Inter IIT Tech Meet 11.0

Expert Answers in a Flash: Improving Domain-Specific QA

Final Evaluation Report

Team ID: 44

Contents

1 Abstract

This report describes the methods used to perform the task of question answering on the given dataset. This problem is termed Extractive Question Answering owing to the task of finding an answer span within a given context paragraph for a given question. Furthermore, we have also been tasked to retrieve the paragraphs which can be used to answer a given question. We conduct a wide variety of experiments to maximize the efficacy of the different approaches devised for the task. The retrieval task is carried out using the "approximate nearest neighbor" algorithm while the question-answering part utilizes large transformer-based architectures. We base our approaches and experiments upon several BERT-like and GPT models to explore the breadth of the problem and thus, we try to achieve commendable results while taking into account the severe computation constraints.

2 Implementation

We utilize the Transformers library [6] by HuggingFace for the implementation of our methods. For the task, we explore several methods under various categories: we begin by exploring the models pre-trained and fine-tuned on the SQuADv2 dataset then progress towards fine-tuning our own variants of BERT-style model on the provided data; following these, we delve further into design choices of additional modules for fine-tuning on the Question Answering task. Following the literature, we concatenate the context and the question together with [SEP] token in our approaches. We provide an overview of the methodology in the following sub-sections.

2.1 Synthetic data generation

We devised a simple approach to generate synthetic question-answer pairs in order to fine-tune the dataset by augmenting it with negative samples. Additionally, we used the NLTK library to augment sentences that had similar meanings to the originals in the provided training dataset. However, when we attempted to fine-tune our models on the synthetically generated pairs, there were very little to negligible gains in the validation accuracy, and additionally, we exceeded the 12-hour training time limit due to the number of samples already pushing the given computational limits with the un-augmented training dataset. There are several possible reasons for this:

1. The NLTK augmented text pairs are semantically not very different from their originals in the embedding space of our models, which means that such an augmentation will fail to be very different from the originals it is constructed from, thus we will simply end up repeating what the dataset already contains.

2. We also grouped together questions and paragraphs from different themes in order to create synthetic negative samples. However, we found that augmenting the dataset with these did not affect the accuracy much after fine-tuning. It is entirely possible that these augmentations are not very challenging for the model to classify as negative in the first place, so the augmentation does not improve the model much. There are two distinct types of negative samples:

(a) Where both paragraphs and questions belong to different themes → easy to classify

(b) Where both paragraphs and questions belong to the same theme - classifying such samples can be a challenging task for the model as we observed through the elevated confidence scores for such examples already present in the dataset. However, augmenting the dataset with such examples is a challenging task as it cannot be done automatically.

In a nutshell, we found synthetic data generation was not appropriate for improving the accuracy of our approach due to pre-training data used by our model.

2.2 Paragraph retrieval task

2.2.1 Methodology

We apply a prefilter on the theme by creating theme wise dictionary which contains: *Df* (the data frame of paragraphs related to that theme) and *Index* (the FAISS index data structure required to get the approximate Nearest Neighbor). As we only search for the paragraph related to the theme of the question we have significantly reduced the search space and this pre-filtering approach also allows to get better results and a defined number of neighbors related to that theme.
We used the FAISS library to get the approximate nearest neighbor. We tried multiple text embedding models and ran them through the same set of randomly selected 30 themes from the training data and compared the paragraph prediction score to get the best model we used in the final colab notebook.
Further, we tested the inference time of our prediction and we chose only to scan 1 paragraph as we were exceeding the time limit if we scanned 2 paragraphs with any optimization like quantization it would result in drop of accuracy by 5 % (as compared to scanning 1 paragraph without quantization) without considering the time limit penalty. We also tried to search nearby cells by assigning the n_probe value as 2 which would increase the chance of getting more accurate results.

We divided our embeddings into clusters according to the number of paragraphs in the theme. We increased number of cells as the number of points increased.

