches of homework problems within our constructed database. We implement a Rewriter-Retriever-Reranker model following standard principles of information retrieval in machine learning, and evaluate the results using a novel evaluation metric.

1 Key Information to include

- Mentor: Archit Sharma
- External Collaborators (if you have any): NA
- Sharing project: NA

2 Introduction

In recent years, information access has become one of the fundamental daily needs of human beings. Whether individuals wish to search for new clothing or ask for directions to the nearest coffee shop, most everyday tasks now require the need for efficient information retrieval. Consequently, multiple information retrieval (IR) systems, such as search engines, have been developed to accommodate the surge in information demand. From general queries to specific applications, these systems play a crucial role in modern life. That being said, last quarter, one of our group members Ronny Junkins built an information retrieval application to help with his studying process for his CS 229 Project. In the educational setting, finding studying resources such as practice problems other than those provided through the course can be often very difficult or time-consuming for students. As a result, he designed an application that, when prompted with a query, searched for and returned the most relevant practice problems from various datasets of questions. However, given the limited resources, time, and information concerning natural language processing, the application still possessed lots of room for improvement. Therefore, our team decided to improve the application's capabilities by re-implementing it using a Rewriter-Retriever-Reranker approach, alongside other natural language processing techniques.

Originally, the CS 229 Project approach involved primarily unsupervised learning methods, including the use of Sentence Bert to generate an embedding space for questions and queries followed by a type of clustering such as k-means or hierarchical clustering. Quantitatively the method seemed to work, but the project struggled significantly with evaluation, and developed no frame of reference for its performance. All types of evaluation depended on labeled training set examples. While the methods used were powerful, they were also relatively unrefined, and left significant room for improvement. We use a version of this model as our baseline, and now attempt to develop models to exceede its performance.

We hypothesize that the Rewriter-Retriever-Reranker approach Zhu et al. (2024) will serve as a better architecture for the application. In summary, the application takes in a query, rewrites it to better align with the practice questions in the dataset, retrieves the most relevant questions, and re-ranks such questions so that they are better ordered by relevancy to the original query. To train and implement the Rewriter-Retriever-Reranker, we utilized the SciQ Dataset Welbl et al. (2017) to serve as our bank of practice problems and additionally generated a synthetic dataset of plausible queries with a Text-to-Text Transfer Transformer Raffel et al. (2023) and Meta's Llama 3 8b Chat model Meta (2024). Additionally, with the queries produced from docTTTTTquery, we rewrote them using Meta's Llama 3 8b Chat model as well with zero-shot prompt engineering: we also briefly used Gemini-flash as a proof of concept for zero-shot learning in our reduced datasetTeam et al. (2024). Once we had our queries synthesized and rephrased, we deployed our synthetic dataset to train and evaluate a retriever model pretrained on the MS-MARCO Dataset Bajaj et al. (2018) from HuggingFace Reimers and Gurevych (2019). Once the retriever had been trained and evaluated, we fed test data from the SciQ Dataset into a reranker model also pretrained on the MS-MARCO and from HuggingFace to determine whether or not re-ranking the relevancy of the documents against each other and the query improved how the questions were provided to a user. From these experiments, we found that for the smaller dataset generated by Gemini-Flash, the baseline model performed significantly better than the other models. However, when we increase the size of the dataset and improve the quality of the synthetic queries, we see significant improvement in Retriever-Reranker models while the performance of the baseline model stagnates.

3 Related Work

In order to keep pace with the growing necessity for information retrieval (IR) systems, alongside the ever-growing volume of data and user demands concerning such applications, it is imperative that modifications and improvements are continually made towards information retrieval systems. Especially with the rise of Large Language Models (LLMs), new opportunities to better the efficiency of such IR systems have become more available. For instance, LLMs can aid in rewriting queries so that they can be better fit to retrieve documents from the data source Zhu et al. (2024). For our project, we decided our own version of a query rewriting using a LLM to improve our retriever's ability to select relevant practice problems.

Additionally, in order to develop IR applications for specific demands, highly specific data surrounding the queries and the documents are necessary to implement an efficient and successful system. However, the datasets to which users have access to are either too general for their intention or not even available. Therefore, new models have been introduced to entirely synthesize datasets such that these IR applications can have the highly specific information they need. For the sake of this project, we chose to deployed Text-to-Text Transfer Transformer (T5) Raffel et al. (2023) as the model to generate our synthetic dataset. Essentially, when finetuned on question answering prompts, T5 can generate synthetic questions for specific documents so that a retriever model can be trained and evaluated.

4 Approach

4.1 Baseline Approach

As a comparison metric, we took our original implementation from the CS229 Final Project framework and simplified it somewhat, removing the unsupervised learning components. This model uses Sentence-BERT (SBERT) Reimers and Gurevych (2019) to embed all questions and queries into an embedding space and manually computes the pairwise distance between each query and each question in the embedding space. It then returns the top k matches based on the highest cosine similarity.

4.2 Main Approach

4.2.1 Query Generation and Rewriting

Baseline Query Generator: An issue we will elaborate on further in the paper is that we are working with a limited dataset in terms of size. We have 13,700 data points which is a relatively tiny amount of data for NLP purposes, and ergo restricts the ability of our model's trained T5 generator to

fully understand language. To act as a baseline which is not contingent on the T5 generator's access to data we deployed an independent LLM to use as query generator. For this we deployed Meta's Llama 3 8b Chat model Meta (2024). We used zero shot learning to extract the topic from each question in our dataset, and then using a template and a rephrase request ("Can you find more questions about <topic>?") to generate a query for the question. For this baseline we did not use a separate rephrase step, as there is already an implicit rephrase built into our zero-shot prompt engineering. Below are two examples of zero-shot generated queries.

Question: What remains a constant of radioactive substance over time?
Generated Query: Are there additional questions about Half-Life available?

Question: What do you call a structure composed of two or more types of tissues that work together to do a specific task?
Generated Query: Can you locate additional questions related to the organ?
As you can see from the second example the results from Llama are not infallible and can have incorrect grammar. However the baseline queries generally capture the correct topic and form a query easily understood by humans.

