Analyzing BERT with Pre-train on SQuAD 2.0

Chenchen Pan
Department of Management Science
Engineering
Stanford University
cpan2@stanford.edu

Liang Xu
Department of Applied Physics
Stanford University
liangxu@stanford.edu

Abstract

BERT achieves the state-of-the-art results in a variety of language tasks. In this project, we replicated the BERT base model, and aim to analyze the source of BERT's strength. There are three possible sources: (1) the pretraining on a large corpus (Wikipedia and BookCorpus); (2) the Transformer architecture; (3) the pretraining on SQuAD contexts, since they are collected from Wikipedia, thus included in the large corpus. In this work, we compared BERT in three different settings: (a) no pretraining; (b) pretraining on a large corpus only; (c) pretraining on a large corpus then SQuAD contexts. From the experiments, we confirm that (2) and (3) are not enough for good performance on SQuAD and (1) is the key source of BERT's strength.

1 Introduction

Machine reading comprehension is a task for a machine to discern the meaning of human language based on the human-performed context. It has gained significant research interests in recent years because of its applications to machine translation, text generation and categorization, question answering and speech recognition, etc. In this paper, we focus the question answering task, where the machine is asked to predict answers from the given questions and context, based on the Stanford Question Answering Dataset, specifically on version 2.0 [5].

The SQuAD 1.1 dataset, provided by Stanford NLP group, consists of 100k+ crowd-sourced questions on a set of Wikipedia articles, and the answer to each question is a segment of its corresponding passage. The challenge on version 1.1 is solved by [3] using Bidirectional Encoder Representations from Transformers (BERT) with test EM and F1 scores exceeding the human performance. Based on the SQuAD 1.1, the SQuAD 2.0 contains 50k new as well as unanswerable questions which are similar to answerable ones. The objective is not only to predict the answer of each question correctly but also to distinguish those questions without appropriate answers, namely unanswerable questions, from others.

2 Related Work

In the early stage of the SQuAD challenge, the researchers has made significant progress on leveraging machine performance to human performance using recurrent neural networks and attention mechanisms. Some important innovations include using Bi-Directional Attention Flow network which obtains a query-aware context representation without early summarization[6], using R-net which cooperates the self-attention mechanism[2], and using QANet which combines self-attention and convolution by adapting ideas from the Transformer[8][7].

Recently, one of the most successful model types to solve the question answering task is pre-trained contextual embeddings (PCE). Unlikely to non-PCE based model which does not require to utilize pretrained weights for word embeddings, PCE uses word embeddings which is achieved by pretrained weights on a large-scale language modeling dataset. Two well-known PCE models are ELMo and BERT. ELMo trains a two-layer bidirectional LSTM for language modeling on a large-scale corpus, and uses this pretrained bi-LSTM as the embedding layer for any desired model which takes text as input[4]. BERT pre-trains the Transformer, and uses the output of the Transformer as the input of fine-tuning procedure[3]. Based on the results of SQuAD 1.1 and 2.0, PCE, typically BERT, shows the promising performance on solving QA tasks and dominates most of the top ranks on the leaderboards.[1]

3 Approach

Limited by computational resources, we used BERT base model (12 layers, 768 hidden size, and 12 self-attention heads) for the following studies. Our goal is to understand and replicate the BERT result with pretraining on SQuAD 2.0, and perform ablation study to clarify the contribution of different components. We would like to analyze the reason behind the BERT's strength. Specifically, we want to know whether the gain comes from: (1) the pretraining on a large corpus (Wikipedia and BookCorpus); (2) the Transformer architecture; (3) the pretraining on SQuAD contexts, since they are collected from Wikipedia, thus included in the large corpus.

We first replicate the BERT model with pre-training on Wiki corpus. And then, we re-run the same model without the provided pre-training contextual embeddings, i.e., we randomly initialized the weights. To Compare the results from the two models, we analyzed their predictions on the two types of questions. We found that on No-Answer questions, the model without pretraining performed better since it almost always predicts "none" for the questions. However, the model with the pretraining performed better on Has-Answer questions. Although the hyperparameters are not fully tuned for the no-pretraining setting due to computational constraint, it shows that the strength of BERT is not just from the architecture.