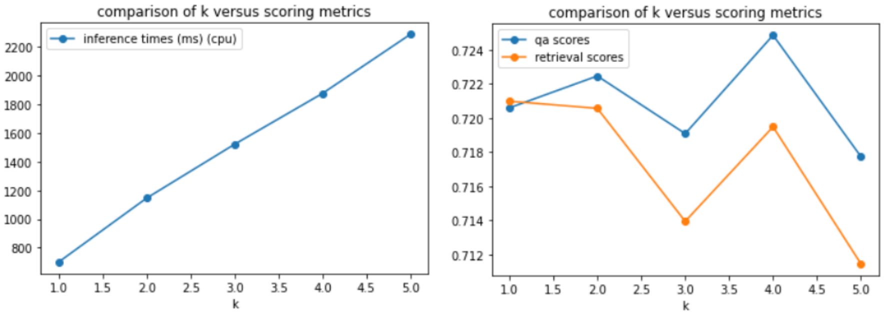


Figure 1: Experiments showing reasoning behind the choice of K in ANN = 1 (best inference time and score)

2.3 Question answering task

2.3.1 GPT-based Methodology

This approach is quite different in the sense that the other approach involves training the parameters of the transformer used. Here, instead of fine-tuning transformer models based on BERT, we take a GPT2 medium model and freeze all its layers. The role of this transformer is to convert the inputs into meaningful embeddings which will be used by two multi-layer perceptrons (MLP). The first MLP is designed to predict the probability of the existence of the answer in a given paragraph for a given question. On the other hand, the second MLP outputs the start and end indices of the answer sentence in the paragraph i.e. two numbers that indicate the position of the answer in the paragraph. The parameters of these MLPs are the ones that will be required to train. A step-by-step procedure of this method is as follows:

1. Data Preprocessing : GPT2 uses a word-piece type tokeniser. Hence, after tokenization, the answer_start values will change. An appropriate function utilizing the offset values of the data has been used to solve this problem. We also add another feature answer_end to the data.

2. Embedding Calculation: These tokenized examples are then passed to the GPT2 to get embeddings of each token. It is important to note that throughout this approach, we keep the GPT2 model as non-trainable.

3. MLP1: GPT2 is an auto-regressive model. This makes the embedding of the last token very important as it can be thought of as a representative of the whole text piece. Hence, this vital piece of information is now fed into the first MLP which outputs the probability of the existence of the answer.

4. MLP2: This feed-forward neural network possesses a bigger architecture as it needs to accomplish a comparatively harder task of determining the start and the end of the answer. So, all the embeddings of the example are fed into this network to get the respective indices.

5. Training: The loss function for the first MLP is the binary cross entropy function while the second MLP uses the general cross entropy function. These losses are then added in a specific ratio and then backpropagation takes place. That ratio is another hyperparameter to tune so as to get the best possible results. It is important to note that for those examples which do not have an answer, the answer_start and answer_end values can be any integer value of choice. Such examples have zero loss by default in the second MLP, thereby rendering these labels insignificant. Any optimizing technique with appropriate hyperparameters can be used to train these neural networks.

Although this approach provides significant gains in performance, we did not use it in our final submission as the inference was slower than the other methods. This problem can be attributed to the massive size of GPT2 models compared to BERT models.

