Comparing Self-supervised Learning Pretext Tasks for Language Modeling

Dean Huang
MIDS
Duke University
tingchiang.huang@duke.edu

Michael Tang
MIDS
Duke University
mt374@duke.edu

Betty Wu
MIDS
Duke University
jiaman.wu@duke.edu

Abstract

Self-supervised learning is rapidly gaining popularity. It enables the model to learn from a large number of unlabeled datasets, and transfer the knowledge learned to perform specific downstream tasks. Self-supervised learning generally has two main stages. The first Pre-training stage trains the model on a large general-purpose unlabeled dataset with various pretext tasks. The second Fine-tuning stage uses the weights learned in the pre-training stage as initial weights and trains the model on a smaller labeled dataset to perform specific tasks. In this project, we compared the impact of different pretext tasks on a downstream task in language modeling. Ablation studies were also implemented to compare the marginal effect of each pretext task and the interaction between pretext tasks. We found that Masked Language Modeling (MLM) performs the best, achieving 0.86 test accuracy; Sentence Order Prediction (SOP) produced the lowest accuracy at 0.77 on the IMDB dataset. We also found that the performance on the pre-training phase is not an indicator of the performance on the specific downstream task.

1 Background

Self-supervised learning is a machine learning method that is rapidly gaining popularity. It is useful because we have large unlabeled datasets readily available. Large labeled datasets are typically unavailable or expensive to obtain. Therefore, self-supervised learning bridges the gap between the abundance of unlabeled data and the need for labeled data for better model performance. This technique enables the model to learn from a large number of unlabeled datasets, and transfer the learned knowledge to perform specific downstream tasks. This paradigm of supervised learning has proven to achieve cutting-edge performance on various downstream tasks in computer vision, natural language processing, and other fields. Self-supervised learning generally has two main stages. The first stage is to train the model on a large general-purpose unlabeled dataset. This stage is referred to as the pre-training stage. During this stage, the model learns the general background knowledge and forms common sense in the system. This stage enables the model to leverage the massive amount of unlabeled data available. At this stage, the model performs self-supervised tasks, also known as pretext tasks. The performance on this task is less important. Rather, we are interested in whether the model learns reasonable intermediate representations or structural meaning that are beneficial to practical downstream tasks. The second stage is fine-tuning. This stage is similar to transfer learning where we use the weights learned in the pre-training stage as initial weights and train the model on a smaller labeled dataset to perform specific tasks. By leveraging knowledge learned from pretext tasks, we expect the model will perform better in the downstream task.

In this project, we are interested in comparing the impacts of different pretext tasks on the downstream task in language modeling. The pretext tasks we compared are Masked Language Modeling (MLM) [1], Next Sentence Prediction (NSP) [1], Sentence Order Prediction (SOP) [2], and Sentence

ECE 661 (2021 Fall) final report.

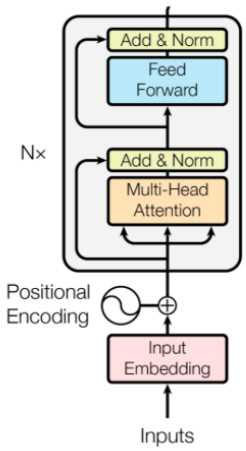

Figure 1: BERT Encoder

Permutation (SP) [3]. In addition, the model architecture explored is the attention-based Transformer encoder model as proposed in Vaswani 2017 [4]. The downstream task is sentiment classification for the IMDB dataset.

