Dense Passage Retrieval For StrategyQA

Omer Levi
omerlevi2@mail.tau.ac.il

Daniel Glickman
glickman1@mail.tau.ac.il

Abstract

Multi-hop reasoning question answering is a challenging task that relies on efficient passage retrieval to select contexts that are relevant for answering the questions. Recent work in open domain question answering shows promising results using a neural retrieval model, over traditional sparse vector methods, such as TF-IDF or BM25. In this work, we train Dense Passage Retrieval(DPR), a dense vector retrieval model, on the StrategyQA, multi-hop reasoning dataset and show it outperforms BM25, improving both context passage recall and accuracy on the dataset. [1]

1 Introduction

The task of question answering can be challenging when multi hop reasoning is required. Many QA(question-answering) models use pre-trained language models that hold a lot of knowledge and can excel in many tasks including QA tasks. Those models are fine tuned for specific tasks and are used successfully as QA models in many cases. However, the use of such closed book language models is often not enough in cases where the questions are more complex. For example, cases where answering questions requires context from different sources, questions that require steps to answer or, questions that require more advanced reading comprehension.

In this work, we focus on the StrategyQA dataset, a QA benchmark that was introduced by Geva et al[1]. The questions in StrategyQA require multi-hop reasoning. The reason that this dataset's questions are considered complex is that the required reasoning steps are implicit in the questions, while in many other multi-hop reasoning datasets, the required steps for answering the questions are mentioned in the questions explicitly.

[1]Our code can be found at https://github.com/omerlevi2/NLP_Project

In Geva et al[1], the authors suggest an approach that requires performing question decomposition as a way to tackle the multi-hop reasoning challenge. Another way to answer the dataset's questions is by providing information that is relevant for answering the questions. This information is question-based, it retrieves information by using the question only and without performing any manipulation on the question. This information is retrieved from a corpus via a IR (information retrieval) model and can add valuable context to the question. The authors present the results of the baseline Roberta models and, they also present the results of models that use the baseline models (Roberta and Roberta*) trained with an addition of 'context' - paragraphs that are retrieved from the Wikipedia dump (from which the questions were composed). The retrieval is done via ElasticSearch that uses a relatively simple IR model(BM25).

Our main goal is to check whether a more advanced IR model can yield better paragraph retrieval and more importantly, if it can yield better context to the QA model. In order to check that, we trained a DPR[3] model, a state-of-the-art IR model on both Natural Questions Dataset [2] and StrategyQA. In the next step, we trained the model that was used as the base model in [1] with the question-based retrieved paragraphs that were provided by our DPR model. We received positive results, with improvements of both recall and accuracy on our test set. Our DPR model improved the dev recall by $\sim 3\%$. Our QA model improved the Roberta*_no_context model by $\sim 4\%$ and the Roberta*_IR model by $\sim 7.5\%$.

2 Dense Passage Retrieval

Open domain question answering typically consists of two stages: First, a retrieval module selects a subset of passages from a large corpus that are rel-

evant for answering the question. Then, a reader module thoroughly examines the retrieved passages to generate an answer. This work focuses on the retrieval component. We implement Dense Passage Retrieval(DPR) and show that using it leads to improvements over sparse retrieval methods, such as BM-25.

DPR is a neural retrieval method in which dense vector representations are learned for both questions and context paragraphs, in a way such that dot-product similarity becomes a good ranking function for retrieval. DPR uses two separate BERT networks for embedding passages and questions, into d-dimensional vectors. Training DPR involves training the encoders such that the dot-product between pairs of questions and supporting passages is maximized. At inference time, the question encoder encodes the input question to a dense vector and this encoding is used to retrieve k passages whose vectors are the closest to the question vector.

