Few-Shot Medical Open QA

Ju Zhang
Stanford University
juzhang@stanford.edu

Noah Kuo
Stanford University
noahkuo@stanford.edu

Abstract

In this paper, we explore the feasibility of few-shot open QA in the medical domain. Few-shot open QA is a variant of standard question answering where there is no gold passage provided and there is no task-specific generator training. It has previously not been explored for biomedical datasets. We propose several different retriever-generator models and analyze the effectiveness using several different metrics. We find that ColBERT retrieval works well for the task, and it can be improved further using additional modifications. We find that the few-shot generator models do not obtain good performance compared to language models specifically trained for biomedical data, but we believe that there are promising results that can be explored further in the future.

1 Introduction

Few-shot open QA is a task within the question answering domain that has its unique challenges and techniques. Research focusing on this area has gained momentum in recent years, as it has broader potential application scenarios. Our project aims to explore the feasibility of few-shot open question-answering (QA) in the medical domain and evaluate the performance of different models on this task.

Traditional QA tasks rely on the user to provide the right context to the system. Generally speaking, each question is associated with a gold passage, and the answer is guaranteed to be a substring located somewhere within the passage. After receiving the gold passage, a reader model tries to predict the span of the answer within the passage and extract the text as the answer (Saini and Yadav, 2017).

In contrast, for open QA, there is no gold passage provided to the model; the model must sift through a collection of passages and decipher where an answer might lie. This is often done with a retriever-reader/generator model, where the retriever identifies which passages in a large corpus are relevant to the question, and the reader/generator then takes the subset of passages and attempts to find or generate the correct answer to the question (Zheng et al., 2021). Few-shot open QA presents an additional challenge, where the generator does not receive any task-specific training. Although these variations can make the overall task more difficult, a few-shot open QA is more widely applicable with more uses than a standard QA model.

Although QA in the medical field has been researched before, few-shot open QA has not. Medical information can be scientifically complex and detailed, and some models may perform poorly on the medical QA tasks, especially when they are not pretrained on the medical corpus. Consequently, it is interesting for us to explore the few-shot open QA for medical information. In this project, We build several retriever-generator models based on different retriever models, including ColBERT, BERT, BM25, and DPR, and compare the performance across them and with a standard QA model pretrained on medical corpus and fine-tuned on QA tasks. Meanwhile, we explore several improvements to the retriever system and assess whether they can lead to better performance.

2 Prior Literature

Past studies have shown the effectiveness of different approaches to open QA. Most rely on a retrieval-reader/generator model, as mentioned previously. Within this framework, there are still many different ways to tackle both the retriever and reader/generator components.

Traditional retriever relies on a ranking function that calculates scores based on certain text properties of a query and a passage. This includes BM25, which ranks passages using a bag-of-words model, calculating a score for each passage based on inverse-document-frequency and word counts of query words (Robertson and Zaragoza, 2009).

Recently, neural ranking models have become in-

creasingly important for information retrieval. One example is the work by Qu et al. 2021, who experiments with a dense passage retrieval system. An effective retriever can be built only using dense representations alone (instead of sparse representations), where embeddings are learned solely from a small number of questions and passages with a dual-encoder. All passages are encoded into output vectors, and then all questions are also encoded into a vector with the same dimensionality using a different encoder. The similarity score is calculated as a simple dot product - encoders are trained to assign higher similarity (larger dot product) to more relevant passages and questions and lower similarity otherwise.

Another group of researchers described the capability of using BERT for passage ranking, and retriever (Nogueira and Cho, 2019). In the system proposed, the query (question) and passage are concatenated together and fed into the BERT model. The output from pooler layer was used to score the similarity between the query and the passage. The passage with the highest score is retrieved. Since the original BERT was only trained on masked language modeling and next sentence prediction, it needs to be trained for the ranking task. In the article, the authors trained the model on the MS MARCO, a dataset sampled from Bing's search query logs (Nguyen et al., 2016). However, this method is computationally-intensive and impractical to use in an online setting. As a result, the authors first used BM25 to retrieve top K passages and then rerank them using BERT to accelerate the retrieval.

