Balancing Performance and Computational Efficiency: Exploring Low-Rank Adaptation for Multi-Transferring Learning

Stanford CS224N Default Project

Caroline Santos Marques da Silva
Department of Computer Science
Stanford University
`cm199204@stanford.edu`

Abstract

This work explores the comparative efficacy of full fine-tuning, linear probing, and Parameter-Efficient Fine-Tuning (PEFT) techniques, with a focus on Low-rank adaptation (LoRA), in training models for natural language processing tasks such as sentiment classification, paraphrase detection, and semantic textual similarity. The study introduces MultitaskBERT, a model leveraging the BERT architecture, which contain task specific layers. The results reveal while LoRA is more computational efficient and tends to be more robust in avoiding overfitting problems, its overall performance did not overcome full fine-tuning approach. Additionally, the project also explores the impact of weighing the loss of each task on the model's accuracy, aiming to enhance multitask learning efficiency and effectiveness.

1 Key Information to include

- Mentor: Anirudh Sriram
- External Collaborators (if you have any): None
- Sharing project: No

2 Introduction

Sequence inductive transfer learning is defined by two stages of model training: pretraining and model adaptation. The first one is responsible for learning general representations of inputs. The second stage is the model adaptation where the learning representations are transferred to downstream tasks. Scaling up a pre-trained model is common practice in order to Increase a model's performance in the model adaptation phase. However, there are a set of techniques that could be explored before changing to a bigger model. This project has the intention to explore a set of techniques related to the adaptation phase to increase the performance of a mini-bert model to a multi-task problem. The selected tasks are sentiment analysis, paraphrase detection and semantic textual similarity.

In order to adapt the model to a downstream task, two techniques are normally used: linear probing (last layer update) or full fine tune. Each of these techniques have disadvantages and advantages. The first one is known for saving more computational resources but with the cost of losing accuracy. However, several papers have studied that this fact is not always true and they are explored in the next section.

In this regard, this project has studied the use of Low-rank adaptation (LoRA)(Hu et al., 2021) technique to strike a balance between full fine-tuning and linear probing methods. LoRA has emerged as a common PEFT technique. As full fine tuning technique might lead to catastrophic forgetting (French, 1999) and overfitting (Yosinski et al., 2014), LoRA can be used to overcome these problems as well as save computational resources. Despite the fact that PEFT techniques have been developed for large language models, this project explores LoRA as a method to update different weights of small sized models.

Additionally, a second problem tackled by this project was the combination of the losses of each task during the training. Several papers explored different approaches in this regard, and they are discussed in the next section.

3 Related Work

3.1 Transfer learning

Full fine-tuning requires more computational resources but usually achieves better results because it allows updating the model's understanding of both low-level and high-level features. In contrast, linear probing requires less computational resources but offers less flexibility since only the last layer is adjusted. Consequently, the model may not adapt as well to the target task, particularly if the task is substantially different from the original training task. In this section, several papers that explore different techniques to improve the model adaptation to a downstream task are discussed. Either exploring different conditions where each of these techniques are a better choice or combine the two approaches to generate a new framework.

The concept of transfer learning in these models relies on the foundational assumption that linear probing can be effective for adapting to downstream tasks. This method is grounded in the belief that the representations learned by the pre-trained model are both rich and general enough to be suitable for a variety of tasks, requiring only minor, task-specific adjustments at the model's end.

In practice, the effectiveness of linear probing can vary significantly based on the similarity between the pre-training and target tasks (Mou et al., 2016). Discrepancies in data distribution, task complexity, and required outputs can necessitate more substantial modifications beyond simple adjustments, challenging the adaptability of pre-trained models solely through linear probing.

In this regard, full fine-tuning technique is well known for leading better accuracy than linear probing in-distribution (Kornblith et al., 2019). However, Kumar et al. (2022) has shown that full fine-tuning can achieve worse accuracy than linear probing out-of-distribution when the pretrained features are good and the distribution shift is large. Additionally, full fine-tuning might lead to overfitting and catastrophic forgetting.

This underscores the importance of exploring additional fine-tuning and adaptation techniques to ensure that transfer learning can be successfully applied across a broader spectrum of tasks and domains.

