Exploring LoRA Adaptation of minBERT Model on Downstream NLP Tasks

Stanford CS224N Default Project

James Joseph Hennessy, Suxi Li
Department of Computer Science
Stanford University
`jjhenn@stanford.edu, suxi2024@stanford.edu`

Mentor: Josh Singh

Abstract

Foundation Models pre-trained on general domain data may not perform optimally on specific tasks. Adaptation and fine-tuning are two major ways to customize pre-trained models with task specific data. This project investigates the enhancement of model performance through LoRA adaptation on a minBERT model for several downstream NLP tasks including sentiment classifications, paraphrase detection, and semantic textual similarity without updating the pre-trained model parameters. We implemented two architectures: one common LoRA adaptation layer for three tasks as well as separate LoRA adaptation layer for each task individually. Our findings suggests the inclusion of LoRA presents potential benefits. However, it does not constitute a universal solution for multitask learning challenges. And task-specific LoRA configuration can achieve better overall performance.

1 Introduction

As large language models gain popularity, businesses across various industries are interested in harnessing their significant potential. Fine-tuning and adaptation are the two primary methods for leveraging large-scale, pre-trained language models, often referred to as Foundation Models, for specific downstream applications. A significant drawback of fine-tuning is that the fine-tuned model retains the same number of parameters as the original model. As Foundation Models grow increasingly larger, the cost of fine-tuning models for each task rapidly becomes prohibitive. Adaptation addresses this challenge by updating only a subset of parameters for new tasks. Proposed by (Hu et al., 2021), LoRA is a low-rank adaptation method that excels in both storage and computation efficiency. By keeping the pre-trained weights in the pre-trained model frozen, LoRA introduces and trains pairs of rank decomposition matrices. These newly learned weights from additional task specific data can then be integrated with the pre-trained weights during inference without introducing higher inference latency.

The primary goal of this project is to implement the minBERT model and improve its performance through LoRA adapatation for the downstream tasks of sentiment analysis, paraphrase detection, and semantic similarity without updating the pretrained model parameters. Specifically, this project will investigate the model performance on the downstream tasks simultaneously. We will assess the outcomes from the baseline, fine-tuned, and low-rank adapted minBERT models.

2 Related Work

2.1 Transformer and BERT

Transformer is a deep learning sequence-to-sequence model architecture that use the innovative self-attention mechanism. Unlike its predecessors, such as RNNs and LSTMs, it does not require data to be processed in order. The Transformer model, introduced in the paper "Attention is All You Need" by Vaswani et al. (2017), relies heavily on the self-attention mechanism to draw global dependencies between input and output, which allows for the model to learn long-distance dependency and enables computation parallelization that speed up training significantly. Since then, Transformer-based language models have dominated the field of NLP, achieving the state-of-art in many tasks. BERT (Devlin et al., 2018) is a large transformer language model pre-trained on a large amount of text. It is designed to pre-train deep bidirectional representations by jointly conditioning on both left and right context in all layers, which enables the model to capture a rich understanding of language context and word relationships. The innovation of BERT lies in its ability to pre-train on a large amount of text and then fine-tune for specific tasks with minimal task-specific architecture modifications. This approach has led to state-of-the-art performances on a wide range of NLP tasks, including question answering, named entity recognition, and sentiment analysis.

2.2 Fine-Tuning and Parameter-Efficient Adaptation

There are a few different strategies for leveraging pre-trained large language models and applying it to a downstream task with more specialized requirements. Fine-tuning is a popular approach that involves continuing the training process of the pre-trained model on a much smaller, task-specific dataset (Devlin et al., 2019). During fine-tuning, the parameters of the entire model can be updated to better perform the target task, such as sentiment analysis, question answering, or document classification. Fine-tuning a highly efficient and effective method. However, as the size of pre-trained models increases, exemplified by GPT-3's 175 billion parameters, performing fine-tuning in the usual way becomes challenging due to large checkpoint it produces and the high hardware barrier to entry since it has the same memory footprint as pre-training.

Parameter efficient adaptation helps to address this challenge by minimizing the number of parameters that need to be updated while fine-tuning the model on a specific task or dataset. This approach is beneficial when computational resources are limited or when the task-specific dataset is relatively small. Various adaptation approaches were proposed. For example, prefix-embedding tuning (Li and Liang, 2021) involves inserting special trainable tokens into the input sequence. They found that by learning only 0.1% of the parameters, prefix-tuning obtains comparable performance. Adapter tuning (Houlsby et al., 2019) introduces adapter layers between the self-attention layer and the subsequent residual connection, offering a modular way to customize a model. Both LoRA and adapter tuning are recognized for their parameter efficiency in model adaptation. However, a limitation of adapter tuning is the introduction of inference latency due to the added adapter layers.