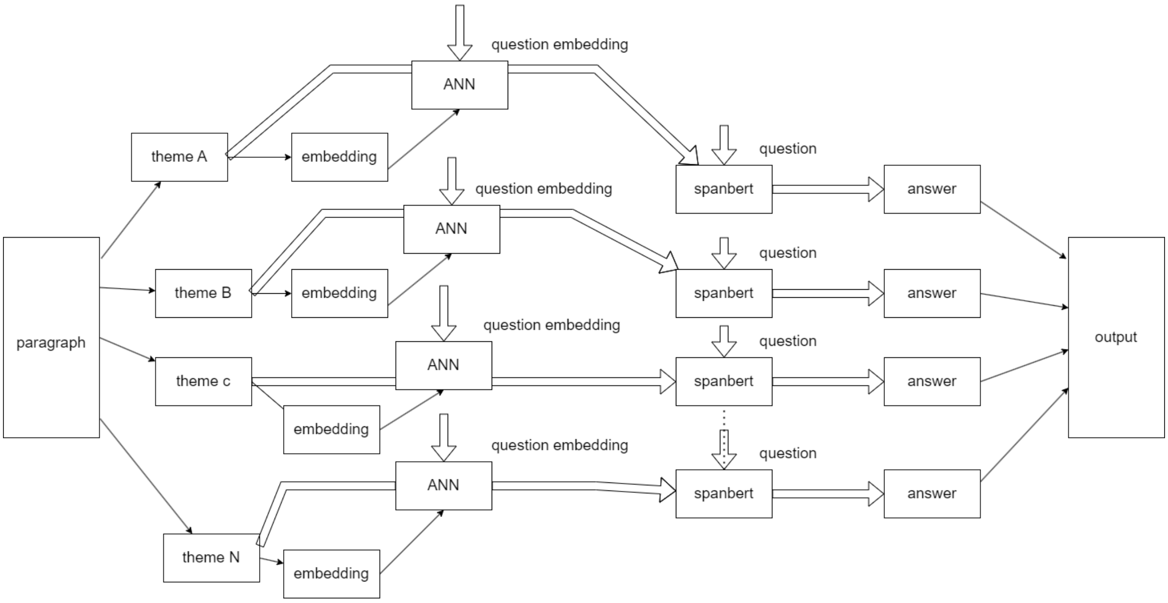
2.3.2 BERT-based Methodology

We follow a two-step approach proposed for performing the task of extractive question answering. Firstly, the text embedding model was used to obtain the embedding of the question. Then, the approximate nearest neighbor was retrieved from the theme dictionary using the FAISS library. The paragraph with the embedding closest to the question embedding was returned as the relevant context for answering the question. Next, the SpanBERT model was utilized to answer the question from the retrieved paragraph. If the confidence score was greater than 0.7, the answer was updated, otherwise, a value of -1 was predicted. This approach demonstrates the potential for utilizing text embedding and large transformer-based models to perform effective extractive question-answering.

2.3.3 Run time improvements

We implemented the SpanBERT model with quantization but it resulted in a drop of 5% in the F1 scores but not enough significant improvement in the inference time. On the other hand, at that cost, we could have a lighter model with a similar inference time.

2.4 Final implemented pipeline



3 Result & runtime analysis

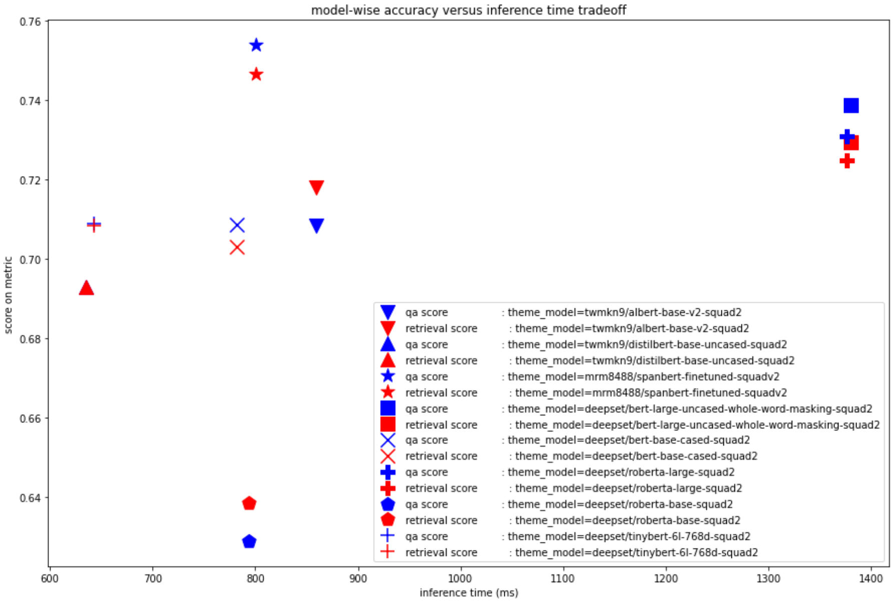


Figure 2: Scatterplot of Retrieval and QA scores against model inference time for different models.

3.1 Data

For evaluation and benchmarking purposes, we split the input data into train and validation sets, with the latter containing one-tenth of the samples given. We compare the models on F1 score and Exact Match scores, being the standard metrics of evaluation for QA, and the given metrics of choice in the problem statement.

For training and testing purposes, examples with token limits exceeding 400 were ignored. The size of the truncated dataset reduced from the original 75056 $\rightarrow$ 74385 samples (671 samples were ignored). We are currently working on techniques to address this which will be covered by us in the next report submission.

