Fine-Tuning BERT for Multi-Task Prediction

Stanford CS224N Default Project

Uma Dayal
Department of Computer Science
Stanford University
`umadayal@stanford.edu`

Abstract

In this project, we delve into the multi-task fine-tuning of BERT (Bidirectional Encoder Representations from Transformers) for three NLP tasks: sentiment analysis, paraphrase detection, and Semantic Textual Similarity (STS). BERT's bidirectional context understanding has revolutionized natural language processing, yet applying it to multi-task learning presents challenges due to potential conflicts between task-specific objectives and data imbalances. We begin by implementing a minimal BERT (minBERT) framework and employing joint loss-based fine-tuning for the tasks. We then integrate Gradient Surgery (PCGrad) to mitigate conflicting gradient directions and explore alternative techniques like cosine-similarity loss and mean-pooling for enhancing STS performance. Our best performing model involves using cosine-similarity loss for STS with mean-pooling. It achieves an overall test score of 0.692, with an accuracy of 0.520 for sentiment classification, 0.721 for paraphrase detection, and 0.667 for STS correlation.

1 Introduction

BERT (Bidirectional Encoder Representations from Transformers) is a transformer-based deep learning language representation model (Devlin et al. (2019)) that has led to great advancements of the domain of natural language processing (NLP). It is especially unique compared to other pre-trained language models because of its deep bidirectional understanding of context in text. Many other language models such as Radford et al. (2018) use unidirectional representations but bidirectional representations could better capture a word's meaning within the full context of the sentence because it considers context from both the left (previous words) and the right (subsequent words) of a given word.

This project explores the implementation and fine-tuning of BERT for three specific downstream tasks: sentiment analysis, paraphrase detection, and Semantic Textual Similarity (STS). Multi-task prediction is an active area of research and is especially challenging because different tasks may have different objectives that could potentially conflict each other. Different tasks may also have varying amounts of data, leading to data imbalance issues. Furthermore, optimizing a model for multiple tasks can be difficult since the loss functions for each task might have different scales and convergence rates, making it hard to find a shared representation that performs well across all tasks.

We start with a minimal implementation of BERT (minBERT) and explore multi-task fine-tuning approaches as laid out by Bi et al. (2022) which uses a joint loss for the three tasks. We then improve on this approach by implementing Gradient Surgery as proposed by Yu et al. (2020). Lastly, we look at other methods such as cosine-similarity fine-tuning and mean-pooling (Reimers and Gurevych (2019)), which vastly improves the performance on the STS task.

2 Related Work

Our approach is largely based on that proposed in the multi-task learning paper by Bi et al. (2022). In this paper, the loss for each task is added together to form a joint loss function. So, in our case, the loss function would look like:

$$\mathcal{L}_{total} = \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_3.$$

However, this approach has the issue of gradient conflicts mentioned in the previous section where the gradient directions of each of the tasks can form an angle of greater than $90°$, leading to conflicting objectives. We avoid this by employing the technique of Gradient Surgery proposed by Yu et al. (2020). Gradient surgery avoids this by projecting the gradient $\mathbf{g}_i$ onto the normal plane of another opposing task's gradient $\mathbf{g}_j$:

$$\mathbf{g}_i = \mathbf{g}_i - \frac{(\mathbf{g}_i \cdot \mathbf{g}_j)}{||\mathbf{g}_j||^2} \cdot \mathbf{g}_j.$$

We also employ cosine-similarity fine-tuning and mean-pooling to improve performance on the STS task. The SBERT framework proposed by Reimers and Gurevych (2019) enhances the capabilities of BERT to derive semantically meaningful sentence embeddings. SBERT is able to improve on the state-of-the-art performance of BERT on the STS task. At the core of their approach is integrating different pooling strategies such as using the CLS token (as in our baseline model), the mean of all output vectors (MEAN-strategy), or the max-over-time of the output vectors (MAX-strategy). They then use cosine-similarity to compare the similarity between the two sentence embeddings, instead of using a regression function to predict the similarity score as in our baseline model. I will be applying these strategies to improve on our baseline.

3 Approach

We build on the BERT framework. The base BERT consists of 12 Encoder Transformer layers which each have the architecture shown in Figure 1. The multi-head attention layer is a key component of the Transformer architecture, transforming hidden states for each sequence element based on others. Unlike fully-connected layers, which process each element separately, the multi-head attention layer combines n different dot-product attention mechanisms to represent each sequence element using a weighted sum of the hidden states of all elements. In BERT, each attention mechanism projects the hidden states into different subspaces, allowing the model to focus on various types of information.

The outputs of these attention mechanisms are concatenated and linearly transformed. A BERT layer, incorporating this multi-head attention, also includes layer normalization and a feed-forward network with a non-linear activation function. The entire process involves layer normalization applied to the self-attention layer's output, combined with a residual connection for stability and performance.

