Optimizing and Improving Question Answering(QA) System Performance using Language Heuristics and Knowledge Distillation

Ayush Bisht
abisht@ncsu.edu

Prasanth Yadla
pyadla2@ncsu.edu

Abstract

Question Answering(QA) systems have emerged as powerful platforms for automatically answering questions asked by humans in natural language using either a pre-structured database or a collection of natural language documents. It comes under the intersection of Natural Language Processing and Information Retrieval.

In this project, we intend to build upon the existing BERT implementation of QA system. We use Knowledge Distillation, a regularization technique to compress learned representation DL models. We also incorporate Data Augmentation techniques to test our model against a varied dataset. Lastly, we perform Post Processing of the inferences with Linguistic Knowledge to make the predictions more reliable and false-positive free.

1 Introduction

There has been significant progress in QA systems. For example, IBM's question answering system, Watson, defeated the two greatest Jeopardy champions, Brad Rutter and Ken Jennings, by a significant margin. QA systems can also be used in chatbots for answering questions related to a specific domain/-topic.

Currently, most QA systems undergo a pre training phase with a corpus or document containing the necessary information. This will be considered as the dialog context. They will then predict answers to posed questions based on what they have been trained on. For example, a model can be trained on the Apollo Space Program paragraph and then be asked questions like "What space station was the manned mission in 1973?".

However, it will take longer to train large paragraphs and corpus, especially if this is happening real-time in chatbots and the user is waiting. It will be immensely helpful if the training time of large context documents can be reduced so that the system can respond immediately. Our general approach is to apply language text augmentation, Post-processing of linguistic knowledge along with optimization and dimensionality reduction techniques in the traditional QA system to improve accuracy and performance.

2 Background and Motivation

We are surrounded by massive amounts of information in full-text documents i.e. web. Usually, we are interested in knowing the answer to our question rather than looking at the document [1]. QA systems are useful in retrieving useful information from the web and providing insights. The QA process can be broken into two parts:

- Information Retrieval: Finding the document containing the answer to the question
- Reading Comprehension: Given the document find the answer to the question

Here, we are concerned with the reading comprehension part of the Question-Answering System.

We primarily rely on the Stanford Question Answering Dataset (SQUAD). SQUAD is a reading comprehension dataset consisting of 10,000+ questions posed by crowd workers on a set of Wikipedia articles, where the answer to each question is a segment of text from the corresponding reading passage.

3 Related Work

For this project, we use BERT as our baseline model. BERT is the state-of-art model for SQuAD 2.0, which is designed to pre-train deep bidirectional representations by jointly conditioning on both left and right context in all layers of Transformers[3]. This model achieves excellent performance on various NLP tasks with two noteworthy pre-training tasks, Masked Language Modeling (MLM) and Next Sentence Prediction (NSP).

Recently, Google AI Language pushed their model into a new level on SQuAD 2.0 with N-gram masking and synthetic self-training. Compared to BERT's single word masking, N-gram masking training enhanced its ability to handle more complicated problems. However, pre-training tasks is usually extremely expensive and time-consuming. For us, data augmentation, as an effective technique to improve the model performance in many NLP tasks [4], can also help break the limitation of dataset size and achieve better performance.

Wei and Zou[4] proposed Easy Data Agumentation technique (EDA) with four simple operations to enhance the training dataset for text classification. In particular, they perform synonym replacement, random insertion, random swap, and random deletion. However, no one has tried this method for a more challenging task: question answering. Inspired by their method, we designed a similar technique for the SQuAD v2.0 dataset. In the QANet paper[6], researchers augmented the data by adding back-translated data and tuned the augmented data ratio to achieve great performance. We decided to use the data augmentation technique[4] to improve the SQuAD 2.0 training set.

Besides, researchers found linguistic patterns and knowledge base can benefit question answering [2]. Their findings inspired us to design a series of linguistic rules to help our model better predict answers.

There also has been work on Training BERT Models with Knowledge Distillation and Augmentation [5]. Our work is largely derived from the same.