As inspiration in desigining our main approach, we drew upon previous techniques from modern day information retrieval systems and new opportunities for LLMs to improve such techniques Zhu et al. (2024). One such way LLMs have been used to improve the quality and efficiency of IR systems is through query rewriting. Suppose we have a query q for which we want to find the k most revelant documents from dataset D. Utilizing an LLM, we can ask it to rewrite q such that the new query $\hat{q}$ has no vocabulary and spelling mistakes, consequently improving the retriever's ability to find relevant documents.

T5 Query Generator: Another idea that had been proposed for finding queries related to each practice problem was to utilize the T5 model to generate synthetic queries based on each problem's answer and support for the answer. Simply put, the T5 model is a encoder-decoder model that can perform a multitude of tasks, including translation and summarization, for which each task is converted to a text-to-text format. That being said, for our intended purposes, we wanted to experiment with using the T5 model for question answering. In order to do so, we attempted to finetune the T5 model on the Stanford Question Answering Dataset (SQuAD) Rajpurkar et al. (2018) where the input text had been formatted with prefixes "<answer>" and "<context>" such that the input text looks like "<answer> **answer of the question** <context> **context of the question**" and used to predict the original question asked. However, due to constraints in GPU RAM and the tedious process of finetuning the model, our group decided that our baseline approach to generating queries would be best.

4.2.2 Retriever

The objective of the Retriever is to find the k most relevant (closest) documents from D to our query q. However, to compare the similarity of q to the documents in D, both q and D must be represented in vector space using sentence embeddings. For our retriever, we utilize a Bi-Encoder the pretrained msmarco-distilbert-base-v4 model for semantic search to encode q and questions from D with SBERT embeddings and compute their distance from each other using cosine similarity Reimers and Gurevych (2019). With these metrics, we collect the top k most relevant documents $d_1, \ldots, d_k$, and re-rank them using a re-ranker.

4.2.3 Reranker

Given the top k most relevant documents $d_1, \ldots, d_k$, it is possible that the retriever may return irrelevant documents. Thus, we employ a Reranker to overcome this by utilizing a pretrained ms-marco-MiniLM-L-6-v2 Cross-Encoder, which takes in the query q and document d_i for $i \in \{1, \ldots, k\}$ simultaneously and produces a score from 0 to 1 characterizing their similarity Reimers and Gurevych (2019). With these scores, we reduce the original k documents to a smaller subset.

5 Experiments

5.1 Data

We used the SciQ Welbl et al. (2017) dataset. This dataset consists of 14,000 crowd sourced science questions. We then use large language models to generate two synthetic datasets, one small synthetic dataset for testing purposes and one larger dataset for more comprehensive evaluation. To generate the small dataset, which consists of 1,291 data points broken down into 1,083 training, 97 testing, and 111 validation examples, we use a LLM, Gemini-flashTeam et al. (2024), to generate a query for each data point in the dataset. A query is a synthesized request to find problems, meant to simulate a real user request for practice questions from the dataset. The SciQ dataset provides answers to all questions in the dataset. We use these answers as inputs to Gemini-flash and give it the task of generating a reasonable user request that could have been seeking the question in the dataset with this answer. An example data point in our dataset is below, of the form (synthetic query, question from dataset, answer): **'Can you find more questions about skeletal structure?', 'What are the location where bones come together?', 'joints'.** Our hybrid dataset perfectly captures out task because it allows the model to see the relationship between a query and a question that is relevant. Lastly, utilizing the T5 model Raffel et al. (2023) finetuned on the SQuAD Dataset Rajpurkar et al. (2018) as discussed in the Approach section, we took answers and supports from such answers to generate questions concerning certain practice problems and treated those synthetic questions as our queries.

5.2 Evaluation method

As our problem and setting are entirely unique, we must develop custom evaluation metrics to determine the quality of results outputted by our model. That is, given a query and the top k results outputted by the model, how do we quantify how satisfied a user would be with those results? We develop multiple evaluation metrics, and assess each based on its consistency with the others.

The primary evaluation metric we deployed was precision of the top-k results as adjudicated by an LLM judge. Due to the nature of our task (IR), it is impossible to have a ground truth top-k documents for each of the infinite possible queries. Even if we were able to manually compute a top-k ground truth, it would be subjective (there is no objective way to rank documents for a query otherwise IR would not be a challenge), a single added document would force each top-k ground truth to be re-evaluated, and there would be no way to evaluate new queries: e.g. when the model is deployed or when working with the sensitive parts of the dataset which we should not directly access(testing or validation splits). Ergo we required a more flexible measure that is robust to unseen queries and documents. Thus we deployed an independent LLM model (known as a LLM judge) to grade each top-k documents our model produces based on its precision. The LLM judge looks at each of the top-k documents produced and gives each a binary score, 1 indicating the document is relevant to the query and 0 when the opposite is true. The precision of the model is the average score given: the closer precision is to 1 the higher the rate the model retrieves relevant documents and the better our model. For our evaluation we used zero-shot learning using the pretrained weights of Mistral-7B-Instruct-v0.3 model from Mistral AIJiang et al. (2023). Note that we intentionally chose not to reuse Llama, as our baseline queries are made with Llama the model likely has bias towards its own queries. This would cause the LLM judge to not evaluate all outputs in a consistent manner. Mistral is a instruct model and thus it preforms excellently at succinctly following directions without adding unnecessary noise(many chat-bots add additional tokens such as "Of course! or "I'd love to help"). Mistral was trained on a a large amount data scraped from the web, giving the model substantial general knowledge and understanding of relevance. Additionally, it is relatively light weight at only 7 billion parameters allowing us to quickly and cost-effectively query the LLM judge. Below are a positive and negative examples of the LLM judge's evaluation.

Query: Are there additional questions about plant hormones?
Question: Phytochromes regulate many plant responses to what?
LLM Score: 1
Note this provide evidence that our LLM judge has a pretty high command over general information. It is able to correctly identify that although phytochromes are essential for plant regulation they are pigments and not hormones.

Query: I'd like to explore more scientific concepts related to oxidants. Can you provide me with additional problems or examples?
Question: What in science are very important so that we can compare experimental data from one lab to another and make sure we all are talking about the same thing?
LLM Score: 0

We further employ two secondary evaluation metrics. The first is SacreBLEU, a variation of the classical BLEU machine language translation metric. This metric essentially counts the exact word and phrase matches between the inputted query and outputted results of the model. We employ this metric because it is reasonable to assume that better results from the model will be on similar topics as the query, and will on average have more word matches than results that are on entirely different topics. For example, given the user query, **'Could you provide me with additional scientific questions related to the vertebral column (backbone)?'**, we expect more accurate results than inaccurate will feature words such as 'vertebrae' and 'backbone'. We of course expect that the SacreBLEU scores will be much lower than typical machine translation BLEU scores as there a multitudes of questions that are accurate results for a given query but that don't use the exact wording of the query.