$$sim(q, p) = EQ(q)^T EP(p)$$

3 Training

We use question and passage encoders similar to the ones in the original DPR paper. Two BERT encoders with output dimensions d=768. We start from encoders that were pre-trained on the Natural Questions(NQ) dataset. NQ is an ODQA dataset with over 300k training examples, and provides a strong starting point for our encoders, which we later fine tune on StrategyQA. Training is done in a way that maximizes the dot product between vectors of relevant pairs of questions and passages. Just like in the original DPR paper, we use in-batch samples as negative samples with the goal of optimizing the negative log likelihood of the positive passage as the loss function:

$$\begin{aligned} & L(q_i, p_i^+, p_{i,1}^-, \cdots, p_{i,n}^-) \\ = & -\log \frac{e^{\text{sim}(q_i, p_i^+)}}{e^{\text{sim}(q_i, p_i^+)} + \sum_{j=1}^{n} e^{\text{sim}(q_i, p_{i,j}^-)}}. \end{aligned}$$

Where q_i is a question embedding vector, p_i^+ is a positive passage, p_i^- is a negative in-batch sample, and sim is the dot product of two vectors.

4 Experiments

4.1 DPR training

We start from a model that is pre-trained on Natural Question(NQ). We then proceed to train the model on NQ questions, but with context adjusted for the StrategyQA Corpus. Finally, we fine tune the encoders on StrategyQA.

Pretrained NQ Encoders We use question and passage encoders similar to the ones in the original DPR paper. Two BERT encoders with output dimensions d=768. We start from encoders that are pre-trained on the Natural Questions dataset. The question and passage encoders are *facebook/dpr-question-encoder-single-nq-base* and *facebook/dpr-ctx-encoder-single-nq-base* respectively, both taken from huggingface.

Adjusting NQ for strategyQA corpus While both NQ and StrategyQA use Wikipedia as a corpus for context passages, strategyQA uses a slightly different corpus than the one used in NQ. Specifically, context passages for strategyQA tend to be shorter. We create a new training set by taking questions from NQ and replacing the context passages with the corresponding passages in the strategyQA corpus. This is done by indexing the corpus with ElasticSearch, and then retrieving the single most similar passage from the strategyQA corpus for each NQ passage. To avoid false negatives we require that there is at least 50% overlap between the two passages. This leaves us with about 100k questions from NQ, with passages adjusted to StrategyQA. We train for 2 epochs with a batch size of 8. Using in-batch negative we have 1 positive passage and 7 negative passages per batch. We use learning rate of 1e-5 and gradient accumulation of 16 of steps.

Fine tuning on StrategyQA Finally, we fine tune on the StrategyQA dataset. The dataset contains less than 2k train questions. Due to it's small size, and limitations to the batch size we could fit, the dataset was very sensitive to overfitting. We adjust the parameters using the validation set, and use a batch size of 6 with 5 in batch-negatives per batch, gradient accumulation of 16 of steps and learning rate of 1e-5. We train for 4 epochs.

4.2 DPR Results

We measure in-batch retrieval precision of our models. In-batch retrieval measures the precision of retrieving the positive context passage out of a batch of passages, and not the whole corpus. We report the results in Table 1. Notice, each batch contains only 6-8 samples, which leads to very high precision, even when just using the pre-trained encoders. We evaluate in-batch since evaluating recall@k re-

quires reindexing of the whole passage corpus, and we could only afford to do so for our final model.

Table 2 shows recall@K for our final model. To measure recall, we index the corpus. That is, for each passage in the corpus, we create a dense embedding using our trained passage embedding. Then, for each question in the StrategyQA dataset, we fetch K passages from the corpus, using dot product similarity, and measure how many of those K passages are in the gold positive passages. We report recall for various K's. Note, that while the original corpus contains 36 million passages, due to computation limitations, we use a smaller corpus which contains the gold passages and another 1 million passages taken at random. We will also mention that while manually examining the retrieved passages, we found the retriever was able to retrieve useful passages, even when these were not the ones in the gold set.

4.3 QA model with DPR IR

Baseline models In StrategyQA[1], the authors used Roberta and fine-tuned it on DROP, 20Q and BOOLQ. The model is trained on DROP with multiple output heads, which are then replaced with a single Boolean output. This model is referred to as Roberta*. This model was used as a baseline model on which different models were trained. One of them is Roberta*_IR_Q, a question-based retrieval model that uses Roberta* as a base model with the addition of question-based retrieved paragraphs. In this work, we also use Roberta* as a baseline model.