4 Proposed Approach

Our first step in the design of the QA system pipeline is to pre-process the data. Our novelty also lies majorly in this step. We use the Stanford Question Answering Dataset (SQUAD 2.0) readily available. An example of this is the following :-

Context :

Beyoncé Giselle Knowles-Carter (/bijnse/ bee-YON-say) (born September 4, 1981) is an American singer, songwriter, record producer and actress. Born and raised in Houston, Texas, she performed in various singing and dancing competitions as a child, and rose to fame in the late 1990s as lead singer of RB girl-group Destiny's Child. Managed by her father, Mathew Knowles, the group became one of the world's best-selling girl groups of all time. Their hiatus saw the release of Beyoncé's debut album, Dangerously in Love (2003), which established her as a solo artist worldwide, earned five Grammy Awards and featured the Billboard Hot 100 number-one singles "Crazy in Love" and "Baby Boy".

Question : When did Beyonce start becoming popular?

Text Span : In the late 1990s

An example data augmentation change would be to **replace "lead" with "leading"**. This replacement will not change the answer span from the context. Similarly, replacing the words with corresponding synonyms should not change the context and would lead to a more richer training dataset. We hope that this would increase the diversity of the existing training dataset and prevent our model from overfitting or output a wrong text span as a prediction.

With this new vocabulary, we generate the embedding for BERT using Glove, Sentence embedding and positional embedding.

Our automated pipeline will fully utilize the BERT model. We finetune the original model by experimenting with a variety of parameters, including learning rate, dropout, batch size and training epochs. Finally throughout our experimentation, we develop a variety of "squad features" utilizing various NLP.

BERT stands for Bidirectional Encoder Representations from Transformers. BERT relies on several layers of Transformer blocks

The exhaustive list hyperparameters we intend to

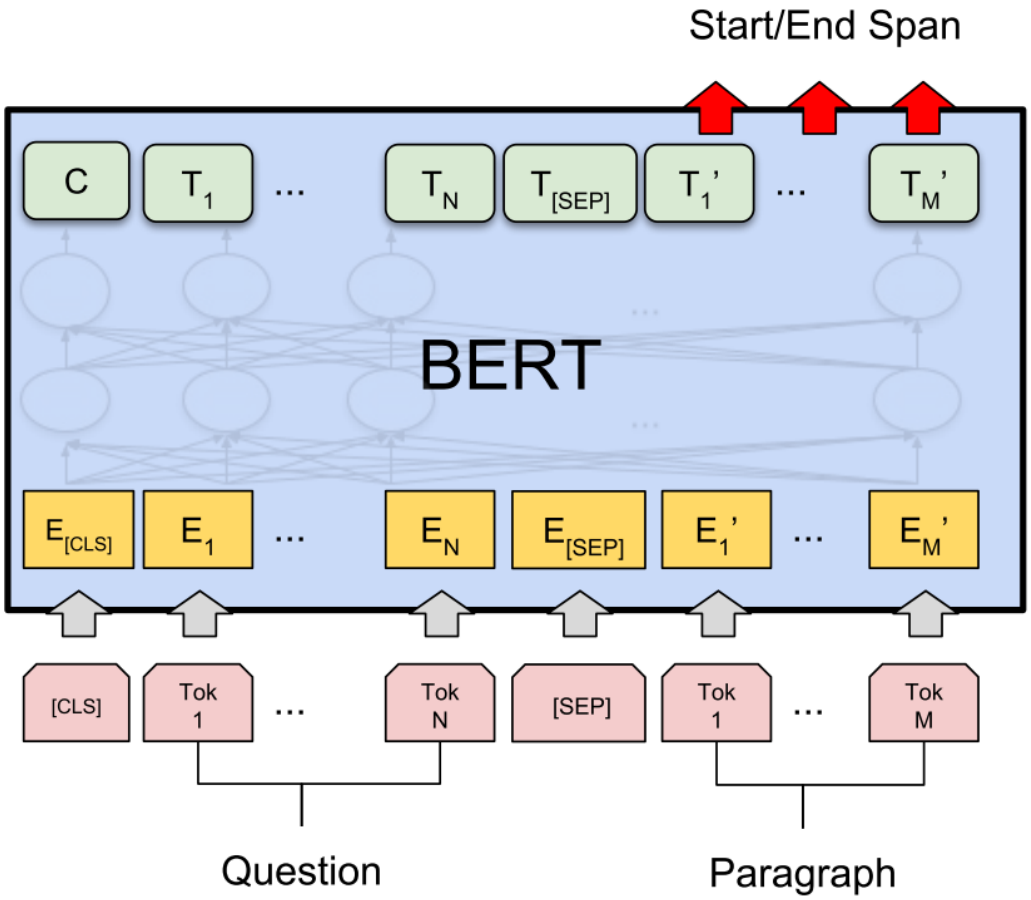

Figure 1: Architecture

tune and use in our BERT model are batch size, max sequence length, learning rate, number of train epochs, base/large pre-trained model, uncased/cased model. We will try to vary each hyperparameter while keeping the others constant and record the evaluation metrics (F1- score and EM score) for each combination and infer out a table for the same. *However, due to computational capability limitations, we restrict our work to using pre-trained models only. We do not perform training and hyper-parameter tuning.*

