Pre-training BERT: Swapped Subject Phrase Detection

Stanford CS224N Default Project

Maggie Wu
Department of Computer Science
Stanford University
`mwu9@stanford.edu`

Abstract

BERT is a transformer-based model that generates contextual word representations (Devlin et al., 2019). These representations can then be fine-tuned and used for downstream tasks such as sentiment analysis, paraphrase detection, and semantic textual similarity detection. In this project, we examine the success of a novel pre-training task aimed at teaching the model topic-level features, where we randomly (with 50% probability) swap subject phrases in two unrelated sentences and have the BERT model predict whether or not they were swapped. Through our experiments, we find that the swapped sentence phrase detection task is more beneficial for downstream task performance than additional pre-training on the masked language model objective using in-task data, and that pre-training on a combination of the two tasks leads to even higher performance and a more generalized model.

1 Key Information to include

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

2 Introduction

One of the most widely-used transformer-based language models, BERT produces contextual word representations that can be fine-tuned and adapted for various downstream tasks. The original BERT model was trained on two different pre-training objectives: a masked language model objective and a next sentence prediction task. The first objective taught the model to predict the correct word at certain masked positions in each sentence/document (or to replace an incorrect word with the correct one), whereas the second was meant to teach the model how to identify relationships between sentences. However, a subsequent work (Liu et al., 2019) showed that the NSP (next sentence prediction) task was actually not beneficial to the BERT model's performance on downstream tasks.

Many downstream tasks that BERT is used for require some kind of relationship between sentences to be identified, and without next sentence prediction, iterations on BERT such as RoBERTa (Liu et al., 2019) no longer use a pre-training task that focuses on higher-level features of text. To replace next sentence prediction, we attempt to create a task to teach the model about higher-level features that also utilizes sentences from different documents. In this work, we propose a new pre-training task, swapped subject phrase detection, in an attempt to teach BERT to recognize if sentence subjects make sense in context, and analyze its effect on downstream task performance as compared to additional pre-training on the masked language model (MLM) objective using in-task data for the downstream tasks.

Through our experiments, we find that additional pre-training on the MLM objective improves performance on downstream tasks, and that pre-training with the swapped sentence phrase (SSP) detection task improves performance even more than the MLM objective does. We also find that pre-training on a combination of the two tasks (with round-robin training) performs the best, and also generalizes better and is less skewed towards a single task.

3 Related Work

The motivation for this work was first inspired by the discovery in RoBERTa (Liu et al., 2019) that next sentence prediction was not a useful pre-training task for downstream task performance, even though two of the three downstream tasks used in this project rely on BERT identifying relationships between sentences. The authors of RoBERTa found that using only the masked language model objective for pre-training was sufficient for SOTA (at the time) results on multiple downstream tasks.

Other pre-training tasks have also been proposed as alternatives or additions to masked language modeling. One such task is Sentence-level Language Modeling (Lee et al., 2020), which aims to capture higher-level features in text at the sentence level by teaching the model to unshuffle a sequence of sentences in order to learn the relationships between the sentences. The authors showed that this led to significant performance improvements on multiple downstream tasks (especially those that considered relationships between sentences) from the base BERT.

Other papers have also explored different variations of token-level tasks similar to the masked language modeling objective. Yamaguchi et al. (2021) explore 5 such tasks including shuffled word detection, random word detection, manipulated word detection, masked token type classification, and masked first character prediction. The authors find that some combinations of these tasks (along with multiple days of additional pre-training) improve on base BERT performance.

The new pre-training task we propose aims to capture features above the token level, although not solely focused on sentence relationships either, by having the model decide if the subject phrases of two sentences should be swapped.

