A Comparative Evaluation of PEFT Techniques: LoRA and Bottleneck Adapters for Binary Classification

Ofek Kirshenboim
Humboldt Universität zu Berlin
ofek.kirshenboim@hu-berlin.de

Arsenii Gulevich
Humboldt Universität zu Berlin
gulevica@hu-berlin.de

Abstract

Parameter-Efficient Fine-Tuning (PEFT) techniques serve as a means to adapt large pretrained language models to downstream tasks while keeping the number of trainable parameters low.

This experiment investigates the effectiveness of various PEFT strategies in the context of a binary text classification task using the CoLA (Corpus of Linguistic Acceptability) dataset (Warstadt et al., 2018). Specifically, we focus on the use of adapter modules—additional trainable layers inserted into the transformer architecture—as introduced by (Houlsby et al., 2019). We begin by establishing a baseline with a BERT model that includes adapters in every transformer layer. Building on this, we evaluate configurations with adapters restricted to only the higher layers of the model, and we further explore the performance of models fine-tuned using Low-Rank Adaptation (LoRA)(Hu et al., 2021) modules with varying parameter counts, and a more recent adapters architecture (Adapter+) introduced by (Steitz and Roth, 2024) for vision transformers (ViT).

According to our experiments, adapter+(Steitz and Roth, 2024) architecture has reached the highest accuracy on the task, LoRA did not perform better even with the same number of parameters. The difference in accuracy amounts to 1-2 percent.

The implementation is available at: `bert-lora-adapter-finetuning`

1 Introduction

Using pre-trained models has proven very effective for many NLP tasks (Howard and Ruder, 2018; Radford et al., 2018). BERT, a Transformer model trained on massive text datasets without supervision, achieved top results on text classification and extractive question answering (Devlin et al., 2019). The goal of this project is to compare the parameter efficiency of different fine-tuning approaches by evaluating how validation accuracy scales with the number of trainable parameters, and to identify which layers contribute most critically to the performance of an adapted model. In many real-world applications, models are often required to handle a stream of diverse tasks. This setting presents a challenge of how to efficiently adapt to new tasks without incurring high computational costs or forgetting previously learned tasks. Houlsby et al. as well as Hu et al. address this problem by introducing parameter-efficient transfer learning approaches that enable compact and extensible models. Their methods allows the models to retain prior knowledge while requiring only a small number of additional parameters for each new task.

In NLP, there are two main ways to use pretrained models: feature-based transfer and fine-tuning. However, Houlsby et al. suggest a different approach using adapter modules. Feature-based transfer works by first training embeddings - numerical representations of test pieces (Mikolov et al., 2013; Yang et al., 2019; Le and Mikolov, 2014) - then using these for other specific tasks (Howard and Ruder, 2018). Fine-tuning takes the weights from an already-trained model and adjusts them for a new task. Both approaches need separate sets of parameters for each new task. Fine-tuning can be more efficient when multiple tasks share the same lower network layers. But Houlsby's adapter method as well as Low Rank Adaptation (LoRA) (Hu et al., 2021) are even more efficient with parameters a seen in figure 1.

However, it is not clear if or Houlsby-style bottleneck adapters perform worse than LoRA in terms of accuracy and parameter efficiency on a binary classification task such as CoLA. Moreover, as pointed out in (Houlsby et al., 2019), the adapter layers only in the last transformer layers seem to contribute the most to the model performance. The question arises if we can strip away adapters in the lower transformer layers without a significant accuracy loss.

Our study finds that LoRA performs worse on the CoLA task than bottleneck adapters and that newer ViT adapters results are transferable to the NLP domain.

2 Related Work

Popular approaches for adapting a pre-trained network to a novel task include:

- **Modular Adaptation** Adapters refer to the concept of incorporating small, trainable modules with minimal parameters into an otherwise frozen neural network. These modules implement bottleneck architectures within transformer layers (Rebuffi et al., 2017; Houlsby et al., 2019).

- **Prompt tuning** Drawing inspiration from prompts, which influence attention patterns across original input tokens, Lester et al. (Lester et al., 2021) introduced prompt tuning: a method that incorporates a collection of trainable tokens into the input sequence, optimized via backpropagation to guide a frozen language model toward specific downstream tasks. Building upon this approach, Li and Liang (Li and Liang, 2021) expanded prompt tuning by introducing learnable tokens at each transformer layer throughout the model, a technique they called prefix tuning.

