Domain Adaptation in Abstractive QA

Zhenjie Jiang
zhejiang@ethz.ch

Antonio Lopardo
alopardo@ethz.ch

Mattei Andrea
amattei@ethz.ch

Abstract

In this paper, we demonstrate that for domain specific QA, the size of the non-parametric memory for RAG can be cut down significantly, for medical domain we managed to reduce the size to less than 5 % compared to the non-parametric memory used in the original RAG implementation. Further, we also managed to replace the encoder for indexing the non-parametric memory and encoding the questions to a domain specific model, which improved the QA performance under the specific domain.

1 Introduction

In this paper, we aim to provide a practical study on how to adapt a SoTA abstractive QA system, namely RAG (Lewis et al., 2021), to a specific domain, in our case, the medical field. We detail which components of the model should be fine-tuned, which should be substituted to domain adapted ones and which can be left as is. To better understand RAG itself, though, it's useful to have some context on the recent developments of the Question Answering task more generally.

Question answering has been a very active area of research in the five years since the release of SQuAD (Rajpurkar et al., 2016). The dataset has facilitated the development of many well-performing systems that, given a context passage and a question, can predict the continuous span in the context passage that, most likely, provides an answer. These models, thus, perform extractive question answering. They deal with language though the task they optimize for is classification.

Yet, the Exact Match (EM) and F1 scores of the best systems both for SQuAD v1 and v2 (Rajpurkar et al., 2018), as of November 11, 2022, have exceeded human performance, and the focus of many researchers in the field has been shifting towards models that can perform extractive QA without direct access to context passages as well as abstractive QA. As opposed to Extractive QA, in abstractive QA no context passage is provided, so models have to generate correct and grammatical answers on their own.

Thus, the challenge that any model tackling this task has to overcome is twofold. Indeed, to come as close as possible to the target answer or answers, not only should it have access to the information needed to answer the question correctly, but it should also be able to produce well-formed sentences.

Large transformer-based models can generate coherent text, especially when using the sequence to sequence paradigm, see BART (**?**) and T5 (Raffel et al., 2020). Some of these models have even been used to write articles or reviews, that while not always being plausible or fully semantically coherent, are an effective display of the models' capabilities. Moreover, the sheer size of some of the more recent transformers-based models suggests that as part of their language modeling task, they encode in their parameters some world knowledge. Several papers support this notion(Petroni et al., 2019) (Roberts et al., 2020), but others highlight the limitations and pitfalls of trusting the knowledge encoded in the models' parameters to be accurate(Wallace et al., 2019) (Forbes et al., 2019). In fact, LMs like BERT have been shown to "hallucinate" knowledge that they have never received in input (Marcus, 2020), making it difficult to rely on them in real-world situations where accuracy is a priority.

Often, in these circumstances, textual or logical knowledge bases (KBs) have been used as sources for the models. Introducing a KB adds different dimensions to the task. What's a practical size for the KB, what should be the format of the data be to make it easier to retrieve it, how can we compare

the query to the data in the KB to avoid retrieving useless information? Are just some of the questions that arise when dealing with KBs for QA.

DrQA (Chen et al., 2017) was one of the first systems that combined non-parametric memory and deep learning to answer open-domain questions from SQuAD using Wikipedia. Its architecture is composed of a retriever used to select relevant passages from the KB and a document reader tasked with selecting the correct span among the retrieved documents based on the question. The model didn't match the F1 scores achieved by competing extractive QA approaches with direct access to context passages, but it was a valuable proof of concept that laid the groundwork for today's SoTA solutions in abstractive QA and QA with external knowledge sources. Most recent systems that use Wikipedia as the non-parametric memory mirror DrQA's architecture.

The most significant difference between the current SoTA and DrQA concerns the inner workings of the retriever. DrQA uses TF-IDF to compare query and articles to retrieve the most relevant ones from the thextual KB, while more recent systems often include the Dense Passage Retriever (DPR) (Karpukhin et al., 2020). DPR uses BERT to encode both the passages in the non-parametric memory and the query. Then, to choose the top-k passages to retrieve, it performs a maximum inner product search between the encoding of the query and the pre-encoded and pre-indexed passages. RAG'architecture includes the DPR as well BART(Lewis et al., 2019). Thus, we found RAG a fitting target for further study, and as we mentioned at the beginning of the introduction, we were interested in how it can be adapted to perform better or be easier to manage when applied to a specific domain.

