NLP Project Report: RAGs for Open Domain Complex QA Group 10

Pratham Johari and **Bertold Kovács** and **Henry Page** and **Danae Savvidi**
Technische Universiteit Delft
{p.johari, B.B.Kovacs-1, h.page, d.n.savvidi}@student.tudelft.nl

Abstract

Retrieval Augmented Generation (RAG) is a promising technique to combat the limitations of Large Language Models (LLMs) for complex question answering tasks. We conduct experiments to study whether providing non-relevant documents, alongside the relevant documents usually provided by RAG methods, can improve the LLM's performance. Our findings indicate that while relevant documents are the most important to perform well on question answering, negative contexts can improve an LLM's performance, as long as they are not distracting 'hard negatives'. We draw on the theory of attention entropy collapse to try to shed light on the reasons behind this phenomenon.

1 Introduction

In recent years, the state of the art for question-answering tasks (QA) has been defined by Large Language Models (LLMs), both for simple QA (Brown et al., 2020) and complex QA (Wei et al., 2022). However, pre-trained LLMs are not without their limitations: in a closed-book setting (that is, without access to an outside knowledge source, and only using the knowledge encoded in the model's parameters) LLMs are unable to use recent knowledge, and are prone to giving inaccurate answers (hallucinating) (Ji et al., 2023). A proposed technique to overcome these weaknesses is Retrieval Augmented Generation (RAG): contexts are retrieved from an outside knowledge source, and the model generates the answer to the query by additionally taking these contexts into account (Lewis et al., 2020).

The intuitive approach is to use the most relevant retrieved contexts for RAG (Gao et al., 2023). However, recent research has cast doubts on whether this approach is the most effective (Cuconasu et al., 2024). In our paper, we study how using non-relevant contexts for RAG impacts an LLMs performance in complex QA tasks, and provide an intuition on why this is the case.

The specific research questions we aim to answer are the following:

- **RQ1:** How does negative contexts impact downstream answer generation performance?
- **RQ2:** Are negative context more important for answer generation than related contexts?
- **RQ3:** Does providing only gold contexts deteriorate the performance compared to mixing with other negative or related contexts?

Our report is structured in the following way: first we summarize related literature in Chapter 2. Then in Chapter 3 we describe the methodology we used including metrics we measured, and Chapter 4 details the exact experiments we conducted and their results. Chapter 5 provides an analysis into the reasons behind our findings, with 6 highlighting limitations of our research. Finally, we summarize our conclusions in Chapter 7. All code related to the project is available at a public github repository.[1]

2 Related Works

2.1 Retrieval Augmented Generation

Traditional models depend on parametric memory, where knowledge is stored within their parameters. This approach often restricts their ability to dynamically update knowledge or offer transparent reasoning behind their outputs. Hybrid architectures, which integrate parametric and non-parametric memory, have emerged as a promising solution to overcome these challenges (Lewis et al., 2020). An example of such an architecture is RAG. By conditioning on retrieved text passages, these models achieve superior performance across a wide range of tasks, including open-domain question answering (QA).

[1] https://github.com/pratham2442000/DSAIT4090_RAG_group_project

A notable strength of RAG lies in its adaptability. By treating the retrieval index as non-parametric memory, the model can be updated with new knowledge simply by modifying the index without requiring retraining. This feature addresses the challenge of keeping NLP systems current with evolving real-world information. Furthermore, RAG's interpretability, stemming from its reliance on retrievable evidence, distinguishes it from purely parametric models, which often lack transparency (Lewis et al., 2020).

2.2 Random vs Hard Negatives

Given a query q and a corpus $\mathbf{C}$, a retriever aims to identify a set of relevant documents D^+. The retrieval process uses a scoring function f that measures the similarity between a query and the documents to identify the most relevant documents.

To evaluate the quality of the retrieved documents, one can predefine a set of relevant documents $\mathbf{R}$ for the given query q. Documents not belonging to $\mathbf{R}$ are considered **negative** documents, denoted as $\mathbf{C} \setminus \mathbf{R}$.

Random negative sampling involves selecting negative documents uniformly at random from the entire corpus (Qu et al., 2021). These documents are assumed to be irrelevant to the query q. In contrast, **hard negatives** are negative documents that are particularly challenging for the model to distinguish from relevant ones. These are instances in the embedding space that closely resemble positive examples. In dense retrieval, incorporating hard negatives during training forces the model to learn more finer distinctions, as these samples have a greater impact on the gradient updates (Yang et al., 2024). Henceforth, when we refer to "hard negatives", we specifically mean the top K most similar to the query q negative documents (as scored by f) (Zhan et al., 2021).