- **Low-rank approaches** Targeting the attention mechanism within transformer layers, (Hu et al., 2021) introduced LoRA, a technique that modifies attention weights through low-rank matrix decomposition. These decomposition matrices can be consolidated with the original attention weights during inference.

In our experiments, we focused on the following specific architectures.

2.1 Adapters

Adapters (Houlsby et al., 2019) are compact modules integrated into transformer layers. They enable network adaptation to new tasks or domains by training only the adapter parameters and classifier, rather than fine-tuning the entire network. These adapters implement a bottleneck architecture with an inner dimension $r \ll d$, where r represents the adapter rank. Specifically, the architecture consists of a down-projection to dimension r using weights $\mathbf{W}_{\text{down}} \in \mathbb{R}^{d\times r}$ and biases $\mathbf{b}_{\text{down}} \in \mathbb{R}^{r}$, followed by a non-linear activation function $\sigma(\cdot)$ (commonly GELU as employed in ViT), and an up-projection with weights $\mathbf{W}_{\text{up}} \in \mathbb{R}^{r\times d}$ and biases $\mathbf{b}_{\text{up}} \in \mathbb{R}^{d}$ that restores the original hidden dimension d of the transformer layer. This configuration produces a base adapter module

$$\text{Adapter}_{\text{base}}(\mathbf{x}) = \sigma(\mathbf{x}\mathbf{W}_{\text{down}} + \mathbf{b}_{\text{down}})\mathbf{W}_{\text{up}} + \mathbf{b}_{\text{up}}\,. \tag{1}$$

The base adapter module can be enhanced further through the incorporation of normalization layers, such as layer normalization (LN). Moreover, the bottleneck output can be modulated by a scaling factor s as

$$\text{Adapter}(\mathbf{x}) = s \cdot \text{Adapter}_{\text{base}}\big(\text{LN}(\mathbf{x})\big)\,. \tag{2}$$

For layer-wise scaling, the factor s is defined as a scalar value, i.e. $s \in \mathbb{R}$, and can be either set as a fixed hyperparameter or optimized during training. Typically, the adapter incorporates a skip connection, resulting in the complete feature transformation:

$$\mathbf{x} \mapsto \text{Adapter}(\mathbf{x}) + \mathbf{x}\,. \tag{3}$$

2.2 Low Rank Adaptation

Concentrating on the attention mechanism within transformer layers, (Hu et al., 2021) introduced Low-Rank Adaptation, which updates attention weights through low-rank decomposition matrices. These matrices can be integrated with the attention weights during inference. The architecture of LoRA bears strong similarity to adapters, with adapters functioning as a generalization of LoRA that operates on the transformer's feed-forward network (FFN).

2.3 Adapter Plus

Through comprehensive analysis of various adapter configurations, (Steitz and Roth, 2024) determined that adapter modules positioned after the transformer layer (Post-Adapter) with learnable, channel-wise scaling and not incorporating layer normalization achieve optimal performance for computer vision tasks.

3 Experiments

This study investigates the following questions: (1) Which parameter-efficient fine-tuning (PEFT) method performs best in terms of accuracy and parameter efficiency? (2) What is the impact of removing adapters from the lower transformer layers on model performance?

3.1 Experimental Setup

Our experimental setup involves a study of parameter-efficient fine-tuning methods on the CoLA task using BERT-base-uncased as the foundation model. We compare three adapter-based methods: LoRA, Houlsby adapters, and Adapter+ through two complementary analyses: (1) parameter efficiency evaluation across varying ranks, reduction factors, and parameter counts to assess performance-efficiency trade-offs, and (2) layer ablation study comparing Adapter+ and Houlsby robustness when systematically removing adapter modules from the initial transformer layers.

Evaluation dataset. We evaluate on the CoLA dataset, following the evaluation protocol established in Houlsby et al. We use the Hugging Face version of the dataset, which contains 8,551 training and 1,043 validation examples after preprocessing. We tokenize inputs using BERT's WordPiece tokenizer with a maximum sequence length of 512 tokens.

Evaluation measures. Each configuration is evaluated across 5 random seeds (42, 43, 44, 45, 46) to ensure statistical reliability. We report validation accuracy and Matthews Correlation Coefficient (MCC) as our primary metrics.

