International Journal of Intelligent Computing and Technology
ISSN: 2457-0249 (Online)
IJICT

Website: https://www.ijict.com January (2026) - Vol. 9, Iss.2 Pages:15–24

Efficient Adaptation of Transformer Models: A Comparison of Full and Parameter-Efficient Fine-Tuning on Limited Data

S Mary Immaculate
Lecturer, Department of Computer Science
Hindustan College of Arts & Science, Chennai, Tamil Nadu, India
G Maria Joyce
Lecturer, Department of Computer Science
Hindustan College of Arts & Science, Chennai, Tamil Nadu, India

Abstract

Pretrained transformer models have achieved state-of-the-art performance in various natural language processing tasks. However, adapting these models using full fine-tuning requires significant computational resources and large datasets. In low-resource environments, full fine-tuning often leads to overfitting and inefficient parameter updates. Parameter-Efficient Fine-Tuning (PEFT) methods offer an alternative approach by updating only a small subset of model parameters while preserving pretrained knowledge. This paper presents a comparative study between full fine-tuning and parameter-efficient methods including LoRA, adapter layers, and BitFit under limited data conditions. Experiments are conducted on small subsets of benchmark text classification datasets. Results demonstrate that PEFT methods achieve comparable performance to full fine-tuning while significantly reducing computational cost and memory usage. The findings suggest that PEFT methods are well-suited for resource-constrained environments and small dataset scenarios.

Keywords: Fine-tuning, Transformer Models, Parameter-Efficient Fine-Tuning, LoRA, Adapters, BitFit.

1 Introduction

Recent advances in natural language processing have been largely driven by pretrained transformer-based architectures such as Google Research BERT and related foundation models. These models learn general language representations from large-scale corpora and can be adapted to downstream tasks such as text classification, sentiment analysis, and question answering. The most common adaptation strategy is full fine-tuning, where all pretrained parameters are updated using task-specific labeled data. While this approach achieves strong performance with sufficient training data, it introduces several practical challenges. Updating millions of parameters requires significant computational resources and increases both memory consumption and training time. Furthermore, in scenarios where only limited training data is available, full fine-tuning often results in overfitting and unstable convergence. To address these limitations, Parameter-Efficient Fine-Tuning (PEFT) methods have been proposed as an alternative. Rather than updating all parameters, PEFT methods introduce small trainable components or selectively update specific parameters while keeping the majority of the model frozen. Popular PEFT techniques include Low-Rank Adaptation (LoRA), adapter layers, and bias-only tuning (BitFit). While preserving competitive performance, these techniques drastically reduce the number of trainable parameters. Despite the growing interest in PEFT methods, systematic comparisons under strictly limited data conditions remain limited. Many studies evaluate these approaches on large datasets without explicitly analyzing their behavior in small-data scenarios. This paper presents a structured comparison between full fine-tuning and multiple PEFT approaches under controlled small-dataset conditions. The study focuses on classification performance, computational efficiency, and overfitting behavior.

2 Literature Review

Fine-tuning pretrained transformer models has become a standard approach in natural language processing tasks. Transformer architectures such as BERT introduced powerful contextual representations that significantly improved performance across multiple NLP tasks including text classification, sentiment analysis, and question answering [1]. In traditional transfer learning settings, models are adapted to downstream tasks through full fine-tuning, where all model parameters are updated using task-specific labeled data. Although full fine-tuning provides high flexibility and strong task adaptation, it requires substantial computational resources and large training datasets. Updating millions of parameters increases training time, memory usage, and the risk of overfitting when the available dataset is small [2]. To address these challenges, researchers have proposed Parameter-Efficient Fine-Tuning (PEFT) methods. These techniques aim to adapt pretrained models while updating only a small subset of parameters, thereby reducing com-

putational cost and improving efficiency. Recent surveys highlight the growing importance of PEFT methods for adapting large-scale transformer models [3]. One widely used PEFT technique is Low-Rank Adaptation (LoRA), which introduces low-rank matrices to approximate weight updates during training. By learning low-dimensional parameter updates while keeping the original weights frozen, LoRA significantly reduces the number of trainable parameters while maintaining strong performance. Another approach involves adapter layers, which insert small neural modules between transformer blocks. During training, only the adapter parameters are updated while the pretrained backbone model remains frozen. Adapter methods provide stable training and efficient parameter usage while preserving the knowledge learned during pretraining. BitFit is an even simpler approach that updates only the bias parameters of the transformer model. Despite its simplicity, BitFit has been shown to perform competitively on several NLP tasks while requiring extremely few trainable parameters. Recent studies have demonstrated that PEFT methods can achieve performance comparable to full fine-tuning while significantly reducing training cost [4]. However, most existing work evaluates these approaches using large datasets. Comparisons under strictly limited data conditions remain relatively unexplored. Therefore, this study focuses on analyzing the effectiveness of PEFT methods when training data is scarce.