The last evaluation metric we consider is Cosine Precision at k = 5. This metric is calculated as follows. Consider the test set, which is made up of (synthetic query, target document) pairs. We assume that the target document is the best match for the synthetic query, and combine all target documents in the test set into a small set of subset of documents. We input the query into the model, evaluate on this subset of documents, and compute precision based on whether or not the target document is in the top 5 results.

5.3 Experimental details

For fine-tuning on the small synthetic dataset, we use 3 epochs, 40 warmup steps, a learning rate of 2×10^{-5}, and evaluation every 10 steps. We assume that each document (question) in the original training set should be close to the synthetic query we have generated for it, and hence use Multiple Negatives Ranking Loss as our training objective. Training progress is tracked using the Cosine Precision at k = 5 evaluation metric as described in Section 5.2 evaluated on the validation set. Training time is less than 5 minutes.

For both the small synthetic dataset and the large synthetic dataset, the evaluation task is as follows. The datasets are made up of (synthetic query, document) pairs. We combine all the documents in the training set into a large "corpus" to simulate a large dataset of questions. Then, for each synthetic query in the test set, we have each model return the top 5 results in the corpus matching the query. These results are then passed to the LLM Judge and SacreBLEU as defined above.

For the small synthetic dataset, we evaluate the "msmarco-distilbert-base-v4" model with and without finetuning, with and without the Cross Encoder, and with and without a fixed maximum sequence length of 256 tokens to measure the effectiveness of these components of the model. We further compare with "all-MiniLM-L6-v2", the default Sentence Transformer model, and AllenAI-SPECTER Cohan et al. (2020) to see how our model does compared to these other pretrained models.

For the large synthetic dataset, we use the same hyperparameters as the small dataset but increase the number of warmup steps to 437. Loss is the same and progress is tracked in the same way. Finetuning takes approximately one hour to complete on this training set.

During experimentation on the large synthetic dataset, for the purposes of time and resources, we only choose to evaluate the MSMarco model, as it was the highest performing on the smaller dataset. We again evaluate with and without fine-tuning, and with and without the cross encoder and observe performance.

5.4 Results

When our model and the others were evaluated on the small synthetic dataset, the baseline clearly outperformed all others overall, see Figure 1. Fine-tuning had a very erratic impact on performance, in some cases improving the performance, as with AllenAI-SPECTER, and in other cases worsening performance, as with MSMarco. Reflecting this, the evalution on test set metric was also unpredictable. We can see evidence for why this should be in Figure 4, Figure 5, and Figure 6, which depict how the three different models performed on the evaluation set over the course of training. While it seems as

if the MSMarco made some improvements, these improvements didn't translate to the test set, and it is clear in all models that performance was impacted in a very chaotic way by fine tuning. Adding the cross encoder seemed to improve performance to a limited extent. The MSMarco model overall outperformed the other models excluding the baseline.

On the large dataset, we see a dramatic change, see Figure 2. Excluding the baseline, all scores with LLM Judge and SacreBLEU are significantly higher than on the smaller dataset. Fine tuning the model and the addition of the cross encoder add significant improvements to the score when compared with leaving these pieces out. We notice from Figure 7 that the loss while training on the large synthetic dataset shows significant improvement when compared to pre-training levels, while the performance on the evaluation dataset in Figure 8 similarly shows good improvement. The performance of the baseline model is now quite comparable to our model.

In figure 3, we further confirm these findings by analyzing the performance of the models on the test set before and after training. It seems that MSMarco benefits the most from fine-tuning on the training dataset in both the case of the large dataset and the small dataset, while "all-MiniLM-L6-v2" is shows a significant decrease in performance. It is expacted that all values are very low as this is a very challenging task (see Section 5.2).

We furthermore see that our evaluation metrics LLM Judge and SacreBLEU Score show remarkably good agreement when analyzing which models are providing significantly better results than others. This is comforting because while the LLM Judge is much more powerful, it is also the more opaque evaluation metric while SacreBLEU is very easy to understand. Hence, when the two models agree, we can be confident that scores they are giving the model's results are reasonable.

	LLM Judge	SacreBLEU Score
Baseline	0.503	2.893
MSMarco (No Finetune, No Cross Encoder)	0.105	2.258
MSMarco (No Finetune, Cross Encoder)	0.128	2.304
MSMarco (Finetune, No Cross Encoder)	0.052	2.269
MSMarco (Finetune, Cross Encoder)	0.072	2.219
all-MiniLM-L6-v2, No Finetune, No Cross Encoder	0.076	2.289
allenai-specter, Finetune, No Cross Encoder	0.066	2.286
allenai-specter, No Finetune, No Cross Encoder	0.070	2.379
allenai-specter, Finetune, No Cross Encoder	0.074	2.222

Figure 1: Evaluation of models on Small Synthetic Dataset

6 Analysis

For the following analysis I decided to focus on our best preforming experiment: our model finetuned on all of the training dataset and using the cross encoder(46.9% LLM judge precision). Our model

	LLM Judge	SacreBLEU Score
Baseline	0.492	4.479
MSMarco (No Finetune, No Cross Encoder)	0.400	4.096
MSMarco (No Finetune, Cross Encoder)	0.455	4.041
MSMarco (Finetune, No Cross Encoder)	0.456	4.083
MSMarco (Finetune, Cross Encoder)	0.469	4.029

Figure 2: Evaluation of models on Large Synthetic Dataset

	Pre-Training Eval on Test Set (cosine precision at k = 5)	Post Fine tuning Eval on Test Set (cosine precision at k = 5)
MSMarco (Small Synthetic)	0.0220	0.0257
allenai-specter (Small Synthetic)	0.0183	0.0312
all-MiniLM-L6-v2 (Small Synthetic)	0.0404	0.0294
MSMarco (Large Synthetic)	0.0020	0.0026

Figure 3: Pre and Post Fine tuning performance on Test Set

appears to heavily weigh similarity between the query and document over actual meaning: this is likely an artifact of our relatively small training dataset pool. Below is an example.