4 Approach

For the default project BERT model (Devlin et al., 2019), please refer to the default final project handout for base BERT architecture details. To produce our baseline model for fine-tuning for downstream tasks, we add model heads to our BERT model for each downstream task. For all downstream and pre-training tasks, we create a model head consisting of two dropout + linear layers separated by a ReLU. For the semantic textual similarity task, we also add a sigmoid as the final activation function to squash the logits between 0 and 1. For tasks that take a pair of sentences as input (paraphrase detection and semantic textual similarity), we pass each of the input sentences through the base BERT model individually to get their embeddings, and then concatenate the embeddings to pass through the added model head.

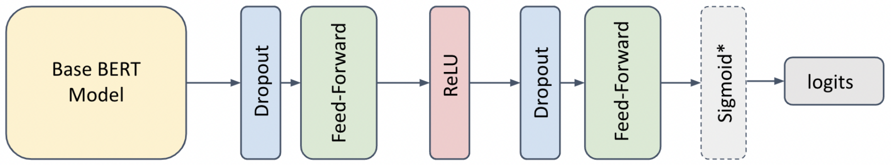

Figure 1: Illustration of the model architecture used for downstream tasks. *Only the semantic textual similarity detection model head uses a Sigmoid activation.

For loss functions, we use cross-entropy for sentiment analysis, binary cross-entropy for paraphrase detection, and MSE loss (with the output logits scaled up to match the range of the labels) for semantic textual similarity. For the additional masked language model pre-training, we use cross-entropy loss on the logits and labels, with large negative weights for non-masked tokens (so that they aren't accounted for in the loss), and binary cross-entropy loss for swapped subject phrase detection.

To fine-tune the BERT model for the three downstream tasks, we do round-robin multi-task fine-tuning. For each step of gradient descent of the optimizer, we calculate the loss for one batch of each downstream task and average the losses to update the model weights.

For the additional masked language model pre-training, we concatenate all downstream task datasets (the Stanford Sentiment Treebank dataset, the Quora dataset, and the SemEval STS Benchmark dataset) into a single large dataset and preprocess all inputs individually. We mask 15% of tokens randomly, with 80% probability of the token being masked, 10% probability of it being replaced with another token, and 10% probability of it being unchanged. The labels are defined as the original tokens for those 15% of tokens, and -100 for the other tokens so that the loss is not calculated using the unaffected tokens.

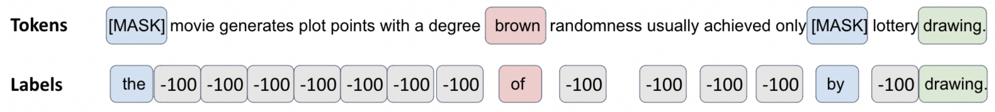
Tokens
[MASK] movie generates plot points with a degree brown randomness usually achieved only [MASK] lottery drawing.
Labels
the -100 -100 -100 -100 -100 -100 -100 of -100 -100 -100 -100 by -100 drawing.

Figure 2: An example of an input/label pair for the masked language modeling objective. The blue tokens have been masked out, the red replaced with a random token, and the green remains unchanged.

For the new swapped subject phrase detection task, we also concatenate all downstream task datasets, but shuffle and then randomly pair inputs from the combined dataset. For those sentence pairs, we use the spacy library to extract the subject phrases of each of the two input sentences. Then, with 50% probability, we swap the subject phrases of the two sentences to create a new pair of inputs. The other 50% of the time, the sentences remain unchanged. We then append a [CLS] token to the beginning of the sentence pair input and a [SEP] token between them, similar to how the original BERT model was trained. The final hidden state of the [CLS] token is used to predict if the subject phrases of the two sentences were swapped.