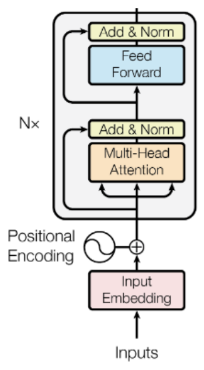

Figure 1: Encoder layer of transformer in BERT (Vaswani et al. (2017))

3.1 Model Architectures

3.1.1 Baseline Model

The baseline model involves passing the input for the three tasks through BERT. The CLS token outputs are finally passed through the three classification heads. The architecture is shown in Figure 2.

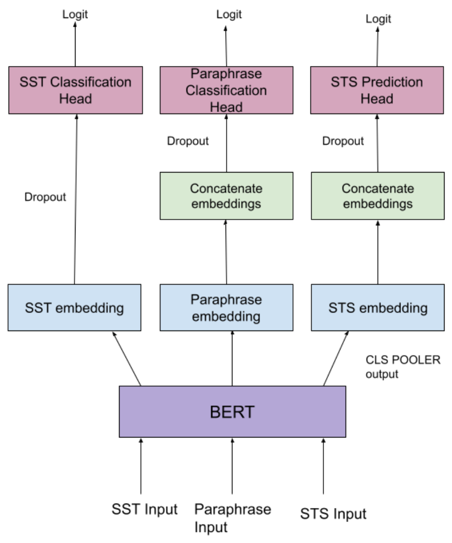

Figure 2: Baseline Architecture

3.1.2 Embedding Cosine Similarity with Mean Pooling

The two main differences in this architecture are that instead of the CLS token, we use the mean pooler output, and instead of passing the STS embeddings through the STS classification head, the logits are the cosine similarities of the embeddings (Figure 3).

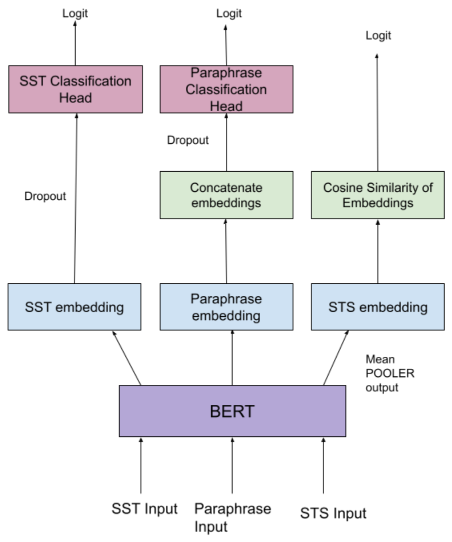

Figure 3: Embedding Cosine Similarity with Mean Pooling Architecture

3.2 Loss Functions

3.2.1 Cross Entropy Loss

The sentiment classification task is a multi-class classification problem and hence, we use the following loss function:

$$\text{Cross Entropy}(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i).$$

The paraphrase detection is a binary classification problem that uses the loss function

$$\text{BCE with Logits}(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N} [y_i \cdot \log(\sigma(\hat{y}_i)) + (1 - y_i) \cdot \log(1 - \sigma(\hat{y}_i))]$$

3.2.2 MSE Loss

We compare the cosine-similarities of the embeddings (appropriately scaled so that they are between 0 and 5 instead of -1 and 1) with the ground-truth STS labels using a Mean-Squared Error (MSE) loss function:

$$\text{MSE Loss}(y, \hat{y}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2.$$

4 Experiments

4.1 Data

For the task of sentiment analysis, we use Stanford Sentiment Treebank (a collection of sentences extracted from movie reviews). This is further parsed with the Stanford parser to form our dataset consisting of unique phrases from those parse trees annotated by 3 human judges to have labels negative, somewhat negative, neutral, somewhat positive, or positive (5 classes). There are 8544 training samples, 1101 validation samples, and 2210 test samples.

For paraphrase detection, we use the Quora dataset which consists of question pairs with labels indicating whether particular instances are paraphrases of one another (binary outputs). There are 141,506 training samples, 20215 validation samples, and 40431 test samples.

For semantic textual similarity, we use the SemEval STS Benchmark dataset consisting of 8,628 different sentence pairs of varying similarity on a scale from 0 (unrelated) to 5 (equivalent meaning). There are 6041 training samples, 864 validation samples, and 1726 test samples.

4.2 Evaluation method

For the sentiment classification and paraphrase detection tasks, we use classification accuracy as our evaluation metric where we evaluate how close the predicted classes are to the ground-truth labels. For the STS task, we use Pearson correlation to evaluate how close the predicted similarity scores to the ground-truth scores.

4.3 Experimental details

For all of our experiments, we run 10 epochs (755 iterations each), fine-tune on the full-model, and use learning rate $1e$-5.

4.3.1 Baseline Experiment

In the baseline experiment, we use the model architecture described in Section 3.1.1. We use BCE with logits loss function for paraphrase detection, cross entropy loss for sentiment classification, and MSE loss for STS (between predicted similarity scores and labels). We then add the losses together as done by Bi et al. (2022) to get a joint loss function.