Our next question is whether the gain is from the pretraining on SQuAD context or from the use of the large scale corpus? Because the SQuAD contexts is collected from Wikipedia, they are included in the large scale corpus, so it is possible that the strength of BERT just comes from pretraining on those contexts. We hypothesize that if this is true, then adding more pretraining on SQuAD contexts should further improve the performance. To test this hypothesis, we first created the SQuAD corpus that extracts all the contexts from the training set. We then initialize the model with the weights pretrained on the large corpus and further pretrain it on the extracted contexts. Next, we train the model on SQuAD training set, and evaluate the performance on dev set. Figure 2 demonstrates the process. After fine-tune and hyperparameter tuning, we compared the results of BERT with extra pretraining on SQuAD with the original one and found that the performance is actually worse (-1.3%).

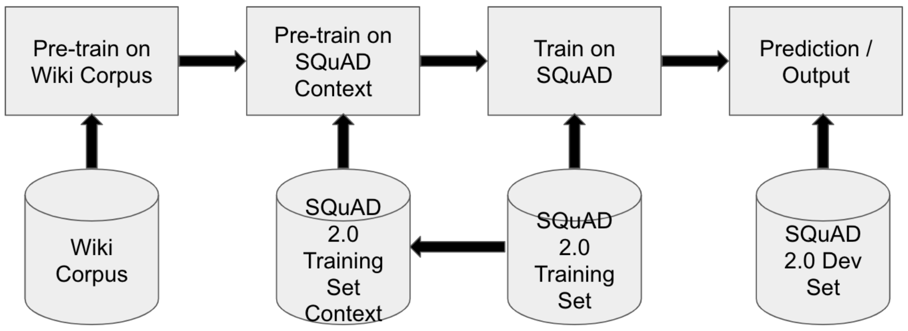


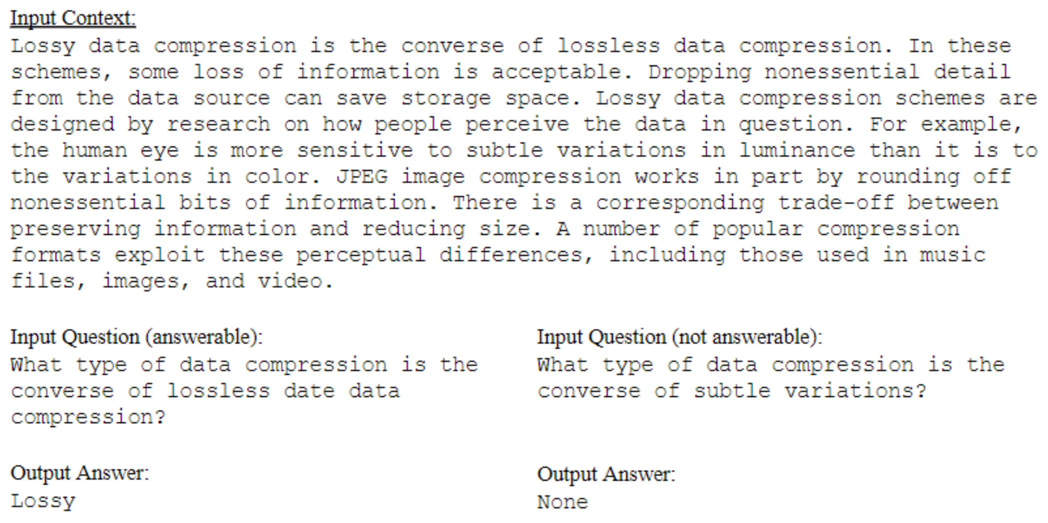
Figure 1: Process Flow

4 Experiments

4.1 Data

SQuAD 2.0 consists of 100k+ question-answer pairs with corresponding passage, and also contains 50k new, unanswerable questions.

Example:



Input Context:

Lossy data compression is the converse of lossless data compression. In these schemes, some loss of information is acceptable. Dropping nonessential detail from the data source can save storage space. Lossy data compression schemes are designed by research on how people perceive the data in question. For example, the human eye is more sensitive to subtle variations in luminance than it is to the variations in color. JPEG image compression works in part by rounding off nonessential bits of information. There is a corresponding trade-off between preserving information and reducing size. A number of popular compression formats exploit these perceptual differences, including those used in music files, images, and video.

Input Question (answerable):

What type of data compression is the converse of lossless date data compression?

Output Answer:

Lossy

Input Question (not answerable):

What type of data compression is the converse of subtle variations?

Output Answer:

None

4.2 Extra Pretraining on SQuAD Corpus

Due to the limit of computational resources, we used BERT base size in all of our experiments. First, we extracted the context paragraph from the training set and performed sentence segmentation with spaCy toolkit to create the SQuAD corpus. Then we replicated the experiments of "masked LM" and "next sentence prediction" on the new corpus. We used the existing BERT checkpoints as the initial checkpoints to start the pre-training. We also reduced the batch size and sequence length to avoid the out-of-memory issue. We tried 2 ways to do the pre-train: 1) pre-train 50k steps on SQuAD context using sequence length 384 with batch size 12; 2) pre-train 90k steps on SQuAD context using sequence length 384 with batch size 12. Table 4 shows the performance of the pre-training.

Model	Masked LM Accuracy	Next Sentence Accuracy
pretraining for 50k steps on SQuAD contexts	0.797	1.0
pretraining for 90k steps on SQuAD contexts	0.850	1.0

Table 1: Pre-train Tasks Results

More training steps on the corpus can achieve higher accuracy on pre-training tasks, however, the final predictions on SQuAD dev set show that the pre-training for more steps actually (See Table 4).

Model	EM Score	F1 Score
pretraining for 50k steps on SQuAD contexts	73.37	76.58
pretraining for 90k steps on SQuAD contexts	72.60	75.87

Table 2: Pre-train on SQuAD Performance

4.3 Results

For comparison purposes, we use the the architecture and model size for (1) the no pretraining; (2) pretraining on Wiki and BookCorpus; (3) pretraining on Wiki and BookCorpus and additional pretraining on SQuAD model. Table 4 shows the model configurations. Figure 2 shows the training loss of the three models on SQuAD.

Hyper-parameters	Values
dropout rate	0.1
vocab size	30522
learning rate	2e-5
hidden size	768
number of hidden layers	12
number of attention heads	12
number of epochs	4

Table 3: Model Configurations

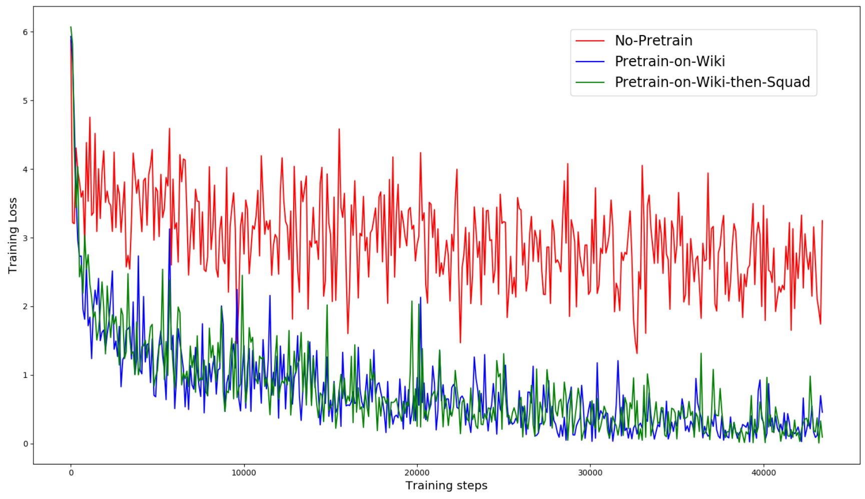


Figure 2: Comparison of training loss on SQuAD

The figure 4 shows the prediction results on SQuAD dev set from the model with different settings.

Model	EM Score	F1 Score
No Pre-train	50.10	50.10
Pre-train on Wiki	74.76	77.90
Pre-train on SQuAD 50k steps	73.37	76.58
Pre-train on SQuAD 90k steps	72.60	75.87

Table 4: Prediction Results

5 Analysis and Discussion

We analyze the number of exact match on the two different types of questions. We compared the results from all three models, (1) BERT with no pre-training; (2) BERT with pretraining on large scale corpus; (3) BERT with additional pretraining on SQuAD. It is shown in Figure 3.