2 Method

The Transformer model was proposed by Vaswani et al., 2017 was to tackle the translation problem. It was designed to address several limitations of the traditional LSTM model such as slow training and vanishing gradient due to long sentences as well as its inability to be parallelized. During Neural Machine Translation (NMT), the encoder section of Transform ingests all input words in one language and generates the corresponding embeddings simultaneously. The decoder section then utilizes the embeddings to produce translation one word at a time in another language. The Transformer model yielded state-of-the-art performance in language translation. However, there had also been a strong demand in other natural language processing tasks such as Question Answering, Text Classification, and Sentiment Analysis, which all require understanding the contextual information of a language. Vaswani 2017 [4] proposed the attention mechanism that eliminates the limitations of LSTM. Transformers networks use the attention mechanism extensively to achieve state-of-the-art performance. Therefore, Bidirectional Encoder Representation from Transformer (BERT) [1] was developed to accomplish the goal of understanding language and learning specific tasks. As mentioned previously, the training of BERT consists of two stages - pretraining and fine-tuning. In pretraining, the model performs self-supervised learning to learn general knowledge of the language such as its context. In the fine-tuning phase, the model is then trained for a specific task by leveraging information it has previously learned.

2.1 Model Architecture

Our project aims to reproduce the pretraining tasks based on the architecture of BERT. This architecture utilizes the encoder section of a Transformer, which effectively learns the context of a language. More specifically, our Transformer model is constructed by stacking three layers of the Transformer encoder. The architecture of a single encoder is illustrated in figure 1. A single encoder consists of four attention heads, where each head has a hidden size of 64 (256/4) and uses a dropout rate of 0.1. The feedforward layer uses the GELU activation as opposed to the regular ReLU activation function, a hidden size of 1024 (256 * 4) and a dropout rate of 0.1.

The input to the model consists of 256 tokens. More specifically, a sequence longer than 256 tokens would be truncated, whereas a shorter sequence would be zero-padded. The tokens are then fed into the input embedding, which transforms each token into a 256-dimensional vector. This input embedding consists of 3 different embeddings as shown in figure 2. The Token Embedding uses the pre-trained 'bert-base-uncased' tokenizer, which has a vocabulary size of 30,522. Therefore, This

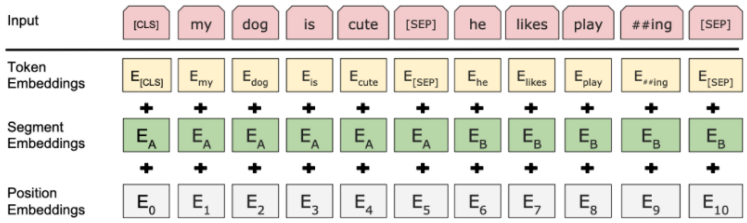

Figure 2: Input Embedding Structure

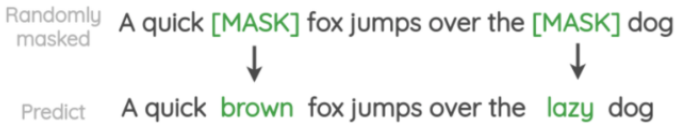

Figure 3: Illustration of MLM. The goal is to predict [MASK] tokens.

embedding layer has a dimension of 30,522 by 256. The Segment Embedding encodes the sentence number and has a dimension of 3 by 256. The Position Embedding indicates the position of a word within a sentence and has a dimension of 256 by 256.

The output dimension varies for different pertaining tasks. For instance, all pretraining tasks except for MLM expect a binary output. Therefore, the output dimension would be 2. In the case of MLM, since the goal is to predict the masked word, the output dimension would be the vocabulary size (30522). In the fine-tuning stage, the output dimension final fully connected layer of the BERT model is changed to 2 to account for the binary outcome in sentiment analysis.