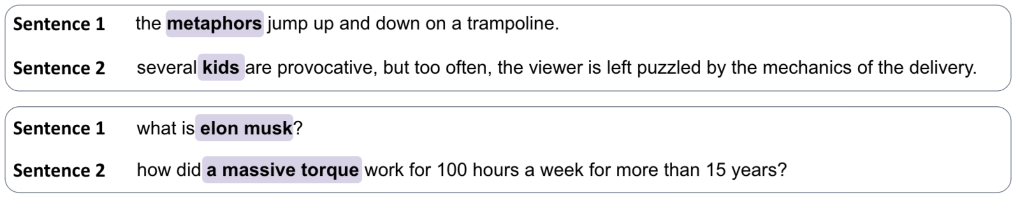
Sentence 1 the **metaphors** jump up and down on a trampoline.
Sentence 2 several **kids** are provocative, but too often, the viewer is left puzzled by the mechanics of the delivery.
Sentence 1 what is **elon musk**?
Sentence 2 how did **a massive torque** work for 100 hours a week for more than 15 years?

Figure 3: Two examples of swapped subject phrase detection input sentence pairs.

In the event that the subject phrases of the two sentences are the same (i.e. "I" is the subject in both sentences), we ensure that the label indicates they are not swapped, in order to avoid feeding the model confounding examples.

5 Experiments

5.1 Data

In our experiments, we use the datasets specified in the handout, including the Stanford Sentiment Treebank (SST) dataset (Socher et al., 2013), the Quora dataset for paraphrase detection, and the SemEval STS Benchmark dataset Agirre et al. (2013) for semantic textual similarity. In the first part of this project, the CFIMDB dataset was also used for sentiment analysis (but not for the multi-task portion of the project).

For multi-task fine-tuning, a dataloader is created for each dataset, and an epoch is defined by the length of the longest dataset (Quora). Due to a large imbalance in the size of the datasets (Quora has 141,498 training examples, SST has 8,544, and STS has 6,040), we implement custom data samplers for SST and STS to repeat examples to match the length of the Quora dataset. Thus, one epoch of training will see each example in the Quora dataset once and examples from the other datasets multiple times.

For the masked language model pre-training task, we similarly repeat examples from the smaller datasets so that they are the same length as the Quora dataset, in order to have the model train on an equal amount of data for all tasks. To do so, we create a custom data sampler to repeat indices of inputs from the smaller datasets to match the length of the Quora dataset. All three training datasets are combined into a single large dataset and batches of inputs are taken from the shuffled dataset. Because masking is done differently for all repeated examples, we don't believe that this training scheme will encourage excessive overfitting on the smaller datasets. For swapped subject phrase detection, we combine all training datasets before shuffling the examples and creating random sentence pairs. We then use the spacy library to find the first subject phrase (in case one input contains multiple sentences) in each original input and swap them half of the time, before concatenating the two and assigning it the corresponding label (0 for unchanged, 1 for swapped). The model is then pre-trained on batches of these sentence pairs. For SSP, we do not repeat examples in one epoch as subject phrases will not change for the same input, which could lead to overfitting.

5.2 Evaluation method

For this project, we use the evaluation metrics specified in the handout, including accuracy for sentiment classification and paraphrase detection, and the Pearson correlation between true and predicted similarity values for semantic textual similarity. For training, we use losses as described in the Approach section above.

The fine-tuning baseline referred to in this paper is a model using the base BERT with model heads for each downstream task as described and shown in the Approach section. This model loads in the pre-trained BERT weights (bert-base-uncased) provided, and fine-tunes for 3 epochs using round-robin training, where 1 epoch sees all inputs from the Quora dataset once (and inputs from the other two datasets multiple times). The model's performance on the dev dataset after being fine-tuned for 3 epochs serves as the baseline performance we use to compare to in our experiments.

5.3 Experimental details

For all experiments discussed here (fine-tuning as well as pre-training tasks), we use a learning rate of 1e-5 and a dropout rate of 0.3. We use a batch size of 32 and fine-tune for 3 epochs. For multi-task fine-tuning, this equates approximately 13,000 steps.