Question Type	Number of questions on Dev Set	Number of Exact Match on Prediction		
		No-pretrain	Pre-train on Wiki	Pre-train on Wiki + SQuAD
Has Answer	5928	146	4270	4239
No Answer	5945	5290	4448	4277

Figure 3: Compare EM Scores of Models on Dev set

We found that the model without pre-training is the worst because it almost always predicts "No Answer" for the questions. Additionaly pretraining on SQuAD actually hurts performance on both question types. This is contradictory to the hypothesis that BERT's superior performance is because it has seen the SQuAD contexts during pretraining.

We then try to analyze the reason why additional pretraining on SQuAD is not helping. We note pretraining on the large corpus as PL, and the one with additional training on SQuAD as PS. We extracted questions that one of them answered correctly but the other got wrong. We first found that a lot PL and PS gives the same answer on 87.56% of the questions, and on 5.64% of the questions only PS got the right answer and on 6.80% of the questions, only PL got the right answer.

To understand what kind of questions are better answered by each model, we trained a classifier to differentiate the questions answered only by PS and the ones answered only by PL. We used logistic regression classifier and applied TFIDF to turn the questions into bag-of-words representations. Then we analyzed the weights of the features, which corresponds to the words. We found that the questions answered only by PS tends to have words that have higher frequency in the SQuAD contexts than those answered only by PL. Specifically, we selected the features / words with highest and lowest weights from the logistic classifier. We then compare the average frequency of the top K words with the highest weights and the top K ones with the lowest weights and plot them in Figure 4.

We can see that the words that indicate PS can answer the question correctly, i.e., the words with highest weights, have larger average frequency. This implicts that the additional pretraining helps on words that appears more frequently in SQuAD but also hurts the less frequent words.

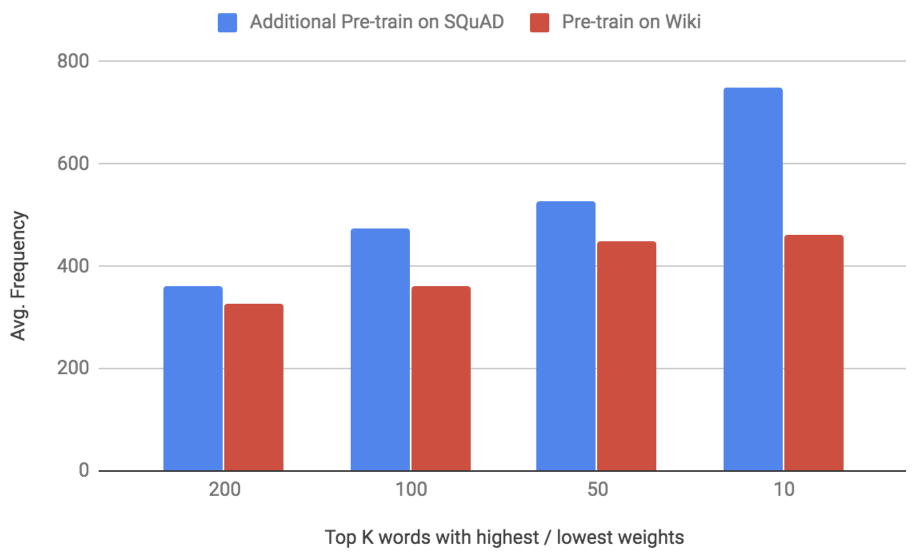


Figure 4: Analysis

6 Conclusion and Future Work

The analysis in this work showed that the pretraining on the large scale corpus is a key source of BERT's strength. And additional pretraining on the domain specific corpus, for example, SQuAD, only helps on questions whose words is frequent on the domain specific corpus.

References

[1] https://rajpurkar.github.io/squad-explorer/.

[2] https://www.microsoft.com/en-us/research/wp-content/uploads/2017/05/r-net.pdf.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[4] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *CoRR*, abs/1802.05365, 2018.

[5] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

[6] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603, 2016.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[8] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. Qanet: Combining local convolution with global self-attention for reading comprehension. *CoRR*, abs/1804.09541, 2018.