2.2 Pre-training

In this project, we experimented with four different pre-training tasks, which are masked language model (MLM), next sentence prediction (NSP), sentence order prediction (SOP), and sentence permutation (SP). All pre-training tasks were trained on the IMDB movie review dataset, which contains 500,000 movie reviews and their corresponding sentiment label ("positive", "negative"). In addition, since all tasks are classification-based, the training objective aimed to minimize the cross-entropy loss. We used the Adam optimizer with a learning rate of 1e-4 and 20 training epochs. Finally, we used a warm start learning rate scheduler where the warm-up was set for the first 10 epochs. The implementation detail for each pretraining task is described below:

Task 1: Masked Language Modeling

Masked language modeling (MLM) is a pre-training task that was popularized by BERT. This pretext task involves randomly masking a word in a sentence with the `[MASK]` token. The model is trained to predict the masked word. Unlike left-to-right language model pre-training, MLM forces the representation learned to consider both the left and the right context. This way we are able to pretrain a deep bidirectional Transformer. To prepare data for training on this task, we take one training example in the dataset, and randomly mask 15% of the tokens that are not padding tokens or other special tokens. See Figure 3. The labels are the tokens for these masked tokens. Note that in order for the model to concentrate on predicting the masked tokens, the gradients updates only apply to the masked tokens.

Task 2: Next Sentence Prediction

Next sentence prediction (NSP) is a pretext task where the goal is to predict whether a sentence follows another sentence in a contextually correct manner. The goal is to help the model understand sentence relationships. More concretely, given a pair of sentences, sentence A and Sentence B, the pre-trained model should correctly identify if Sentence B follows Sentence A. This idea is illustrated

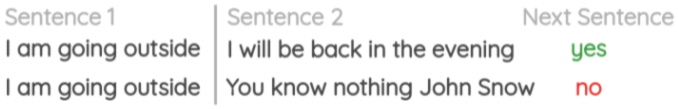

Figure 4: Illustration of NSP. The goal is to predict correct if Sentence 2 follows Sentence 1

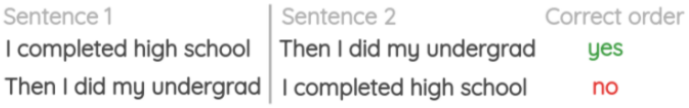

Figure 5: Illustration of SOP. The goal is to predict if two sentences are in the right order.

in Figure 4. In the example, 'I will be back in the evening' is the contextual correct sentence to follow 'I am going outside' as opposed to 'You know nothing John Snow'. Because of the binarized outcome, this pretrain task can be treated as a classification task. Data preparing involves selecting Sentence A and Sentence B from the same monolingual corpus, where Sentence B is the actual sentence that follows Sentence A 50% of the time (positive), and a random sentence is sampled from the corpus as Sentence B for the other 50% of the time (negative).

To achieve this, we created a bag of sentences, where each 'sentence' is a span of at most 5 sentences. In cases where a movie view consists of less than 5 sentences, the entire movie review is treated as a single span. In addition, because we assumed that movie reviews are independent of one another, we implemented the span generation process so that a span could only consist of sentences from the same movie review. For the positive case, we pick 2 consecutive spans from the same movie review. For the negative case, we pick the next available span as Sentence A, and randomly sample Sentence B from the bag. The data were then aggregated using the scheme in figure 4 to prepare for training. Note that `[SEP]` separate token is placed between Sentence A and Sentence B. `[CLS]` taken indicates the start of a training sample. The output is a single value that represents the true class. Therefore, the final layer used in our model is a fully connected layer. The fully connected layer's input size is 256 as defined by our transformer architecture, and the output size is 2, which represents the positive and negative probabilities.

Task 3: Sentence Order Prediction

Sentence order prediction (SOP) is a pretext task with the goal of predicting the ordering of two consecutive segments of texts. SOP focuses primarily on inter-sentence coherence and is designed to overcome the ineffectiveness of the NSP loss proposed in the original BERT. The main reason behind NSP's ineffectiveness is its conflation of topic prediction and coherence prediction in a single task. Overall, topic prediction is much easier to learn than coherence prediction, and the process has a huge overlap with the learning process of MLM. To make the model learn the discourse-level coherence properties, the negative example has the same two consecutive segments as NSP but with their order swapped. This idea is illustrated in Figure 5. In the example, you can see the negative sample having segment order swapped with 'I completed high school' in sentence 2 and 'Then I did my undergrad' in sentence 1. Therefore, the only difference between NSP and SOP for our project was the creation of negative samples.

