Assignment - 2 | NLP

Shamik Basu - 0001035358 | Pratiksha Pratiksha - 0001034021 | Pritikumari Gupta - 0001026995

MSc Artificial Intelligence, Unibo

Abstract

The text extractor and the encoder-decoder are two pre-trained transformer-based modules that we present in this paper as a model for tackling the QaA challenge. Such a model is trained by a single loss. Regarding the average SQuAD F1 score, the results are satisfactory.

1. Introduction

Answering questions that refer to sections that have the necessary information is known as question answering (QA). Additionally, the history of prior question-and-answer turns may be used to generate the response.

In our project, we employ a model made up of the Text Extractor (TE) and the Encoder-Decoder (ED) modules. We are encoding the most likely text span (answer span) and feeding it to do encoder decoder model for the generation of the answer. This strategy is used since the passage may be quite long and it will aid the encoder-decoder in discovering the interesting information. Both modules are constructed using a transformer-based pre-trained design.

The CoQA dataset, utilizing the specified training-test split and further dividing the training into training-validation, with proportions of 0.80.2, is the dataset being considered. Questions that cannot be answered are removed.

The pre-trained transformer-based designs DistilRoBERTa and BERTTiny were both employed. Three distinct random seeds have been examined for both models. There have been a total of 12 distinct studies conducted because the conversational history could be taken into account or not.

2. System Description

The Text Extractor and the Encoder-Decoder are the two modules that make up our model (Fig. 1).

The text extractor is a transformer-based encoder, with a linear layer and Softmax on top. Basically, the linear layer calculates the scalar significance scores out of the input text's contextual embeddings, which the encoder computes.

The Encoder-Decoder is a modified pre-trained transformer-based encoder-decoder that can now accept the text's span as the encoder's second input. By passing the importances through a linear layer and adding the output to the input of each encoder block, the spans are introduced into the model. For each block of the encoder, we have opted to utilize a separate linear layer.

Both modules are constructed using a pretrained transformer-based architecture from Hugging Face (Wolf et al., 2020), either DistilRoBERTa or BERTTiny. The Hugging Face encoder-decoder has been changed to accept input that includes the text's span and use them in each encoder block in order to implement the ED. The code for DistilRoBERTa (HuggingFace, 2020b) and BERTTiny (HuggingFace, 2020a) has been extracted from the Hugging Face and modified to suit our needs.

The used loss function consists of a single loss.

The first setback concerns the TE's virtue. It calculates the discrepancy between the passage's actual length and the answer span given to it by the extractor.

The ED's calculates the discrepancy between the generated and actual response. This is calculated using the cross-entropy loss, which is the same common loss function used by encoders and decoders.

It's important to note that the gradient of the loss also passes through the Text extractor,instructing it to simply identifying the span.For validation and test, the network employ the predicted span,

According to intuition, we start by predicting the answer span, at first the predicted span will contain <unk> but gradually during training it will start to predict the correct span and feed it into the encoder decoder. The encoder decoder will just generate the final answer.

3. Experimental Setup and Results

The Adam optimizer has been utilized for the training process, with learning rates of 1e-03, batch sizes of 1, and 3 epochs. The utilization of the conversational history has also been examined, and three distinct random seeds have been set. Using the average SQuAD F1 score across all instances, 12 separate experiments were run and evaluated on the validation and test sets. Table 1 summarizes the findings.

The setup with seed 42 and without using the QaA history has the highest validation score for DistilRoBERTa, also the configuration with seed 42 and without using the QaA history has the highest validation score for BERTTiny.

Even under the worst scenarios, the Berttiny model is highly effective.

	DistilRoBERTa				BERTTiny			
	Validation SQuAD F1		Test SQuAD F1		.Validation SQuAD F1		Test SQuAD F1	
	History	No history	History	No History	History	No history	History	No history
Seed 42	0.72323	0.73256	0.72222	0.73228	0.73787	0.74345	0.72323	0.74672
Seed 2022	0.72522	0.73166	0.72526	0.73159	0.72052	0.74425	0.72436	0.74263
Seed 1337	0.72467	0.73128	0.72310	0.73224	0.71467	0.74268	0.72635	0.74102

Table 1: Squad F1 Scores for DistilRoberta and Berttiny

The results from the source cnn are less than ideal, although those from the other sources are not as problematic. It is observed that the model is failed to predict the correct answer if the ground truth answer span is too large. The model performed very well if the ground truth answer is just one or two words long.