2 Related Work

In this section we will detail more extensively papers and models mentioned in the introduction as well as some related architectures that we will compare to RAG, namely T5 and BART.

DrQA The DrQA model has a split architecture containing both a document retriever and a document reader, as we mentioned in the introduction. The document retriever uses the Term Frequency — Inverse Document Frequency (TF-IDF) technique to compare passages and the query as well as hashed bigram features to return the top five Wikipedia articles related to the question. The document reader encodes all the paragraphs in the articles it receives from the retriever using a bidirectional RNN. While the query encoding is simply a weighted average of the word-embeddings of its words. The starting and ending positions are predicted using a bilinear term, $\exp(\mathbf{p}_i \mathbf{W} \mathbf{q})$, with different weight matrices for the start and end token. The DrQA paper is worth mentioning also because it provides useful best practices on how to deal with Wikipedia articles, like removing lists or tables and ignoring disambiguation pages.

Dense Passage Retrieval Compared to DrQA DPR uses dense passage and query embeddings for retrieval. The DPR works Wikipedia articles too, preprocessed similarly to DrQA but split in passages of 100 words to make encoding more practical. Passage encoding and indexing can take very long depending on the size of the textual KB, but it has be done only once in a preprocessing step. At testing time, meanwhile, the authors reported that DPR is close to 50 times faster than BM25 (Robertson and Zaragoza, 2009). The query and the passages are encoded using BERT, and the similarity score between any passage and the question is simply the dot product of the two encodings. Maximum inner product search is used to retriever the most relevant documents. The BERT models that encode the query and passages don't share the same parameters and are finetuned with supervised data to improve DPR's performance before training the rest of the model.

REALM Retrieval-Augmented Language Model Pre-Training or REALM (Guu et al., 2020) is a retrieval augmented QA model similar to RAG, but instead of using a pre-trained DPR, the BERT models that encode questions and passages are trained along with the rest of the model. REALM uses MIPS to retrieve the best documents, but training the entire model at the same time adds complexity. The model "refreshes" the index of the encoded passages by asynchronously re-embedding and re-indexing all documents every several hundred training steps. As we will explain later, RAG takes a different approach, pretraining the DPR without continuously updating its passage encoder and index during training.

T5 T5 is a transformer-based sequence to sequence model that, as one of the LMs with

Model \ Metric	F1	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-Lsum
with context in input					
BART-Large	34.04	38.15	26.96	35.89	35.96
T5-Large	31.46	36.47	25.97	34.06	34.03
RAG-Original	33.2	37.74	25.78	35.30	35.33
without context in input					
BART-Large	3.37	3.52	0.00	3.50	3.49
T5-Large	21.03	25.07	11.25	22.41	22.13
RAG-Original	27.64	32.30	17.36	29.26	29.20
RAG-MedWiki	26.78	31.30	16.94	28.70	28.67
RAG-MedWiki++	27.4	32.17	17.40	29.31	29.39
RAG-MedWiki++ & Bio_ClinicalBERT	28.29	32.37	18.30	29.66	29.57
RAG-Medline plus & Bio_ClinicalBERT	27.98	32.45	17.55	29.79	29.85

Table 1: BioASQ Results

the most parameters, is a fitting representative for abstractive QA systems that don't use non-parametric memory. The model performs very well on TriviaQA (Joshi et al., 2017), WebQuestions (Bordes et al., 2014), and Natural Questions (Kwiatkowski et al., 2019) and has also shown significant generative abilities, to produce new answers for squad, for example. T5 is also interesting because of the fully text-to-text approach used to train it or finetune it for any task.

BART RAG uses BART as its conditioned generator network. BART follows the sequence to sequence paradigm and performs competitively on extractive QA tasks with access to context passages. But this model's forte is in language generation, abstractive dialogue, question answering, and summarization tasks. BART's SoTA rouge scores on XSum make it particularly suitable for RAG since we expect the model to generate new, original text from the 100-word passages in its non-parametric memory. To isolate BART's contribution to RAG's performance, we will run baselines with this model.

3 Methods

This section will detail in full RAG's architecture and describe some of the adaptations we applied to make the model work better in MedicalQA.