Based on the premise that different layers encapsulate distinct types of information (Yosinski et al., 2014), Howard and Ruder (2018) propose a fine-tuning technique that explores the impact of fine-tuning model layers with varying learning rates.They propose that these layers should have individualized fine-tuning settings. Additionally, they propose a gradual unfreezing approach during the fine-tuning process to avoid catastrophic forgetting. Their findings demonstrate that progressively unfreezing the layers from top to bottom, combined with the application of a slanted triangular learning rate, improves performance in text classification tasks. This method is referred to as discriminative fine-tuning.

Peters et al. (2019) asked how to best adapt the pretrained model to a given target task. Their results across diverse NLP tasks with two state-of-the-art models show that the relative performance of fine-tuning vs. feature extraction depends on the similarity of the pretraining and target tasks.

Peters et al. (2019) also investigate how information evolves across different layers during fine-tuning in machine learning models, using two evaluation techniques: diagnostic classifier performance and mutual information (MI) between representations and labels. Diagnostic classifiers show that fine-tuning generally enhances performance across all layers, with single sentence tasks like sentiment classification seeing a gradual increase in performance towards the last layers, whereas sentence pair tasks STS exhibit a plateau in performance beyond the fourth layer. This suggests that for sentence pair tasks, crucial information is not predominantly in the top layers, highlighting the effectiveness of fine-tuning. Regarding MI, they compute the MI between hidden activations and random labels and random representations and random labels. They have found that MI for fine-tuned representations rises gradually through the intermediate and last layers for the sentence pair task, while for the single sentence classification tasks, the MI rises sharply in the last layers.

Given the advantage and disadvantage of linear probing and full fine-tuning, we propose to use LoRA technique to adapt the model to downstream task as well as linear probing and full fine-tuning. Our assumption is that LoRA can improve generalization performance by constraining the model complexity, which prevent overfitting, especially in scenarios with limited training data Hu et al. (2021). Additionally, LoRA significantly lowers computational complexity and memory usage.

3.2 Multi-Task Learning

The previous section explored the concept of transfer learning for model adaptation to a downstream task. However, this project is focused on multi-task learning. Whether a single task or multi-task problem, the training process for the model remains consistent, involving two main phases: pretraining and adaptation. As the focus of this project is the model adaptation, this section is dedicated to explore model adaptation within a multi-task learning paradigm.

Humans possess the ability to continually learn new tasks without compromising the performance of previously learned tasks. Ideally, the expectation for deep learning models remains the same. Additionally, the learning of multiple tasks jointly helps to transfer the knowledge contained in a task to other tasks, with the hope of improving the generalization performance of all the tasks.

In the field of multi-task learning in machine learning several works explored different approach such as which tasks should be learned together and how to combine the loss values.

Huq and Pervin (2022) argue that to train a model in multi-task learning settings we need to sum the loss values from different tasks and address specific weights to the losses that puts more emphasis on difficult tasks. Basically, their approach is to address automatically a weight to each task specific loss by taking the sum of the loss values for each task and use it to figure out the ratio of how much a single task's loss value contributes to the total loss.

A different approach was studied by Kendall et al. (2018). Their approach relies on weighs multiple loss functions by considering the uncertainty of each task. The approach of Huq and Pervin (2022) has shown better results than the approach of Kendall et al. (2018).

In this project, the losses function are not sum-up because the idea of sequential learning was used. However, we explore with similar technique of Kendall et al. (2018) a method to weigh the losses.

4 Approach

4.1 Goal

The primary objective of this project is to identify the optimal model training configuration that achieves superior performance across three tasks, sentiment analysis, paraphrasing, and similarity detection, by updating a fewer number of parameters than the full fine-tuning model training configuration ('finetune') and more parameters than the linear probing model training configuration ('pretrain').

The secondary objective is to study the effect of weighing the loss of each task on the model's accuracy.

4.2 Model Architecture

In this project, the MultitaskBERT model has been developed leveraging the BERT architecture to perform three distinct natural language processing tasks: sentiment classification, paraphrase detection, and semantic textual similarity.

MultitaskBERT integrates on the configuration to either retain the pre-trained parameters, fine-tune them, or apply LoRA (Low-Rank Adaptation) adjustments for task-specific enhancements.

The model employs a conditional parameter adjustment strategy where the training behavior of BERT parameters is governed by the specified configuration. In the 'pretrain' mode, BERT's parameters are frozen, making them non-trainable and thus the model operates in a feature extraction manner.

