Improving minBERT and Its Downstream Tasks

Stanford CS224N Default Project

Kevin Yang
Department of Computer Science
Stanford University
keviny23@stanford.edu

Madhumita Dange
Department of Computer Science
Stanford University
madhumid@stanford.edu

Abstract

We re-implemented the minBERT model with multi-head self-attention and transformer layers. Using pre-trained parameters loaded, we performed evaluation on both sentiment analysis only and multi-task pipeline. We also optimized various aspects of the BERT model targeting at higher computational performance on multiple tasks. Targeting for robustness and generalization, our particular approach mainly focused in these three aspects: hyperparameter tuning, multi-task setting, and model size and layers. We also conducted comprehensive numerical experiments to compare the performance of the model before and after each optimization step by step. While a lot of more study is needed in this area, the BERT mechanism has the potential to achieve both efficiency, performance, and generality in the future.

1 Key Information to include

- Mentor: David Lim <dslim@stanford.edu>
- External Collaborators (if you have any): NA
- Sharing project: NA

2 Introduction

Pre-training on language models has been proven to be one of the most effective way to improve various natural language processing tasks (Dai and Le, 2015; Peters et al., 2018; Howard and Ruder, 2018). In 2019, BERT (Bidirectional Encoder Representations from Transformers) were proposed to further pre-train deeper bidirectional representations from unlabeled text. Unlike other language representation models, BERT has been designed to fine-tune for various natural language tasks such as language inference, question answering and so on without substantial task-specific architecture modifications other than an additional output layer. BERT has been showed by numerical examples to achieve empirically powerful performance with conceptually simplicity (Devlin et al., 2019).

While BERT proposed a brand new pre-training mechanism that has been adopted and extended later on to many task and applications as one of the many big milestones of nature language processing, it also has its limitations. For example, it only evaluated the performance based on GLUE datasets, SQuAD v1.1 datasets, SQuAD 2.0 datasets, and SWAG dataset. The overall mechanism can be further improved, generalized, and/or adopted to many other datasets and applications.

In this paper, we aims at optimizing and extending BERT model. Our work aims to propose and test potential improvements to the model, thereby contributing to the ongoing advancements in the field of nature language programming.

3 Related Work

There have been many existing work on improving BERT machanism and/or adopting BERT to more broadly applicable tasks. Among these, several work looked at fine-tuning the BERT model on different datasets such as SMART (Jiang et al., 2020) and Sentence-BERT (Reimers and Gurevych, 2019), and additional pre-training to get richer and more robust embeddings such as the Stanford Natural Language Inference corpus (Bowman et al., 2015) and the Multi-Genre Natural Language Inference (MultiNLI) corpus (Williams et al., 2018). Another natural way to improve BERT model was to further pre-train with target-domain data, for example, Sun et. al. conduced comprehensive numerical examples to looks at fine-tuning BERT on on text classification task and then provided a general solution on BERT fine-tuning (Sun et al., 2019). In Sentence-BERT (SBERT), the similarity between two embeddings was also computed using their cosine similarity to further improve the BERT model by reducing the effort for finding the most similar pair from 65 hours with BERT to about 5 seconds while maintaining the accuracy (Reimers and Gurevych, 2019). Henderson et. al. applied a different loss function such as Multiple Negatives Ranking Loss for natural language response suggestion (Henderson et al., 2017). To address the over-fitting cased by aggressive fine-tuning and improve BERT on unseen data, Jiang et. al. proposed a learning mechanism for robust and efficient to attain better generalization performance through Principled Regularized Optimization and also proposed a class of Bregman proximal point optimization methods (Jiang et al., 2020). Many other work have studied the BERT model by adding multi-task settings (Stickland and Murray, 2019; Bi et al., 2022; Yu et al., 2020). Gao et. al. proposed SimCSE as a contrastive learning framework to work with both labeled and unlabeled and data (Gao et al., 2021).

4 Approach

In order to optimize the BERT model, We started with re-implementing it. We leveraged the starter code and a minimalist implementation of the BERT model (minBERT) in the given project GitHub repository with the following command:

git clone https://github.com/timothydai/minbert-default-final-project.git

Particularly, we completed the implementation of the minBERT Model containing Multi-head Self-attention, the Transformer Layer, and the Adam Optimizer. We also implemented Sentiment Classification with BERT embeddings. After successfully re-implemented the above models, we are delving deeper into the minBERT model to identify areas for potential improvement. Our eventual goal was on enhancing the model's robustness and generalization capabilities, thereby improving its applicability across a wider range of tasks and datasets. We will discuss various approaches that we applied in details below.