2.3 ADORE

Zhan et al. (Zhan et al., 2021) focus on enhancing the training of dense retrieval (DR) models by addressing issues related to negative sampling strategies. They point out hard hard negative sampling is more effective than random negative sampling. However, the widely adopted static hard negative sampling method, where negatives are pre-retrieved and remain fixed during training, is prone to instability.

To overcome these limitations, they propose a training method called ADORE (Algorithm for Directly Optimizing Ranking performance). ADORE is a query-side training approach for DR models that uses dynamic hard negatives and LambdaLoss to optimise IR metrics (NDCG, MAP, RR). It focuses on training the query encoder while leveraging a pre-trained document encoder (along with static document embeddings).

During each training iteration t, hard negatives are dynamically retrieved using encoder with parameters ϕ_t. This involves encoding a batch of queries and selecting the top-ranked documents as negatives. The training process considers all pairs of relevant documents and hard negatives, incorporating the product of rank loss and LambdaLoss to directly optimize the ranking metric. Here, ΔM represents the change in the IR metric being optimised (as specified by LambdaLoss). The pairwise loss function is defined as:

$$L(d^+, d^-) = \Delta M \cdot \log(1 + e^{f(q,d^-) - f(q,d^+)}) \quad (1)$$

This loss is averaged over the number of pairs and the batch size of the queries.

Experimental results on benchmark datasets demonstrate that both ADORE achieve significant performance gains over competitive baselines like ANCE and knowledge distillation methods (Zhan et al., 2021).

3 Methodology

3.1 Dataset and Codebase

To conduct our experiments, we utilized Dexter (Prabhu and Anand, 2024), a toolkit to evaluate zero-shot retrieval capabilities of dense and sparse retrieval models. We also used the 2WikiMultiHopQA[2] dataset, which contains compositional questions requiring multi-hop reasoning over Wikipedia articles. The dataset provides annotated oracle contexts for each question. It contains a total of 190k questions and has a corpus size of 570k documents. For computational reasons, we select the first 1200 questions from dev.json as the test set.

To extract relevant contexts for each query, we used Contriever, a pre-trained dense retriever. Additionally, we trained a retrieval model using ADORE for comparison in later experiments. The retrieved contexts were provided as input to a generative LLM, the Llama model on HuggingFace (Meta,

[2]https://github.com/Alab-NII/2wikimultihop

2024), which is the second best ranking LLM[3] after the paid GPT versions which we could not fund for our project.

3.2 Metrics

The metrics we use to evaluate our model are **Exact Match (EM)** and **Cover Exact Match (CEM)**. EM is a strict metric that considers an answer correct only if it matches the ground truth exactly. This enforces precision and ensures generated answers are concise, accurate, and unambiguous. For instance, in response to the question *What is the capital of the Netherlands?*, the answer *Amsterdam* would be correct under EM, while a longer but valid response like *The capital of the Netherlands is Amsterdam.* would not. EM serves as a benchmark for the model's ability to produce precise outputs without unnecessary verbosity.

To complement EM, we use CEM, which evaluates whether the ground truth answer is a substring of the generated response. This metric accommodates linguistic variation, allowing answers like *The capital of the Netherlands is Amsterdam.* to be classified as correct. CEM reflects the practical utility of answers in real-world scenarios, where phrasing often varies. While it does not penalize verbosity or ambiguity, the inclusion of EM ensures that precision is still evaluated on a certain level.

A possible more nuanced evaluation technique would be one using contextual embeddings, such as BERTScore (Zhang et al., 2019). While this could lead to a more accurate evaluation, this technique is computationally intensive, and due to a lack of resources is not included in our paper.

In our experiments, we analyse the performance of RAG systems by varying the proportions of relevant and non-relevant contexts (random or hard negatives) provided to the language model. Below, we describe our methodology for data preparation, sampling strategies, and implementation in detail.

3.3 Retrieval Process

For each query, we retrieve the top 100 relevant documents using Contriever. From these 100 documents, we select the top k relevant documents depending on the experiment. This approach is more computationally efficient as it avoids repeated retrievals for different experiments, as the process of retrieving documents is computationally expensive.

[3]https://artificialanalysis.ai/leaderboards/models

3.4 LLM Answer Processing

The LLM's generated responses often included the entire input context in addition to the answer. This necessitated a post-processing step to extract only the answer portion which we will refer to as cleaned response. Without this clean-up, our evaluation metric (CEM) would have registered a match whenever the answer was present in the context, regardless of whether the LLM's generated answer was correct or even present within the response itself. But we are not evaluating the performance of the LLM, but the retrieving methods, if the answer is in the context, we should assume that a better LLM can generate the correct response. We provide the results of this cleaned version in the Appendix A.

4 Experiment Setup

As shown on Table 1 and 3, we conduct extensive experiments. We analyse the performance with different mixes of relevant and random as well as relevant and hard negative documents. We evaluate this for both 12 total documents as well as 24 total documents (so any improvement is really due to the type of document, and not due to an increased number of documents provided). We also compare simple relevant contexts with ADORE relevant contexts. The results shown are the percentage of answers where we reach a CEM, over 1200 queries.

4.1 Experiments 1 & 2: Relevant and Oracle Contexts

In Experiment 1, we evaluate the impact of providing the LLM with relevant contexts to answer questions. This done by selecting the top-k relevant documents retrieved by Contriever for each query, where $k \in \{1, 3, 5\}$. This setup establishes a baseline for the following experiments where we explore the influence of different context types on the LLM's performance.

By design, the top-k retrieved documents by Contriever are expected to be the most relevant to the query. Therefore, Experiment 1 helps us understand how well the LLM performs when it has access only to documents related to the question.

In Experiment 2, we evaluate the impact of using the oracle contexts in the LLM. Oracle contexts refer to the ground truth documents that contain the answer to the query. These are constructed by concatenating the relevant Wikipedia articles for the entities mentioned in the query (Ho et al., 2020).

Here again, we use top-k where $k \in \{1, 3, 5\}$.

By providing the oracle contexts, we aim to establish an upper bound on the performance achievable by the LLM when it has access to perfect information. This allows us to compare the performance of the LLM in Experiment 1 (using retrieved contexts) with the ideal scenario where it has access to the exact documents containing the answer.

4.2 Experiments 3 & 4: Random and Hard Negatives

In these experiments we introduce non-relevant contexts and incorporate negative documents (indicated by the on the database) into the input in two forms:

1. **Random Negatives:** these are randomly selected documents from the collection that have no relevance to the query.

2. **Hard Negatives:** these are documents close to the query in vector space but not part of the ground-truth set. We sample hard negatives from the remaining $100 - k$, documents in the top-100 retrieved set to simulate realistic retrieval errors. In this case, k is the number of relevant documents we have retrieved mixed with h hard negatives as input to the LLM.

We gradually increase the proportion of non-relevant (negative) documents in increments of 25% to study its effect. This results in configurations such as 100% relevant (e.g., 12 relevant, 0 negatives), 75% relevant (e.g., 9 relevant, 3 negatives), and so on. We repeat these experiments for two total document counts: 12 and 24, to examine the impact of both the proportion of negatives and the overall document count on the RAG's answer generation performance. Notably, 24 added contexts already represent a large input for the LLM, which cannot effectively handle more due to computational constraints. The document counts were selected to ensure they can be easily divided into increments of 25%. Lastly, due to the computational demands of these experiments, smaller increments, such as 10%, were not possible given the time constraints of this project.

4.3 Experiment 5: Adore

The ADORE training process involved fine-tuning the pre-trained query encoder using the dev set provided by Dexter. We optimised for Mean Average Precision at rank 10 (**MAP@10**). This metric was chosen because the dataset only included binary relevance labels. While metrics like NDCG (Normalized Discounted Cumulative Gain) can accommodate binary relevance, they are less ideal in this context because its design is primarily geared toward graded relevance, where documents can have varying degrees of relevance. In the case of binary relevance, NDCG's nuanced weighting of rank positions becomes less impactful, reducing its ability to provide meaningful differentiation between ranking systems. Additionally, NDCG may overemphasize the importance of higher-ranked results, which is less suitable for scenarios like ours where the primary goal is to retrieve multiple relevant documents rather than focus solely on their rank order. Additionally, Mean Reciprocal Rank (MRR) was not a perfect fit for this problem, as it prioritises the highest-ranking relevant document, whereas in complex question-answering tasks, we focus on retrieving multiple relevant documents rather than just the top-ranked one.

For ADORE, the query encoder was trained using the AdamW optimiser alongside a linearly decreasing learning rate, starting at 5×10^{-6}, as a form of regularisation. Additionally, a batch size of 32 was used for training. These hyperparameter choices align with the approach described in the original paper (Zhan et al., 2021). The model was trained for a total of 50 epochs.

5 Evaluation and Analysis

Looking at the results of our experiments shown in Table 1 and Table 3, some interesting tendencies seem to present themselves. We first point these out, then answer our research questions.

5.1 Experiment 1 & 2

Providing oracle contexts tends to outperform relevant contexts. This is according to our expectations: oracle contexts are the 'ideal' inputs, and essentially serve as an upper bound on how well the LLM can perform with retrieved information. However, while oracle contexts are typically hand-picked, human errors or biases could result in suboptimal choices, and the retriever might occasionally find more effective contexts. Although this is rare, based on the metrics we observe, the top 3 contexts retrieved with the Contriever gave a better exact match than the top 3 oracle contexts which could be related to this phenomenon.

Cover Exact Match metrics are higher when

Experiment	Cover Exact Match %			Exact Match %		
	Top 1	Top 3	Top 5	Top 1	Top 3	Top 5
Relevant	40.67%	44.08%	47.75%	1.75%	**1.75%**	0.58%
Oracle	**43.50%**	**52.42%**	**63.42%**	**2.33%**	0.92%	**0.75%**
Adore	39.92%	48.08%	52.58%	2.00%	0.83%	0.25%

Table 1: Comparison of CEM and EM performance across Experiment 1 (Relevant), Experiment 2 (Oracle), and Experiment 5 (Adore). The highest percentage in each column is highlighted in bold.

Metric	Performance	
	Contriever	ADORE
NDCG@1	0.42417	**0.74833**
NDCG@2	0.38580	**0.67515**
NDCG@3	0.34182	**0.62467**
MAP@1	0.04266	**0.07521**
MAP@2	0.06460	**0.12010**
MAP@3	0.07620	**0.15494**
RECALL@1	0.04266	**0.07521**
RECALL@2	0.07536	**0.13142**
RECALL@3	0.09527	**0.17760**

Table 2: Comparison of performance metrics (NDCG, MAP, and Recall) between Contriever (base model) and Contriever trained with ADORE. The higher values are bolded to highlight the improvements with ADORE.

more documents are provided. The intuitive answer is that having more contexts which contain information about the answer helps us answer better - this result is hardly surprising.

Exact Match metrics are higher when less documents are provided. This result is more surprising. We propose the explanation that having longer and more varied inputs possibly prompts the LLM to also provide more verbose, less focused answers - which are punished by the strict Exact Match metric.

5.2 Experiment 3 & 4

A larger portion of relevant documents tends to produce better results. For both 12 and 24 document scenarios, and both random and hard negatives, we see the tendency that increasing the percentage of relevant documents also improves the result. The intuitive explanation is that relevant documents provide more information on the correct answer - and the 'directing away' that negatives can help with is not informative enough to warrant losing direct information over.

Having only hard negatives performs better than having only random negatives, for both the 12 and 24 document scenario. This possibly happens because in the case of no relevant document provided, hard negatives lose their weakness of being 'distracting' (since there is no correct answer to distract from), but they can possibly help in directing the answers away from some wrong choices it possibly would have made.

When mixed with relevant contexts, random and hard negatives perform similarly well, with with random negative contexts having a slight upper hand. This is somewhat in line with the findings of Cuconasu et al. (2024), who also found that while adding random contexts might even improve performance, even the addition of a few hard negative contexts drastically reduces performance. Our findings show a less drastic drop in performance, but still show that mixes with hard negatives perform worse than mixes with random contexts.

Exact Match is an uninformative metric for our task. Our results with exact match end up being uninformative and uninteresting, as this metric is way too strict for the current task, with essentially only luck being the differentiating principle between the seemingly correct answers, and the correct answers it actually accepts. While we still present our evaluation with this metric, our analysis is only based on the Cover Exact Match metric.