In the study published in 2021, the researchers proposed a system that involves using the ColBERT ranking model for passage retrieval in Open QA (Khattab et al. 2021). ColBERT is a model previously introduced in the another paper for document retrieval (Khattab and Zaharia, 2020) . ColBERT encodes passages into a matrix of token-level embeddings and relies on contextual late interaction to estimate the relevance scores between a query and a document. ColBERT performs much better on several retrieval tasks and metrics compared to all other non-BERT baseline models and is much more efficient than BERT since it does not use an all-to-all interaction. The model also performs very well for both retrieval and end-to-end OpenQA compared to previous baselines. ColBERT retrievers achieve high success@20 scores that are several points above BM25, DPR, and single-vector ColBERT. The large variant of ColBERT-QA attains state-of-the-art performance on various datasets for end-to-end Open QA, including SQuAD.

Despite these new recent methodologies, Open QA has not been fully explored in the medical domain. However, there have been some forays into the standard question answering task for medical knowledge. For example, in Tech et al. 2019, researchers employed a hierarchical neural retrieval model that uses a deep attention mechanism at multiple levels. The model uses hierarchical self-attentions in three places: the first self-attention layer is in the queries over different words in a sentence to give higher priority to more important words while creating a pooled representation of the query. The second self-attention layer is in the documents over different words in a sentence to give higher priority to more important words. Lastly, a third self-attention layer is used over different sentences in a document to successively select the document features that are most relevant to the query. Finally, the representations from queries and documents are used to compute a similarity score that could be used to rank documents given a query. The authors discovered that their proposed model outperformed all the baseline models by a considerable amount.

Other techniques have also been attempted for biomedical question answering (Du et al., 2022). This includes multiple data augmentation strategies, a model weighting strategy, and adversarial training. One example of data augmentation was TextRank for summarization (Barrios et al., 2016). The authors used this method to generate summaries of each context to increase the number of shorter contexts in training data. With regard to the model weighting strategy, the authors tried to develop a method to combine two models: QANet and BioBERT. In the beginning, the two models were trained respectively. In order to make the final prediction, the authors introduced a model-weighting function that takes the prediction probability and cosine similarity between the generated answers from QANet (Yu et al., 2018) and BioBERT (Lee et al., 2019). For adversarial training, the authors used a method to generate adversarial samples on the BioASQ training dataset. This method works by inserting interference sentences into the context. This method involves applying semantics-altering perturbations to the question, creating a fake an-

swer that has the same type as the original answer, and finally combining the altered question and fake answer into declarative form.

3 Data

For our project, we use the BioASQ 10b dataset, which includes approximately 4234 questions and answers. This dataset was prepared for the 2022 BioASQ challenge, which is a yearly-held competition on large-scale biomedical semantic indexing and question answering (Tsatsaronis et al., 2015).

All of the questions in the dataset ask about specific medical knowledge based on a collection of abstracts of medical publications, and the answers, which are manually curated by a group of biomedical experts, are given in different formats. Based on the format of answers, this dataset can be classified into four subtypes: answers with a simple Yes vs. No, answers with a list of short phrases or words, answers with a factoid string, and answers that summarize multiple snippets.

Type	Factoid	List	Summary	Yes/No
Number	1252	816	1018	1148

Table 1: **Distribution of types of questions/answer pairs.**

Table 1 shows the distribution of QA pairs by type. By assessment, we discovered that only the factoid question and list questions are suitable for our intended few-shot openQA experiments. For yes/no questions, it may involve using some classifiers. For summary questions, it may involve evaluation based on different metrics such as ROGUE and BLEU scores. Consequently, we choose to filter out the yes/no and summary questions. The final datasets we obtain are composed of 1252 factoid questions and 816 list questions. Without differentiating between the two types, we aggregate and process them together.