4.1 Hyperparameter tuning

While some defaults for various hyperparameters were provided in the repo, these do not necessarily lead to the best results. Therefore, the first approach we have taken was to perform a hyperparameter search to find the best hyperparameters for the minBERT model we implemented. With our completed model using both pretrained and finetuned embeddings, we conducted comprehensive numerical experiments on selected datasets with different hyperparameters. The model was then evaluated based on Accuracy metric on different dataset. We summarize the numerical result in the Section 5.1.

4.2 Multi-task setting

We also embarked on enhancing computational performance across multiple tasks. We started this with the implementation of a classifier pipeline that trains the minBERT implementation to simultaneously perform the sentiment analysis, the paraphrase detection, and the semantic textual task. After evaluate the original pipeline, it is then further optimized by leveraging multi-task setting. Specifically, rather than fine-tuning minBERT on the loss function of individual tasks, we

alternatively make use of multi-task learning to update BERT simultaneously.

As suggested by (Bi et al., 2022), one way of doing this is by adding together the loss functions on the tasks of sentiment analysis, paraphrase detection, and semantic textual together, instead of just using the loss function of sentiment analysis loss function as shown in Equation 1:

$$L_{Total} = L_{SentimentAnalysis} + L_{ParaphraseDetection} + L_{DemanticTextual} \tag{1}$$

However, using multi-task learning is not always effective depending on how the model would be fine-tuned. One challenge we faced was that gradient directions of different tasks might have conflict with each another. To further improve the multi-task setting, we re-implemented the Gradient Surgery method proposed by Yu et. al. (Yu et al., 2020). Specifically, we project the gradient of one particular task $\mathbf{g_i}$ onto the normal plane of the gradient of another conflicting task, $\mathbf{g_j}$ following equation 2:

$$\mathbf{g_i} = \mathbf{g_i} - \frac{\mathbf{g_i} \cdot \mathbf{g_j}}{\|\mathbf{g_j}\|^2} \cdot \mathbf{g_j} \tag{2}$$

The Gradient Surgery method provided a simple and general approach to avoid conflict interference between task gradients and thus leads to substantial gains in efficiency and performance.

The end-to-end multitask fine-tuning model is later evaluated and compared based on the original setting. The result are present in the Section 5.1.

4.3 Optimization on model size and layers

After completing the hyperparameter optimization and multi-task setting, we still identify rooms of improvement. Therefore, we looked at the structure of the model to see if we can try increasing layer size or the number of layers.

We did this in three aspects. First, we added a bi-LSTM layer after minBERT model as a common layer to improve the performance of the model on all tasks. Additionally, we also added linear layers for each tasks as additional task specific layer. These two optimization did improve the overall score on the multi-task pipeline, but we also start to see some over-fitting when compares to the dev set and test set. Therefore, we added additional dropout layers for each tasks to improved score on validation and test set.

After these optimization of bi-LSTM, task specific linear layers and the dropout layers, we conduct numerical experiment and the result is present in Section 5.1.

5 Experiments

In this section we discuss the experiments in details from the following four aspects.

5.1 Data

We leveraged these datasets:

Stanford Sentiment Treebank (SST) dataset: The Stanford Sentiment Treebank dataset are extracted from movie reviews that contains 11,855 single sentences. The dataset was parsed with the Stanford parser and includes a total of 215,154 unique phrases. The dataset is annotated by 3 human judges. For each phrase, a label of negative, somewhat negative, neutral, somewhat positive, or positive is provided. We will utilize BERT embeddings to predict these sentiment classification labels. For the

SST dataset we have the following splits:
• train (8,544 examples)
• dev (1,101 examples)
• test (2,210 examples)

CFIMDB dataset: The CFIMDB dataset consists of 2,434 highly polar movie reviews in which review a binary label of negative or positive has been provided. We will also utilize BERT embeddings to predict these sentiment classifications. For the CFIMDB dataset we have the following splits:
• train (1,701 examples)
• dev (245 examples)
• test (488 examples)

For Sentiment analysis task, since the label of SST is from 0 to 4, we thus relabeled CFIMDB so that the original positive (y = 1) are now marked as 4.

5.2 Evaluation method

The different versions of the minBERT model we implemented and optimized were being evaluated against the baseline methods with the default hyperparameters and setting. This evaluation utilized the metrics derived from our experimental results and compare them with the scores for the original BERT model. The evaluation has been performed in terms by averaging accuracy across the three downstream tasks using methods such as similarity, accuracy, and/or GLUE score.

5.3 Experimental details

After re-implementing the BERT model, we conducted numerical experiments on our primary task, sentiment analysis, with the original default hyper-parameters as baseline. We then aimed at optimizing the model with various different hyper-parameters that were shown in the Results following method in Section 4.1.

Using both pre-trained and fine-tuned embedding on the SST and the CFIMDB datasets, we also evaluated the original classifier pipeline that trains the BERT implementation to simultaneously perform sentiment analysis, paraphrase detection, and semantic textual similarity that we implemented. As described in Section 4.2 as part of the extension, we also optimized the multi-classifier leveraging multi-task setting and combined the loss function on all three tasks during the training process of both pretraining and finetuning on all three datasets. We also implemented the Gradient Surgery method to address the problem of conflicting interference between task gradients for substantial gains in efficiency and performance. The comparison before and after this optimization were shown in the next section.

Lastly, we also followed the method in Section 4.3 to add additional optimization on model structure. Specifically, we added a general bi-LSTM layer, task specific linear layers and the dropout layers. The performance of the model before and after these optimizations are present in the next section.

All the experiments in the paper are conducted on Google Cloud Platform using Compute Engine API Service on 1 x NVIDIA T4 GPU VM in Zone us-west3-b with Image pytorch-2-0-gpu-v20231105-debian-11-py310. We thank Google for providing the $300 credits and the course operator for providing the additional $50 credits.

5.4 Results

As discussed in the Section 4.1, the first optimization we conducted was hyperparameter tuning. Table 1 illustrates the accuracy of the BERT model with the default setting, along with the new

hyperparameters being evaluated sequentially and one at a time. The accuracy on the dev set were given for each experiment when only one hyperparameter was changed. As we could see from the table, our current approaches of parameter tuning does not change that much when it compares to the original setting. One interesting findings we observed is that by flipping the learning rate of pretraining and finetuning, the accuracy dropped significantly. This indicates the hypothesis that for this particular case the learning rate in pretraining needs to be larger then the one in finetuning.

Dev Accuracy	**Pretrain-SST**	**Finetune-SST**	**Pretrain-CFIMDB**	**Finetune-CFIMDB**
Default	0.402	0.525	0.784	0.971
lr=1.00E-05,1.00E-03	0.316	0.262	0.576	0.502
P dropout=0.1	NA	0.52	NA	0.976
P dropout=0.5	NA	0.523	NA	0.967
batch=16	NA	0.52	NA	0.971
batch=64	NA	0.515	NA	0.959
seed=21111	NA	0.517	NA	0.955
seed=100	NA	0.528	NA	0.963

Table 1: Sentiment Analysis on default and different hyper-parameters

As discussed in Section 4.2, we also optimized the multi-classifier with a multi-task setting by implementing Combined Loss and Gradient Surgery. Table 2 compares the performance of the model before and after the optimization. Particularly, when the multi-classifier was only trained on sentiment analysis (SST) loss and its dataset, the model performance was way worse for paraphrase detection task (PD), semantic textual similarity (STS) task, and overall. On one hand, with the Combined Loss function and training on all three datasets, though the performance on sentiment analysis dropped only slightly, the performance for the other two tasks and overall increased significantly. This indicates that the Combined Loss method enabled the minBERT model to learn on top of all three tasks, so the model is trained towards the better performance of the three tasks overall, instead of just the single task of sentiment analysis. Furthermore on the other hand, after the Gradient Surgery is implemented, the model improved further an all three tasks, especially on paraphrase detection task and semantic textual similarity task. By alleviating the potential problem of conflicting interference between task gradients, the model achieved significantly towards the performance in multi-task setting.

Model	**Original**	**Combined Loss**	**Combined Loss w. Gradient Surgery**
SST dev Acc.	0.532	0.520	0.571
PD dev Acc.	0.385	0.625	0.826
STS dev Cor.	0.130	0.557	0.871
Overall dev Score	0.494	0.641	0.759

Table 2: Multi-classifier before and after the multi-task optimization

Lastly as described in Section 4.3, we also added additional optimization on model structure by adding a general bi-LSTM layer after the minBert model, after than we also added task specific linear layers and the dropout layers. The performance of the model before and after each optimization steps are shown in Table 3. As illustrated below, the improvement from bi-LSTM on the Overall dev score is very minimum, and even with the task specific linear layer and dropout later the improvement was also very small and it looks like only sentiment analysis was improved and the overall score even dropped. This indicates that he beneficial impact from both a more complicated learning mechanism led by the bi-LSTM and task specific linear layers, and also the regularization induced by the dropout layer to address overfitting was not that helpful. This proves the already near-optimal performance of BERT mechanism as claimed in the original paper (Devlin et al., 2019).

