Parameter-Efficient Fine-Tuning of RoBERTa on AGNEWS using LoRA

Ron Regi Zacharia, Vinayak Naveen

rrz2014@nyu.edu, vn2259@nyu.edu

Abstract

This project explores parameter-efficient fine-tuning (PEFT) using Low-Rank Adaptation (LoRA) for text classification on the AGNEWS dataset, under the constraint of fewer than one million trainable parameters. Traditional fine-tuning of large language models like RoBERTa is often computationally expensive. LoRA addresses this by freezing the pre-trained model weights and injecting trainable low-rank matrices, significantly reducing memory and compute costs. Using the roberta-base model as the backbone, we fine-tune it using LoRA adapters to classify short news articles into four categories: World, Sports, Business, and Sci/Tech. The LoRA configuration is applied to selected attention submodules with tunable hyperparameters including rank (r) and scaling factor. We preprocess the AGNEWS dataset, train the LoRA-enhanced model using Hugging Face's Trainer API, and monitor training dynamics with custom logging. Our model achieves competitive performance while adhering to the strict parameter limit. We evaluate the final system on a held-out validation set, visualize performance via confusion matrices and learning curves, and generate predictions for Kaggle leaderboard submission. This project demonstrates that with careful adaptation and hyperparameter tuning, LoRA enables scalable and effective fine-tuning of transformers in resource-constrained scenarios.

Github Code

Link for the code = https://github.com/ronrzach/dl_proj2_g2

Introduction

Transformer-based language models such as BERT and RoBERTa have become the cornerstone of modern NLP systems due to their exceptional performance on a wide range of downstream tasks. However, full fine-tuning of these models requires significant computational resources, making it impractical for scenarios with hardware constraints. To address this, Parameter-Efficient Fine-Tuning (PEFT) techniques have been proposed, which selectively adapt a small number of parameters while keeping the majority of the model frozen.

In this project, we explore Low-Rank Adaptation (LoRA), a popular PEFT technique, to fine-tune a pre-trained roberta-base model on the AGNEWS dataset for text classification. The dataset consists of over 120,000 short news articles categorized into four topics: World, Sports, Business, and Sci/Tech. LoRA introduces trainable low-rank matrices into the attention layers of the transformer, allowing us to stay within a strict constraint of under 1 million trainable parameters.

Our implementation involves preprocessing the AGNEWS dataset using Hugging Face's datasets and tokenizers libraries, applying LoRA adapters via the peft library, and fine-tuning the model using the Trainer API. To monitor training, we integrated a custom TrainerCallback that logs loss values periodically. Evaluation is performed using validation accuracy and visualized using training curves and confusion matrices. Inference is run on an unlabelled test set, and final predictions are exported for Kaggle leaderboard submission.

This project demonstrates that LoRA is not only effective in reducing training cost but also maintains strong classification performance, making it a practical choice for resource-constrained NLP applications.

Methodology

Dataset

We use the AGNEWS dataset provided through the Hugging Face `datasets` library. It is a widely-used benchmark for topic classification of short news articles. The dataset contains a total of 120,000 training examples and 7,600 test examples, each consisting of a news headline and a corresponding label from one of four categories: **World**, **Sports**, **Business**, and **Sci/Tech**.

The dataset is loaded using the `load_dataset('ag_news', split='train')` function. We manually split 640 samples from the training set to form a validation set, which is used to monitor the model's performance during fine-tuning. The remaining samples form the training set.

Each data point contains a `text` field and a `label` field. The label is an integer ranging from 0 to 3, which is mapped to the corresponding class name using the `ClassLabel` feature provided by the dataset. To verify class balance, we

plot the label distributions in both the training and validation sets prior to model training.

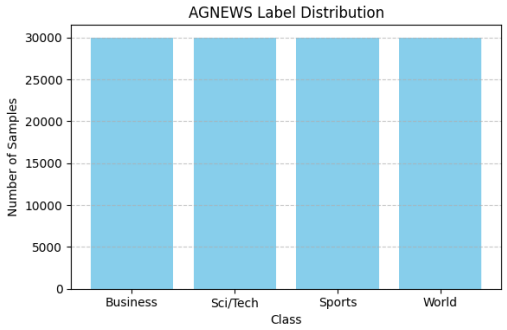

Figure 1: Normalized Confusion Matrix for the CIFAR-10 Classification Task.