3.2 Evaluation Protocols

F1 score is defined as the harmonic mean of precision and recall over the answer span, i.e., it represents the proportion of overlap between the true answer window and the predicted one. accuracy is the number of classifications a model correctly predicts divided by the total number of predictions made.

3.3 Results

We evaluated models on the basis of the evaluation protocols shown above. The graph of the same is shown above.

4 Conclusion and Future Work

Large pre-trained language models are a great avenue to explore in resource-constrained settings where training from scratch is not possible. In this report we have explored the use of such models for the task of extractive Question Answering where the appropriate paragraph retrieval is also desired. We explored a variety of pre-trained models from encoder-only and decoder-only families finetuned on SQuADv2 dataset and the provided dataset. We also experimented with a variety of techniques, viz., synthetic data generation, model quantization and custom model finetuning to see how these aid in improved performance in a resource-constrained environment.

As for future work, the utilization of more complex synthetic data generation techniques for Question Answering and finetuning the paragraph retrieval task using contrastive learning. Furthermore, going multilingual can really help a wide variety of people, especially in the Indian context; multilingualism can be dealt with naively by back-translating context paragraphs and question to english, and the more sophisticated approaches can include multilingual models or mixture of expert models. there are several avenues that can be explored to further improve the performance of the extractive question answering task. For example, incorporating additional sources of information, such as external knowledge databases, could lead to more comprehensive answers. Additionally, fine-tuning the models on domain-specific data could increase their accuracy and relevance. Another potential direction is to extend the current approach to perform abstractive question answering, which involves generating new text to answer a question rather than selecting a span of text from a given context. These and other possibilities represent exciting opportunities for future research in the field of question answering.

5 Related Works

The most recent works that have significantly improved the performance of Question Answering systems are the model built based on the BERT [1] architecture. Thus, we have tried to exhaustively explore the performance of these SOTA models with various fine-tuning techniques on our given dataset. BERT (Bidirectional Encoder Representations from Transformers) is a deep learning model developed by researchers at Google in 2018 that is based on the Transformer architecture. BERT is designed to pre-train deep bidirectional representations of text by considering both the left and right contexts of the input data. This allows the pre-trained BERT model to be fine-tuned with a single additional output layer to achieve state-of-the-art performance on various natural language processing (NLP) tasks like Question Answering in our case.

On the other hand, RoBERTa [3] builds on BERT and modifies key hyperparameters, removing the next-sentence pretraining objective and training with much larger mini-batches and learning rates. ALBERT [2], yet another model based on BERT modifies the key design choices regarding the parameters in BERT to reduce the model size and obtain a higher performance on the Question Answering task than the previous two models. The mentioned models achieve great performance on the Question Answering task on the SQuAD [5] and SQuADv2 [4] dataset; however, simply fine-tuning these masked language models on these datasets using just linear projection of model representations is a sub-optimal approach due to the distributional shift in the training and the fine-tuning data. Several methods build upon the representations learned by these models to address this issue for improving performance on the SQuADv2 dataset, a vivid demonstration being the use of SA-Net on top of ALBERT encodings.

The BERT-family models are often augmented with the Verifier module from SG-Net [7] which adds an extra answer verifier to predict if the question is answerable and re-weighs the logits of the model based on the score of the said verifier. Following the same principles, Zhang at al. [8] propose Retro-Reader that integrates two stages of reading and verification strategies in pursuit of a better architecture design for the verifier module. The top performing models are ensembles of one or a combination of the approaches mentioned in this section, however, we limit our discussion to single-model approaches and the verifier extensions thereof, considering the constraints put on the training and inference time.

References

[1] Toutanova K. Devlin J. Chang M. Lee K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2018.

[2] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, 2019.

[3] M. Goyal N. Liu, Y. Ott. RoBERTa: A Robustly Optimized BERT Pretraining Approach. 2019.

[4] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*. Association for Computational Linguistics, 2018.

[5] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. The Association for Computational Linguistics, 2016.

[6] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics.

[7] Zhuosheng Zhang, Yuwei Wu, Junru Zhou, Sufeng Duan, Hai Zhao, and Rui Wang. Sg-net: Syntax-guided machine reading comprehension. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 2020.

[8] Zhuosheng Zhang, Junjie Yang, and Hai Zhao. Retrospective reader for machine reading comprehension. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 2021.