Conversely, the 'finetune' mode allows all parameters to be updated during training, leveraging the full capacity of BERT for adaptation to the tasks at hand. The 'lora' option introduces a targeted modification of certain layers within BERT using Low-Rank Adaptation, specifically tuning specific modules (attention layers and/or point-wise feed forward layer) to enhance model performance on the tasks without extensive retraining of the entire model.

For task-specific adaptations, MultitaskBERT incorporates additional layers on top of the shared BERT encoder. For sentiment classification, it adds a dropout layer followed by a linear classification layer tailored to predict five levels of sentiment. Paraphrase detection and semantic textual similarity tasks are addressed through a pipeline that first concatenates the input sequences and the result is sent to the shared BERT encoder. After that, a similar approach to the one taken for sentiment classification is applied to the similarity and paraphrase tasks: a dropout layer for regularization and, finally, a classification layer that outputs the task-specific predictions are added.

The forward method of MultitaskBERT is designed to process a batch of sentences, extracting pooled embeddings from the BERT model, which serve as input to the task-specific layers.

4.3 Baseline

The linear probing multitask model training configuration was chosen as the baseline. The model candidates utilized various LoRA model training configurations and two weighing losses configuration. Additionally, all model candidates will be compared with the fully fine-tuned model.

5 Experiments

In order to test which training techniques could achieve the best performance, the following experiments were done:

Model	Description
Baseline LP	Linear Probing (LP)
FT	Full Fine-Tuning
Candidate 1	Full Fine-Tuning and weighted losses(wl).
Candidate 2	Linear Probing and weighted losses (wl).
Candidate 3	LoRA, wl, $r = 4$ and $lr = 1e^{-3}$ and weighted losses (wl)
Candidate 4	LoRA, $r = 4$ and $lr = 1e^{-3}$
Candidate 5	LoRA, $r = 4$ and $lr = 1e^{-5}$
Candidate 6	LoRA, $r = 8$ and $lr = 1e^{-5}$

Table 1: Model Descriptions

The hyperparameter of the loss were: $10L_{sts}, 4L_{sst}$ and $1L_{paraphrase}$. The criterion for these parameters were based on the uncertanty of each task as the tasks sst and sts have more possibles values than paraphrase task (binary classification). In addition, the weight were also chosen to approximate the values of the loss during the training.

5.1 Data

The dataset used were Quora, SemEval STS Benchmark and Stanford Sentiment Treebank (SST)

5.2 Evaluation method

Sentiment and paraphrase tasks were evaluated by computing the accuracy and Similarity task was evaluated by the person correlation

The behavior of the loss during the training was analyze as well and consider as evaluation method.

5.3 Experimental details

One epoch in the training process is defined as follow:

- For each task specific dataset, the batches are generated and their are used to train the model by updating the shared weights and the task specific layer according to the task of each batch.
- For each batch, a forward pass through the model is performed to compute the predictions. Depending on the task (sentiment analysis, paraphrase detection, semantic textual similarity), the model uses different pathways (i.e., each task has its own linear layers and dropout layer) to process the input data and generate the output.
- Once the predictions are obtained, the loss function specific to the task is applied to the share weights and to the task specific layers. The choice of loss function depends on the nature of the task. Cross-entropy for sentiment analysis, binary cross-entropy for paraphrase detection, mean square error for semantic textual similarity.

All candidates and baseline were trained with: : 6.604 examples from STS, 8.544 from SST and 141.506 from Quora dataset.

5.4 Results

This section presents the results of all experiments define at the table 1

The Figure 1 depicts the loss behavior during the training and The Figures at 2 depicts the accuracy for the training set and dev set. By analyzing both results we can summarize:

- Both candidate 2 and the baseline LP exhibited a flat loss curve. According to Figure 2, both the training and development accuracies were unsatisfactory.
- For candidate 1 (Full fine-tuning), the approach to weighting the loss demonstrated an improved descent behavior. However, this did not translate into enhanced accuracy, as illustrated in Figure 2. In fact, the model exhibits overfitting.
- Candidate 3 (Lora) displayed divergent behavior at epoch 7, attributable to the implementation of weighted losses.
- Candidates using Lora with a learning rate of 10^5 showed a more favorable loss descent pattern compared to the Full Tuning (FT) model. Nonetheless, their overall accuracy did not surpass that of the FT model. However, as Figure 3 indicates, Lora models tend to avoid overfitting, unlike the FT model.