3.1 Overview - RAG

We mentioned a few details of RAG already; its major components are thetextual KB or non-parametric memory as the authors refer to it in the model's paper, the Dense passage Retriever, and finally, the BART generator. We use the RAG-Sequence model that computes the probability $p(y)$ of the target sequence given the input sequence, marginalizing over the retrieved documents. In practice, when using BART, each retrieved passage is concatenated to the input sequence consistently with the seq2seq paradigm. During training, the objective is to minimize the negative marginal log-likelihood of each target, $\sum_j -\log p\left(y_j \mid x_j\right)$. The authors use Adam to optimize the generator and query encoder. At test time, the decoding process is more complex as there is the need to run separated beam searches for each retrieved documents, but this improves the model generative capacity. So, ultimately, RAG's architecture and operation are comparatively straightforward. A retriever extracts knowledge from an indexed non-parametric memory using a similarity metric calculated with the input sequence. Then a generator network is conditioned on the extracted knowledge and the input sequence to produce a coherent and correct output. As opposed to other transformer-based LMs or QA systems, getting RAG to work involves more preliminary steps. First, there is the need to design the non-parametric memory. As we will show in our experiments, Wikipedia or subsets of it work well, though, the DPR can accommodate any textual source. Books, manuals other sites could all work. The size of the non-parametric memory can

Model \ Metric	F1	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-Lsum
without context in input					
BART-Large	0.85	4.94	0.11	4.89	4.93
T5-Large	5.19	6.92	2.43	5.88	5.94
RAG-Original	0.47	2.38	0.02	2.35	2.36
RAG-MedWiki	0.32	1.76	0.02	1.72	1.74
RAG-MedWiki++	5.35	7.84	2.36	6.49	6.61
RAG-MedWiki++ & Bio_ClinicalBERT	6.14	8.03	3.15	6.78	6.79
RAG-Medline plus & Bio_ClinicalBERT	6.06	8.02	3.05	6.83	6.86

Table 2: MedQuAD Results, Note: answers from 'BART-Large' and 'RAG-Original' were essentially just gibberish, since RAG also uses BART as the generator, this might be a limitation of BART, see Section 5.3.

be a concern, especially if we add more sources to Wikipedia, due to the preprocessing and indexing steps needed to use the textual KB, but these steps need to be performed only once. Indexing the non-parametric memory involves using pre-trained word or sentence embedding models. These models are readily available, and we show that domain-specific ones can help improve retrieval quality in closed-domain QA. But they need to be fine-tuned on this specific retrieval task using QA pairs that reference relevant context passages in the textual KB. Negative sampling is used to ensure that only pertinent documents are retrieved and that the encodings don't collapse to uniformity. The generator network could be a pretrained seq2seq architecture. Still, it needs to be fine-tuned to deal with relatively long input sequences, depending on the size of the passages in the non-parametric memory. In the following subsections, we will discuss more in detail what we used when running our experiments.

3.2 Non-parametric memory or textual Knowledge Base

Wiki One of the textual knowledge bases we considered was the used in the original RAG implementation. It consists of english Wikipedia stripped of non-textual data and utility pages like the disambiguation ones.

Downloading and processing the knowledge base requires around 150GB of disk space.

MedWiki The total size of the Wikipedia textual knowledge base is around 2 million pages. This results in large memory requirements for the model, rendering it hard to train and operate.

In order to solve this problem, we narrowed down our interest to MedicalQA and made another smaller knowledge base by selecting a subset of the Wikipedia. This subset was generated by selecting the pages contained in the category "Healthcare" and its sub-categories.

The final size of the subset is around 800MB, and requires 6GB of disk space after being processed for RAG, which is around 4 % of the original one.

MedWiki++ Although Wikipedia can provide a large general textual knowledge base, there is no guarantee of its reliability. To improve the quality of the information available to our model we extended MedWiki by adding the source pages of the contexts of the MedQuAD dataset.

Moreover, this allows also to evaluate how accurate is the ability of the retriever to search a specific page especially in the presence of a potentially inconsistent and more diverse textual knowledge base.

The source pages were cleaned from the HTML tags and put entirely in the knowledge base. The result is a small set of 4300 documents that occupies around 16MB.

Medline Plus We also experimented with producing a totally different textual knowledge base that contains only information from reliable and official sources. To do this we scrapped the entire medical encyclopedia along with the glossaries of drugs and supplements of the NIH's Medline website.

To further improve performance, the pages were split by paragraph. Each paragraph form a single document. In this way the we mitigate the partition of the documents into chunks of 100 words

that RAG performs.

The size of the textual Knowledge Base after being processed for RAG was less than 5GB, even smaller compared to the medical subset of Wikipedia.