Query: What other questions are there about Elevation?
Document Retrieved: Screws move objects to a higher elevation by increasing what?

Although both query and document contain the word "elevation", the LLM judge correctly judges the document as irrelevant. The important distinction here is that the word "elevation" is not the key topic when used in the context of a screw raising objects(the real topic is likely somewhere along the lines of simple machines). The typical human would use the topic "elevation" to get more questions about heights on the earth's surface and not "elevating" objects. An example of conflation of structure is below.

Query: What other questions are there about Elevation?
Document Retrieved: Altitude is height above what?

The words "elevation" and "altitude" are used in very similar structures, as both words are used to describe position in the world and are typically surrounded by descriptors like above, below, high, and low. Despite their similar use, the words are not synonymous. Altitude refers to height positions in the atmosphere and elevation to heights on the surface(a hiker reaches an elevation and a plane flies at an altitude). The model's selection of the above document provides substantial evidence that the model weighs similar structure and use heavily when it should be paying high attention to the meaning of the words. Our model conflates word matches and similar structure with high

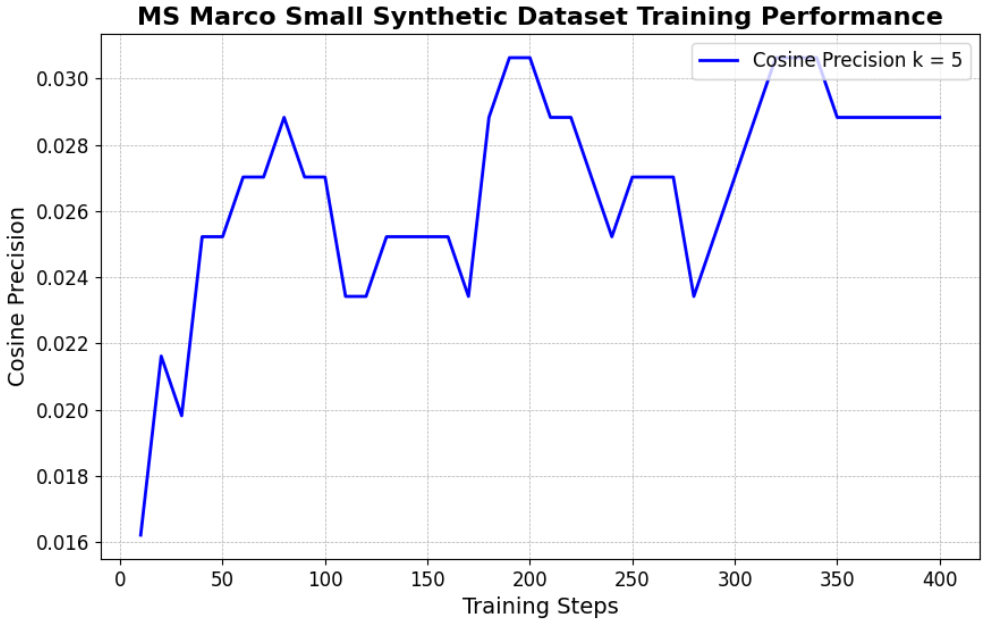

Figure 4: MS Marco Performance on Evaluation Set during Training

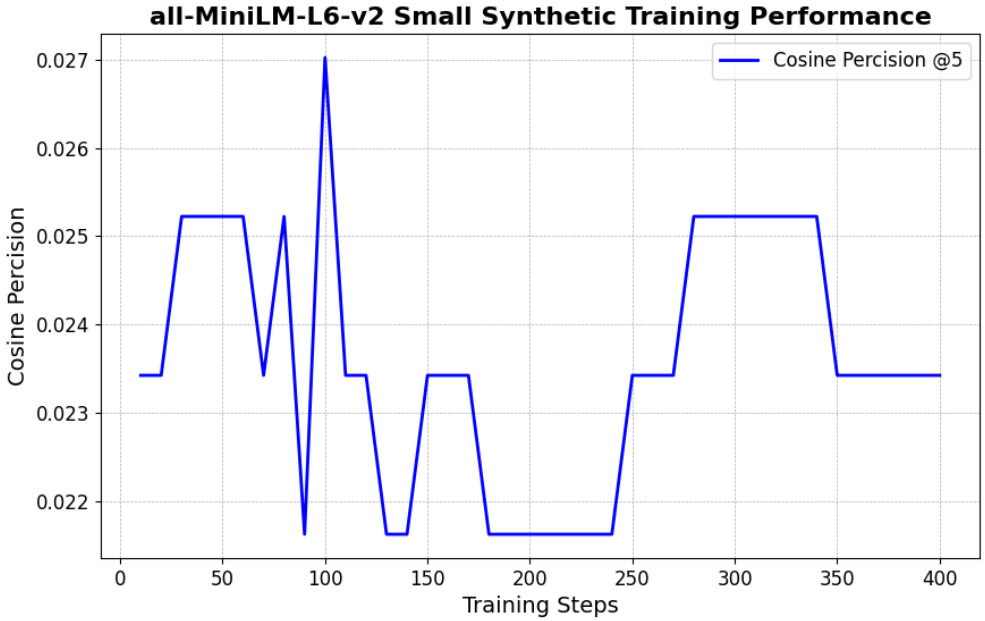

Figure 5: all-MiniLM-L6-v2 Performance on Evaluation Set during Training

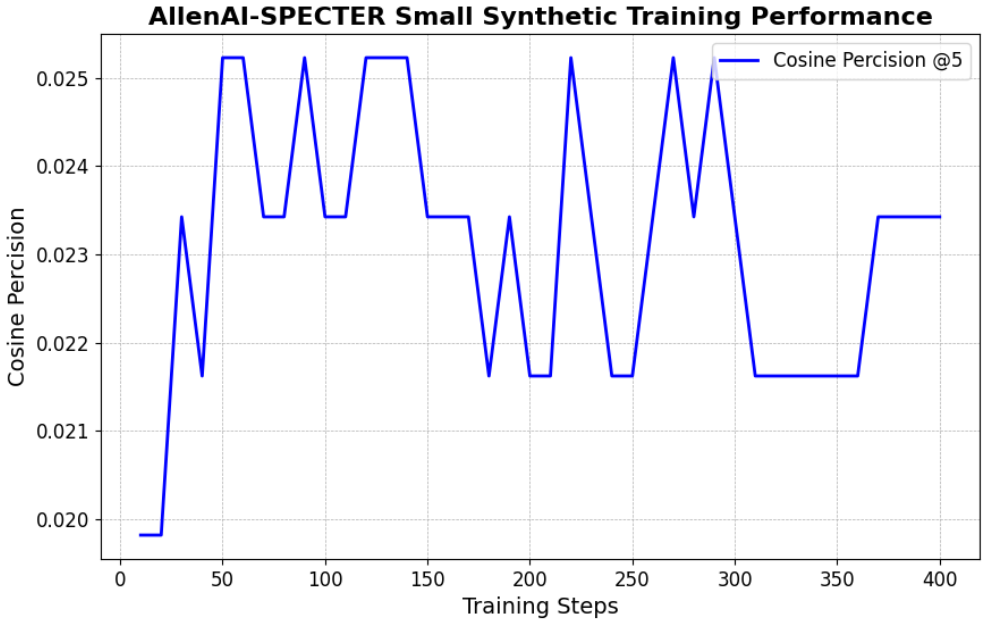

Figure 6: AllenAI-SPECTER Performance on Evaluation Set during Training

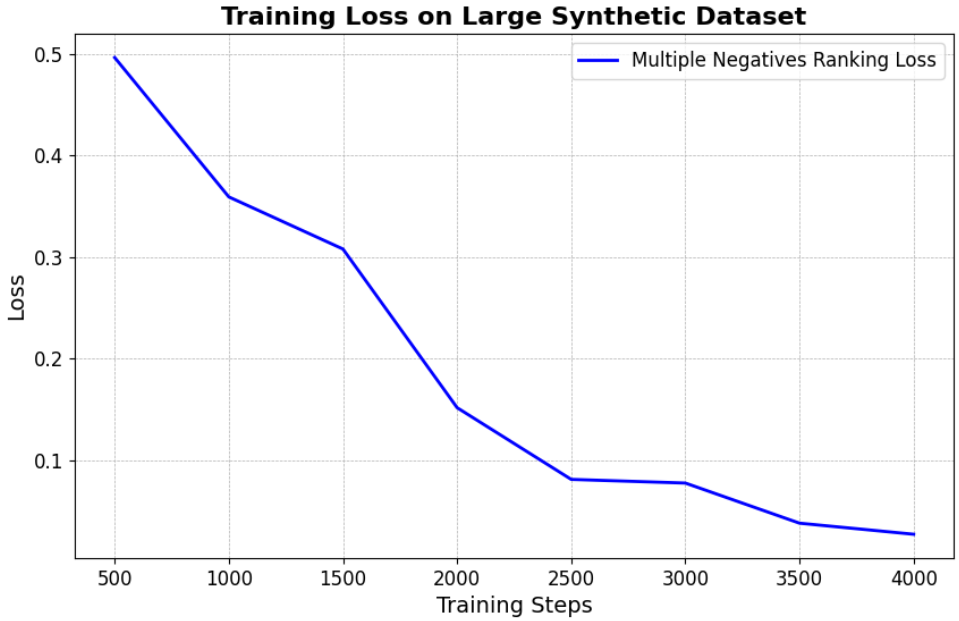

Figure 7: MS Marco loss during fine-tuning on the large synthetic dataset

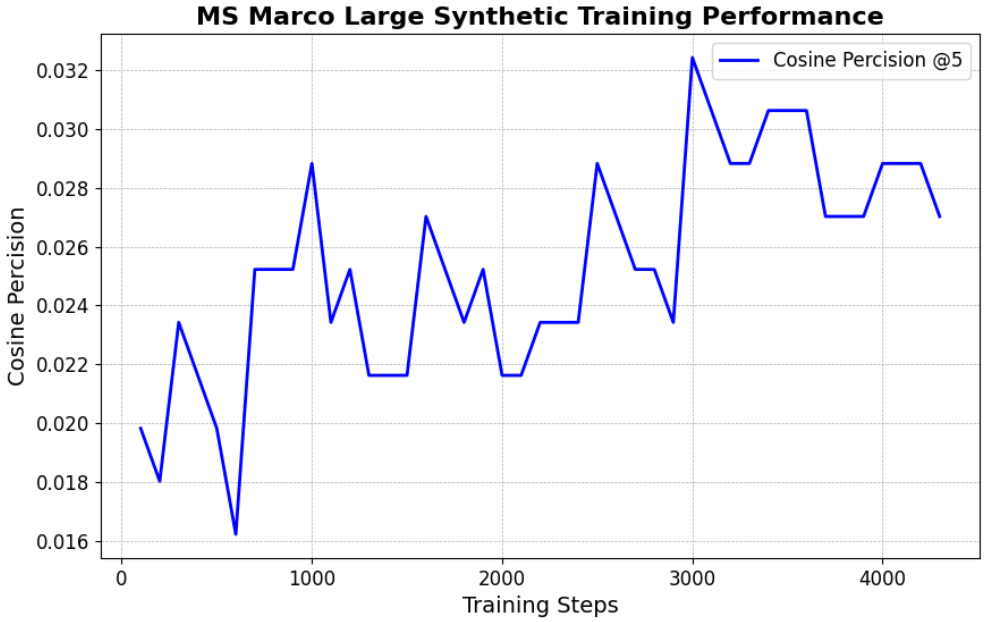

Figure 8: MS Marco Performance on Evaluation Set during Training

relevance, which is an important behavior to move away from in the future. Although it makes sense how our model could easily fall into the pitfall of over emphasizing structure and word matches. Our model has a very limited knowledge base(small training set of 10,000 examples) and a small number of parameters, thus it would be impossible for it to memorize any knowledge base without extreme over fitting on the training set. Thus, because structure and word similarity can be computed without having to memorize any knowledge it makes sense that our model so heavily emphasizes the aforementioned qualities. To fix this in the future we can drastically expand our model's size, giving it the ability to memorize the meanings of key topics, as well as expanding our training set, exposing our model to a wider range of concepts and meanings.