2.3 Low-Rank Structure

Low-rank structure is very common in large and high-rank matrices. Those heavily over-parametrized neural networks usually can be closely approximated with fewer parameters without significant loss in performance (Oymak et al., 2019). The basic idea is to decompose a large, potentially high-rank matrix into the product of two or more smaller matrices, where the inner dimension is the rank of the approximation. This rank is typically much smaller than the original dimensions of the matrix, leading to a significant reduction in the number of parameters.

3 Approach

3.1 Baseline

The baseline for the sentiment classification task with the vanilla minBERT model is provided in the handout. Accuracy is used to evaluate the model performance for sentiment analysis and paraphrase detection on the SST/CFMIBD and Quora datasets, respectively for each task. For semantic textual similarity, we use the Pearson score evaluated on the SemEval STS dataset.

3.2 LoRA Adapatation

LoRA updates weights on a low "intrinsic rank" during adaptation. For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d \times k}$, LoRA constrains its update to the weights with a low-rank decomposition $W_0 + \Delta W = W_0 + BA$, where $B \in \mathbb{R}^{d \times r}$, $A \in \mathbb{R}^{r \times k}$, and the rank $r \ll \min(d, k)$. During training, only A and B will be updated, while W_0 is frozen and does not receive gradient updates. For $h = W_0 x$, the modified forward pass yields:

$$h = W_0 x + \Delta W x = W_0 x + BAx \tag{1}$$

3.3 Task Specific Multi-LoRA Architecture

We adapted the pre-trained BERT model using LoRA to simultaneously perform three tasks: sentiment analysis, paraphrase detection, and semantic textual similarity. Given LoRA's performance sensitivity to specific tasks and datasets, we further tailored our implementation with separate LoRA adaptations for each task to optimize model performance.

4 Experiments

4.1 Data

Four major datasets will be used in this project. For sentiment analysis task, we will use Stanford Sentiment Treebank (SST) and CFIMDB datasets. The SST dataset consists of 11,855 single sentences extracted from movie reviews. The dataset was parsed with the Stanford parser and includes a total of 215,154 unique phrases from those parse trees. Each phrase has a label of negative, somewhat negative, neutral, somewhat positive, or positive. The CFIMDB dataset consists of 2,434 highly polar movie reviews. Each movie review has a binary label of negative or positive. For paraphrase detection task, we will use a subset of the Quora dataset that consists of 400,000 question pairs with labels indicating whether instances are paraphrases of one another. For semantic textual similarity task, we will use SemEval STS Benchmark dataset that consists of 8,628 different sentence pairs of varying similarity on a scale from 0 (unrelated) to 5 (equivalent meaning).

4.2 Evaluation method

For sentiment analysis and paraphrase detection, we will use accuracy to evaluate our model performance. For semantic textual similarity, the output is a value between 0 and 5. We will use the Pearson correlation score to evaluate our model performance. We also explored establishing a baseline performance for our model by using a pretrained model, such as DistilBERT, to obtain sentence embeddings and calculate the cosine similarity between them. We found the resulting cosine similarities to be very high, typically above 0.9, even for two seemingly different sentences. After careful evaluation and discussions with the teaching assistant, we decided to use our model's performance prior to any extensions as the evaluation baseline.

4.3 Experimental details

Our experiments are methodically structured to assess the efficacy of multitask learning within a BERT-based framework, followed by the integration and evaluation of Low-Rank Adaptation (LoRA). The deliberate and incremental approach enabled a clear understanding of the impact of each component and configuration within the multitask BERT model. We outline our procedural approach for these experiments below.

Baseline Multitask Configuration: The baseline setup involves utilizing a BERT model with frozen pretrained parameters, augmented with three task-specific layers dedicated to sentiment classification, paraphrase detection, and semantic textual similarity. The model operates under the multitask paradigm, with simultaneous updates across all task-specific layers.

Fine-tuning Procedure: Building upon the baseline, we proceeded to fine-tune the pretrained BERT parameters using a learning rate of $1e-5$ across 10 epochs of training. This step was executed with the aim to understand the impacts of parameter adjustments on the model's task-specific performance. The task-specific layers were also updated by the finetuning process.

LoRA Integration Process: The experimental trajectory then advanced to the integration of a common LoRA layer across all tasks. The LoRA layer was incorporated while keeping the pretrained BERT parameters frozen, to discern the individual contribution of the LoRA mechanism to the multitask learning process.

