Reading Comprehension Question-Answering Models Using SQuAD2.0

Chenxi Liu, Nuo Tian, Mary Yu, Ziyang Zhang

Dec 15th, 2020

Georgetown University

Abstract

In this project, we learn about the theory and applications of some modern deep learning models in the field of natural language processing (NLP). We finetune BERT (Devlin et al., 2019)
, ALBERT (Lan et al., 2020), ELECTRA (Clark et al., 2020) and GPT2 (Radford et al., 2019) to solve reading comprehension and question answering problems. We compare the difference of these models and evaluate their performance using the metric of F1 and EM on the **SQuAD2.0 (**Rajpurkar et al., 2018**)**.

1 Introduction & Purpose Statement

Recurrent neural networks and its variants, like gated recurrent unit (Cho et al., 2014), long-short term memory (Hochreiter, 1997), had been the golden-standard deep learning methods for natural language modeling for a long time. The rise of attention mechanisms in natural language application significantly enhance the performance of these recurrent models (Luong et al., 2015). However, the inherently sequential nature of these methods limits parallelization within training examples, which becomes critical at longer sequence lengths (Vaswani et al., 2017).

In recent years, deep learning methods in the field of NLP have been undergoing a revolution. Transformer, a model architecture that completely gets rid of recurrence and instead relies entirely on an attention mechanism to draw global dependencies between input and output, is a game-changer in the deep NLP world. It uses self-attention which relates different positions of a single sequence in order to compute a representation of the sequence. The Transformer allows for significantly more parallelization, which makes the training of massive models with hundreds of millions free parameters a reasonable task (Vaswani et al., 2017). Based on the idea of Transformer, many architectures, like BERT and GPT, emerge and become the new state of the art in various NLP tasks.

In this project we study machine reading comprehension and question answering problems in NLP. We finetune several modern language models including BERT, ALBERT, ELECTRA and GPT-2, which are developed based on Transformer instead of recurrent models. The chosen models are expected to have the ability to read contextual articles and then answer questions regarding it.

2 Related Work

2.1 BERT

BERT, also known as Bidirectional Encoder Representations from Transformers, is a well-known state of the art language model for natural language processing and it is a trained Transformer Encoder stack.

BERT uses a masked language model (MLM), which masks random tokens from the input. The

ultimate goal is to guess the id of the masked word's original vocabulary based on its context. This approach is different from some previous works because MLM offers the option of using both the left and the right context in order to perform guessing, which enables the training of a deep bidirectional Transformer. Apart from MLM, another task for BERT was to use Next Sentence Prediction (NSP) to understand the relationship between two sentences. This is very important for tasks such as question-answering. A pre-trained binarized next sentence prediction task was used to accomplish this goal.

BERT uses the WordPiece embeddings that comes with a 30,000-token vocabulary for its input representation. The sentences are separated with a special token ([SEP]) and packed into a sequence that starts with a special classification token ([CLS]). Later on, a learned embedding is applied to indicate the sentence that each token belongs. Finally, input representations are calculated for each token by adding up the token, segment, and the token's position embedding.

The implementation of BERT is separated into two sections: pre-training and fine-tuning. The same architectures are used in both procedures except for the output layer, and they all begin with the same pre-trained model parameters. All of these parameters are then fine-tuned according to the specific task for which the model will be used, in our case, question-answering through context. (Devlin et al., 2019)

2.2 ALBERT

ALBERT stands for A Lite BERT, which uses the similar backbone architecture as BERT. However, ALBERT uses two parameter-reduction techniques to increase the training speed of BERT and lower the memory consumption. The first technique is factorized embedding parameterization, which is splitting the embedding parameters into two smaller matrices while projecting the one-hot vectors to the hidden space instead of direct projection. The second technique is cross-layer parameter sharing. ALBERT shares all parameters across layers to improve parameter efficiency. Apart from these two techniques, ALBERT also uses a sentence-order prediction (SOP) instead of NSP in BERT in order to achieve inner-sentence modeling. SOP focuses more on modeling the coherence of the inner sentences instead of topic prediction. It was shown that SOP is able to solve some NSP tasks, however NSP cannot solve SOP tasks at all. Therefore, ALBERT models tend to improve and perform better for tasks that involve multi-sentence encoding. (Lan et al., 2020)