Post processing using linguistic knowledge: During prediction time, the probability for a text span from word i to word j being the answer is :

$P(i, j) = softmax(startlogit(i) + endlogit(j))$
where, i $\leq$ j

If we were to incorporate linguistic knowledge to the predictions, an example definition would be the following: For "When" questions, if Text(i) belongs to ['before', 'after', 'about', 'during' ., etc] which are common prepositional words used when answering "when" questions.

5 Experimental Design

There are two Question answering (QA) models being used in this project:- Fine-tuned BERT and fine-tuned DistilBERT. Both models are trained on SQUAD2.0 dataset. We have used the dev dataset to perform inference the two fine tuned models and compare their performances. We have also used the augmented dev set along with linguistic post-processing on each model and reported observations on the same.

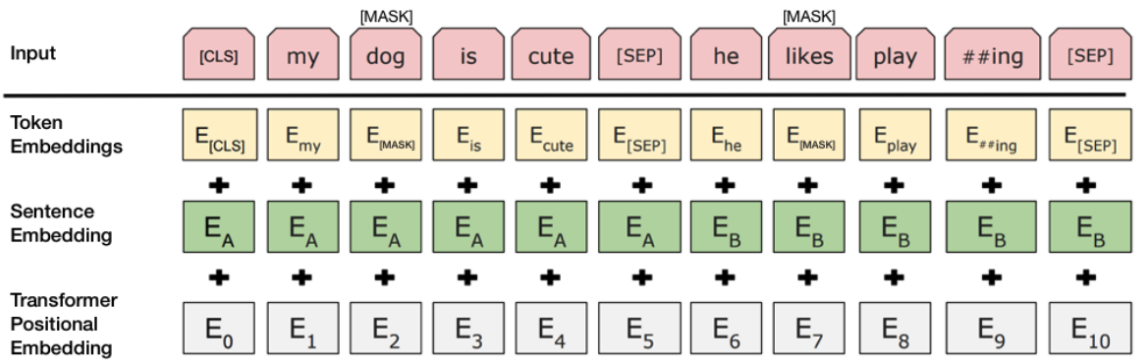

Figure 2: Word Embedding in BERT

5.1 Dataset Collection

We plan to apply our model to Stanford Question Answering Dataset (SQUAD v2.0). The input to the model is a question with a context paragraph, and the output should be the span of the text in the paragraph that can answer the question. Some features of SQUAD include the following:-

- It is a closed dataset meaning the answer to the question is a part of the context and is a continuous span. So the task can be simplified as finding the start and end index of the answer in the context.
- The distribution of the question length is centered(median) around 10 words, and the distribution of the context length is centered around 110 words

5.2 Experimental Setup

1. **Programming Language**: Python 3.8
2. **Cloud Platform**: Google Collab with GCP, TPU (Jupyter notebook)
3. **Libraries Used**: Tensorflow, Pytorch, Pandas, Numpy
4. **Environment**: Python Virtual Environment,
5. **Package Manager**: Pip
6. **Version Control**: Git

5.3 Systems Design

We extract the questions, context and question ids from the SQUAD 2.0 dev set. Each question and its associated context is tokenized using the model based tokenizer before being fed to the model as inputs. The model generates the probability distribution for the start and end indexes of the answer. We extract the combinations of all the answers that can be generated using a start, end index pair. We then select the top 5 answers (the one with the highest start, end logit sum, higher the sum means higher the probability). We also add an empty string as a possible answer for the question. We select the most probable answer as our prediction and store in a json file. We also save the difference between the null prediction and the best text prediction in a null_odds file. These values are used to calculate the null threshold which helps us improve our model scores.