Variations in LoRA Configurations: To meticulously evaluate the influence of the LoRA component, we explored various LoRA configurations. This involved altering the rank parameter and adjusting the alpha values and dropout rates. These experiments are key to pinpointing the LoRA configuration that harmonizes best with the multitask framework.

Task-Specific LoRA Adaptations: Following the common LoRA layer experiments, we tailored the LoRA adaptation to each task independently. This customization is indicative of our pursuit to optimize the interaction between LoRA and the idiosyncrasies of each task within the multitask setting.

Training Time and Computational Resources: Our LoRA experiments were conducted for 1 epoch each to ensure comparability across configurations. We also noticed a plateau in the loss curve after even 1 epoch, which persisted in following epochs. To avoid unnecessary training time after understanding this, we continued using 1 epoch in following experiments. The training was performed on 1 NVIDIA T4 GPU on a Google Cloud VM, with an average training time of 12 minutes per epoch.

4.4 Experimental Results

Our experimental journey encompassed an array of model configurations, each evaluated on their performance across our three distinct NLP tasks. Detailed results and our observations are reported below.

4.4.1 Multitask Baseline and Fine-Tuning Performance

In our experiment, we first established a multitask baseline by leveraging a pre-trained BERT model with its parameters frozen, while simultaneously updating three task-specific layers. As reported in Table 1, the pre-trained model achieves a decent accuracy on three tasks, especially for paraphrase detection mainly due to the much larger training data set. Subsequently, we conducted a fine-tuning experiment where, unlike the baseline, the BERT pre-trained parameters were allowed to update during training alongside the three task-specific layers.

Table 1: Multitask Baseline and Fine-tuning Performance

Metric	**Sentiment Accuracy**	**Paraphrase Accuracy**	**Similarity Pearson**
Training Data Size	8,544	141,506	6,041
Multitask Baseline	0.447	0.692	0.277
Fine-Tuning	0.405	0.761	0.168

The results from this phase of experiments suggest that a larger training dataset for a specific task significantly enhances the task-specific accuracy of the model. This observation aligns with the general understanding that more data can lead to better model performance, given that the model has enough capacity to learn from the increased data. The results also indicate potential conflicts among the tasks when trained simultaneously. This is evidenced by the increased accuracy on paraphrase detection but deteriorated performance on other two tasks. This phenomenon, known as negative transfer, suggests that the tasks may compete for model capacity, leading to compromised performance on some tasks.

4.4.2 LoRA Configurations

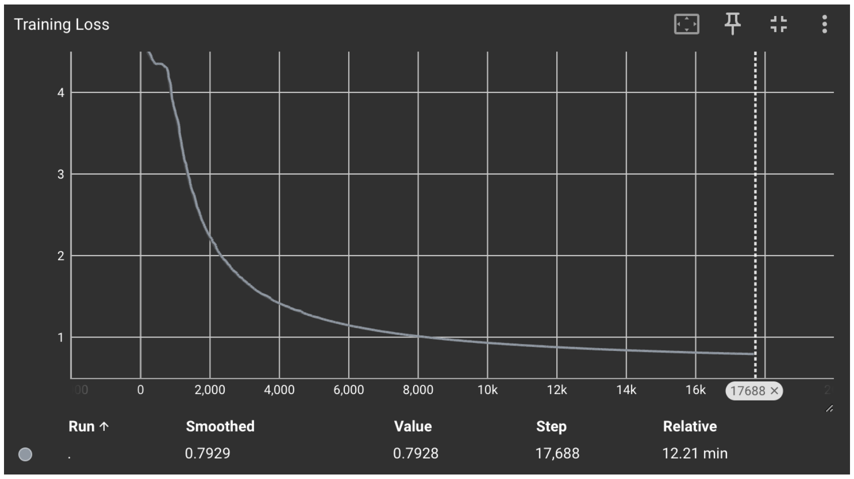
In this comprehensive experiment, we investigated the effects of different LoRA configurations on the pre-trained BERT model's performance across these three pre-defined tasks. The primary objective was to explore how variations in the rank (r) parameter influence the model's multitasking capabilities. While additional experiments were conducted to assess the impact of other LoRA parameters such as the alpha value, dropout rate, and target modules, it was found that the model's performance was relatively less affected by these factors. For the sake of brevity, we focus exclusively on the outcomes associated with varying the rank parameter. As a general rule derived from empirical success in research, it is suggested to use an alpha value twice as large as the rank value. This heuristic ensures controlled yet significant updates to pre-trained weights. LoRA adapatation was applied to both value and query matrices in all the experiments.

