Enhancing BERT:The Effects of Additional Pretraining Using Downstream Task Relevant Datasets

Stanford CS224N Default Project

Chase Nwamu
Department of Computer Science
Stanford University
cnwamu@stanford.edu

Abstract

This paper is a direct extension to the work of "Bidirectional Encoder Representations from Transformers", or BERT. BERT is a revolutionary transformer based model that outputs context-aware word representations developed by google engineers in 2018. BERT representations have been shown to be so robust, that with just an additional layer, it can provide strong results for a number of NLP tasks. BERT representations and the underlying transformer method of self attention have achieved benchmark results on tasks including sentiment analysis, named entity recognition, and paraphrase detection. Since BERT embeddings are so effective downstream for so many tasks, they certainly holistically capture a lot of the nuances of words and language. Therefore, generally speaking, finding ways to improve upon BERT embeddings for specific downstream tasks other than just adding additional layers and fine-tuning them are non trivial. However, there is one way that is somewhat apparent. Although BERT was trained on wikipedia articles, most downstream tasks have far more specific sets of words that are applicable to them. Therefore, additionally pretraining BERT on datasets similar to those used for downstream tasks is one way that BERT embeddings might be improved. My project addresses this, and seeks to quantify exactly how much improvement can be seen from doing this.

1 Key Information to include

- Mentor: Sonia
- External Collaborators (if you have any): N/A
- Sharing project: No

2 Introduction

In this paper, we attempt to we explore the effect of running additional pretraining on existing BERT embeddings on the success of 3 downstream tasks: Sentiment Analysis, Paraphrase Detection, and Textual Similarity Detection. For each of the three tasks, we pretrain the model additionally on a dataset that is similar to the dataset that will be used for downstream task evaluation - That is we take the dataset that will be used for evaluation from the same place as the dataset that will be used for additional pretrain (e.g. Quora, Stanford Sentiment Treebank, etc).

Additional pretraining has been run on language representation models for downstream tasks with varying levels of success, but there are two reasons that additional pretraining on BERT specifically, which there is less literature on, is incredibly significant. Firstly, BERT is bidirectional, as opposed to a lot of of its unidirectional language representation predecessors. This means that any token can

attend to any other token in a given layer in the self attention architecture, so representations are learned far differently than other types of language models. Therefore, it is certainly conceivable that additional pretraining on a downstream task relevant dataset will affect results differently than other models. Secondly and likely more importantly, the original BERT was trained on the masked language model objective. The masked language model objective randomly masks tokens in the input, then the model attempts to predict the hidden token based only on its context. So additional pretraining on the MLM objective helps to develop full context aware representations, not just one way ones. In short, additional pretraining might have the power to significantly positively affect the downstream results of NLP tasks. Furthermore, this would be an important result because MLM does not require any labeled data, and just requires large bodies of text. Since there is so much text on the internet, additional pretraining on this objective is relatively tractable.

3 Related Work

3.1 Language Model Pretraining

The concept of pre-trained word embeddings has been widely recognized as foundational for building advanced NLP systems. These embeddings, which encapsulate deep semantic meanings, are crucial because they provide a rich, pre-trained context that enhances model performance without the need for training from scratch. Research has shown that both sentence and paragraph embeddings significantly boost the models' capabilities in handling various language processing tasks Le and Mikolov (2018). Moreover, the development of embeddings that capture context dynamically has marked a significant step forward, offering more depth and flexibility in how models understand and interact with text. In addition to pre-training with unsupervised data, transfer learning with a large amount of supervised data has also been shown to acheive solid results on even more 'complex' downstream tasks such as machine translation Tomas Mikolov and Dean (2013) BERT is trained on the Masked Language Model Task and the Next Sentence Prediction Task using a broad cross-domain dataset. Unlike earlier bidirectional language models (biLMs) that combine two unidirectional models (i.e., left-to-right and right-to-left), BERT introduces a Masked Language Model that predicts words which have been randomly masked or altered. BERT represents the first fine-tuning based representational model to achieve top-tier results across a variety of NLP tasks, highlighting the substantial efficacy of the fine-tuning approach. In our paper, we expand upon the pretraining techniques of BERT for the purposes of our three downstream tasks.