Task 4: Sentence Permutation

Sentence Permutation (SP) was used in BART[3] as one of the pretext tasks. The goal of Sentence permutation is to learn how sentences in one paragraph are ordered. Given one document, the document is divided into sentences based on full stops. These sentences are then shuffled randomly. The model is trained to predict the permuted order of these sentences, as illustrated in Figure 6[1].

[1]Figure 3, Figure 4, Figure 5, Figure 6 are adapted from https://amitness.com/2020/05/self-supervised-learning-nlp/

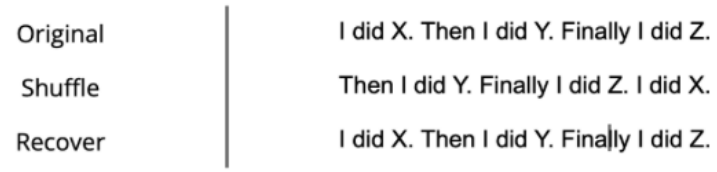

Figure 6: Illustration of SP. The goal is to predict the correct order.

2.3 Fine-tuning

The fining-tuning is similar to a typical supervised learning task. This phase consists of taking the pre-trained model from each task and re-train it on the IMDB dataset with the task of predicting whether the sentiment is "positive" or "negative". Each data example is tokenized with the pre-trained BERT uncased tokenizer with 256 tokens and 256 embedding size. The hyperparameters are the same as the pre-training tasks.

3 Experiment results

It is important to note that in order for results to be comparable across different pretext tasks, each model training used the exact same set of hyperparameters.

3.1 Performance Metrics

For the pre-training phase, the accuracy of MLM was calculated by dividing the sum of correct mask predictions with the total number of mask predictions made, while the accuracy of NSP/SOP/SP was calculated by dividing the sum of correct sentence predictions with the total sentence predictions made. If the pre-train tasks are hybrid, meaning having two pre-train tasks, the overall accuracy was calculated by averaging the accuracies of those two tasks. For the fine-tuning phase, the sentiment classification accuracy was calculated by dividing the sum of correct sentiment predictions with the total sentiment predictions made.

3.2 Experimental Setup

For both pre-train and fine-tune phases, we train for a total of 20 epochs and 10 epochs on the IMDB movie review dataset respectively. During training, we monitored both the loss and accuracy for the training set and the validation set, making sure the loss is reducing and the accuracy is increasing as the number of epochs increases.

3.3 Pre-train Performance Comparison

Since MLM is making word level prediction instead of sentence level prediction, it is not appropriate to compare MLM with NSP/SOP/SP. Therefore, we will only conduct performance comparison for sentence level pre-train tasks and for hybrid pre-train tasks. Overall, for sentence level pre-train tasks, NSP was able to achieve a higher validation prediction accuracy given the same number of epochs. The result matches our expectation because out of all the sentence level pre-train tasks, NSP is the easiest to learn, given the task can achieve decent results simply by excelling in topic prediction. In contrast, both SOP and SP required excelling in coherence prediction, which is more difficult to learn than topic prediction. Since SP requires predicting the correct order for more than two sentences, SP has the lowest validation prediction accuracy out of all the sentence level pre-train tasks. For hybrid pre-train tasks, MLM + NSP has the highest accuracy and MLM + SP has the lowest accuracy, which matches the result of sentence level pre-train tasks comparison.

3.4 Fine-tune Performance Comparison

Among all the pre-train tasks, MLM + SOP and MLM have the highest sentiment classification test accuracy and SOP has the lowest sentiment classification test accuracy.