As Figure 1 shows, the dataset is provided in JSON format containing multiple fields, only part of which are relevant to the QA task. Noticeably, instead of providing a whole paragraph of context, the data provides a list of snippets, which are all excerpts from various abstracts of medical publications. This is different from the benchmark SQuAD dataset, where for a single question, multiple answers and one context passage are given (Rajpurkar et al., 2016). To simplify the experiment, we decide to concatenate all text from snippets together

{'body': 'List signaling molecules (ligands) that interact with the receptor EGFR?',
'documents': [...],
'triples': [...],
'ideal_answer': ['...'],
'exact_answer': [['epidermal growth factor'],
['betacellulin'],
['epiregulin']],
'concepts': [...],
'type': 'list',
'id': '55046d5ff8aee20f27000007',
'snippets': [{'offsetInBeginSection': 1085,
'offsetInEndSection': 1199,
'text': 'the epidermal growth factor receptor (EGFR) ligands, such as epidermal growth factor (EGF) and amphiregulin (AREG)',
'beginSection': 'abstract',
'document': 'http://www.ncbi.nlm.nih.gov/pubmed/24323361',
'endSection': 'abstract'},
{'offsetInBeginSection': 1139,
'offsetInEndSection': 1247,
'text': ' EGFR ligands epidermal growth factor (EGF), amphiregulin (AREG) and transforming growth factor alpha (TGFα)',
'beginSection': 'abstract',
'document': 'http://www.ncbi.nlm.nih.gov/pubmed/24124521',
'endSection': 'abstract'},
...]}

Figure 1: **A list-type question sample from BioASQ.** The question (red), answer (blue) and context (brown) are highlighted. Data fields irrelevant to the QA task has been hidden for better visualization.

to generate a single paragraph of text. The resulting passages are often significantly longer than the SQuAD passages. As the preliminary strategy, we truncated the passage to 1024 tokens to accelerate experiments, preserve memory, and prevent certain models' incapability of ingesting overly long sequences.

Since we aim to perform few-shot open QA task, we divide the data into a 10/10/80 train/dev/test split with a fixed random seed. Note that the number of test examples will far outweigh the other two partitions due to the type of task.

4 Model

4.1 General Model Architecture

Our few-shot open QA system is based on the retriever-generator model structure, which is composed of a passage retriever and text generator, shown as figure 2. The retriever retrieves the most-relevant passage from the context corpus based on the question (query). Then a prompt is built using a few shots of the BioASQ training subset and the retrieved passage, which is next fed into the generator system. The generator model tries to generate the correct answer based on the built prompt.

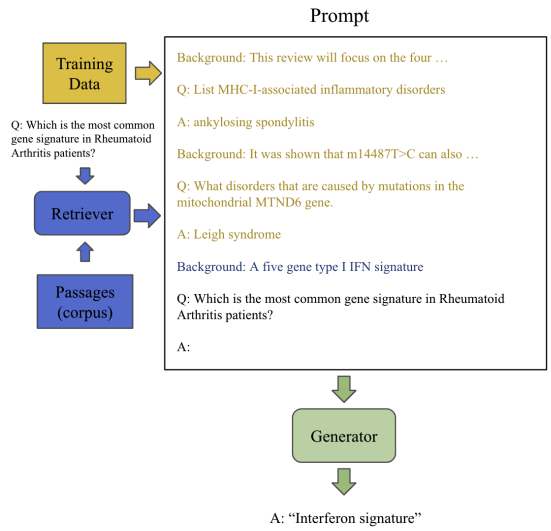

Figure 2: **Model architecture overview.** The passage retrieved by the retriever is highlighted in blue and text contents from train split is highlighted in yellow.

4.2 Generator Model

4.2.1 GPT-Neo

For our generator, we use the GPT-Neo model pre-trained by EleutherAI in this study (Black et al., 2021). The GPT-Neo model is similar to GPT-2 by OpenAI (Radford et al., 2019), although it utilizes local attention in every other layer using a window size of 256. The model is pre-trained on the Pile dataset rather than on biomedical language specifically. The model comes with several versions with different sizes of parameters. The version we use in this study is the one with 125M trainable parameters, which is a smaller size compared to other GPT-Neo versions that can include up to 2.7B parameters. We choose to use this smaller version for the sake of fast testing our hypotheses instead of achieving the state-of-art benchmarks. Therefore, GPT-Neo-125M is used in all retriever-generator systems we implemented.