3.2 Multitask Learning

The field of multi-task learning Collobert and Weston (2008) has been explored to improve the efficiency and effectiveness of text classification systems. By adapting models like BERT to handle multiple tasks at once, significant strides have been made. This method not only makes the training process more efficient by reducing redundancy but also improves the ability of these models to apply their learning to a broader range of tasks effectively. In our paper, after we pretrain BERT on texts we are interested in, we turn to multitask learning as our method of fine-tuning.

4 Approach

In this paper, our overarching approach is as follows. We import preset BERT embeddings from a library, then we run multitask fine-tuning on them using the 3 tasks of sentiment analysis, paraphrase detection, and textual similarity detection. We evaluate the results of all 3 tasks on test sets, and these accuarcy scores (and textual similarity correlation score for textual similarity) act as a baseline. Then, for the experimental portion, we take the same preset BERT embeddings, and further pretrain them using 3 datasets that are similar to the test sets for each or the 3 downstream tasks. Then we fine-tune the updated BERT weights on the 3 downstream tasks. We compare the accuracy on each of the respective tasks with and without pretraining on all three corpuses.

4.1 BERT Model

BERT is an instrumental part of our project, and a basic understanding of its mechanisms is crucial. Firstly, all sentence embeddings that we fed into BERT for training were tokenized by a BERT function, then padded to be the same length. Additionally, embeddings passed into BERT were the concatenation of the token embeddings themselves, positional embeddings to include information about where in the sentence the token occurs, and separation embeddings to include information about which tokens belonged to which sentences. The BERT embedding layer has a dimensionality of 768, which is also then the dimensionality of the output embedding prior to any fine-tuning or linear layers being applied. In total it is made up of 12 Encoder Transformer Layers, which contain the following: multi-head attention, followed by an additive and normalization layer with a residual connection, a feed-forward layer, and a final additive and normalization layer with a residual connection. Although all parts of the network are important, the self attention mechanism is at the core of the transformer architecture. It transforms the hidden states for each element of a sequence based on the other elements of a sequence. The self attention mechanism used in this project is the same as that used in the original BERT paper Jacob Devlin and Toutanovan (2018)

4.2 MLM Pretraining

Both the original BERT model, from which we derive our initial embeddings, and our additional pretraining efforts utilize the Masked Language Modeling (MLM) objective. In MLM, a certain percentage of the input tokens are masked randomly and the model is trained to predict these masked tokens. Specifically, in the original BERT architecture, 15 percent of the word piece tokens in each sequence are selected for masking during training. The final hidden vectors corresponding to these masked tokens are then input into a softmax layer that predicts the original tokens from the entire vocabulary.

To ensure consistency between the pre-training phase and the subsequent fine-tuning phase, the tokens selected for masking undergo a specific substitution process rather than being directly replaced with the [MASK] token. During the generation of training examples, 15 percent of the token positions are chosen at random; of these, 80 percent are replaced by the [MASK] token, 10 percent are replaced with a random token from the vocabulary to introduce noise and help the model generalize better, and the remaining 10 percent are left unchanged. This approach helps in preventing the model from merely learning to predict the presence of [MASK] tokens and instead enables it to more effectively learn contextual relationships between tokens, enhancing its ability to understand and process language in a way that is beneficial for both the pre-training and fine-tuning stages. This method not only boosts the model's predictive accuracy during pre-training but also improves its adaptability and performance in downstream NLP tasks during fine-tuning.

4.3 Downstream tasks

The three tasks that this project evaluates, and therefore, fine-tunes on are sentiment analysis, paraphrase detection, and textual similarity detection. For fine-tuning all 3 of these tasks, we add a linear layer to the end of the model, then train proper classification for each of the 3 tasks respectively in the same loop. For sentiment analysis, our sentences were all labeled 1-5 (according to their sentiment) with the numbers corresponding to negative, somewhat negative, neutral, somewhat positive, and positive. For paraphrase detection, the input data point is two sentences, and their label y is 1 or 0 for if sentence 2 is a paraphrase of sentence 1. For textual similaritiy, input pairs of sentences (x) are labeled with a similarity score between 0 and 5 (6 labels).

5 Experiments

This section contains the following.

5.1 Data

In our project, we used 3 distinct datasets for the three downstream tasks. For sentiment analysis, we used the Stanford Sentiment Treebank dataset (SST), which contains 8,544 train examples, 1,101 dev examples, and 2,210 text examples. The examples contained in SST are all movie reviews. For paraphrase detection, we trained and evaluated on a Quora dataset, which had 283,010 train examples, 40,429 dev examples, and 80,859 test examples. Lastly, our textual similarity dataset, SemEval Benchmark Dataset, 6,040 contained train examples, 863 dev examples, and 1,725 test examples. We trained on all data in the train datasets, and evaluated on all data in the test data sets, but due to time constraints did not tune parameters using the dev sets.

5.2 Evaluation method

We will use two metrics two primary metrics for evaluation of the model. The first metric, which is to be used on sentiment analysis and paraphrase detection is just accuracy - the amount of correctly predicted y labels in evaluation over the total amount of predicted y labels (For sentiment analysis this will be correctly predicted sentiment values, and for paraphrase detection this will be correctly identified paraphrases or not paraphrases) The other metric we will use, Pearson Correlation, is for evaluating textual similarity. We take the Pearson correlation of the true similarity values against the predicted similarity values.

5.3 Experimental details

In both our baseline experiments without pretraining, and our full experiment with pretraining included, we used a learning rate of 1E-3. We used a constant batch size of 32 examples, and trained for 6 epochs in both pretraining and finetuning. We used a dropout probability of 0.3 for all layers involving it. In the multitask fine-tuning portion of the experiment, we trained batches in the following order: sentiment analysis, paraphrase detection, then textual similarity.

For our project, we ran one baseline experiment, and 4 test experiments. The baseline experiment we ran did NOT involve pretraining on the MLM objective. We simply imported the BERT weights, fine-tuned them for the three tasks over 6 epochs, then recorded the relevant metrics. The first test experiment involved pretraining on the masked language model objective (predicting words using context) using ONLY the Stanford Sentiment Treebank dataset, meaning only using the movie reviews as the corpus for training for 6 epochs. Then after the preset BERT weights had been updated, we fine tuned them for another 6 epochs on the downstream tasks. Similarly, for the second and third test experiments, we pretrained the preset BERT weights on the Quora and SemEval (parphrase and textual similarity) datasets respectively, then fine tuned them on the downstream tasks. Lastly, for the 4th and final test experiment, we pretrained using MLM on all 3 datasets (SST, Quora, SemEval), then fine-tuned and collected the results.

5.4 Results

The table below contains the results for our baseline experiment

Sentiment Accuracy	**Paraphrase Accuracy**	**Similarity PC**
0.272	0.409	0.518

Table 1: Baseline Results

Here are the results for the test portion of the experiments:

-	Sentiment Accuracy	Paraphrase Accuracy	Similarity PC
SST	0.332	0.389	0.515
Quora	0.338	0.524	0.801
SemEval	0.301	0.377	0.566
All Datasets	0.407	0.531	0.860

Table 2: Full table of results training on different datasets

6 Analysis

The broadest takeaway that can be drawn from the test data is that training on all 3 datasets increased the downstream performance on all 3 tasks. From the baseline of no pretraining to pretraining on all 3 datasets, sentiment accuracy increased by 33.1 percent, paraphrase accuracy increased by 26.7 percent, and the textual similarity Pearson Correlation increased by 39.7 percent. With these drastic improvements on just 6 epochs of pretraining, it is clear that additional pretraining of the BERT weights (that have already had a massive amount of training) with relevant datasets helps downstream performance. Further confirming this theory is the fact that from the results you can see a direct improvement in the downstream task based on the dataset it was pretrained. To see this, we must inspect the experiments where we trained on just one dataset (SST, Quora, or SemEval). Out of the 3 individual datasets, training on Quora (the paraphrase dataset) yields the greatest jump in performance for all 3 downstream tasks. However, the Quora dataset contains approximately 100 times more train examples than the average of the other datasets (about 283,000 train examples). So it is likely that the Quora dataset boosting the 3 metrics by the most is due to the sheer volume of words and word contexts seen in the Quora dataset compared to the others. For the other two datasets, which had a similar size, they both contributed to a significantly higher jump in performance on their own downstream task(from the baseline). When trained on the SST dataset alone, sentiment accuracy reached 33.2 percent, whereas when trained on the similarly sized SemEval alone, it reached just 30 percent. Conversely, when trained on the SemEval dataset alone, our similarity Pearson Correlation (for textual similarity) reached 0.566, while when it was trained on the SST dataset alone it reached just 0.515.

7 Conclusion

The conclusion we can draw from our data is that training pretraining a BERT model on a corpus that is similar to the one used in fine-tuning and evaluation **will** indeed increase its performance in the relevant downstream task. The greatest limitation of this work is that we did not normalize for the large jump in data between the SST/SemEval Datasets, and the Quora Dataset. Because of this, it is harder to draw a direct conclusion about the effect of Quora dataset on downstream performance compared to the other two. It is worth noting, however, that the amount of new things that can be learned from a corpus do not scale linearly. Additionally, excluding the Quora dataset, the numbers in the table match up with what we would expect to see, meaning that for boxes concerning the other two datasets, the corresponding task performs significantly better when trained on the correct dataset. If we could take this project further, next steps would certainly be to run the same experiments with the Quora dataset normalized to be the size of the others. Additionally, we would expand our experiment to additional downstream tasks.

8 Ethics Statement

This project and its implications pose some ethical risks. One of the largest ethical issues that arises with BERT in general, and therefore with trying to improve the word representations is bias in the corpuses it is trained on. Almost any body of text generated by a human will contain the biases that that person has. Because BERT creates such complex and high dimensional representations of words, it is almost certain that whatever negative bias exists regarding the word will appear somewhere in its representation. Secondly, as with many machine learning systems, a lack of transparency, which might lead to a lack of accountability, is a large concern. Because the multi layer transformer structure that underlies BERT is so complex, it is likely impossible to sort of step through the model and see where word representations are actually coming from. In fact, there is likely little clarity as to what specific elements within the word representations themselves mean. This is dangerous because if a word representation were ever to cause harm in anyway, it would be hard to work backwards and discover the issue. As far as solving these problems, the first problem of bias and fairness can only be attacked by trying to ensure that all corpuses that the model is trained on are as bias free as possible, which is an incredibly different task. The second sort of black box issue which exists in essentially all of deep learning is difficult to solve, and we must ensure that we always understand the extent to which we are relying on ML, and the consequences if a system outputs something incorrectly.

References

TRonan Collobert and Jason Weston. 2008. Deep neural networks with multitask learning. In *In Proceedings of the 25th international conference on Machine learning*.

Kenton Lee Jacob Devlin, Ming-Wei Chang and Kristina Toutanovan. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *In Proceedings of the 25th international conference on Machine learning*.

Quoc Le and Tomas Mikolov. 2018. Distributed representations of sentences and documents. In *International Conference on Machine Learning*.

Kai Chen Greg S Corrado Tomas Mikolov, Ilya Sutskever and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*.