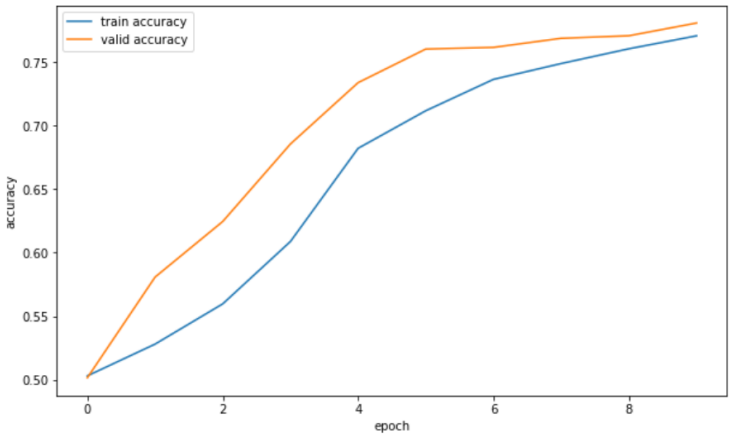

Figure 7: SOP Finetuning accuracy by epoch.

Pre-train + Fine-tune		
Pre-train Tasks	Best Validation Prediction Accuracy	Sentiment Classification Test Accuracy
Baseline	-	0.861
MLM	0.035	0.860
NSP	0.578	0.858
SOP	0.510	0.773
SP	0.250	0.851
MLM + NSP	0.297	0.849
MLM + SOP	0.273	0.861
MLM + SP	0.249	0.85

Besides SOP, the training accuracy and validation accuracy for all pre-train tasks began to diverge at some point of time, showing signs of overfitting. Even though SOP had the lowest test accuracy, 0.773, Figure 7 shows both the training accuracy and validation accuracy continued to increase even after epoch 10, showing signs of the model underfitting. Therefore, we can probably obtain higher test accuracies if we increase the number of epochs.

3.5 Ablation Performance Analysis

The result of our experiment showed that besides MLM + SOP, all the hybrid pre-training tasks were not able to outperform its individual pre-training counterparts. One major reason might be due to the training dataset that is used to train the pre-training tasks. Given the limited computational resources, we utilized IMDB reviews for training instead of a larger, more complex corpus. The average length of each review was short and the review lacked diversity in phonemes, morphemes, lexemes, syntax, and context. Therefore, we were unable to train our model to understand various linguistic structures. If we would have trained our model on a larger and more diverse corpus, we believed the hybrid tasks would definitely outperform their individual counterparts.

4 Conclusions

To sum up, for the pre-training stage, NSP and SOP had high validation prediction accuracy for sentence level prediction tasks and MLM + NSP had the highest validation prediction accuracy for hybrid pre-train tasks. However, for the fine-tune stage, MLM and MLM + SOP produced the best performance with test accuracy equal to 0.86. This shows the performance of the pre-training stage is not an indicator of the performance of the downstream task, as shown by MLM + SOP. SOP had the worst performance in the fine-tune stage with test accuracy equal to 0.77. The poor performance of SOP was mainly resulted from underfitting, which could be addressed by longer training epochs. To further boost the performance of each pre-train task, we have to perform hyperparameter tuning for each task. However, the most optimal way to boost the performance of each pre-train task is to perform pre-training on a larger dataset.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. volume abs/1810.04805, 2018.

[2] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942, 2019.

[3] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019.

[4] 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.

A Timeline and task allocation

Timeline		
Date	Goal	Assigned Member
November 12th	MLM Pre-training and Finetuning	Betty Wu
November 15th	NSP Pre-training and Finetuning	Michael Tang, Betty Wu
November 18th	SOP Pre-training and Finetuning	Michael Tang, Betty Wu
November 22th	SP Pre-training and Finetuning	Michael Tang, Betty Wu
November 25th	MLM + NSP Pre-training and Finetuning	Dean Huang
November 28th	MLM + SOP Pre-training and Finetuning	Dean Huang
December 3rd	MLM + SP Pre-training and Finetuning	Betty Wu