6 Analysis

This project's findings indicate that adjusting the loss weight did not enhance task accuracy, particularly for the sentiment classification task, which proved to be the most challenging. The techniques employed in this study failed to achieve an accuracy exceeding 0.54 on the development dataset for this task. To improve accuracy, alternative strategies could be considered, such as diversifying the data or expanding the model size specifically, increasing the number of attention layers to capture a broader array of feature relationships. The loss behavior observed suggests that the model may be memorizing rather than learning from the training data, a tendency that appears to intensify with an increase in the proportion of learnable parameters.

However, the results obtained with LoRA align with existing literature regarding its efficacy in preventing overfitting.

As for the process of manually adjusting loss weights based on task uncertainty, it proved insufficient. An interesting follow-up experiment could involve applying the technique suggested by Huq and Pervin (2022) to study whether sequence learning could benefit from their approach as well.

7 Conclusion

- The model exhibits descent values of accuracy in semantic textual similarity (STS) and paraphrase identification tasks (more than 0.8). However, its performance is poorly in the sentiment classification task (SST) (less than 0.54).

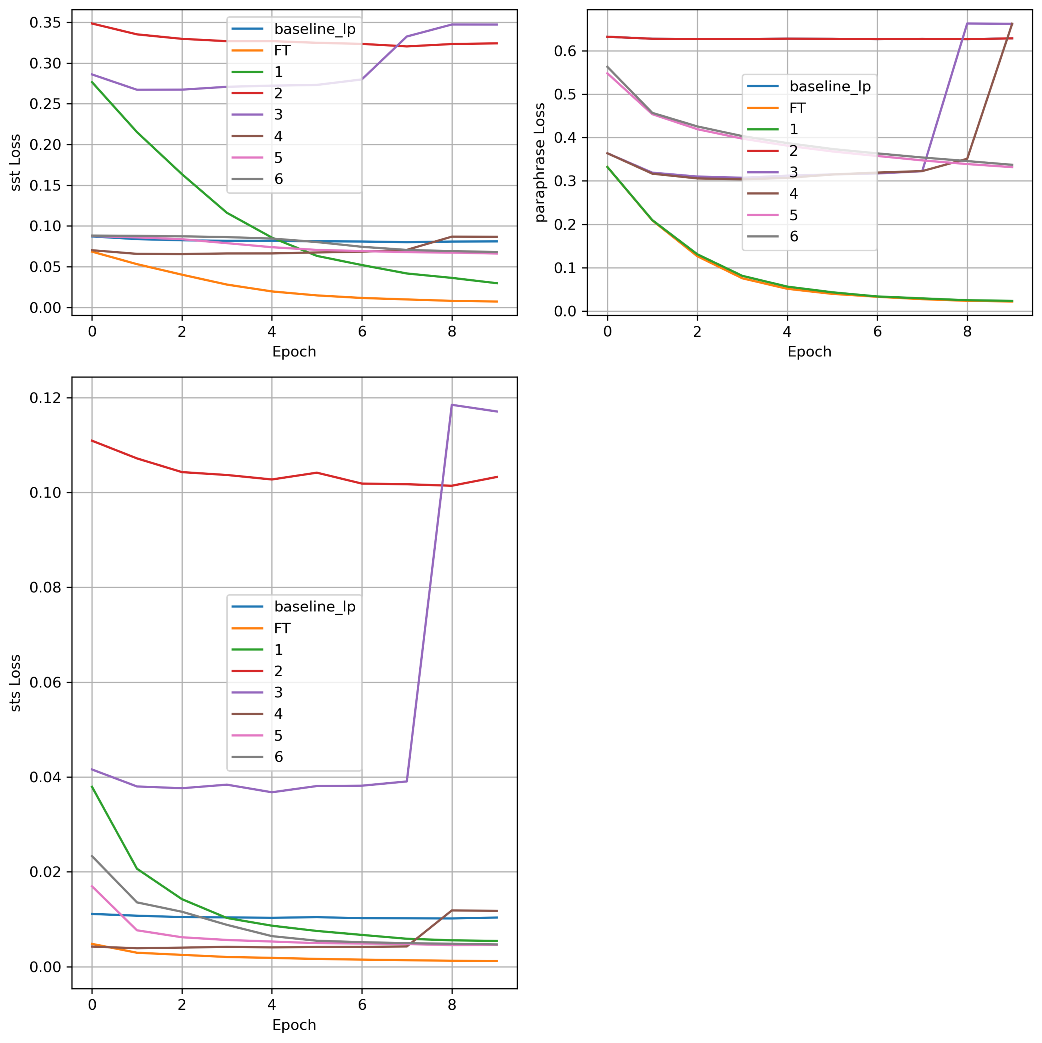
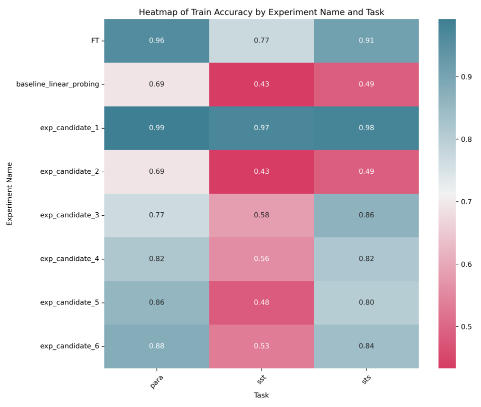
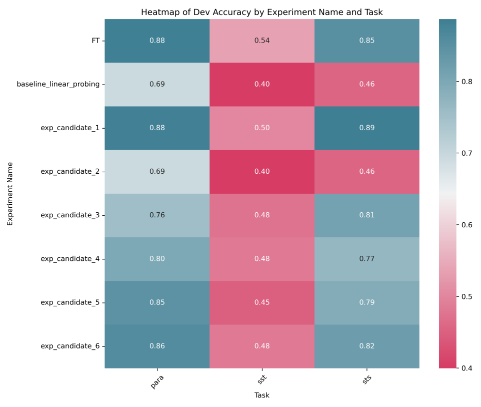