4.3.2 Cosine-similarity

In this experiment, we use the same loss functions for paraphrase detection and sentiment classification, but for STS, we compute the MSE loss between the (scaled) cosine-similarity scores of the embeddings and the labels. We again use a joint loss function (multi-task fine-tuning).

4.3.3 Cosine-similarity + Mean Pooling

We use the architecture in Section 3.1.2, which has a similar setup to the previous experiment except we now use mean-pooling instead of the CLS token.

4.3.4 Cosine-similarity + Mean Pooling + External Similarity Feature

This is similar to the previous experiment, except to improve the paraphrase detection performance, we concatenate the cosine-similarity to the two embeddings so that it is included as an external feature. Since two sentences that are paraphrases of each other would have high similarity scores, we're hoping that adding the cosine-similarity score as a feature would give the model more information to predict this.

4.3.5 Cosine-similarity + Mean Pooling + External Similarity Feature + PCGrad

We do the previous experiment but now with gradient surgery (PCGrad) to counter the conflicting gradients.

4.4 Results

The single-task SST accuracy for minBERT was found to be 0.515. We compare our SST performance to this. For the multi-task performance, we compare the accuracies and correlations of the different tasks with those of the baseline experiment.

The dev results are shown in the table below. We submitted our best performing experiment (Cosine-similarity + mean pooling + PCGrad) to the test leaderboard submission and got a SST test accuracy of 0.520, paraphrase test accuracy of 0.721, and STS test correlation of 0.667. The **overall test score** was **0.692**.

	SST dev acc	**Paraphrase dev acc**	**STS dev corr**
Baseline	**0.514**	0.722	0.360
Cosine-Sim	0.514	0.722	0.458
Cosine-Sim + mean-pool	0.502	0.722	**0.703**
Cosine-Sim + mean-pool + feature	0.510	0.**728**	0.598
Cosine-Sim + mean-pool + feature + PCGrad	0.507	0.706	0.686

Table 1: Results Table

The best **overall dev score** I got was **0.692** with the Cosine-Sim + mean-pool + PCGrad experiment.

5 Analysis

We see that the addition of cosine-similarity fine-tuning and mean-pooling led to the greatest improvement from the baseline, though it did lead to a decrease in performance for the SST task. Cosine-similarity fine-tuning by itself did not lead to any change in SST accuracy and paraphrase accuracy but lead to improvement in STS correlation because while cosine similarity provides a more nuanced measure of similarity for STS, it does not directly impact sentiment analysis or paraphrase detection tasks.The fact that mean pooling significantly enhanced STS performance indicates that averaging embeddings captures contextual information better than the CLS token for similarity tasks. It, however, led to worse SST accuracy, which could possibly be because averaging embeddings can smooth out nuances in meaning conveyed by specific words, leading to loss of important sentiment cues.

In terms of paraphrase accuracy, adding cosine-similarity as a feature led to the greatest improvement in paraphrase detection, but led to decreases in performance for both SST and STS. This could be because the addition of cosine similarity as a feature may lead to overfitting on the paraphrase detection task. If the model becomes too specialized in predicting paraphrases based on the combined features, it may lose its generalization ability for the STS task. In case the decreased performance was due to conflicting gradients, we did another experiment using gradient surgery, but, interestingly, this led to an improvement in STS but was worse for SST and paraphrase detection.

In general, we see that the SST accuracy for all our experiments cannot surpass that of the single-task baseline (0.515) and does not improve from the baseline accuracy. BERT might be struggling more with task compared to the other task because of the difficulty in capturing the subtleties in sentiments.

6 Conclusion

After experimenting with different combination of multi-task fine-tuning techniques, we concluded that combining cosine-similarity embedding loss with mean-pooling yielded the most substantial overall performance improvements, particularly enhancing Semantic Textual Similarity (STS). However, these enhancements came at the cost of reduced accuracy in sentiment analysis (SST), highlighting the challenge of balancing different task requirements in a shared model. Gradient surgery (PCGrad) provided only marginal benefits, suggesting its limited effectiveness when tasks share significant similarities. ur experiments also revealed that while these advanced techniques improved paraphrase detection and STS, they failed to enhance SST performance, underscoring the complexity of capturing sentiment nuances in a multi-task framework. Future research could focus on developing specialized approaches to boost SST accuracy without compromising the performance of other tasks, potentially involving more nuanced integration of task-specific features or adaptive loss balancing strategies.

7 Ethics Statement

The model might learn and propagate biases, such as racial or gender biases, present in the training data, leading to discriminatory outcomes. For instance, sentiment analysis could discriminate against certain demographic groups if the training data contains biased language or opinions. One way to mitigate this is to further train the model on more diverse datasets with less bias against minority demographic groups.

Furthermore, the model could be used to generate or spread misinformation, especially in applications involving text generation or sentiment analysis. This could be mitigated by hard-coding certain checks that moderate the validity of the content generated.

References

Qiwei Bi, Jian Li, Lifeng Shang, Xin Jiang, Qun Liu, and Hanfang Yang. 2022. Mtrec: Multi-task learning over bert for news recommendation. In *Findings*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning.