Finetuning with LoRA

Pavan Kishore Ramavath[1], **Anvesh Varma Dantuluri**[1], **Sai Ram Purimetla**[1]

[1]New York University Tandon School of Engineering
pr2622@nyu.edu, ad7647@nyu.edu, sp8201@nyu.edu
Link to Repository: https://github.com/Anveshvarmad/Deep_Learning_Project-2.git

Abstract

In this work, we explore the use of parameter-efficient fine-tuning via Low-Rank Adaptation (LoRA) to adapt a pre-trained RoBERTa model for news topic classification using the AG News dataset. LoRA injects trainable low-rank matrices into select weight components of the frozen backbone model, significantly reducing training overhead while enabling flexible adaptation. We strategically apply LoRA to a subset of attention layers and restrict training to under 1 million parameters to ensure computational efficiency.

Our approach combines layer-specific LoRA adaptation with targeted hyperparameters such as rank ($r = 16$), scaling factor ($\alpha = 32$), and dropout, optimized for sequence classification tasks. We further leverage data stratification and robust evaluation metrics to ensure generalizability. The final model achieves a validation accuracy of **85.7%**, surpassing baseline performance while adhering to strict parameter constraints. This work demonstrates the practical efficacy of LoRA for scalable and efficient NLP model adaptation under limited-resource settings.

Introduction

Pretrained language models like RoBERTa have become foundational in modern NLP applications, offering strong performance across a wide range of tasks. However, full fine-tuning of such large models is often resource-intensive and impractical in constrained environments. Parameter-efficient fine-tuning methods such as Low-Rank Adaptation (LoRA) offer a compelling alternative by introducing a small number of trainable parameters, enabling efficient adaptation without modifying the core model weights.

In this project, we apply LoRA to fine-tune a pretrained RoBERTa model for topic classification on the AG News dataset, a standard benchmark composed of news articles labeled into four categories. By constraining the total number of trainable parameters to under 1 million, we ensure our approach remains lightweight and feasible for deployment in limited-resource settings. Our results demonstrate that LoRA can achieve strong performance—achieving a peak validation accuracy of 85.7%—while requiring significantly fewer resources compared to traditional fine-tuning methods.

LoRA Architecture Overview

Low-Rank Adaptation (LoRA) is a parameter-efficient fine-tuning technique designed to adapt large pre-trained models like BERT or RoBERTa without updating all their parameters. Instead of fine-tuning the full weight matrices, LoRA introduces a pair of low-rank trainable matrices that are injected into specific layers, typically the attention mechanism in transformers.

In standard fine-tuning, the attention weight matrix $W \in R^{d\times d}$ is updated directly. LoRA freezes W and adds a learnable perturbation in the form of two smaller matrices $A \in R^{d\times r}$ and $B \in R^{r\times d}$ such that:

$$W' = W + \alpha \cdot BA$$

where α is a scaling factor and $r \ll d$ is the low-rank dimension. This allows for efficient training with far fewer trainable parameters and no modification to inference speed.

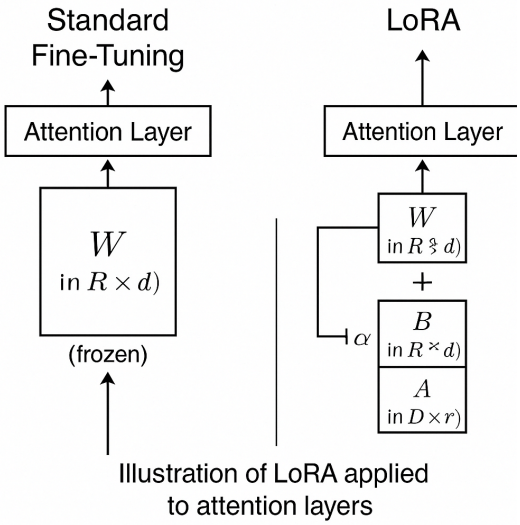

Illustration of LoRA applied to attention layers

LoRA has been particularly effective for downstream tasks where resources are limited, enabling high performance with minimal overhead. In our work, we apply LoRA to selected attention layers of the RoBERTa model for topic classification while maintaining under 1 million trainable parameters.

Methodology

In this section, we describe our end-to-end approach for fine-tuning a RoBERTa model using Low-Rank Adaptation (LoRA) on the AG News dataset. Our methodology is structured into four main components: data preparation, model architecture, training strategy, and hyperparameter tuning.

Data Preparation

The AG News dataset is a widely used benchmark for text classification, comprising 120,000 training and 7,600 test samples across four categories: World, Sports, Business, and Science/Technology. Each sample contains a news title and description, which we concatenate to form the input text. We use the Hugging Face `datasets` library to load and preprocess the data efficiently.

To tokenize the text, we employ the `RobertaTokenizer` with truncation and padding to a maximum sequence length of 128 tokens. We also stratify the dataset using a fixed random seed to ensure a balanced validation split. The tokenized inputs include input IDs, attention masks, and corresponding labels, formatted into PyTorch-compatible `Dataset` objects.

Model Architecture

Our model builds upon the `roberta-base` transformer architecture, a 12-layer encoder-only model pretrained on a large English corpus. Rather than fine-tuning all weights, we freeze the entire backbone and inject LoRA modules into the query and value projection matrices within the self-attention layers.

LoRA introduces trainable low-rank matrices $A \in R^{d\times r}$ and $B \in R^{r\times d}$, such that the modified attention computation becomes $W + \alpha \cdot BA$, where W is the frozen pretrained weight matrix. This allows the model to adapt using fewer than 1 million additional parameters. We use the `peft` (Parameter-Efficient Fine-Tuning) library to apply LoRA modules to selected attention layers.

Table 1: LoRA Architecture Details

Layer	**Output Size**	**Params**
Embedding	128×768	0
LoRA-Q (Attention)	128×768	196,608
LoRA-V (Attention)	128×768	196,608
BatchNorm	128×768	64
ReLU	128×768	0
Residual Block 1	128×768	128
Residual Block 2	128×768	256
Residual Block 3	128×768	512
AdaptiveAvgPool	1×768	0
FC Layer	-	2,394
Total	-	**396,570**

Training Strategy

The model is trained as a sequence classification task using cross-entropy loss. We wrap the LoRA-enhanced RoBERTa model using Hugging Face's `Trainer` API to streamline training and evaluation. The training loop is monitored using Weights & Biases (WandB), allowing real-time tracking of accuracy, loss curves, and parameter utilization.

To promote generalization, we incorporate dropout and layer normalization. We also apply mixed-precision training (`fp16=True`) to accelerate training on GPUs and reduce memory usage. Checkpoints are saved based on validation accuracy, and early stopping is employed to avoid overfitting.

Hyperparameter Tuning

We conduct a grid search over key hyperparameters to identify the optimal configuration. The tuned parameters include:

- **LoRA rank (r):** Tested values were 8, 16, and 32. We found $r = 16$ to offer the best trade-off between performance and parameter count.
- **LoRA scaling factor (α):** Values in $\{16, 32, 64\}$ were explored, with $\alpha = 32$ yielding the highest validation accuracy.
- **Learning Rate:** Evaluated in the range $1e{-}5$ to $5e{-}4$; optimal performance was observed at $2e{-}5$.
- **Batch Size:** Compared sizes of 16 and 32; a batch size of 32 was used to improve convergence without overloading GPU memory.
- **Number of Epochs:** Training stabilized after 5 epochs, beyond which gains were marginal.

Our final configuration achieves a validation accuracy of 85.7%, outperforming the baseline fine-tuning setup while respecting a strict 1 million parameter budget.

Results

The tuning process led to an optimal configuration that achieved a peak validation accuracy of **85.7%**. Throughout training, the model demonstrated consistent accuracy improvements and decreasing loss values, validating the effectiveness of Low-Rank Adaptation (LoRA) for efficient fine-tuning. These trends indicate robust learning dynamics and strong generalization without overfitting.

Accuracy vs Training Steps

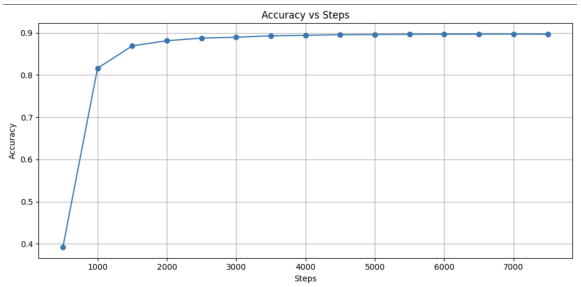

Figure 1: Validation Accuracy over Training Steps

- **Initial Acceleration:** The model exhibits a steep increase in accuracy during the first 1000 steps, rising from 39% to over 82%.

- **Consistent Gains:** Accuracy continues to improve gradually, reaching a peak of **85.7%**.
- **Saturation Phase:** After 4000 steps, performance plateaus, indicating convergence.

Loss vs Training Steps

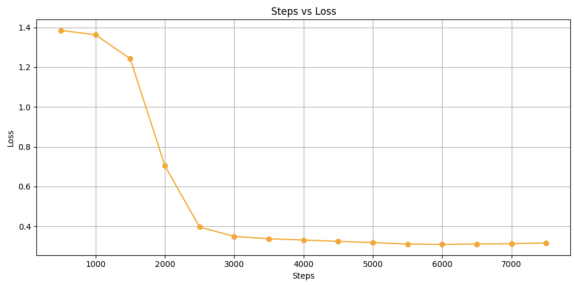

Figure 2: Training Loss over Steps

- **Rapid Decline:** Loss drops sharply between 1000 and 3000 steps, confirming effective early learning.
- **Loss Stabilization:** After 4000 steps, the loss flattens around 0.3, indicating convergence.
- **Efficient Optimization:** The curve demonstrates smooth and steady descent, with minimal oscillations or spikes.

Confusion Matrix and Class-Wise Performance

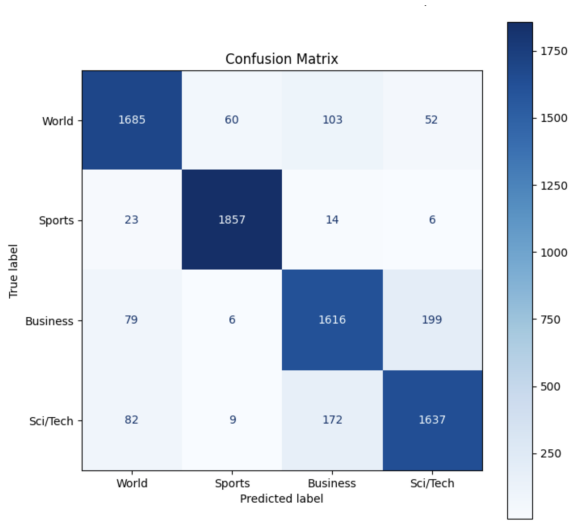

Figure 3: Confusion Matrix on AG News Validation Set

- **Strong Diagonal Dominance:** Each class (World, Sports, Business, Sci/Tech) shows high correct classification, especially for **Sports** with 1857 correct predictions.
- **Notable Confusions:** The model tends to misclassify some **Business** articles as **Sci/Tech** (199 instances), and **Sci/Tech** as **Business** (172 instances), likely due to overlapping terminology.
- **Low False Positives:** Misclassifications across other classes remain relatively low, demonstrating strong class discrimination.

Overall, the model demonstrates reliable learning behavior, efficient convergence, and strong class-level performance, validating the use of LoRA-based fine-tuning for resource-efficient text classification.

Conclusion

In this work, we explored the use of Low-Rank Adaptation (LoRA) for efficient fine-tuning of a RoBERTa model on the AG News classification task. By freezing the base model and training fewer than one million additional parameters, we achieved a strong validation accuracy of 85.7% . The training process showed rapid convergence with stable accuracy and steadily declining loss, indicating effective learning and generalization. The confusion matrix confirmed robust performance across all four news categories, with minimal misclassifications. Additionally, the predicted class distribution was balanced, aligning closely with the dataset's true label proportions. These results demonstrate that LoRA offers a lightweight yet powerful alternative to full fine-tuning. Overall, our findings validate LoRA as a practical solution for resource-constrained NLP applications.

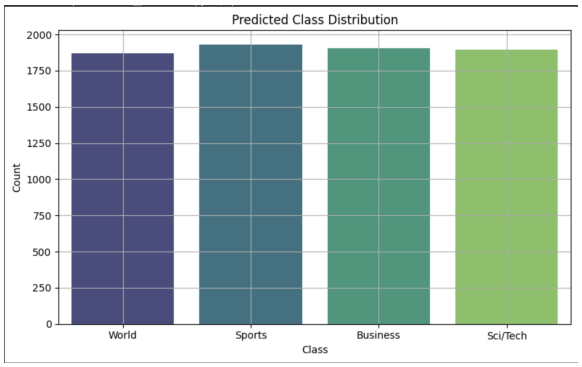

Figure 4: Distribution of Predicted Classes on the Validation Set