3.3 Retreiver

We have described the role of the retriever in the overview; now we detail the specific configurations we experimented with.

Dense Passage Retriever (DPR) The standard RAG configuration for the DPR uses the bert-base-cased model available on HuggingFace's model hub. It uses different instantiation of this model for both the query encoder and the passage encoder. The QA pairs with references to relevant contexts needed to train these encoders come from the Natural Questions dataset. The finetuned retriever performs very well on the validation set with above 95% accuracy on relevant passage retrieval.

DPR with Bio_ClinicalBERT Encoder We experimented with changing the query and passage encoders in the DPR to different versions of Bio_Bert(Lee et al., 2019) to make the model more suited to MedicalQA. Bio_BERT is essentially the same as the bert-base-cased model of the standard configuration that has later been pre-trained on a biomedical domain corpora of PubMed abstracts, and PMC full-text articles. The specific version of Bio_Bert that performed best is Bio_Clinical_BERT (Alsentzer et al., 2019). It was also pre-trained on electronic health records. We finetuned Bio_Bert on the retrieval task using a subset of health-related questions from Natural Questions. We selected this subset by only keeping questions that had as relevant context passages health-related Wikipedia articles present in our WikiMed non-parametric memory. The accuracy of this domain specific retrievers was lower on our validation set of health QA pairs, but they made the RAG model perform better end-to-end

End to End training We also tried to apply some of the work that researchers at NVIDIA have recently published to train the entire RAG model synchronously (Sachan et al., 2021), that is including the passage encoder. However, the added training time on our datasets made it hard to compare the results fairly.

3.4 Generator

RAG uses the BART-large pretrained model available on HuggingFace's model hub, finetuned on the natural questions datasets along with the rest of the RAG architecture. We tried to use T5 as an alternative Seq-to-Seq model, but it performed worse in all of our testing. We left the generator as is, but we compare RAG's performance to T5 and BART throughout our benchmarks.

4 Datasets

All the textual KBs we used for the models are composed of documents organized in CVS files. Each document is a tuple containing the title and body of a page.

4.1 BioASQ

The BioASQ dataset (Tsatsaronis et al., 2015) was built in the context of a challenge to develop a system for indexing and retrieving information from biomedical articles.

The dataset is made for abstractive QA, and it consists of questions along with a set of PubMed papers to be used as context for answering. In addition, the dataset also contains at least one golden answer per question, which are annotated by professional medics and the portions of text that should be used for generating it. For some questions, the dataset also provides exact/extractive answers.

For the purpose of the project, we uses the 2021 version of the dataset and focuses on abstractive QA.

4.2 Natural Questions

Natural Questions is general QA datasets, it contains over 300k manually annotated questions. All of them were based on real anonymized Google's search engine queries. The answers were annotated using as a source a Wikipedia page that shows up in the first 5 results of the associated queries.
The answers are presented in 2 ways: the first is a long form answer while the second is a short form one (e.g. YES/NO, TRUE/FALSE, etc...). For our task, we considered only the long form answers.

4.3 MedQuAD

MedQuAD (Ben Abacha and Demner-Fushman, 2019) is a SQuAD-like QA dataset. Unlike the previous two, it was conceived for patients and/or

customers settings. It contains a set of extractive question and answer pairs derived information available on websites of the US's National Institute of Health (NIH). Therefore, their style and register are more accessible to the wider public.

Originally, the datasets contained around 47k pairs, however, only a subset of them were usable. Some answers were removed due to copyright issues (although article from US government are in the public domain, part of these websites was made by third companies who retained copyright of them). Another chunk of them had missing contexts, therefore, it would not be possible to evaluate them correctly.

4.4 MedInfo

Similar to MedQuAD, MedInfo (Ben Abacha et al., 2019) is SQuAD-like dataset. The QA pairs of the dataset come from anonymized patient queries sent to MedLine Plus. Its focus is on commercial drugs, and the sources used to annotate the answers follow the guidelines of the Food and Drug Administration (FDA).
Given the relative small size of the dataset (around 700 QA pairs), we used this dataset for evaluating the models already fine-tuned on other datasets.

5 Experiments

5.1 Overview

Our experiments involve both running models with contexts supplied in the input, and without contexts in the input, with models fine-tuned for three epochs. The main focus of our experiments is QA without the contexts in the input sequence. Although having contexts supplied in the input yields better results, the VRAM usage also increases proportionally with respect to the size of the contexts during both training and inference and the task is significantly less complex. In the case of MedQuAD (Ben Abacha and Demner-Fushman, 2019), the contexts were too long, and we weren't able to feed them as input with the questions. Further, in the real world, context is usually not supplied with the questions.