Models and hyperparameters. Base Model: We employ BERT-base-uncased (110M parameters, 12 layers, 768 hidden dimensions, 12 attention heads) as our foundation transformer model for all experiments. Adapter Configurations: For LoRA, we test ranks of [1, 4, 16, 32, 64, 128] with alpha value of 16, bias disabled, and a dropout rate of 0.1 to explore the rank-performance trade-off. For Houlsby and Adapter+ methods, we use reduction factors of [8, 16, 32, 64] (the factor by which the dimension of the transformer layer is reduced to the inner dimension of the bottleneck adapter module), Swish activation function for Houlsby and GELU activation function for Adapter+, with no dropout within adapter modules. The leave_out parameter controls which transformer layers exclude adapters, enabling our layer ablation experiments. Training Hyperparameters: All models are trained for 8 epochs using the AdamW optimizer with a learning rate of 2e-5 and batch size of 16. Hyperparameter Selection: The choice of 8 epochs provides sufficient training without overfitting on the CoLA task, as evidenced by convergence patterns in preliminary experiments. Reduction factors of 8, 16, 32, and 64 represent part of the range evaluated in the original Houlsby paper, enabling direct comparison with established baselines.

3.2 Results

The experimental results are presented in Figures 1 and 2, which illustrate the comparative performance of the three parameter-efficient fine-tuning methods across different configurations and parameter counts.

Overall Performance Comparison. Adapter+ demonstrates superior performance compared to Houlsby adapters and LoRA in most experimental conditions. As shown in Figure 1, Adapter+ achieves validation accuracies ranging from 81.25% to 82.38% on the CoLA dataset, consistently outperforming Houlsby adapters (80.02% to 81.50%) and LoRA (80.02% to 80.61%). This performance advantage is maintained across different numbers of trainable parameters, suggesting that Adapter+ provides more effective parameter utilization for the given task.

Robustness to removal of Adapter Modules The analysis of robustness to first layer removal reveals distinct behavioral patterns between the methods. Figure 2 demonstrates that Houlsby Adapters exhibit greater resilience to the removal of initial layers.

Parameter Efficiency. The relationship between trainable parameters and performance reveals that LoRA maintains relatively stable performance across different parameter configurations, with accuracy fluctuating minimally between 79.9% and 80.5%. However, this stability comes at the cost of lower absolute performance compared to the adapter-based methods.

3.3 Conclusion

Our experimental results indicate that LoRA performance remained stable across the range of rank values investigated, despite the corresponding changes in the number of trainable parameters. Houslby-style bottleneck adapters achieved performance levels comparable to those of LoRA. Among the parameter-efficient fine-tuning approaches evaluated, Adapter+ demonstrated modest but consistent improvements over both LoRA and Houslby-style adapters, which tells us that the ViT improvements are transferable into the NLP domain.

Adapters Architecture	ACCURACY	MCC
Adapter+ Adapters		
+ *RF - 8*	81.25 ± 0.19	0.54 ± 0.001
+ RF - 16	81.73 ± 0.18	0.55 ± 0.010
+ RF - 32	**82.38** ± **0.22**	0.57 ± 0.010
+ RF - 64	82.13 ± 0.55	**0.58** ± **0.010**
Houlsby Adapters		
+ *RF - 8*	80.02 ± 0.28	0.52 ± 0.005
+ RF - 16	80.94 ± 0.20	0.54 ± 0.010
+ RF - 32	80.94 ± 0.04	0.54 ± 0.006
+ RF - 64	81.50 ± 0.09	0.56 ± 0.004
LORA		
+ R - 1	80.61 ± 0.27	0.53 ± 0.010
+ R - 4	80.31 ± 0.23	0.52 ± 0.003
+ R - 16	80.40 ± 0.15	0.52 ± 0.010
+ R - 32	79.90 ± 0.16	0.52 ± 0.005
+ R - 64	80.48 ± 0.18	0.52 ± 0.003
+ R - 128	80.02 ± 0.004	0.51 ± 0.003

Table 1: Performance of different PEFT methods on the CoLA dataset. Each score is reported as mean ± standard deviation across 5 random seeds (42, 43, 44, 45, 46). RF refers to the reduction factor, and R refers to the rank used in LoRA.