Splitting the dataset As mentioned before, due to limited resources, we could not train our DPR model using the entire corpus of 36 millions paragraphs. Therefore, we could not use the real test set of StrategyQA. We used the validation set that was created and used by the authors as our clean test set and compared our results to the authors' results on that dataset.

Training the model:Roberta*_DPR

In order to compare our results to the two baseline models, Roberta* and Roberta*_IR_Q, we used the same code and configurations to train our model which is similar to Roberta*_IR_Q, with the change of using question-based context that is provided by our DPR model. Before training, we use our DPR model to retrieve the top ten paragraphs for every question in the dataset. Just like the authors of [1] did, we store those paragraphs in a file that serves as cache, so that during training, the model can get the relevant context to the questions directly from memory. Note that, due to a lack of resources, we did not perform a hyperparameter search. The only things that we tested using the validation set are the batch size, the number of K paragraphs that the model should use as context for each question and the number of epochs to run. We ended up using the same configurations that the authors used in [1] with one change- we found that it is optimal to provide the model three paragraphs out of the ten that were retrieved for each question. After tweaking on the validation set, we ran the last training on the full training set and received great results. As shown in Table 3 ,the accuracy on the test set outperforms both Roberta* and Roberta*_IR_Q, that achieved accuracy of 65.9% and 62.4% respectively. Our model achieves an accuracy of 69.99%. In the paper [1] the authors used a naive IR using ElasticSearch that under-performed its base model. This implies that the context that was given to the model did not help and might even caused the opposite effect. We can see from our experiments and results that our model outperformed its base model which shows that the provided context helps. It's important to mention that for each question we used only three paragraphs as added context and our recall@3 was $\sim 12\%$. This leads us to the most important conclusion- Our DPR model helps by providing relevant paragraphs and shows that an improvement can be obtained also with paragraphs that can provide good context and that are not necessarily the gold paragraphs.

5 Conclusion

We presented a DPR based QA model that utilizes both the strength of a large LM and the power of a deep learning IR model. We showed that a trained IR model that uses advanced deep learning techniques can provide better context and can also outperform the recall score of widely used TF-IDF based models that lack contextual understanding. We showed that our model outperforms the IR model that was used by the authors in [1], in both recall and precision score but moreover, our model is able to provide high quality context even in cases where the gold paragraphs from which the questions were composed is not retrieved. In fact, our results are close to the Roberta*_Oracle_P model that uses the gold paragraphs as context. Due to

Model	Dataset	Precision
(1) Bert Pretrained	Adjusted NQ	0.9693
(2) Bert Pretrained(1) trained on Adjusted NQ	Adjusted NQ	0.9725
(3) Pretrained(1) fine-tuned on StrategyQA	StrategyQA	0.9179
(4) Pretrained trained on Adjusted(2), then fine tuned on StrategyQA	StrategyQA	0.9607

Table 1: In-batch precision

K	Recall
1	0.063
2	0.101
3	0.122
5	0.161
7	0.176
10	0.204

Table 2: Recall@K

Model	Accuracy	recall@10
Roberta*_no_context	0.659	-
Roberta*_IR_Q	0.6244	0.174
Roberta*_DPR (our model)	0.699	0.204

Table 3: Accuracy and recall@10 on StrategyQA

lack of resources we had to narrow our search space of context paragraphs. We believe that by working on the original full corpus and performing a more extensive hyper-parameter search, our model will be able to provide even better context and therefore, better results. For future work, we would like to experiment training the DPR model using more robust negatives sampling methods. We believe that this is one of the main keys to improving this model. We also want to test our model with a full indexed corpus. We believe that the bigger the search space for paragraphs is, the better the context that the DPR model will provide.

References

[1] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, Jonathan Berant 2021.Did Aristotle Use a Laptop? A Question Answering Benchmark with Implicit Reasoning Strategies

[2] 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 (TACL).

[3] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering

Student Details

Daniel Glickman 311241194 Omer Levi 305215071