7 Conclusion

From our results we can surmise that the largest limiting factor in our model's performance is dataset size. Over the course of our research experiments we evaluated on a small subset of our data, roughly 1,300 examples, to serve as our project milestone before utilizing the full dataset, roughly 13,000 examples, for our final experimentation. From this ten fold increase in data we observed LLM judge precision also scale from an average of 7.5% on our finetuned models to an average of 44.3% precision on our finetuned models. That is around a 6 fold improvement for a ten fold increase in data. Additionally, looking at our loss over training iterations graph we can clearly see that loss was still decreasing with additional training. Today's cutting edge models use trillions of training tokens and a trillion or so parameters. Our model was trained on a little over a million tokens and utilizes a few billion parameters, and the fact our best experiment achieved only 3% less precision than the baseline with our extreme data, model size, and training times limitations is extremely promising. Thus we reasonably expect that as we increase our dataset side our model's performance will easily eclipse that of the baseline. Additionally, comparing the outputs of our various evaluation metrics, reveals that our novel metrics are consistent. There is a positive correlation between our LLM judge precision and SacreBLEU providing evidence that they both are preforming a good job at evaluating the quality of retrieved documents. This is positive news because each evaluation metric has its own benefits and drawbacks depending on the nature of the query and documents it is evaluating. The knowledge that either will give you comparable results will be critical in future research.

8 Ethics Statement

The LLM judge we deployed was trained on a large amount of data scraped from the web. However, Mistral AI has yet to disclose how said data was obtained. It is incredibly likely that the data was taken without permission of the authors. Even though we did not directly take the work of individuals without permission, the use of our API credits to pay for use of the model encourages the developers of the model to continue to take data in the future. Secondly because we do not know how Mistral acquired its data we cannot guarantee that their dataset was free from bias, data collected in mass from the open internet has the propensity to contain deplorable content. Thirdly Mistral is un-moderated, which makes it easier for us as there is less post processing involved, but also means there is no filter between unsavory responses from Mistral and our model. This is heavily mitigated by us only asking yes or no questions and rephrase requests, but it is still possible for bias to seep into our model from Mistral. Additionally continuing on the topic of moderation, our model has no content moderation. If a user enters a very toxic query(e.g. "Find more question about the benefits of eugenics on society") our model will do its best to find documents that fulfils the users sentiment. Ideally our model should reject the query on the basis of its toxic nature. The lack of moderation creates substantial risk for what is known as an "echo chamber" in literature. If you enter a biased query it is likely that your top-k documents retrieved will contain the same bias, and therefore reinforcing the users belief in said bias. For the sake of society it is better that our model prioritizes retrieving an unbiased set of questions pertaining to the topic rather thantrying to fulfil the query as best as possible.

References

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

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S. Weld. 2020. SPECTER: Document-level Representation Learning using Citation-informed Transformers. In *ACL*.

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

Meta. 2024. Introducing meta llama 3: The most capable openly available llm to date. *Meta AI*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Association for Computational Linguistics (ACL)*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Gemini Team, Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry, Lepikhin, Timothy Lillicrap, Jean baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, Ioannis Antonoglou, Rohan Anil, Sebastian Borgeaud, Andrew Dai, Katie Millican, Ethan Dyer, Mia Glaese, Thibault Sottiaux, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, James Molloy, Jilin Chen, Michael Isard, Paul Barham, Tom Hennigan, Ross McIlroy, Melvin Johnson, Johan Schalkwyk, Eli Collins, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, Clemens Meyer, Gregory Thornton, Zhen Yang, Henryk Michalewski, Zaheer Abbas, Nathan Schucher, Ankesh Anand, Richard Ives, James Keeling, Karel Lenc, Salem Haykal, Siamak Shakeri, Pranav Shyam, Aakanksha Chowdhery, Roman Ring, Stephen Spencer, Eren Sezener, Luke Vilnis, Oscar Chang, Nobuyuki Morioka, George Tucker, Ce Zheng, Oliver Woodman, Nithya Attaluri, Tomas Kocisky, Evgenii Eltyshev, Xi Chen, Timothy Chung, Vittorio Selo, Siddhartha Brahma, Petko Georgiev, Ambrose Slone, Zhenkai Zhu, James Lottes, Siyuan Qiao, Ben Caine, Sebastian Riedel, Alex Tomala, Martin Chadwick, Juliette Love, Peter Choy, Sid Mittal, Neil

Houlsby, Yunhao Tang, Matthew Lamm, Libin Bai, Qiao Zhang, Luheng He, Yong Cheng, Peter Humphreys, Yujia Li, Sergey Brin, Albin Cassirer, Yingjie Miao, Lukas Zilka, Taylor Tobin, Kelvin Xu, Lev Proleev, Daniel Sohn, Alberto Magni, Lisa Anne Hendricks, Isabel Gao, Santiago Ontanon, Oskar Bunyan, Nathan Byrd, Abhanshu Sharma, Biao Zhang, Mario Pinto, Rishika Sinha, Harsh Mehta, Dawei Jia, Sergi Caelles, Albert Webson, Alex Morris, Becca Roelofs, Yifan Ding, Robin Strudel, Xuehan Xiong, Marvin Ritter, Mostafa Dehghani, Rahma Chaabouni, Abhijit Karmarkar, Guangda Lai, Fabian Mentzer, Bibo Xu, YaGuang Li, Yujing Zhang, Tom Le Paine, Alex Goldin, Behnam Neyshabur, Kate Baumli, Anselm Levskaya, Michael Laskin, Wenhao Jia, Jack W. Rae, Kefan Xiao, Antoine He, Skye Giordano, Lakshman Yagati, Jean-Baptiste Lespiau, Paul Natsev, Sanjay Ganapathy, Fangyu Liu, Danilo Martins, Nanxin Chen, Yunhan Xu, Megan Barnes, Rhys May, Arpi Vezer, Junhyuk Oh, Ken Franko, Sophie Bridgers, Ruizhe Zhao, Boxi Wu, Basil Mustafa, Sean Sechrist, Emilio Parisotto, Thanumalayan Sankaranarayana Pillai, Chris Larkin, Chenjie Gu, Christina Sorokin, Maxim Krikun, Alexey Guseynov, Jessica Landon, Romina Datta, Alexander Pritzel, Phoebe Thacker, Fan Yang, Kevin Hui, Anja Hauth, Chih-Kuan Yeh, David Barker, Justin Mao-Jones, Sophia Austin, Hannah Sheahan, Parker Schuh, James Svensson, Rohan Jain, Vinay Ramasesh, Anton Briukhov, Da-Woon Chung, Tamara von Glehn, Christina Butterfield, Priya Jhakra, Matthew Wiethoff, Justin Frye, Jordan Grimstad, Beer Changpinyo, Charline Le Lan, Anna Bortsova, Yonghui Wu, Paul Voigtlaender, Tara Sainath, Shane Gu, Charlotte Smith, Will Hawkins, Kris Cao, James Besley, Srivatsan Srinivasan, Mark Omernick, Colin Gaffney, Gabriela Surita, Ryan Burnell, Bogdan Damoc, Junwhan Ahn, Andrew Brock, Mantas Pajarskas, Anastasia Petrushkina, Seb Noury, Lorenzo Blanco, Kevin Swersky, Arun Ahuja, Thi Avrahami, Vedant Misra, Raoul de Liedekerke, Mariko Iinuma, Alex Polozov, Sarah York, George van den Driessche, Paul Michel, Justin Chiu, Rory Blevins, Zach Gleicher, Adrià Recasens, Alban Rrustemi, Elena Gribovskaya, Aurko Roy, Wiktor Gworek, Sébastien M. R. Arnold, Lisa Lee, James Lee-Thorp, Marcello Maggioni, Enrique Piqueras, Kartikeya Badola, Sharad Vikram, Lucas Gonzalez, Anirudh Baddepudi, Evan Senter, Jacob Devlin, James Qin, Michael Azzam, Maja Trebacz, Martin Polacek, Kashyap Krishnakumar, Shuo yiin Chang, Matthew Tung, Ivo Penchev, Rishabh Joshi, Kate Olszewska, Carrie Muir, Mateo Wirth, Ale Jakse Hartman, Josh Newlan, Sheleem Kashem, Vijay Bolina, Elahe Dabir, Joost van Amersfoort, Zafarali Ahmed, James Cobon-Kerr, Aishwarya Kamath, Arnar Mar Hrafnkelsson, Le Hou, Ian Mackinnon, Alexandre Frechette, Eric Noland, Xiance Si, Emanuel Taropa, Dong Li, Phil Crone, Anmol Gulati, Sébastien Cevey, Jonas Adler, Ada Ma, David Silver, Simon Tokumine, Richard Powell, Stephan Lee, Kiran Vodrahalli, Samer Hassan, Diana Mincu, Antoine Yang, Nir Levine, Jenny Brennan, Mingqiu Wang, Sarah Hodkinson, Jeffrey Zhao, Josh Lipschultz, Aedan Pope, Michael B. Chang, Cheng Li, Laurent El Shafey, Michela Paganini, Sholto Douglas, Bernd Bohnet, Fabio Pardo, Seth Odoom, Mihaela Rosca, Cicero Nogueira dos Santos, Kedar Soparkar, Arthur Guez, Tom Hudson, Steven Hansen, Chulayuth Asawaroengchai, Ravi Addanki, Tianhe Yu, Wojciech Stokowiec, Mina Khan, Justin Gilmer, Jaehoon Lee, Carrie Grimes Bostock, Keran Rong, Jonathan Caton, Pedram Pejman, Filip Pavetic, Geoff Brown, Vivek Sharma, Mario Lučić, Rajkumar Samuel, Josip Djolonga, Amol Mandhane, Lars Lowe Sjösund, Elena Buchatskaya, Elspeth White, Natalie Clay, Jiepu Jiang, Hyeontaek Lim, Ross Hemsley, Zeyncep Cankara, Jane Labanowski, Nicola De Cao, David Steiner, Sayed Hadi Hashemi, Jacob Austin, Anita Gergely, Tim Blyth, Joe Stanton, Kaushik Shivakumar, Aditya Siddhant, Anders Andreassen, Carlos Araya, Nikhil Sethi, Rakesh Shivanna, Steven Hand, Ankur Bapna, Ali Khodaei, Antoine Miech, Garrett Tanzer, Andy Swing, Shantanu Thakoor, Lora Aroyo, Zhufeng Pan, Zachary Nado, Jakub Sygnowski, Stephanie Winkler, Dian Yu, Mohammad Saleh, Loren Maggiore, Yamini Bansal, Xavier Garcia, Mehran Kazemi, Piyush Patil, Ishita Dasgupta, Iain Barr, Minh Giang, Thais Kagohara, Ivo Danihelka, Amit Marathe, Vladimir Feinberg, Mohamed Elhawaty, Nimesh Ghelani, Dan Horgan, Helen Miller, Lexi Walker, Richard Tanburn, Mukarram Tariq, Disha Shrivastava, Fei Xia, Qingze Wang, Chung-Cheng Chiu, Zoe Ashwood, Khuslen Baatarsukh, Sina Samangooei, Raphaël Lopez Kaufman, Fred Alcober, Axel Stjerngren, Paul Komarek, Katerina Tsihlas, Anudhyan Boral, Ramona Comanescu, Jeremy Chen, Ruibo Liu, Chris Welty, Dawn Bloxwich, Charlie Chen, Yanhua Sun, Fangxiaoyu Feng, Matthew Mauger, Xerxes Dotiwalla, Vincent Hellendoorn, Michael Sharman, Ivy Zheng, Krishna Haridasan, Gabe Barth-Maron, Craig Swanson, Dominika Rogozińska, Alek Andreev, Paul Kishan Rubenstein, Ruoxin Sang, Dan Hurt, Gamaleldin Elsayed, Renshen Wang, Dave Lacey, Anastasija Ilić, Yao Zhao, Adam Iwanicki, Alejandro Lince, Alexander Chen, Christina Lyu, Carl Lebsack, Jordan Griffith, Meenu Gaba, Paramjit Sandhu, Phil Chen, Anna Koop, Ravi Rajwar, Soheil Hassas Yeganeh, Solomon Chang, Rui Zhu, Soroush Radpour, Elnaz Davoodi, Ving Ian Lei, Yang Xu, Daniel Toyama, Constant Segal, Martin Wicke, Hanzhao Lin, Anna Bulanova,