When the contexts are not supplied in the input, the larger transformers model such as T5-large(Raffel et al., 2020) is still capable of answering the question to a reasonable degree by relying on the knowledge stored within the model itself, as shown in Table 1. But the comparatively smaller model such as BART-Large struggles, which is likely due to the reduced capacity for maintaining knowledge.

The RAG models are equipped with non-parametric memory. And in our testing, RAG-based models beat T5 by over 5% in all mertics when tested on BioASQ as shown in Table 1, Table 3. The addition of the textual Knowledge Base significantly increases the performance over just using the transformers model alone. Although, the textual knowledge bases contain passages in the medical domain, they are not the exact contexts of the specific questions, thus our RAG-based model do not require the user to supply the contexts either as part of the question or the knowledge base. This approach overcomes the limitation of more traditional transformer models without non-parametric memory, since with a textual knowledge base the model can automatically retrieve the information it needs from the knowledge base when answering the questions without the need to store it directly in its weights. The results also show that for domain specific QA, the size of the Knowledge Base can be cut down significantly as demonstrated in section 3.2 without any major impact on the performance. 'RAG-MedWiki', 'RAG-MedWiki++' and 'RAG-Medline plus' all had comparative performance to the original RAG, despite using textual knowledge bases that are less than 5% of the size of the original non-parametric memory. The 'Medline plus' Knowledge Base specifically demonstrates that perhaps the quality of the textual knowledge base is more crucial than its size.

As shown in the Results, for domain specific QA, the performance can be further improved by using domain specific encoders for passage retrieval as demonstrated with 'RAG-MedWiki++ & BioClinicalBERT'. This modification improved the perfomance by about 1% in BioASQ (Table 1).

We have also experimented with replacing the generator of RAG from BART to a medical domain specific model such as 'pegasus-pubmed' (Zhang et al., 2020). However, the results were much worse compare to using BART in the majority of the cases. One possible reason for such poor performance might be that this domain specific model was pretrained for a very specific task, and lacks the ability to generalize to more abstract tasks.

Model \ Metric	F1	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-Lsum
without context in input					
T5-Large	6.94	8.97	1.86	7.86	7.96
RAG-Original	8.83	10.83	1.98	9.20	9.30
RAG-MedWiki	8.52	10.67	2.05	9.16	9.23
RAG-MedWiki++	8.90	11.05	2.31	9.48	9.58
RAG-MedWiki++ & Bio_ClinicalBERT	8.75	10.00	1.86	8.75	8.77
RAG-Medline plus & Bio_ClinicalBERT	8.99	10.69	2.08	8.86	9.32

Table 3: MedInfo Results, Note: For MedInfo experiments, we used the same model trained for BioASQ in order to test how well does the models generalize to other similar datasets. 'BART-Large' is excluded, as it already fails at answering questions in BioASQ when the context is not supplied (the answers were completely gibberish).

5.2 BioASQ

The results are presented in Table 1.

In our experimental setup, we use the ideal answer which is an abstractive answer produced by human annotator as the gold label for the questions. Some questions have multiple ideal answers, for simplicity, we only use the top answer. For the experiments that involved having contexts in the input, the input is formatted as 'context: the_context question: the_question'.

Because we are using the 2021 version of the dataset, some of the newer topics covered by the questions are not at all available in any of the textual knowledge bases used for RAG, which certainly limited RAG's performance. A newer and more up-to-date non-parametric memory can certainly improve the performance of RAG.

5.3 MedQuAD

The results are presented in Table 2.

Questions in MedQuAD (Ben Abacha and Demner-Fushman, 2019) are often paired with very long and specific answers, with which all our models struggled. 'RAG-Original' performed very poorly, which is likely partially a result of using the BART generator, which wasn't capable of capturing the very long answers, which is also mirrored in the poor performance of 'BART-Large'.

'MedWiki++' contains the passages from the contexts of MedQuAD, which significantly boosted the performance of RAG, even though the contexts were not directly paired with the questions. The 'Medline plus' (see section 3.2) textual knowledge base, which does not contain contexts from MedQuAD also performed very well. This shows that, as long as the knowledge base contains information regarding the question, RAG can perform well. This also shows that the Knowledge Base has to contain some relevant information in order to be effective, and the information contained in Wikipedia alone was often not detailed enough for many of the questions in MedQuaD.