5.3.1 Data augmentation

Our implementation is that we take the list of contexts from SQUAD 2.0 dataset. We take each context, ignore determiners, proper nouns and punctuation. With the remaining words, with 50% probability, we pick a synonym of the word and replace the word with a randomly selected synonym. We do that for all the words over the context. In this way, we form the augmented dataset using synonymn and random replacement.

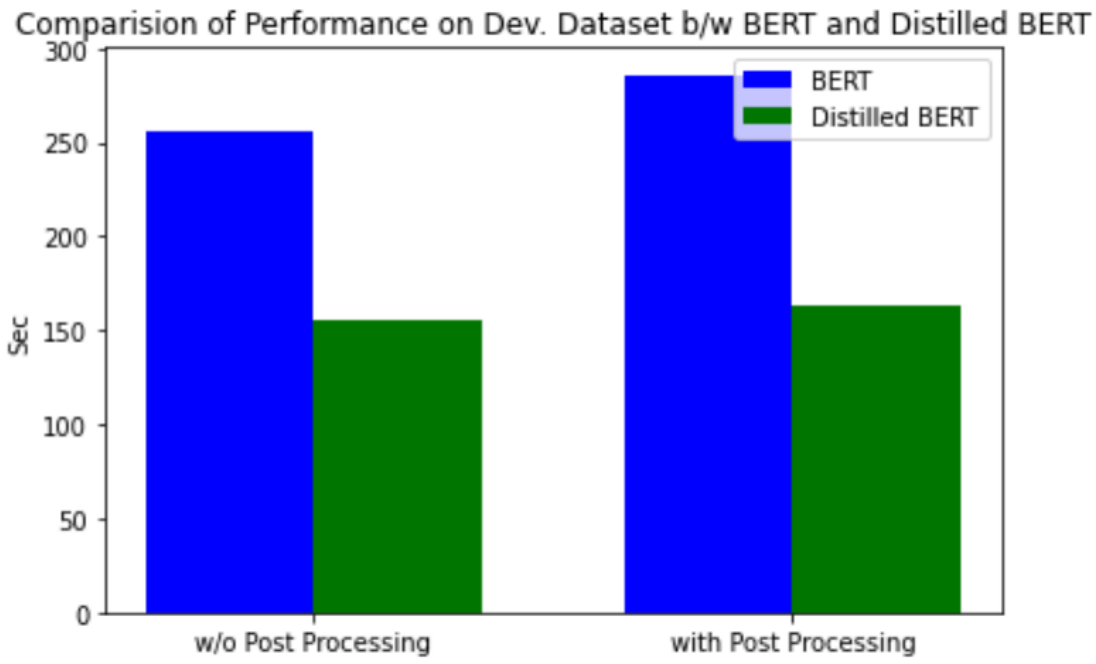

Figure 3: Performance Comparison

5.3.2 Post Processing

When considering post processing, we simply check the question type (why, when, where, other etc.), and if an answer associated with it consists of specific words, we increase the probability of that answer. This way, an answer which might have a low probability due to inaccuracy in a model, will have its probability increased and may be included in the best set of answers. Thereby, it will be chosen as the answer span of the question given the context.

6 Evaluation and Results

Following are the results obtained from each configuration of the model and dataset and postprocessing steps :-

6.1 Performance Results

The time measured is for inference over the SQUAD 2.0 dev set containing questions and contexts. The units are seconds.

- BERT without Post Processing: **256.47 sec**
- Distilled BERT without Post-Processing: **155.74 sec**
- BERT with Post Processing: **286.12 sec**
- Distilled BERT with Post Processing: **162.94 sec**

Performance Results Analysis

We find that in the case of without Post-Processing, Distilled BERT takes less time to infer, compared to the standard BERT. This is consistent with the theoretical backing that Distilled BERT is faster. In the case of Post-Processing as well, we find that Distilled BERT takes less time than the standard BERT model. In this case, Post-Processing has incurred some cost in time.

6.2 Analysis of Post Processing using Linguistic Knowledge

We can find the results of this experiment in Figure 4. When comparing the EM score with Distilled BERT and BERT, we find that the former is slightly less than the latter. This is consistent with the theory that Distillation compromises slightly on accuracy, to achieve better performance than BERT.