For additional pre-training on the masked language model objective, we use a batch size of 32 and train for 1 epoch, or approximately 25,000 steps. For the swapped sentence phrase detection task, we use a batch size of 32 and train for 1 epoch, or approximately 5,000 steps. For multi-task pre-training, where we use round-robin training for the two pre-training objectives, we repeat data for the swapped sentence phrase task to match the size of the MLM dataset for each epoch, but use a batch size of 8 due to memory limitations. We also train for 1 epoch, or approximately 100,000 steps.

When doing additional pre-training, we freeze the layers of the base BERT model for the first 500 steps of training in order to train the task-specific model heads first and avoid drastic updates to the BERT weights early in training when it adapts to the new domain and task.

The main experiments we run and discuss are:

- Pre-training for 1 epoch on the masked language model objective before fine-tuning for 3 epochs on the downstream tasks
- Pre-training for 1 epoch on the swapped sentence phrase detection task before fine-tuning for 3 epochs
- Multi-task (round-robin) pre-training for 1 epoch on both MLM and swapped sentence phrase detection, and then fine-tuning for 3 epochs

We acknowledge that for a more comprehensive comparison of all methods, further experimentation with more fine-tuning on all training schemes and comparisons between amount of pre-training with an equal number of steps versus epochs would be useful. A few experiments that we run to see the effects of pre-training for more equal numbers of steps are included and discussed in the results below.

5.4 Results

Although not discussed in length in this paper, the first part of this project involved implementing parts of the base BERT model and AdamW optimizer, and training the base model for SST with frozen and unfrozen BERT weights. The experiments done for that portion of the project are below, where we compare performance on SST for the base BERT trained on the SST dataset, and across all downstream tasks for a multi-task BERT trained on only SST and then on all downstream task datasets (but without round-robin training).

Experiment	SST Dev. Acc.	Para Dev. Acc.	STS Dev. PC
Base BERT Pretrain (SST only)	0.399	–	–
Base BERT Finetune (SST only)	**0.526**	–	–
Multi-task BERT Finetune (trained on SST only)	0.516	**0.507**	0.003
Multi-task BERT Finetune (trained on all)	0.487	0.482	**0.320**

These results make sense with a fully-trainable BERT trained only on SST performing the best on SST, the multi-task BERT trained only on SST doing the second-best on SST, and the multi-task BERT trained on on all datasets being the most generalizable model with better-than-random performance on two out of three tasks. None of the models did significantly better than random guessing on the paraphrase detection task, which may have been due to training without the round-robin method. For this model, we train on all batches of one dataset at a time in each epoch, and paraphrase detection was not the last task to be trained on, so the model may have forgotten information from that part of training by the time it's evaluated. Of the two multi-task BERT models, only the one trained on all datasets had reasonable performance on STS, with the other model having very poor performance. This could be due to STS simply being a much harder task, and since the model is only trained on SST, the model has no knowledge of similarity between sentences.

My results for the 3 main experiments listed in section 5.3 above (plus the fine-tuned baseline using round-robin multi-task training) are shown in the table below. Accuracy is reported on the train and dev datasets for SST and Paraphrase Detection, and the Pearson Correlation on the train and dev datasets is reported for STS. Bolded scores are for the highest dev set scores in each task, with paraphrase detection having two, one that does not consider the additional experiments below the horizontal line, and one that does.

The two additional experiments included below the horizontal line in the table include pre-training on MLM for 5,000 steps to match the length of pre-training on SSP for 1 epoch, and pre-training on SSP for 25,000 steps to match the length of pre-training on MLM for 1 epoch. These experiments were run to gain some insight on how the unequal number of steps between pre-training tasks affected downstream task performance.