Model	Original	With bi-LSTM	With bi-LSTM and task specific
SST dev Acc.	0.517	0.507	0.521
PD dev Acc.	0.826	0.827	0.817
STS dev Cor.	0.871	0.872	0.871
Overall dev Score	0.759	0.757	0.758

Table 3: minBERT performance before and after the model structure optimization

In the next section we will continue to discuss the result and provide some insights and intuition behind the model performance, and the limitation and potential future work.

6 Analysis

As indicated in the previous section, we aimed at improving minBERT and our focus was mainly in three aspects: hyperparameter tuning, optimization on multi-task setting, and optimization on model size and layers.

While hyperparameter tuning does not help that much, the original hyperparameters seem to be already optimized by early studies on BERT. //

Our exciting findings appeared to be mostly in multi-task setting. As illustrated in Section 5.4 and particularly in Table 2, the performance on paraphrase detection task and semantic textual similarity task improved significantly just by adding combined loss function. Particularly, the dev accuracy on paraphrase detection task almost doubled and the dev correlation on semantic textual similarity is now four times compared to the original setting. Though the model is training slightly less towards the optimal model on sentiment analysis task only, the overall dev score increased by about 30 percent. Additionally with Gradient Surgery method, the overall dev score and all other matrices improved even more significantly, indicating the problem of conflicting interface of task gradients was one of the main challenge to be solved. With Gradient Surgery taking projections of gradient on one with each other, the overall score increased from 0.641 to 0.759, with a 18.41 percentage improvement. Moreover, the performance on paraphrase detection and semantic textual similarity improves even more obvious. These interesting results calls for even more studies in the multi-task setting on BERT in the future and particularly we would like to see the extension of BERT on even more tasks.

Though the improvement on optimization on model structure are very slight, the additional bi-LSTM layer, and task specific linear layers and dropout layers proves the already near-optimal and efficiency of the original BERT model. Furthermore, this approaches also makes the model even more complicated and thus more computational expense was required. One interesting findings we observed during the process is when we training the model with default hyperparameter settings, our GPC instance returned low memory errors. The training process can only be done when we change our batch size from 8 to 4 and even after that, training in the optimized model takes more time than before. This is not a good sign to us seems the original goal of BERT model is to provide a general yet efficient mechanism and making the model more complicated and adding more task specific layer seems to be defeating the purpose. This really brings the interesting and open-ended question on how to trade-off the model performance with efficiency, and yet more studies are expected in this particular topics.

On relating to the model efficiency, one limitations on our study is that we only considered the model performance on its final outputs as the only evaluation method. Future research can potentially add running time as another key indicator when compares to overall performance of the model in terms of both solution quality and computational expense.

7 Conclusion

In this paper, we re-implemented the minBERT model with multi-head self-attention and transformer layers. Using pre-trained parameters loaded, we performed evaluation on both sentiment analysis only and multi-task pipeline. We also optimized various aspects of the BERT model targeting at higher computational performance on multiple tasks. Aiming at improving the model robustness and generalization, we looked at the improvement of minBERT model specifically on hyperparameter tuning, optimization on multi-task setting, and optimization on model size and layers. Though our contribution to the model improvement was mainly achieved with adding combined loss function and the application of Gradient Surgery, we also discussed the corresponding problem on computational expenses and efficiency. Though more research could be done particularly at the direction of multi-task setting and the trade off between solution quality and running time, the BERT mechanism has the potential to achieve both powerful performance, efficiency and generality on even more tasks in nature language processing.

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 of the Association for Computational Linguistics: ACL 2022*, pages 2663–2669, Dublin, Ireland. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

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.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun hsuan Sung, Laszlo Lukacs, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2177–2190, Online. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

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, Hong Kong, China. Association for Computational Linguistics.

Asa Cooper Stickland and Iain Murray. 2019. BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5986–5995. PMLR.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings*, page 194–206, Berlin, Heidelberg. Springer-Verlag.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 5824–5836. Curran Associates, Inc.