2.1 Full Fine -Tuning

Transformer architectures significantly improved NLP performance after the introduction of pretrained models such as BERT. Traditional adaptation involves updating all model parameters for downstream tasks. While full fine-tuning provides flexibility and strong performance, it requires large computational resources and may generalize poorly when training data is limited. Various regularization and optimization techniques have been proposed to improve stability, but the challenge of updating large numbers of parameters remains.

2.2 Parameter-Efficient Fine-Tuning

Parameter-efficient fine-tuning methods reduce the number of trainable parameters while preserving pretrained knowledge. Adapter Methods Adapter methods introduce small bottleneck layers between transformer blocks. Only adapter parameters are trained, while the original model remains frozen. LoRA LoRA decomposes weight updates into low-rank matrices, reducing memory usage and improving efficiency. BitFit BitFit updates only bias parameters within transformer layers, making it one of the most parameter-efficient approaches.

2.3 Fine-Tuning with Limited Data

Learning from limited data is a common challenge in NLP. Small datasets often lead to overfitting and poor generalization. Few-shot learning and data augmentation techniques attempt to address these challenges, but they often increase training complexity. Comparisons between full fine-tuning and PEFT methods in small-data scenarios remain limited, motivating this study.

3 Methodology

3.1 Model Architecture

A pretrained transformer model is used as the backbone architecture in this study. Transformer models consist of multiple stacked layers that include self-attention mechanisms and feed-forward neural networks [5]. These components allow the model to capture contextual relationships between words and generate meaningful representations for downstream tasks. The pretrained model serves as a strong initialization, enabling efficient adaptation to new tasks with limited training data. In this work, the pretrained transformer model is adapted using four different fine-tuning approaches: full fine-tuning, Low-Rank Adaptation (LoRA), adapter layers, and BitFit. Each method modifies the model in a different way, allowing a comparison between full parameter updates and parameter-efficient approaches. The objective is to evaluate the performance and efficiency of these methods when applied to small datasets.

3.2 Fine-Tuning Methods

3.2.1 Full Fine-Tuning

Full fine-tuning updates all model parameters during training. This approach allows the model to fully adapt to the target task by modifying the entire set of pretrained weights. As a result, full fine-tuning often achieves high performance because the model can learn task-specific representations. However, updating all parameters requires significant computational resources and memory. In addition, full fine-tuning may lead to overfitting when the training dataset is small, as the model may learn patterns specific to the training data rather than generalizable features [6].

3.2.2 LoRA

Low-Rank Adaptation (LoRA) is a parameter-efficient fine-tuning method that introduces low-rank matrices to approximate weight updates. Instead of modifying the original model parameters, LoRA adds small trainable matrices that capture

task-specific information while keeping the pretrained weights frozen. This significantly reduces the number of trainable parameters and lowers memory usage during training. LoRA also allows faster training compared to full fine-tuning while maintaining competitive performance, making it suitable for resource-constrained environments. LoRA introduces low-rank matrices to approximate parameter updates while keeping pretrained weights frozen.

3.2.3 Adapter Layers

Adapter layers are small neural modules inserted between transformer layers. During training, only the adapter parameters are updated while the original model parameters remain unchanged. This approach reduces the number of trainable parameters and provides stable training behavior. Adapter layers allow the model to adapt to new tasks while preserving the knowledge learned during pretraining. Due to their moderate parameter size, adapter methods provide a balance between efficiency and performance.

3.2.4 BitFit

BitFit is a lightweight fine-tuning method that updates only the bias parameters of the transformer model. Since bias terms represent a very small portion of the total parameters, BitFit is highly efficient and requires minimal computational resources. This method allows rapid model adaptation with very low memory usage. However, because only a small number of parameters are updated, BitFit may achieve lower performance compared to other fine-tuning methods on more complex tasks. BitFit updates only the bias parameters of transformer layers, significantly reducing the number of trainable parameters [7].