Experiment	SST Train/Dev. Acc.	Para Train/Dev. Acc.	STS Train/Dev. PC
Fine-tuned Baseline	0.997 / 0.501	0.855 / **0.790**	0.946 / 0.489
MLM + Fine-tuning	0.991 / 0.500	0.842 / 0.775	0.922 / 0.529
SSP + Fine-tuning	0.997 / 0.508	0.859 / 0.788	0.939 / 0.544
MLM + SSP + Fine-tuning	0.991 / **0.510**	0.845 / 0.787	0.930 / **0.557**
MLM (5k) + Fine-tuning	0.997 / 0.504	0.901 / 0.781	0.981 / 0.434
SSP (25k) + Fine-tuning	0.984 / 0.476	0.863 / **0.804**	0.959 / 0.453

These results are promising because we can see that the best overall dev set performance was achieved by the combined MLM and SSP pre-training scheme, with the second highest overall performance from the SSP pre-training scheme. The MLM scheme had the next highest overall score after that, which illustrates that there's benefit from doing additional pre-training with MLM, an even greater benefit from doing SSP pre-training, and the most benefit from doing both.

We observe that even though the MLM scheme with only 5,000 steps was trained for a shorter amount of time, it actually overfit to the training data the most. We also see that the fine-tuned baseline and 25,000 steps of SSP did the best on the paraphrase detection, but both did worse on the other two tasks (generalized the least). This may be partially due to the imbalance in amount of training data between downstream tasks, since paraphrase detection had the most unique examples, and would be less prone to overfitting, especially with more time spent pre-training, as in the SSP (25k) case.

The first addition of MLM pre-training showed little change in SST accuracy and a decrease in paraphrase detection accuracy, but a large increase (4 points) on STS, so the objective likely helped the model become more generalized, and was able to improve average performance across the 3 tasks. Using SSP pre-training was an improvement on MLM in all 3 tasks, and out of the first 3 models in the table had the best performance on SST and STS (and had similar performance to the baseline on paraphrase detection). The combination of MLM and SSP pre-training mainly saw a substantial increase in STS performance, with a fairly significant 7 point increase from the baseline, and a 1.3 point increase from just SSP, which again indicates a more generalizable model.

Of the two additional experiments, MLM (5k) mainly saw a much worse performance in STS, with high overfitting during training. SSP (25k) performed the poorest on SST but the best on paraphrase detection.

Due to limitations of the test leaderboard, we only have test accuracies for two experiments, as shown below. We place at 101 on the leaderboard, although the goal of this project was not to produce the highest performing model possible using any means, but rather to evaluate the benefit of a new pre-training task.

Experiment	SST Test Acc.	Para Test Acc.	STS Test PC
SSP + Fine-tuning	0.523	0.786	0.512
MLM + SSP + Fine-tuning	**0.547**	**0.787**	**0.532**

We see that the multi-task pre-training with MLM and SSP far outperforms using SSP alone, with a 2 point increase in the STS score and a 2.4 point increase in the SST score. This implies that the multi-task pre-training is particularly helpful in generalizing the BERT model to multiple downstream tasks.

These results actually exceed expectations, as we were unsure how effective it would be to attempt to teach the model about subject phrases in sentences, and if it would be more helpful than masked language modeling. However, it appears that the SSP task does in fact teach the model features of the data that MLM does not, leading to better generalization and overall performance of the model on downstream tasks.

6 Analysis

In this section we examine the qualitative performance of our top-scoring model, which was pre-trained on both MLM and SSP.

6.1 Sentiment Analysis

Manual inspection of predictions on inputs showed that the vast majority of incorrect predictions were off by one. Additional training on the STS task or a larger training dataset for STS could therefore be helpful in aligning more of the off-by-one examples. Other than off-by-one errors, the main failure mode of the model was with contradictions/irony and context. One example is the input sentence:

"a beautifully made piece of unwatchable drivel."

which had the label 2 but was predicted as 4 by our model. The model most likely placed more weight on "beautifully made" even though we as humans see that the sentiment is dominated by calling the movie "unwatchable drivel." The model is unable to pick up on this subtlety and is unable to determine which phrase is more important for the sentiment of the sentence.

Another example of this is the model predicting the sentiment score of the sentence:

"don’t be fooled by the impressive cast list - eye see you is pure junk."

as 3, while the label is 0. This is similar to the previous example, where the sentence has one comlimentary phrase and one critical, where the critical phrase encodes most of the sentiment of the sentence, but the model is unable to identify it.