5.3 Experiment 5

ADORE performs better than normal relevant documents. Table 2 provides a comparative analysis of the previously discussed IR metrics between two models: the base retriever (Contriever) and the enhanced retriever trained with ADORE. Across all evaluated positions (1, 2, and 3), the ADORE-trained retriever demonstrates significant improvements over the base model. These results illustrate that ADORE effectively enhances the retriever's

Total #Docs	% of Relevant	Cover Exact Match %		Exact Match %	
		Random	Hard	Random	Hard Negatives
12	0%	44.92	46.25	0.00	0.00
	25%	50.33	51.67	0.00	**0.17**
	50%	54.83	54.50	0.00	0.00
	75%	56.50	56.08	0.00	0.08
	100%	**57.17**		**0.17**	
24	0%	46.92	48.25	0.00	0.00
	25%	55.92	53.00	0.00	**0.08**
	50%	**59.50**	56.17	0.00	0.00
	75%	**59.50**	57.42	0.00	0.00
	100%	58.58		0.0	

Table 3: Comparison of experiments with different ratios of relevant and non-relevant documents. In Experiment 2, relevant contexts are mixed with random contexts, and in Experiment 3, relevant contexts are mixed with hard negatives. The bold values represent the best performance for each configuration. The total document count shows the number of documents (relevant + non-relevant), and the relevant/non-relevant ratio is indicated for each mix.

ability to prioritise and retrieve relevant documents, especially at higher-ranking positions. In table 1, we can see that ADORE is never able to achieve the best performance as oracle contexts have better performance. However, in certain cases, the method achieves up to a 4.83% improvement in performance compared to the baseline. This result was anticipated, as dynamic hard negatives are retrieved and used to update the query encoder during training. The loss function (1) adjusts the weights of the query encoder to ensure that the embeddings of hard negatives are 'pushed away' from the embeddings of relevant corpus elements. In essence, ADORE enhances the embedding capacity, enabling it to better capture the semantic information of the text (Liu et al., 2024).

One potential reason for the relatively modest performance gains observed is that ADORE specifically optimizes for hard negatives. Since the query encoder is employed for negative selection at each iteration, it produces adaptive negatives for the same queries (Zhao et al., 2024). Some studies suggest that combining hard negative samples with easy negative samples may yield more effective semantic retrieval models (Guo et al., 2022).

5.4 Answering our Research Questions

- **RQ1: Adding random negative contexts can help performance, but having a large portion of any negative context is worse than only relevant contexts.** For explaining this phenomenon, we turn to the explanation theorized in Cuconasu et al. (2024) inspired by Zhai et al. (2023): 'attention entropy collapse'. Attention entropy essentially measures how 'spread out' the attention of a transformer is. Previous studies found that when this value is really low, and we achieve 'attention entropy collapse', the LLM can generate nonsensical outputs based on only a few, or possible a single token. Providing various non-relevant contexts can act against this by increasing attention entropy.

- **RQ2: Relevant contexts are more important for answer generation, than negative ones.** Comparing situations where we only have relevant contexts against one where we have only random documents or hard negative documents, we find that we perform much better in the only relevant scenario. Furthermore, it seems to be a general tendency to perform better with a higher portion of relevant documents in the mix for mixes as well. This is unsurprising: while negative contexts can act against 'attention entropy collapse', we still need to be able to find the correct answer, in which negative contexts do not seem to help.

- **RQ3: Providing only gold contexts does perform worse than a mix of gold contexts and random contexts,** but better than any mix of gold and hard negative contexts. Our

explanation is the same again: having gold contexts to help us find the correct answer, and negative contexts to act against 'attention entropy collapse' is optimal. Hard negatives are not helpful in this task, since not only do they not provide the answer, their contribution towards the attention entropy is minimal.

6 Limitations

During our evaluation, we only evaluate a specific dataset, and a specific LLM: while we expect to get similar results from other benchmarking settings and models, it would make our findings more robust if we did our experiments over different dataset - model combinations. Furthermore, we only run one experiment for each setting due to a lack of computational resources, making our results more subject to random particularities.

As mentioned in subsection 3.2 the metric we use (CEM) does have limitations as well, as it does not adequately punish unnecessarily verbose or ambiguous answers. A possible metric to overcome these weaknesses would be BERTScore, a contextual embedding based evaluation metric technique, which we did not use due to computational constraints.

7 Conclusions

Our research investigates whether providing non-relevant documents, alongside the relevant documents usually provided by RAG methods, can improve the LLM's performance on complex question-answering tasks.

Our findings indicate that negative contexts can improve an LLM's performance on complex QA tasks, as long as they are not distracting "hard negatives". This suggests that a mix of relevant and negative contexts can provide a more informative training signal for the LLM, allowing it to better distinguish between relevant and irrelevant information.

- **Performance with gold contexts (perfect information):** When provided with the exact documents containing the answer (oracle contexts), the LLM achieved the highest performance (63.42% CEM for $k = 5$). This suggests that the LLM is capable of answering complex questions accurately when given complete and relevant information.