Table 1: Comparison of Fine-Tuning Methods

Method	Trainable Parameters	Memory Usage	Training Speed	Performance	Overfitting Risk
Full Fine-Tuning	100%	High	Slow	High	High
LoRA	Low	Low	Fast	High	Low
Adapter Layers	Medium	Medium	Medium	High	Low
BitFit	Very Low	Very Low	Very Fast	Medium	Low

Table 2: Fine-Tuning Parameter Updates

Method	Updated Parameters	Frozen Parameters
Full Fine-Tuning	All Weights	None
LoRA	Low-Rank Matrices	Original Weights
Adapter Layers	Adapter Modules	Backbone Model
BitFit	Bias Parameters	All Other Weights

Table 3: Advantages and Disadvantages of Fine-Tuning Methods

Method	**Advantages**	**Disadvantages**
Full Fine-Tuning	High accuracy, Full flexibility	High memory usage
LoRA	Efficient training, Low memory	Rank tuning required
Adapter Layers	Stable training	Extra layers added
BitFit	Very efficient	Lower performance

4 Experimental Setup

4.1 Datasets

The experiments were conducted using subsets of the AG News text classification dataset, which is a widely used benchmark dataset for evaluating natural language processing models. The dataset contains news articles categorized into four classes: World, Sports, Business, and Science/Technology. The dataset is publicly available and commonly used for benchmarking transformer-based text classification models. To simulate low-resource learning scenarios, smaller subsets of the dataset were created. Three dataset sizes were considered: 100 samples, 500 samples, and 1000 samples. These subsets were randomly selected from the original dataset while maintaining class balance across categories. Before training, standard text preprocessing steps were applied. These steps include text normalization, tokenization using the transformer tokenizer, removal of unnecessary whitespace, and truncation or padding of sequences to a fixed input length. The data was then split into training and validation sets using an 80:20 ratio to ensure consistent evaluation. The AG News dataset was selected because it provides clear category labels and is frequently used for benchmarking transformer-based models. Its balanced structure and manageable text length make it suitable for evaluating the effectiveness of fine-tuning methods under limited data conditions. Each dataset was split into training and validation sets to ensure consistent evaluation across all fine-tuning methods. The smaller dataset sizes help analyze how different methods perform when training data is limited. This setup allows a fair comparison between full fine-tuning and parameter-efficient fine-tuning methods in terms of performance and generalization [8].

4.2 Training Configuration

All models were trained using the same training configuration to ensure a fair comparison between methods. The AdamW optimizer was used because it is widely applied in transformer model training and provides stable convergence. A learning rate of 5e-5 was selected to balance training stability and convergence speed. The batch size was set to 16 to maintain efficient GPU memory usage while ensuring stable training. The models were trained for a maximum of 20

epochs. Early stopping was applied to prevent overfitting by stopping training when the validation performance stopped improving. This helps ensure that the models do not overfit the small datasets.

4.3 Evaluation Metrics

The performance of the models was evaluated using standard classification metrics, including accuracy, precision, recall, and F1-score. These metrics provide a comprehensive evaluation of model performance across different dataset sizes. Accuracy measures the proportion of correctly predicted samples out of the total number of samples. Precision measures the proportion of correct positive predictions, while recall measures the ability of the model to identify positive samples. The F1-score represents the harmonic mean of precision and recall and provides a balanced measure of model performance. Accuracy is calculated as: Accuracy = Number of Correct Predictions / Total Number of Predictions These evaluation metrics allow a consistent comparison between full fine-tuning and parameter-efficient fine-tuning methods.

5 Results

5.1 Performance Comparison

The experimental results demonstrate how different fine-tuning methods perform under limited data conditions. As shown in Table I, model performance improves as the dataset size increases from 100 to 1000 samples. This behavior is expected because larger datasets provide more training examples that allow the model to learn more generalized patterns. Among the evaluated methods, LoRA consistently achieves the highest accuracy across all dataset sizes. This indicates that low-rank parameter updates effectively capture task-specific information while preserving the knowledge contained in pretrained weights. Adapter layers also demonstrate strong performance, producing results close to LoRA while maintaining stable training behavior. Full fine-tuning performs competitively when more training data is available, but its performance slightly decreases in smaller datasets. This is likely due to overfitting, since updating all model parameters using limited data may cause the model to memorize training examples rather than learn general features. BitFit achieves the lowest accuracy among the methods, particularly on the smallest dataset. However, it still provides reasonable performance despite updating only bias parameters. This highlights the efficiency of parameter-efficient methods in scenarios where computational resources are limited.These results support the research objective of evaluating efficient adaptation strategies for transformer models under limited data conditions. Full fine-tuning performs competitively on larger datasets but achieves slightly lower accuracy on

smaller datasets. This occurs because full fine-tuning updates all model parameters, which increases the risk of overfitting when training data is limited. BitFit shows the lowest accuracy among the methods, especially on the smallest dataset. However, its performance improves as the dataset size increases, indicating that it can still learn useful patterns despite updating only bias parameters.