6.2 Paraphrase Detection

One of the main failure modes appears to be with sentences that differ in one or two words that have completely different meanings. For example, the sentence pair:

"do you need intelligence to succeed in life?"

"do we need to be obsequious to succeed in life?"

has label 0 (is not a paraphrase), but our model predicts 1, likely due to the sentences being largely similar besides the actual qualities the sentences are referring to ("intelligence" and "obsequious"). This implies that our model is still fairly flawed in identifying the similarity of important word embeddings, which could potentially be improved through additions like cosine similarity loss. One more example of this failure mode is the sentence pair:

"is there any evolutionary advantage of baldness?"

"was there any evolutionary advantage for beards?"

which have label 0 but the model predicted as 1, again due to the sentences mainly differing in the object being asked about ("baldness versus "beards").

We also observed a fairly common failure with examples that had somewhat ambiguous inputs. For example, the sentence pair:

"what are the best and profitable ways for saving money?"

"what are your best ways to save money?"

had label 0 but was predicted to have label 1 by our model. However, it could be argued that these questions are in fact similar enough to be paraphrases, and there are many such examples where the label for a question pair is 0 but the questions are very similar in meaning. These examples may be somewhat confounding, as a human could likely make the same prediction as our model.

Of the main successful examples, our model does fairly well with rephrasings of questions, which are likely a majority of the paraphrased questions. For example, the sentence pair:

"will we ever rid the world of terrorism?"

"will terrorism in the world ever stop?"

is a rephrasing of the same question, and our model is able to correctly identify that it is a paraphrasing.

6.3 Semantic Textual Similarity

The main failure mode we observe is possibly a direct result of our swapped subject phrase detection pre-training. Many incorrectly classified examples have the same subject and differing predicates, making them highly unrelated, but with our model predicting that they are related.

Two examples of this behavior include the sentence pair:

"a dog swims through the water."

"a dog wearing a cape is running through the snow."

which has a label of 1.0 and a prediction of similarity of 3.6 by our model, and the sentence pair:

"china stocks open higher monday"

"china stocks close lower on thursday"

which also has a label of 1.0 and a prediction of 3.4 by our model. We hypothesize that our model thought the pairs were much more similar than they were due to the subjects of the sentences being the same, and placing too much weight on that. We found it very interesting how large of a potential impact a single epoch of pre-training with SSP had on the model's performance, and consider experiments with the amount of SSP pre-training for reducing this behavior as work for the future.

7 Conclusion

The main takeaways from this work are that additional pre-training using the same masked language model objective that BERT was originally trained on, using in-task data, was beneficial for performance on multiple downstream tasks, and that the proposed pre-training task of swapped subject phrase detection was even more beneficial (also using in-task data), with a combination of the two (using multi-task pre-training) producing the highest overall performance scores out of all these configurations. Qualitative analysis of results shows that the swapped subject phrase detection task may bias the model towards that part of sentences in downstream tasks, and further experimentation would be needed to determine the ideal amount of SSP pre-training. This work is primarily limited by the number of experiment configurations tested: ideally, different combinations of different amounts of time spent pre-training and fine-tuning the model would provide more well-defined insights on the benefits of each pre-training task. Such exploration would be an avenue for future work, and derivatives of the SSP task that focus on other aspects of sentences (or larger sequences of text, such as topic sentences in a paragraph) would be interesting to explore as well.

References

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, Atlanta, Georgia, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Haejun Lee, Drew A. Hudson, Kangwook Lee, and Christopher D. Manning. 2020. Slm: Learning a discourse language representation with sentence unshuffling.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Atsuki Yamaguchi, George Chrysostomou, Katerina Margatina, and Nikolaos Aletras. 2021. Frustratingly simple pretraining alternatives to masked language modeling.