Our generator tokenizes the input prompt, then generates text to complete the prompt from the retrieved context using the HuggingFace GeneratorMixin. Generated probabilities are also calculated.

4.3 Baseline Models

4.3.1 BM25 Retrieval with GPT-Neo generator

Our baseline system uses the BM25 ranking function to rank the relevance of all passages relative to the input question. The passage with the highest BM25 score is retrieved for building the text prompt for the GPT-Neo generator.

4.3.2 BioBERT

Another baseline model we use is BioBERT, with modifications for the QA task. BioBERT has nearly identical architecture as the original BERT but is pre-trained on large-scale biomedical corpora using the same masking and next sentence prediction training strategy as BERT (Lee et al., 2019). There are a few modifications from the original BioBERT model in order to be used for question answering. An additional softmax layer is added in order to transform the final output state into probabilities for start and end tokens. The model is fine-tuned using SQuAD. Using this model, researchers obtained the best performance on the 7th BioASQ Challenge, an earlier version of the question-answering dataset described earlier (Yoon et al., 2019). This baseline model obtains similarly strong results on our own dataset but does not necessarily fit the few-shot open QA task introduced earlier. We chose to provide the correct gold passages when we ran experiments with the BioBERT language model in order to obtain a "ceiling" for our experiments, so experiments on BioBERT would not be considered to be open QA. We do not expect few-shot open QA models to outperform BioBERT QA with gold passages, but we simply use this as a reference point.

4.4 Retrieval Models

We experiment with several different neural retrieval models in order to compare performance between various methods. We use the same generator model described in the baseline models in order to isolate the impact of different retrieval models.

4.4.1 ColBERT

Our first system uses ColBERT retrieval without modifications. We follow the process described in Khattab et al. 2021 to index all passages included in the BioASQ dataset. To index all passages, we use the checkpoint of the ColBERT model that was pre-trained on MS MARCO but not finetuned on a medical corpus.

4.4.2 ColBERT with Normalized Answer Scoring

We experiment with improving the ColBERT retriever using normalized answer scoring. Our

baseline system assumes that the passage with the top-ranking according to the retriever should be used for the answer. For answer scoring, we assign a score to answers from different passages by calculating the conditional probabilities $P(passage|question)$ and $P(answer|passage, question)$. As an additional step, we also normalize answer scoring by length, which may help reduce the retriever's bias against long answers.

4.4.3 Dense Passage Retrieval

We also experiment using dense passage retrieval (DPR) method (Qu et al. 2021). In this method, retrieval is implemented using only dense representations, where embeddings are learned from a small set of questions and passages by a dual-encoder stack. We utilize the HuggingFace transformers library to implement the model and work with a DPR model pretrained by Facebook AI on Google's Natural Questions dataset.

To retrieve the passage from all candidates, we first calculate k-dimensional passage embeddings for all passages. Then, for a given question, we also generate a k-dimensional embedding and calculate the dot product between the question embedding and all passage embeddings. The retriever selects passages with the highest dot product. The passage embeddings do not change for each question, allowing them to be pre-computed for efficiency.

4.4.4 BERT Retrieval

We also implement BERT for the passage retrieval system as described in the prior literature section. In the system, a given question is concatenated to one passage in the corpus and fed into a BERT model. The pooler output generates a logit score that represents the similarity between the question and passage, which is used to rank the passage's relevancy.

We utilize the HuggingFace transformers library to implement the model and work with a BERT model pretrained by NBoost on MS MARCO dataset (Thienes and Pertschuk, 2019). Since an all-interaction BERT retrieval model is computationally expensive, our hardware and budget in this project do not support testing the model in full on the whole BioASQ test subset. We only choose 250 questions from the test subset for performance evalutation.