Table 2: LoRA Configurations and Model Performance

LoRA	r	alpha	dropout	Sentiment Accuracy	Paraphrase Accuracy	Similarity Pearson
Y	2	16	0.1	0.451	0.681	0.278
Y	4	16	0.1	0.421	0.681	0.281
Y	8	16	0.1	0.423	0.677	0.276
Y	16	16	0.1	0.428	0.654	0.309
Y	32	16	0.1	0.439	0.679	0.287
Y	32	32	0.1	0.439	0.679	0.287
Y	64	32	0.1	0.420	0.686	0.272

The results indicate a nuanced relationship between the LoRA configuration and the model's performance across different tasks as shown in table 2. Notably, increasing the rank (r) beyond 2 generally led to improvements in sentiment classification accuracy up to a certain point (r=32). However, the best sentiment classification accuracy was achieved when r=2, suggesting a nonlinear relationship between rank and model performance. The paraphrase detection task showed less sensitivity to changes in rank, with accuracies remaining relatively stable across different configurations. This suggests that the task might be less affected by the low-rank structure introduced by LoRA. For semantic textual similarity correlation, the highest score was achieved with r=16.

We also monitored the training loss closely using the Tensor board. An typical example is shown below with behavior that we observed in most of the experiments.



Run ↑	Smoothed	Value	Step	Relative
.	0.7929	0.7928	17,688	12.21 min

Figure 1: LoRA Adaptation Training Loss

4.4.3 Task-Specific LoRA Adaptation Performance

In this experiment, we explored the efficacy of implementing task-specific LoRA configurations within a multitask learning framework using a pre-trained BERT model. The primary objective was to ascertain whether different tasks within the multitask setup could benefit from a unique LoRA configuration, potentially leading to enhanced overall model performance. The model was trained for one epoch across all tasks, with varying LoRA configurations applied specifically to each task. The configurations varied in terms of the rank (r) and the application of LoRA to the BERT model's Value and/or Query layers, with some configurations also exploring the impact of adjusting the dropout rate.

Table 3: Performance of LoRA Adaptation on Sentiment Classification

	Rank	Dropout	Target Module	Accuracy
Sentiment Classification	4	0.1	Value, Query	0.44
	4	0.1	Value	0.441
	2	0.1	Value, Query	0.452
	2	0.2	Value, Query	0.439

Table 4: Performance of LoRA Adaptation on Paraphrase Detection

	Rank	Dropout	Target Module	Accuracy
Paraphrase Detection	2	0.1	Value, Query	0.701
	2	0.1	Value	0.671
	2	0.05	Value, Query	0.696
	2	0.2	Value, Query	0.702

Table 5: Performance of LoRA Adaptation on Semantic Similarity Pearson Correlation

	Rank	Dropout	Target Module	Pearson
Semantic Similarity Pearson	16	0.1	Value, Query	0.288
	16	0.1	Value	0.283
	32	0.1	Value, Query	0.300
	32	0.05	Value, Query	0.295

The results indicate that each task may have a distinct optimal LoRA configuration. For instance, sentiment classification achieved the highest accuracy with a rank of 2 applied to both Value and Query layers, whereas paraphrase detection peaked at a rank of 2 under the same conditions. Semantic textual similarity showed the best correlation with a rank of 32 applied to both Value and Query layers.

5 Analysis

Upon reviewing our results, we observed that integrating LoRA into our model configurations does not consistently outperform the baseline model across all tasks. While certain configurations with LoRA showed slight improvements in sentiment classification and semantic textual similarity, they did not deliver a clear advantage in paraphrase detection. These mixed outcomes led us to conjecture that the benefit of LoRA might be context-dependent, enhancing performance on certain tasks while being less effective or neutral on others.

Specifically, we noticed that increasing the rank parameter r within LoRA adaptations does not uniformly translate to better performance. This countered our original hypothesis that larger matrices would automatically increase performance by capturing more sophisticated patterns. The optimal rank setting seems to differ across tasks, underlining the necessity for task-specific tuning when applying LoRA in a multitask framework. The complexity added by LoRA, indicated by the alpha parameter, similarly did not correlate directly with improved results, suggesting that there is a nuanced balance between model complexity and task performance.