	Distilled BERT (dev set)	BERT (dev set)
With Post processing	EM Score :- 64.625 F1 Score :- 68.530	EM Score :- 69.434 F1 Score :- 73.823
Without Post Processing	EM Score :- 64.575 F1 Score :- 68.547	EM Score :- 69.342 F1 Score :- 73.800

Figure 4: Post Processing Results

However, with post processing, we slightly get better Exact Match (EM) score and F1 score. This validates that Linguistic Post-Processing helps filter out better results.

The total improvements on the EM and F1 scores are limited because we only performed post processing for the "when", "Where", "Whose", "Which" questions. Only about 14% of the questions in the dataset are these types as shown in Fig 6 . For the other types of questions such as "What" or "How", it is hard to think of a good linguistic rule to apply on the predicted answers since the answers to these questions can have varied forms. The figure below shows the distribution of question types in SQUAD 2.0.

6.3 Data Augmentation Analysis

In general, on Augmented Data, (results on figure 5) we find that the EM Score and F1 score (in 50s) less than the standard vanilla Dev. Data (in 60s). This shows that the model does not perform as effectively on the augmented data. This is true, because the models were trained on the vanilla data, and changing the testing dataset would change it's inference and accuracy on it as well.

7 Limitations

The limitations with respect to Implementation in General are as follows :-

1. Training of BERT with SQUAD 2.0 Dataset is inherently compute intensive. This is because each question has answer spans that vary and training on every combination would be time consuming. Therefore, we use pre-trained SQUAD 2.0 trained BERT models available online. We use case and uncased, along with distilled and native BERT.

2. The questions starting with When, where, whose and which amounts to 15 - 20% of questions, in post processing. We could include more variety of question rules linguistically.

There are problems arising from a linguistic standpoint as well :-

1. Context: : ...there were rich and well socially standing Chinese while there were less rich Mongol and Semu than there were Mongol and Semu who lived in poverty and were ill treated.

    - Question: There were many Mongols with what unexpected status?
    - Answer: lived in poverty and were ill treated vs. **Prediction**: less rich.
    - Analysis: This error is caused by model's lack of knowledge in linguistic structure. Human never makes this mistake because the predicted answer and the human answer are indeed semantically similar at the first glance. However, the head of "less" is not "rich" but "Mongols", meaning "less" and "rich" is not forming a phrase here. Thus, a human will never choose "less rich" as an answer, but BERT does not understand the underlying linguistic structure and lead it to make the mistake.
    - Fix :- There is an easy fix to this error. We can parse the sentence using dependency parse and check whether the predicted words are in the same clause. If not,

	Distilled BERT (aug. dev. data)	BERT (aug. dev. data)
With Post processing	EM :- 51.579, F1 :- 55.440	EM :- 51.983, F1 :- 57.488
Without Post Processing	EM :- 51.065, F1 :- 55.481	EM :- 51.966, F1 :- 57.508

Figure 5: Results from Augmented Data

we can directly abandon the possible answer as a candidate, and search an answer from other predicted possible answers.

2. Context: ...the Commission has a monopoly on initiating the legislative procedure, although the Council is the de facto catalyst of many legislative initiatives. The Parliament can also formally request the Commission to submit a legislative proposal but the Commission can reject such a suggestion...

   - Question: Who is the sole governing authority capable of initiating legislative proposals?
   - Answer: the Commission vs. Prediction: the Council
   - Analysis: the model is sort of weak at distinguishing two close entities in the paragraph, even though the difference between "the Commission" and "the Council" is quite obvious to human readers. Both "the Commission" and "the Council" are predicted candidates by the model. The model maybe does not understand the transition word "although", which signals a contrast relationship.
   - Fix: We can fix this by checking which candidate answer is the actor of "initiating the legislative procedure" in the context. We can obtain the original linguistic structure of the context and pick the one that can be the actor of action in the query.

8 Conclusion and Future Work

From doing a comparitive study between Distilled BERT and BERT in various environments, Data and configurations, we arrive at the following points.
BERT for QA:

- It takes more time and resources to train and infer using the BERT model.
- It is more accurate, having slightly higher EM score than Distilled BERT.
- Linguistic Post Processing helps improve the EM score by a slight margin
- Inference on Augmented Data (without Training on the same) would slightly reduce the EM score

Distilled BERT for QA:

- Since it's a lighter weight version of BERT, it computes inference faster than BERT.
- It compromises accuracy, as the EM score is slightly less than BERT.
- Linguistic Post Processing helps improve the EM score by a slight margin
- Inference on Augmented Data (without Training on the same) would slightly reduce the EM score