Adrià Puigdomènech Badia, Nemanja Rakićević, Pablo Sprechmann, Angelos Filos, Shaobo Hou, Víctor Campos, Nora Kassner, Devendra Sachan, Meire Fortunato, Chimezie Iwuanyanwu, Vitaly Nikolaev, Balaji Lakshminarayanan, Sadegh Jazayeri, Mani Varadarajan, Chetan Tekur, Doug Fritz, Misha Khalman, David Reitter, Kingshuk Dasgupta, Shourya Sarcar, Tina Ornduff, Javier Snaider, Fantine Huot, Johnson Jia, Rupert Kemp, Nejc Trdin, Anitha Vijayakumar, Lucy Kim, Christof Angermueller, Li Lao, Tianqi Liu, Haibin Zhang, David Engel, Somer Greene, Anaïs White, Jessica Austin, Lilly Taylor, Shereen Ashraf, Dangyi Liu, Maria Georgaki, Irene Cai, Yana Kulizhskaya, Sonam Goenka, Brennan Saeta, Ying Xu, Christian Frank, Dario de Cesare, Brona Robenek, Harry Richardson, Mahmoud Alnahlawi, Christopher Yew, Priya Ponnapalli, Marco Tagliasacchi, Alex Korchemniy, Yelin Kim, Dinghua Li, Bill Rosgen, Kyle Levin, Jeremy Wiesner, Praseem Banzal, Praveen Srinivasan, Hongkun Yu, Çağlar Ünlü, David Reid, Zora Tung, Daniel Finchelstein, Ravin Kumar, Andre Elisseeff, Jin Huang, Ming Zhang, Ricardo Aguilar, Mai Giménez, Jiawei Xia, Olivier Dousse, Willi Gierke, Damion Yates, Komal Jalan, Lu Li, Eri Latorre-Chimoto, Duc Dung Nguyen, Ken Durden, Praveen Kallakuri, Yaxin Liu, Matthew Johnson, Tomy Tsai, Alice Talbert, Jasmine Liu, Alexander Neitz, Chen Elkind, Marco Selvi, Mimi Jasarevic, Livio Baldini Soares, Albert Cui, Pidong Wang, Alek Wenjiao Wang, Xinyu Ye, Krystal Kallarackal, Lucia Loher, Hoi Lam, Josef Broder, Dan Holtmann-Rice, Nina Martin, Bramandia Ramadhana, Mrinal Shukla, Sujoy Basu, Abhi Mohan, Nick Fernando, Noah Fiedel, Kim Paterson, Hui Li, Ankush Garg, Jane Park, DongHyun Choi, Diane Wu, Sankalp Singh, Zhishuai Zhang, Amir Globerson, Lily Yu, John Carpenter, Félix de Chaumont Quitry, Carey Radebaugh, Chu-Cheng Lin, Alex Tudor, Prakash Shroff, Drew Garmon, Dayou Du, Neera Vats, Han Lu, Shariq Iqbal, Alex Yakubovich, Nilesh Tripuraneni, James Manyika, Haroon Qureshi, Nan Hua, Christel Ngani, Maria Abi Raad, Hannah Forbes, Jeff Stanway, Mukund Sundararajan, Victor Ungureanu, Colton Bishop, Yunjie Li, Balaji Venkatraman, Bo Li, Chloe Thornton, Salvatore Scellato, Nishesh Gupta, Yicheng Wang, Ian Tenney, Xihui Wu, Ashish Shenoy, Gabriel Carvajal, Diana Gage Wright, Ben Bariach, Zhuyun Xiao, Peter Hawkins, Sid Dalmia, Clement Farabet, Pedro Valenzuela, Quan Yuan, Ananth Agarwal, Mia Chen, Wooyeol Kim, Brice Hulse, Nandita Dukkipati, Adam Paszke, Andrew Bolt, Kiam Choo, Jennifer Beattie, Jennifer Prendki, Harsha Vashisht, Rebeca Santamaria-Fernandez, Luis C. Cobo, Jarek Wilkiewicz, David Madras, Ali Elqursh, Grant Uy, Kevin Ramirez, Matt Harvey, Tyler Liechty, Heiga Zen, Jeff Seibert, Clara Huiyi Hu, Andrey Khorlin, Maigo Le, Asaf Aharoni, Megan Li, Lily Wang, Sandeep Kumar, Norman Casagrande, Jay Hoover, Dalia El Badawy, David Soergel, Denis Vnukov, Matt Miecnikowski, Jiri Simsa, Praveen Kumar, Thibault Sellam, Daniel Vlasic, Samira Daruki, Nir Shabat, John Zhang, Guolong Su, Jiageng Zhang, Jeremiah Liu, Yi Sun, Evan Palmer, Alireza Ghaffarkhah, Xi Xiong, Victor Cotruta, Michael Fink, Lucas Dixon, Ashwin Sreevatsa, Adrian Goedeckemeyer, Alek Dimitriev, Mohsen Jafari, Remi Crocker, Nicholas FitzGerald, Aviral Kumar, Sanjay Ghemawat, Ivan Philips, Frederick Liu, Yannie Liang, Rachel Sterneck, Alena Repina, Marcus Wu, Laura Knight, Marin Georgiev, Hyo Lee, Harry Askham, Abhishek Chakladar, Annie Louis, Carl Crous, Hardie Cate, Dessie Petrova, Michael Quinn, Denese Owusu-Afriyie, Achintya Singhal, Nan Wei, Solomon Kim, Damien Vincent, Milad Nasr, Christopher A. Choquette-Choo, Reiko Tojo, Shawn Lu, Diego de Las Casas, Yuchung Cheng, Tolga Bolukbasi, Katherine Lee, Saaber Fatehi, Rajagopal Ananthanarayanan, Miteyan Patel, Charbel Kaed, Jing Li, Shreyas Rammohan Belle, Zhe Chen, Jaclyn Konzelmann, Siim Põder, Roopal Garg, Vinod Koverkathu, Adam Brown, Chris Dyer, Rosanne Liu, Azade Nova, Jun Xu, Alanna Walton, Alicia Parrish, Mark Epstein, Sara McCarthy, Slav Petrov, Demis Hassabis, Koray Kavukcuoglu, Jeffrey Dean, and Oriol Vinyals. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. *ArXiv*, abs/1707.06209.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zhicheng Dou, and Ji-Rong Wen. 2024. Large language models for information retrieval: A survey.