2.3 ELECTRA

ELECTRA stands for "Efficiently Learning an Encoder that Classifies Token Replacements Accurately". It is actually a change in pretraining approach compared to BERT. There are basically no changes done to the underlying BERT model. While describing the BERT model, we recognized its use of Masked Language Modeling (MLM) for pretraining, which is masking random tokens and trying to guess the id of its original vocabulary. ELECTRA, on the other hand, approaches this task differently. Instead of masking, ELECTRA uses replacements on some of the tokens. The replacements are usually sampled from the output of a small masked language model. Then, instead of trying to identify the id and identity of the masked tokens (like in MLM), ELECTRA trains the model on trying to identify whether each token in the input was replaced or not. In this case, the model learns from all of the input tokens, instead of only the masked subset. This pretraining approach reduces the computational costs, as well as the efficiency. The small MLM trained to replace input tokens is called the generator, and the model that tries to identify the replaced tokens is called the generator. (Clark et al., 2020)

2.4 GPT

GPT-2 (a successor to GPT) stands for Generative Pre-Training. It is a model built from Transformer decoder blocks and is trained on a dataset of 8 million web pages, which contains naturally occurring demonstrations of many tasks across diverse domains. Unlike BERT, which is trained to solve multiple tasks, GPT-2 is trained with a simple objective: predict the next word given the context, thus it is an unsupervised autoregression language model. This training objective makes GPT-2 able to generate conditional synthetic sentences with unprecedented quality. More surprisingly, GPT-

2 can do quite a good job in many language modeling tasks like question answering, reading comprehension, summarizing, and translation, even though it has no task-specific training. While the performance of GPT-2 on these downstream tasks is not comparable to state-of-the-art, it shows that these tasks can still somehow benefit from unsupervised learning techniques. (Radford et al., 2019)

3 Dataset

Stanford Question Answering Dataset (SQuAD) is a reading comprehension dataset, consisting of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or *span*, from the corresponding reading passage, or the question might be unanswerable. **SQuAD2.0** combines the 100,000 questions in SQuAD1.1 with over 50,000 unanswerable questions. In order to do well with SQuAD2.0, models must not only be able to deal with answering the questions when possible, but also be able to determine which questions have no answer supported by the paragraph/context provided.

4 Method

4.1 BERT, ALBERT, ELECTRA

Since BERT, ALBERT, ELECTRA models are all based on the Transformer framework, all of it have been trained with the goal of question answering (part of the goals). The training example will look like this:

(<|startoftext|> [CONTEXT]: Some context paragraphs. <|endoftext|>, <|startoftext|> [QUESTION]: Some question about the context paragraph. <|endoftext|>,[ANSWER]: start-index and end-index of the context)

In order to train based on the format above, we first need to convert the answer (text) into two indexes based on the length of the text and the start-indexes (provided). We also allow the text indexes to be off by 1 to 2 characters which hopefully improve our model. Those models could learn and adjust themselves based on the relationship between answer indexes, question and context.

For testing, we provide entry like this:

(<|startoftext|> [CONTEXT]: Some context paragraphs. <|endoftext|>, <|startoftext|> [QUESTION]: Some question about the context paragraph. <|endoftext|>

4.2 GPT

GPT-2 is trained using unsupervised learning, thus the training set is just a long corpus without any labels. Because GPT-2 is not trained for the goal of question answering, we need to preprocess the training data to fit the required format for the task. Reading comprehension and question answering tasks involve the following parts: context, question, answer. We use special tokens to indicate what the following content after an input text should be. The format of an training (fine tuning) example looks like:

<|startoftext|>
[CONTEXT]: ***Some context paragraphs.***
[QUESTION]: ***Some question about the context paragraph.***
[ANSWER]: ***Some answer to the question.***
<|endoftext|>

All training examples should be preprocessed in the above format and are concatenated to be a long corpus for fine tuning use. The GPT-2 model can learn the pattern and try to understand the relationship between these indication tokens (**[CONTEXT], [QUESTION], [ANSWER]**) and the text following them.
At prediction time, the context and the question will be provided as the warm-up text prefix to the model, with the format of:

<|startoftext|>
[CONTEXT]: ***Some context paragraphs.***
[QUESTION]: ***Some question about the context paragraph.***
[ANSWER]:

GPT-2 has a strong capacity to fit different kinds of long-term patterns, even undesirable ones. In the raw dataset, those unanswerable questions are always after answerable ones. This pattern is just the order the dataset is arranged by and has nothing to do with the QA task. However, this undesired pattern will be learned by the GPT-2 model, leading to overfitting. So, we randomly shuffle the questions to mitigate this problem, which leads to the rise of the F1 and EM score by 0.2.

5 Results and Discussion

To assess the quality of the models for SQuAD question answering tasks, we use two metrics: exact match (EM) and F1 score, which are computed on individual question and answer pairs. If one question has multiple correct answers, then we use the maximum score over all possible correct answers that are computed. Lastly, the overall EM and F1 scores are computed for each model by averaging over all individual question and answer pairs.