4.4.5 T5 Doc2Query Passage Augmentation

We additionally perform T5 Doc2Query Passage Augmentation in combination with all models discussed prior and assess the impact on the model performance (Cheriton, 2019). Doc2Query (Nogueira et al., 2019) is a method that uses a sequence-to-sequence transformer to augment the passage in order to improve the retriever's performance. In our experiment, we use the T5 model to predict top-K questions for each passage might answer and append them at the end of the original passage. The augmented documents are then re-indexed by the retriever model. We utilize the HuggingFace transformers library to implement the model and work with a T5 Doc2Query model pretrained on the MS MARCO dataset. Note that we have not fine-tuned this model and the questions predicted are queries from Bing search.

5 Methods

For our experimental approach, we pre-process data using the methods described in the Data section. Next, we index all passages to be retrieved. For each question, we use our retriever and generator model to generate the top answer and probability.

For factoid questions, correctness is measured the same as other standard QA tasks. For list questions, the question has a list of responses. These are treated as multiple correct options, similar to SQuAD. Metrics used are Average EM, Macro Precision, Macro Recall, and Macro F1.

EM is "exact match," which equals to 1 if the generated answer exactly matches one of the answers and 0 otherwise. Average EM is the average of EM scores across all questions. Macro Precision and Recall is simply the average precision/recall of all questions. Macro F1 is the average F1 of all questions (F1 is twice the product of precision and recall, divided by the sum of precision and recall).

We also perform quantitative analyses on the retrieval system using success@K and mean reciprocal rank (MRR@K). Success@K represents the percentage of prompts where the top-K passages retrieved include one of the answer substrings. MRR@K represents the average reciprocal rank (RR) of all queries, where RR is defined as the reciprocal of the rank of the first relevant passage in the top-K results. We chose $K = 5$ for both metrics since our number of candidate passages is relatively small.

	Success@5	MRR@5
BM25	0.5915	0.4804
BM25 w/Passage Augmentation	0.6393	0.5196
Dense Passage Retrieval	0.5184	0.3571
Dense Passage Retrieval w/Passage Augmentation	0.5444	0.3849
ColBERT	**0.7970**	0.7667
ColBERT w/Passage Augmentation	0.7958	0.7682
BERT	0.7894	**0.812**

Table 2: Retrieval Metrics

All experiments are conducted in an environment with Ubuntu 18.04 and Python 3.9.7, on an Amazon AWS EC2 allocated with a single NVIDIA Tesla T4 GPU with 16GB graphic memory.

6 Results

We analyze all of the retrieval model performance based on success@5 and MRR@5, which result is presented in Table 2. We find that ColBERT performs the best for Success@5 and BERT is the best for MRR@5. Note that due to resource limitation, the BERT-based retrieval is evaluated on approximately 15% of cases in the test subset only, which means the number may be subject to randomness.

Results from analyzing the full question answering system are displayed in Table 3 and 4. Table 3 contains the results from systems that do not use passage augmentation, while Table 4 contains results from systems that do. We find that ColBERT and BERT models have comparable top performance before further improvement, and utilizing answer scoring in ColBERT notably improves the system's performance.

To understand the respective contributions from the retriever and generator to the overall system performance, we also compared how the GPT-Neo generator model performed with a QA model, BioBERT, that had been undergoing task-specific training when both were given gold passages. We also compared our version of GPT-Neo, which had 125M parameters, with a larger version that contained 1.3B parameters in this scenario. The results are presented in Table 5.

7 Analysis

First, there is a clear advantage for the retriever system based on ColBERT compared to BM25 baseline and DPR systems. As we can observe from Table 2, ColBERT retrieval achieved the highest success@5 performance, while BERT with passage augmentation had the highest MRR@5 performance on the dataset. However, the BERT retriever requires intensive computing and runs much slower than ColBERT (data not shown), even making full-scale testing on the model performance given our project resource infeasible. Another interesting finding from Table 2 is that passage augmentation has a noticeable positive impact on the performance of the BM25 and DPR systems but does not make as much of a difference for the ColBERT retriever.

As a result, it is within our expectation that the retriever-generator model based on ColBERT and BERT leads to the best performance across the board. Table 3 and Table 4 show that ColBERT and BERT systems are similar in all metrics (average EM, macro precision/recall/F1) and are both superior to the BM25 and DPR based systems. The best system for our dataset is ColBERT with answer scoring but without passage augmentation. In general, although passage augmentation improves retrieval metrics, it seems to have a slightly negative impact on the generator since the performance of the overall system went down in most cases. We observe that for the retriever metrics alone, passage augmentation increases the Success@5 score and decreases MRR@5 score for ColBERT. Therefore it calls for further research on the impact of passage augmentation on ColBERT retriever, such as evaluating metrics like NDCG as well as testing on other datasets.

Another interesting finding comes from analyzing the standalone performance of the generator models given gold passages in Table 5. The larger version of GPT-Neo with 1.3 billion parameters performed favorably compared to the more lightweight version with 125 million parameters, which is to be expected. However, the lightweight version of GPT-Neo actually performed worse when given the gold passages instead of the passage retrieved by

	Average EM	Macro F1	Macro Precision	Macro Recall
BM25	0.0211	0.0928	0.0867	0.1636
Dense Passage Retrieval	0.0097	0.0807	0.0760	0.1415
ColBERT	0.0314	0.1180	0.1090	0.2057
ColBERT w/ Answer Scoring	**0.0773**	**0.1641**	**0.1719**	**0.2268**
BERT	0.0280	0.1182	0.1131	0.2016

Table 3: Results for Retrieval Models without Passage Augmentation

	Average EM	Macro F1	Macro Precision	Macro Recall
BM25	0.0151	0.0807	0.0753	0.1425
Dense Passage Retrieval	0.0121	0.0803	0.0745	0.1412
ColBERT	0.0278	0.1131	0.1025	0.2038
ColBERT w/ Answer Scoring	0.0725	0.1505	0.1537	0.1989

Table 4: Results for Retrieval Models with Passage Augmentation

	Average EM	Macro F1	Macro Precision	Macro Recall
GPT-Neo	0.0224	0.1112	0.0989	0.2080
GPT-Neo Large	0.1190	0.2888	0.2762	0.4246
BioBERT	**0.2592**	**0.4801**	**0.4859**	**0.5945**

Table 5: Generator Metrics given Gold Passages

the models based on ColBERT. We surmise that two reasons may cause it: first, the generator model does not need to extract the answer from the gold passage; second, many of the answers in BioASQ are not exact text excerpts from the gold passage but more like a summarized concept from the passage. Therefore a "relevant" passage retrieved may give the generator better heuristics than the gold passage. Finally, both these models also did not perform as well as BioBERT, which achieved the highest performance by far. This also makes sense, given that BioBERT receives task-specific training and has achieved state-of-the-art performance on past versions of the BioASQ dataset. The few-shot nature of most of our models clearly leads to a lower performance compared to other models.

8 Conclusion

This paper establishes the performance of few-shot open QA models on biomedical datasets and compares various different approaches for passage retrieval on this task. Our main finding is that systems that are based on ColBERT have the highest performance as well as very efficient computation, and making small modifications such as answer scoring can improve results. On the other hand, other modifications such as passage augmentation helped retrieval in some cases but adversely affected overall performance in other cases. We also find that few-shot models do not perform nearly as well as language models that have been trained on biomedical corpora and QA tasks, such as BioBERT. Further research could be done to identify other types of performance improvements on ColBERT. For example, fine-tuning the retrieval system could improve the performance of the overall system, although it would come at the expense of computing resources. Another potential idea would be to extend some of the same ideas and retriever models used to other types of information besides biomedical data. It could also be used on a general dataset like SQuAD. Additionally, rank fusion is an interesting direction in which we can explore whether a combination of the retrieval results from several models can lead to performance improvement.