Table 4: Performance Comparison

Method	**100 Samples**	**500 Samples**	**1000 Samples**
Full FT	78%	85%	90%
LoRA	80%	87%	91%
Adapter	79%	86%	90%
BitFit	75%	83%	88%

These results suggest that parameter-efficient fine-tuning methods can achieve performance comparable to full fine-tuning while using fewer parameters.

5.2 Efficiency Comparison

The efficiency comparison evaluates the computational requirements of each method in terms of parameter usage, memory consumption, and training time. Full fine-tuning requires updating all model parameters, which leads to high memory usage and slower training. This makes it computationally expensive, especially for large transformer models. In contrast, parameter-efficient methods update only a small portion of the model parameters, which reduces memory usage and training time. LoRA requires fewer parameters than full fine-tuning and allows faster training while maintaining strong performance. Adapter layers require a moderate number of parameters and provide a balance between efficiency and performance. BitFit requires the smallest number of parameters and achieves the fastest training time, making it the most efficient method.

Table 5: Efficiency Comparison

Method	**Parameters**	**Memory**	**Training Time**
Full FT	High	High	Slow
LoRA	Low	Low	Fast
Adapter	Medium	Medium	Medium
BitFit	Very Low	Very Low	Very Fast

These results show that parameter-efficient methods significantly reduce computational cost compared to full fine-tuning.

6 Overfitting Analysis

Overfitting occurs when a model learns patterns specific to the training data instead of general patterns that apply to new data. This problem is more common when training data is limited. The results indicate that full fine-tuning shows higher variance on small datasets, which suggests overfitting. Since full fine-tuning updates all model parameters, the model may memorize the training data instead of learning general features. Parameter-efficient methods demonstrate more stable performance across different dataset sizes. Because these methods update fewer parameters, they are less likely to overfit and tend to generalize better to unseen data. This stability makes parameter-efficient methods particularly suitable for small dataset scenarios. Overall, the results show that parameter-efficient fine-tuning methods provide a good balance between performance and efficiency while reducing the risk of overfitting. The results show that parameter-efficient fine-tuning (PEFT) methods provide a good balance between performance and efficiency on small datasets. LoRA achieves the best overall performance, while adapter layers provide stable training. BitFit is the most efficient method but may result in slightly lower accuracy. Full fine-tuning performs well with larger datasets but requires more computational resources and is more prone to overfitting on small datasets. In contrast, PEFT methods update fewer parameters, which helps improve generalization and reduces training time and memory usage.

7 Conclusion

This paper compared full fine-tuning and parameter-efficient fine-tuning methods on small datasets. The results show that PEFT methods achieve comparable accuracy while reducing computational cost. LoRA provides the best trade-off between performance and efficiency, while adapter methods offer stable training and BitFit provides high efficiency. Full fine-tuning is more suitable for larger datasets, whereas PEFT methods are better suited for small datasets and limited-resource environments. Overall, parameter-efficient fine-tuning is an effective approach for adapting transformer models in practical applications.

References

[1] K. Lv *et al.*, "Full Parameter Fine-Tuning for Large Language Models with Limited Resources," *arXiv preprint arXiv:2306.09782*, 2023.

[2] Z. Han *et al.*, "Parameter-Efficient Fine-Tuning for Large Models: A Comprehensive Survey," *arXiv preprint arXiv:2403.14608*, 2024.

[3] K. P. V. Srinivasan *et al.*, "Comparative Analysis of Different Efficient Fine-Tuning Methods of Large Language Models in Low-Resource Setting," *arXiv preprint arXiv:2405.13181*, 2024.

[4] D. Zhang *et al.*, "Parameter-Efficient Fine-Tuning for Foundation Models," *arXiv preprint arXiv:2501.13787*, 2025.

[5] O. Razuvayevskaya *et al.*, "Comparison Between Parameter-Efficient Techniques and Full Fine-Tuning: A Case Study on Multilingual News Article Classification," *PLOS ONE*, vol. 19, no. 5, 2024.

[6] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL*, 2019.

[7] E. Hu *et al.*, "LoRA: Low-rank adaptation of large language models," in *Proc. ICML*, 2022.

[8] T. Ben-Zaken, Y. Goldberg, and S. Ravfogel, "BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language models," in *Proc. ACL*, 2022.

How to cite this article:

S Mary Immaculate & G Maria Joyce, "Efficient Adaptation of Transformer Models: A Comparison of Full and Parameter-Efficient Fine-Tuning on Limited Data", International Journal of Intelligent Computing and Technology (IJICT), Vol.9, Iss.2, pp.15-24, 2026.