5.1 EM Score

EM represents Exact Match, which is calculated with following logic: if the characters of the model's prediction match exactly the same with the characters of the ground truth answers (provided by the dataset), the EM score is 1; otherwise, the EM score is 0. If the model does not predict with any text, or if there is any character that does not match with the ground truth answer, the score would be 0 automatically since this is an all-or-nothing metric.

5.2 F1 Score

F1 score is widely used in classification problems. It is suitable for our question and answering model because we treat precision and recall equally: computed over the individual words in the prediction against those in the ground truth answers. Precision is the ratio of the number of shared worlds to the total number of words in the prediction, and recall is the ratio of the number of shared words to the total number of words in the ground truth answers.

$$F_1 = \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

5.3 Model Performance

One example for demonstration purpose:

For this example, the question is 'Which type of climate may have allowed the rainforest to spread across the continent?' and the context about this question is 'Following the Cretaceous–Paleogene extinction event, the extinction of the dinosaurs and the wetter climate may have allowed the tropical rainforest to spread out across the continent. From 66–34 Mya, the rainforest …… evolution of a broad diversity of species.'

Model	Answer(before fine-tune)	F1 Score (before fine tune)	Answer(after fine-tune)	F1 Score (after fine tune)
BERT	'wetter'	1.0	'wetter'	1.0
ALBERT	'tropical'	0.154	'wetter'	1.0
ELECTRA	''	0	'Wetter climate'	0.667 (~1.0)
GPT2	'After the extinction'	0	'The wetter climate may have allowed for the spread of the rainforest to spread out across the continent.'	0.846

SQuAD2.0 validation set:

Model	EM before fine-tune	F1 before fine-tune	EM after fine-tune	F1 after fine-tune
BERT	0.366	0.407	0.230	0.258
ALBERT	0.656	0.683	0.379	0.420
ELECTRA	0.312	0.325	0.257	0.327
GPT	0.012	0.06	0.508	0.577

From the example provided on the top, it shows that all the models perform better after the fine tune for the questions that have a ground truth answer (in other words, answerable questions). However, the BERT, ALBERT, and ELECTRA after fine-tune have problems when working with non-answerable questions, and drag down the model performance. The size of the models is also a problem during training. Since these models are pretty large, it's hard to train and finetune them using the computation power we have at hand. We had to use a lower epoch in order to finish the finetuning process in a reasonable amount of time, and it may be another reason that causes the EM and F-1 scores to be lower. In the future, we would like to dive in tuning the models to make them feasible with questions that are not answerable. Based on the working theory behind these three

models, it will be interesting to see whether longer training times will improve its performance on identifying the unanswerable questions, or maybe it's a better choice to add a 'score' to the output values that helps to identify the feasibility of the answers.

For the GPT-2 model, the EM and F1 score before fine tuning is really low. This is because the GPT-2 model is not specifically trained to solve question answerings tasks. Interestingly, however, after fine tuning, the GPT-2 model has a significant improvement. This shows that GPT-2 has the ability to adapt to new tasks very quickly. Trained on text corpus from a large number of websites with naturally occurring demonstrations of many tasks across diverse domains, this unsupervised learning model can solve different NLP tasks with very quick task-specific fine tuning. It seems that GPT-2 has the ability to "understand" natural language. For those unanswerable questions, the answer is set to be "unanswerable". We find this is helpful for the GPT-2 model to better handle unanswerable questions than use empty string as the answer. This may be because the word "unanswerable" contains semantic meaning and more information about the question than "". For further study, it is interesting and exciting to explore whether the performance of these GPT models can scale up simply by using bigger models and more training data and how big the model should be for them to have human-level language understanding ability.

References

Clark, Kevin, et al. "Electra: Pre-training text encoders as discriminators rather than generators." arXiv preprint arXiv:2003.10555 (2020).

Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

Lan, Zhenzhong, et al. "Albert: A lite bert for self-supervised learning of language representations." arXiv preprint arXiv:1909.11942 (2019).

Radford, Alec, et al. "Language Models are Unsupervised Multitask Learners" (2019)

Rajpurkar, Pranav, et al. "Know What You Don't Know: Unanswerable Questions for SQuAD" ACL 2018

Cho, Kyunghyun, et al. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation" EMNLP 2014.
Sepp Hochreiter; Jürgen Schmidhuber. "Long short-term memory" 1997

Luong, Minh-Thang, et al. "Effective Approaches to Attention-based Neural Machine Translation " EMNLP 2015

Vaswani, Ashish et al. "Attention Is All You Need" NIPS 2017

.