When compared to BERTTiny, the model DistilRoBERTa performs slightly poor.The Text extractor appears to be fairly inconsistent and untrustworthy for worst answers. Even when taken devoid of context, the majority of the provided replies have little connection to the inquiries.

Finding the necessary information in the passage may be the most challenging task and the performance-limiting factor, as demonstrated by the fact that both models during training showed a slight increase of training loss during the third epoch. Most of the faults in both models come from questions that has the ground truth answer too long.

4. Discussion

The best DistilRoBERTa configuration according to the validation score is the one with seed

42 and without using the QaA history, while the best BERTTiny configuration is the one with the seed 42 and not using the QaA history. The model Berttiny is particularly good even in the worst cases. The less satisfactory results refer to the source cnn, while the ones of the other sources are less serious. Some errors are just an interpretation which is correct, although different from the expected one. The majority of the errors are that the models are not able to predict for long ground truth answer span.. The model DistilRoberta is particularly bad with respect to Berttiny. The Text Extractor seems to be pretty inconsistent and unreliable for what concerns the worst errors. Most of the given answers are completely unrelated to the questions even out of context . During training, both the models showed a slight increase of the training loss during the third epoch when the teacher forcing supervision is being completely removed, suggesting that finding the relevant information in the passage is the hardest task and the limiting factor for the performance. For both models, most of the errors are obtained from questions that has long ground truth answer span.

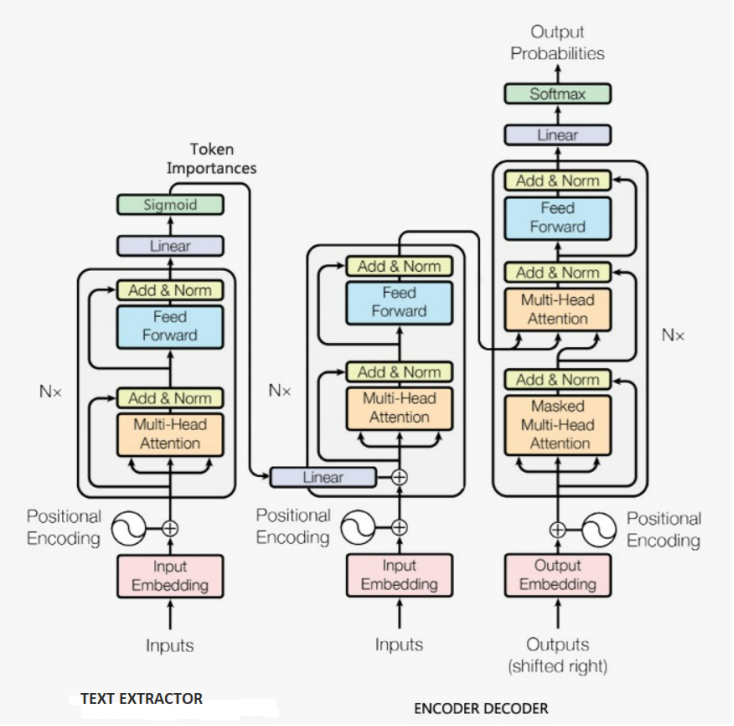
5. Conclusion

Overall, based on the SQuAD F1 scores, the Berttiny model produces pretty respectable outcomes. Even taking into account the "worst" mistakes, the model yields logically sound conclusions. A few of the generated replies from the DistilRoberta model, on the other hand, are completely out of context with the questions. Increasing the training epochs could be a deciding element in order to get better results.

References

[1] https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/packages/corpora/dependency_treebank.zip

6. Appendix



(Fig.1 The diagram of the model)