There is much **scope for future work**. Some of them include the following :

1. Training on Augmented Data would be an important step for future work, so that it can handle all forms of data.

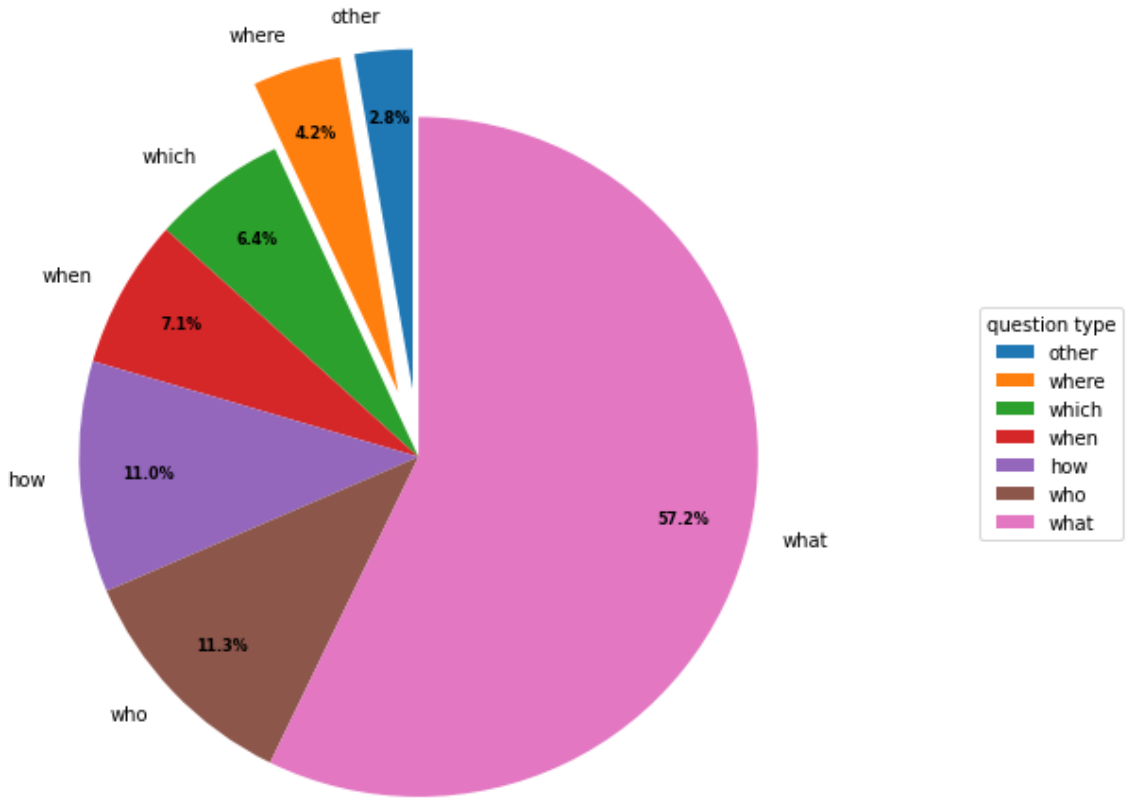

Figure 6: Question Types

2. Incorporating other types of questions in post-processing would also improve the accuracy and EM score of the model inference. Some work has already been done in this regard. We take the help of Penn TreeBank Discourses to tag the question text. Once we obtain the tags, it is more easier to identify the type of question.

9 Individual Contributions

- Ayush Bisht :- Inference and Performance/EM Score comparison on DistilBERT and BERT with performance measures.

- Prasanth Yadla :- Incorporation of Post Processing using Linguistic Knowledge and Augmentation of Data using random synonymn replacement

References

[1] @akshaynavalakha. Nlp question answering system, 2019.

[2] D. Chen, A. Fisch, J. Weston, and A. Bordes. Reading wikipedia to answer open-domain questions, 2017.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[4] J. Wei and K. Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks, 2019.

[5] H. J. Wen Zhou, Xianzhe Zhang. Ensemble bert with data augmentation and linguistic knowledge on squad 2.0, 2019.

[6] A. W. Yu, D. Dohan, M.-T. Luong, R. Zhao, K. Chen, M. Norouzi, and Q. V. Le. Qanet: Combining local convolution with global self-attention for reading comprehension, 2018.