Figure 1: Loss by epoch for the model candidates.

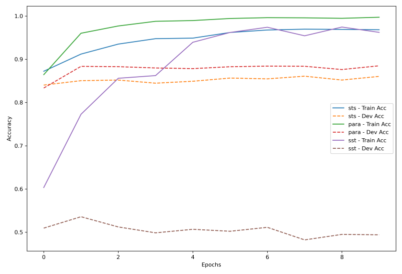


(a) Train set

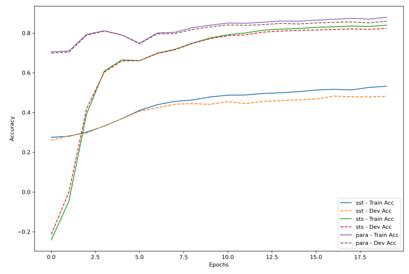


(b) Dev set

Figure 2: Heatmap of Accuracy by Experiment Name and Task



(a) FT



(b) Candidate 6

Figure 3: Train and Dev Accuracy vs Epochs for Each Task

- Given that none of the explored techniques sufficiently improved SST accuracy, alternative strategies might include augmenting the dataset or increasing the number of attention layers in the base model.
- Although full fine-tuning without loss weighting achieved superior accuracy, the LoRA technique mitigated overfitting. Nonetheless, its overall accuracy on the development dataset was 4.76% inferior.
- Further exploration can be done in the approach to weighting losses. Investigating alternative hyperparameter values or experimenting with different methodologies, such as the one proposed by Huq and Pervin (2022), could offer better results and insights.

References

Robert French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3:128–135.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Aminul Huq and Mst. Tasnim Pervin. 2022. Adaptive weight assignment scheme for multi-task learning. *IAES International Journal of Artificial Intelligence (IJ-AI)*, 11(1):173.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics.

Simon Kornblith, Jonathon Shlens, and Quoc V. Le. 2019. Do better imagenet models transfer better?

Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. 2022. Fine-tuning can distort pretrained features and underperform out-of-distribution.

Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. 2016. How transferable are neural networks in nlp applications?

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks?

A Appendix

A second model architecture was explored before running the model architecture explained in this project. The model architecture described in the previous section achieved better results. Therefore, it was chosen to generate all the results reported in this project.

The abandoned model architecture comprised additional layers on top of the shared BERT encoder. For sentiment classification, it included a dropout layer followed by a linear classification layer tailored to predict five levels of sentiment. Paraphrase detection and semantic textual similarity tasks were addressed through a pipeline that first concatenates embeddings from pairs of sentences, followed by a compression layer (linear layer) to reduce dimensionality, a dropout layer for regularization, and finally, a classification layer that outputs task-specific predictions.

The results described at the project milestone were generated from the abandoned model architecture.