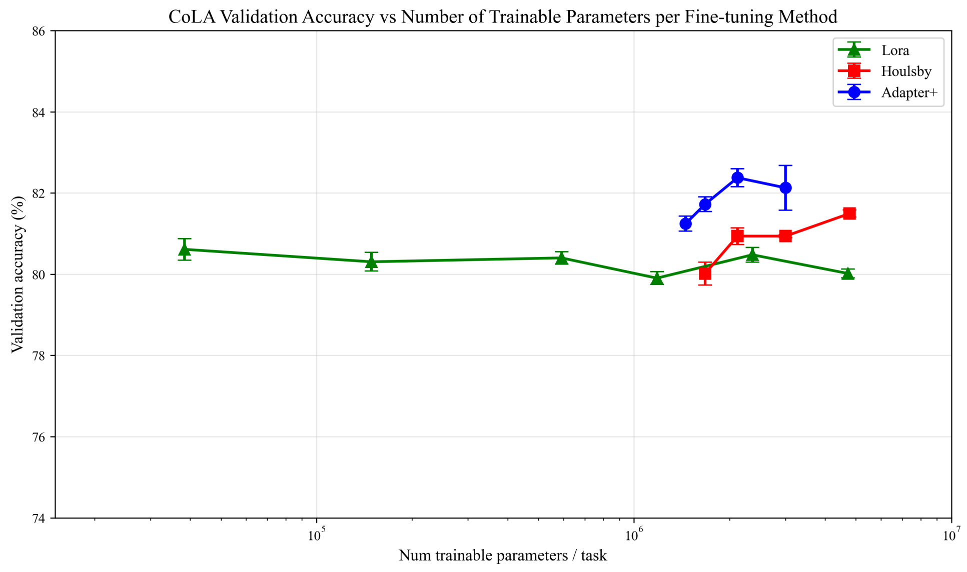

Figure 1: Performance comparison of different PEFT techniques on the CoLA task

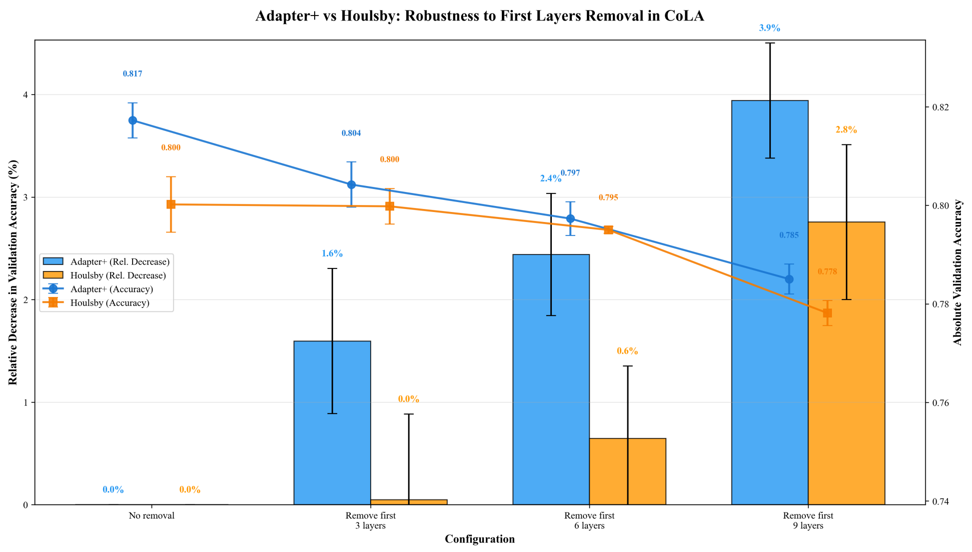

Figure 2: Comparison Adapter+, Houlsby: Ablation of adapter modules from transformer layers.

Several limitations constrain the scope of this investigation. The study focused on a specific binary classification task using a small-scale English-language dataset, which can limit the generalizability of the findings in different tasks, languages, and data scales. Additionally, the hyperparameter optimization was not exhaustive.

References

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.

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.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

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.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

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

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters.

Jan-Martin O. Steitz and Stefan Roth. 2024. Adapters strike back.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, et al. 2019. Multilingual universal sentence encoder for semantic retrieval. *arXiv preprint arXiv:1907.04307*.

4 Source Code

`bert-lora-adapter-finetuning`