In our experimentation of LoRA configurations, we specifically scrutinized the interplay between rank (r), alpha, and dropout parameters. This meticulous examination was anchored in the hypothesis that each task's unique characteristics and complexity would resonate differently with the structural adjustments induced by LoRA. For sentiment classification, a task that inherently involves navigating nuanced linguistic expressions and context, we observed that a lower rank ($r = 2$), coupled with a moderate alpha value (16) and a dropout rate of 0.1, yielded the highest accuracy. This configuration suggests that minimal yet precise updates to the model's attention mechanism are pivotal in capturing the subtle distinctions within sentiment-laden text, thereby enhancing classification performance without overwhelming the model with excessive parameter adjustments.

Conversely, for the paraphrase detection task, the stability across varying rank settings underscored a less pronounced sensitivity to the low-rank structure, potentially due to the binary nature of the task that hinges on comparative rather than nuanced understanding. The relative insensitivity to rank variations could imply that paraphrase detection benefits more from the semantic understanding embedded in the original BERT weights than from the adaptability offered by LoRA's low-rank updates.

Semantic textual similarity, characterized by the need to gauge fine-grained semantic nuances, exhibited optimal correlation at a higher rank ($r = 16$), indicating that a more complex low-rank update facilitates a richer representation of semantic relationships. However, the performance dip observed at ranks beyond 16 serves as a testament to the diminishing returns and potential overfitting associated with overly complex adaptations, emphasizing the critical balance between model expressiveness and over-parameterization.

This granular analysis of LoRA configurations revealed the task-dependent optimal tuning of rank, alpha, and dropout parameters, showing the need for balance between maintaining the pre-trained model's generalizability and introducing targeted adaptability. For sentiment classification, a conservative approach with lower rank values proved beneficial, while semantic textual similarity called for a moderately higher complexity via increased rank.

Our quantitative findings led us to understand that while the inclusion of LoRA presents potential benefits, it does not constitute a universal solution for improving performance in multitask learning.

Summary of Key Observations

- Larger training datasets substantially improve task-specific model accuracy
- There is potential conflict among tasks when trained simultaneously
- LoRA integration does not guarantee performance increases relative to the baseline multitask model and is highly dependent on hyperparameter selection
- Task-specific LoRA configurations can optimize performance for individual tasks, but may harm performance of other tasks

6 Conclusion

In this project, we delved into the integration and analysis of LoRA layers within a multitask learning framework based on BERT. Our journey began with establishing a baseline multitask classifier, which we incrementally enhanced by introducing and refining LoRA layers for each specific task. We successfully implemented a custom and functional LoRA framework for linear layers (see the $/lora$ directory in our project folder) and conducted extensive experiments where we tuned the hyperparameters of LoRA layers to attempt achieving better performance.

Each experimental phase built upon the insights gained from the preceding one, enabling a cumulative enhancement in our approach. We ventured to show that alternatives to fine-tuning such as LoRA are indeed viable, but found that true performance gains are highly contingent on the precision of their configurations.

Our work was not without its limitations. Time was a significant constraint, as developing a fully custom LoRA library proved to be a much more extensive undertaking than initially anticipated. As we came to learn while developing this extension, existing codebases to integrate LoRA across an entire pretrained transformer model often span span more than $3,500$ lines of high-complexity code and are maintained by a host of top-tier ML engineers. Our custom implementation allows for adaptation of feedforward linear layer weights, but future work would involve creating these implementations for the Query, Key, Value and Output projection matrices.

For future work, we propose the development of a streamlined version of these extensive libraries. This minified edition would focus solely on the essential components necessary for the implementation of LoRA. Such a focused library would not only facilitate a deeper understanding of LoRA's mechanics but also enhance accessibility for researchers and practitioners wishing to explore this alternative finetuning approach.

Furthermore, a great area for future research would be in better understanding the interactions between LoRA configurations and task-specific performance by, for example, developing automated testing suites to identify optimal hyperparameters. The combinatorial complexity of hyperparameter selection for LoRA explodes for pretrained models with many intermediate Transformer layers and model adapters, so finding a method to do this automatically and at low-cost (i.e. without the need to fully train on each hyperparameter set) would be fascinating and highly useful. We would have certainly loved such a tool during our experimental process.

In conclusion, our project underscored the potential of innovative fine-tuning replacements in the form of LoRA. The critical takeaway is the importance of task-specific configuration sensitivity, which emerges as a cornerstone for successful multitask learning. Despite not having achieved increased performance results, the knowledge we have amassed sets an exciting foundation for future experimentation in building and optimizing Transformer models.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. *arXiv:1902.00751 [cs, stat]*.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Samet Oymak, Zalan Fabian, Mingchen Li, and Mahdi Soltanolkotabi. 2019. Generalization guarantees for neural networks via harnessing the low-rank structure of the jacobian. *arXiv preprint arXiv:1906.05392*.

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*, pages 5998–6008.