5.4 MedInfo

The results are presented in Table 3.

We used the MedInfo dataset to further test the models that were trained on BioASQ. We wanted to see how well fine-tuning on BioASQ translates from dataset to dataset under the same domain for the same task.

The answers provided by MedInfo are not annotated by human experts and aren't always grammatical, which caused the scores to suffer, as our RAG models generates abstractive answers. However, in general the answers generated by the models are reasonably well-formed, and provided revelant answers to the questions, however they have a tendency to be too short and not fully informative. Surprisingly, 'RAG-MedWiki++ & Bio_ClinicalBERT' did outperform the other variants of RAG, which might be caused by the fact that 'Bio_ClinicalBERT' was only pretrained for a subsection of the medical domain, and does not generalize well across the board.

6 Discussion

RAG's practicality In our preliminary tests on open-domain QA datasets like Natural Questions the size of the knowledge base was a critical obstacle. Solving this issue is an important step to enhance the practicality in the wild and the economic viability of knowledge base-enhanced QA

models such as RAG.
In our experiments, we showed that in case of domain-specific applications, it is possible to mitigate this problem by using custom-made textual knowledge bases while retaining or even improving the performance of the RAG model.
We expect there to be still margins for improvements on RAG but due to its requirements, experimenting with this model can be very expensive. Our adaptation and testing took more than four hundred hours of compute time on NVIDIA V100s and A6000s.

RAG's Advantage One benefit of RAG is that the architecture is quite modular, the knowledge base, the retriever, the encoder and the generator are all individually replaceable, which makes it easy to modify for different scenarios.
The MedQuAD experiment clearly showed that Bio_ClinicalBERT was able to perform better than standard BERT in case of same knowledge base. Therefore, this approach is advisable for fields where there are sufficient data for having a corpus big enough to train a custom BERT model such as the medical field or the legal one.

7 Conclusion

In this paper, we show some practical and effective steps to adapt a SoTA abstractive open-domain QA model, RAG, to a specific domain. We perform the QA on the medical domain and detailed how changing the non-parametric memory can make using the model more practical, and how domain adapted query and passage encoders can improve performance.

8 Future Work

One possible future improvement for the model is to train a BART like generator from scratch, that is truly designed for the architecture and optimised for the specific domain, enhancing the answer formation.

Another possible field of future works is enhancing the underlying knowledge base. The custom knowledge base is relative small. In future works, we will expand it by adding more high quality documents. The NIH websites could be used as the main source of these documents because it fulfills the requirements of reliability while being in the public domain.

[1]

References

Emily Alsentzer, John R. Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew B. A. McDermott. 2019. Publicly available clinical bert embeddings.

Asma Ben Abacha and Dina Demner-Fushman. 2019. A question-entailment approach to question answering. *BMC Bioinformatics*, 20(1).

Asma Ben Abacha, Yassine Mrabet, Mark Sharp, Travis Goodwin, Sonya E. Shooshan, and Dina Demner-Fushman. 2019. Bridging the gap between consumers' medication questions and trusted answers. In *MEDINFO 2019*.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question answering with subgraph embeddings.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions.

Maxwell Forbes, Ari Holtzman, and Yejin Choi. 2019. Do neural language representations learn physical commonsense? *CoRR*, abs/1908.02899.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*.

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*.

[1]Code Repository: https://github.com/AntonioLopardo/MedRag

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks.

Gary Marcus. 2020. The next decade in ai: Four steps towards robust artificial intelligence.

Fabio Petroni, Tim Rocktäschel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *CoRR*, abs/1909.01066.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. 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.

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

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? *CoRR*, abs/2002.08910.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Devendra Singh Sachan, Mostofa Patwary, Mohammad Shoeybi, Neel Kant, Wei Ping, William L. Hamilton, and Bryan Catanzaro. 2021. End-to-end training of neural retrievers for open-domain question answering. *CoRR*, abs/2101.00408.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R. Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artiéres, Axel-Cyrille Ngonga Ngomo, Norman Heino, Eric Gaussier, Liliana Barrio-Alvers, Michael Schroeder, Ion Androutsopoulos, and Georgios Paliouras. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC Bioinformatics*, 16(1):138.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP models know numbers? probing numeracy in embeddings. *CoRR*, abs/1909.07940.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.