- **Performance with relevant contexts:** When the LLM was only provided with the top retrieved documents by Contriever (assumed to be mostly relevant), the performance dropped slightly (47.75% CEM for $k = 5$). This indicates that the retrieval process can introduce some noise, making it more challenging for the LLM to identify the answer.

- **Performance with mixed contexts (relevant + negative):** Interestingly, including a mix of relevant and negative contexts (random or hard negatives) led to a slight improvement in performance compared to using only relevant contexts. This effect was observed for both 12 and 24 total documents provided to the LLM.

- **Impact of hard negatives:** The inclusion of hard negatives did not significantly improve performance compared to random negatives. This suggests that the benefit of negative contexts comes from providing a broader range of information, rather than specifically introducing difficult-to-distinguish negatives.

Overall, our findings suggest that providing a mix of relevant and negative contexts can be beneficial for RAG systems, as long as the negative contexts are not overly challenging for the LLM to distinguish from relevant ones. This approach can help the LLM learn a more robust representation of the information space, leading to improved performance on complex QA tasks.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. 2024. The power of noise: Redefining retrieval for rag systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 719–729.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Jiafeng Guo, Yinqiong Cai, Yixing Fan, Fei Sun, Ruqing Zhang, and Xueqi Cheng. 2022. Semantic models for the first-stage retrieval: A comprehensive review. *ACM Trans. Inf. Syst.*, 40(4).

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

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

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Qi Liu, Bo Wang, Nan Wang, and Jiaxin Mao. 2024. Leveraging passage embeddings for efficient listwise reranking with large language models.

Meta. 2024. Llama 2 - 7b chat hf. Accessed: 2024-12-19.

Venktesh V Deepali Prabhu and Avishek Anand. 2024. Dexter: A benchmark for open-domain complex question answering using llms. *arXiv preprint arXiv:2406.17158*.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5835–5847, Online. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Zhen Yang, Ming Ding, Tinglin Huang, Yukuo Cen, Junshuai Song, Bin Xu, Yuxiao Dong, and Jie Tang. 2024. Does negative sampling matter? a review with insights into its theory and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(8):5692–5711.

Shuangfei Zhai, Tatiana Likhomanenko, Etai Littwin, Dan Busbridge, Jason Ramapuram, Yizhe Zhang, Jiatao Gu, and Joshua M Susskind. 2023. Stabilizing transformer training by preventing attention entropy collapse. In *International Conference on Machine Learning*, pages 40770–40803. PMLR.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1503–1512. ACM.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Wayne Xin Zhao, Jing Liu, Ruiyang Ren, and Ji-Rong Wen. 2024. Dense text retrieval based on pretrained language models: A survey. *ACM Trans. Inf. Syst.*, 42(4).

A Results of Cleaned LLM Response

Experiment	Cover Exact Match %			Exact Match %		
	Top 1	Top 3	Top 5	Top 1	Top 3	Top 5
Relevant	31.50%	28.25%	26.25%	1.75%	**1.75%**	0.58%
Oracle	29.50%	26.83%	28.50%	**2.33%**	0.92%	**0.75%**
Adore	**32.00%**	**30.17%**	**30.08%**	2.00%	0.83%	0.25%

Table 4: Same as table 1 but with cleaned answers as discussed about in Section 3.4. Comparison of CEM and EM performance across Experiment 1 (Relevant), Experiment 2 (Oracle), and Experiment 5 (Adore). The highest percentage in each column is highlighted in bold.

Total #Docs	% of Relevant	Cover Exact Match %		Exact Match %	
		Random	Hard	Random	Hard Negatives
12	0%	20.17	18.67	0.00	0.00
	25%	22.83	23.42	0.00	**0.17**
	50%	26.33	26.00	0.00	0.00
	75%	28.67	27.25	0.00	0.08
	100%	28.00		0.17	
24	0%	11.42	16.92	0.67	0.00
	25%	22.67	20.00	0.00	0.08
	50%	21.83	20.83	0.08	0.00
	75%	22.42	22.67	0.08	0.00
	100%	21.83		0.0	

Table 5: Same as table 3 but with cleaned answers as discussed about in Section 3.4. Comparison of experiments with different ratios of relevant and non-relevant documents. In Experiment 2, relevant contexts are mixed with random contexts, and in Experiment 3, relevant contexts are mixed with hard negatives. The bold values represent the best performance for each configuration. The total document count shows the number of documents (relevant + non-relevant), and the relevant/non-relevant ratio is indicated for each mix.