Known Project Limitations

The availability of a quality medical QA dataset has annoyed the medical NLP/NLU community for a long time. It also applies to our research here. We rely on the BioASQ dataset, which is probably one of the few quality biomedical QA datasets that can fulfill our requirement for few-shot openQA tasks. Nevertheless, it contains only 4,000+ questions, much smaller than the size of SQuAD, which contains more than 100K questions. In addition,

given the structure of the dataset and the nature of the task, we were limited to only using a subset of the BioASQ dataset. We did not focus on classifying "yes/no" answers or the "summarization" answers, which makes the available dataset smaller to approximately 2,000+ questions. Furthermore, for the list questions, we considered answers to be an "exact match" if they matched one of the answers in the list. However, in the BioASQ task, answers needed to contain multiple of the answers in the list. Finally, the passage in the dataset comes in multiple snippets for a given question, and there is no standard way to process the dataset. We took a simple method that concatenates all snippets and truncates that to 1024 tokens. In this sense, the logical relation between concatenated snippets may not make sense at all and may negatively impact the system's performance.

We were also limited by the computing resource for this research and have not had a chance to assess many computational intensive experiments. For example, we did not employ the GPT-Neo-3.7B model for the generator model, which will be very difficult for our single T4 GPU. The results were obtained using a more lightweight and efficient GPT-Neo-125M model unless otherwise mentioned, which meant that stronger performance could have been achieved with other generator models. Another example is that our evaluation of BERT is not complete in the sense that we only ran its experiments on 15% of the test subset and did not proceed to test that on augmented passage. A smarter implementation of the model may help reduce the running time but having good computing power will be very helpful in completing the experiment.

Authorship Statement

Ju Zhang assembled the dataset and ran the BM25, DPR, ColBERT, BERT, and passage, augmentation models. He also made figures and literature reviews of medical domain QA research papers. Noah Kuo ran the BioBERT tests and wrote most of the project report, and made all result tables. He also did the literature review of open QA research papers. Some project code also made use of the Open QA HW code, in which modules like answer scoring were implemented during the homework or provided by the CS224U teaching team.

References

Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization. *ArXiv*, abs/1602.03606.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow. If you use this software, please cite it using these metadata.

David R. Cheriton. 2019. From doc2query to doctttttquery.

Yongping Du, Jingya Yan, Yiliang Zhao, and Xingnan Jin. 2022. Improving biomedical question answering by data augmentation and model weighting.

Omar Khattab, Christopher Potts, and Matei Zaharia. 2021. Relevance-guided supervision for OpenQA with ColBERT. *Transactions of the Association for Computational Linguistics*, 9:929–944.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over BERT. *CoRR*, abs/2004.12832.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. *CoRR*, abs/1611.09268.

Rodrigo Nogueira, Wei Yang, Jimmy J. Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *ArXiv*, abs/1904.08375.

Rodrigo Frassetto Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with BERT. *CoRR*, abs/1901.04085.

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.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3:333–389.

Anjali Saini and P. K. Yadav. 2017. A survey on question –answering system. *International Journal of Engineering and Computer Science*.

Ming Zhu Virginia Tech, Ming Zhu, Virginia Tech, Aman Ahuja Virginia Tech, Aman Ahuja, Wei Wei Google, Wei Wei, Google, Chandan K. Reddy Virginia Tech, Chandan K. Reddy, and et al. 2019. A hierarchical attention retrieval model for healthcare question answering: The world wide web conference.

Cole Thienes and Jack Pertschuk. 2019. Nboost: Neural boosting search results. https://github.com/koursaros-ai/nboost.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, et al. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16(1):138.

Wonjin Yoon, Jinhyuk Lee, Donghyeon Kim, Minbyul Jeong, and Jaewoo Kang. 2019. Pre-trained language model for biomedical question answering. *CoRR*, abs/1909.08229.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. *CoRR*, abs/1804.09541.

Dequan Zheng, Jing Yang, and Baishuo Yong. 2021. Open domain question answering based on retriever-reader architecture. *Business Intelligence and Information Technology*.