Model Architecture

Our model architecture is based on the `roberta-base` transformer, a pre-trained encoder from the BERT family known for its strong performance on text classification tasks. To enable parameter-efficient fine-tuning, we modify the model using Low-Rank Adaptation (LoRA), which injects lightweight trainable matrices into specific attention submodules of the transformer without modifying the original weights.

Key Design Components:

a) Base Model: RoBERTa

We use `roberta-base`, a 12-layer transformer encoder model from the Hugging Face `transformers` library, pre-trained on large English corpora. The model includes approximately 125 million parameters and is well-suited for text classification tasks. We utilize the `RobertaForSequenceClassification` class, which adds a linear classification head on top of the encoder. The output dimension is set to 4 to match the four AGNEWS categories. We explicitly define an `id2label` mapping to preserve interpretability of model predictions and metrics.

b) LoRA Integration

To enable parameter-efficient fine-tuning, we apply Low-Rank Adaptation (LoRA) through the `peft` library. Instead of updating the full set of transformer weights, LoRA inserts trainable low-rank matrices into the model's self-attention layers. Specifically, we target the `query`, `key`, and `value` projection matrices in each attention head. These modules are wrapped with low-rank matrices A and B such that the adaptation update ΔW is approximated as AB. This reduces memory and computational cost, while preserving the representational power of the base model.

c) LoRA Configuration

Our LoRA configuration is defined with a rank $r = 2$, scaling factor $\alpha = 8$, and dropout rate of 0.05. The rank determines the dimensionality of the low-rank updates, while α scales their contribution when added to the frozen weights. We disable bias updates by setting `bias='none'` and define the task type as `SEQ_CLS` to signal that LoRA is being used for sequence-level classification. This setup allows for lightweight adaptation while preserving generalization from the pre-trained model.

d) Parameter Count Monitoring

A critical design constraint for this project is maintaining fewer than 1 million trainable parameters. To ensure this, we programmatically calculate both the total number of model parameters and the number of trainable parameters after LoRA injection. All base model parameters are frozen, and only the LoRA adapters and classification head remain trainable. We report the trainable parameter count and its proportion relative to the total model size to ensure compliance with project constraints. Our final model maintains full classification capabilities while using less than 1% of the base model's parameters, demonstrating the efficiency of LoRA.

Results

Evaluation Metrics

The performance of our LoRA-adapted RoBERTa model is primarily measured using classification accuracy on a held-out validation set containing 640 samples. The validation accuracy is computed using Hugging Face's `evaluate` library, which internally utilizes `sklearn.metrics.accuracy_score`. Throughout the training process, accuracy is evaluated every 50 steps.

Our final model achieves a validation accuracy of **90.63%**. This exceeds the project baseline target of 80%, confirming that Low-Rank Adaptation, when properly configured, can deliver competitive performance even under strict parameter constraints. The high validation accuracy demonstrates the model's ability to generalize well to unseen samples despite having only 704K trainable parameters.

Parameter Efficiency

A core constraint of this project is that the model must contain fewer than one million trainable parameters. This requirement is met through the use of LoRA, which injects low-rank matrices into selected attention submodules while keeping the base model weights frozen.

Our final configuration resulted in **704,260 trainable parameters**, which is just **0.56%** of the full model size of approximately 125 million parameters. These parameters are exclusively located in the LoRA adapters and the classification head, leaving the rest of the RoBERTa backbone untouched. This confirms that the model is both memory- and compute-efficient, satisfying the constraint while still achieving high performance.

Training Dynamics

We fine-tuned the model for 1 epoch with a maximum of 1200 training steps, using Stochastic Gradient Descent (SGD) with a learning rate of 0.1 and a batch size of 16. Logging was enabled for both training loss (every 10 steps)

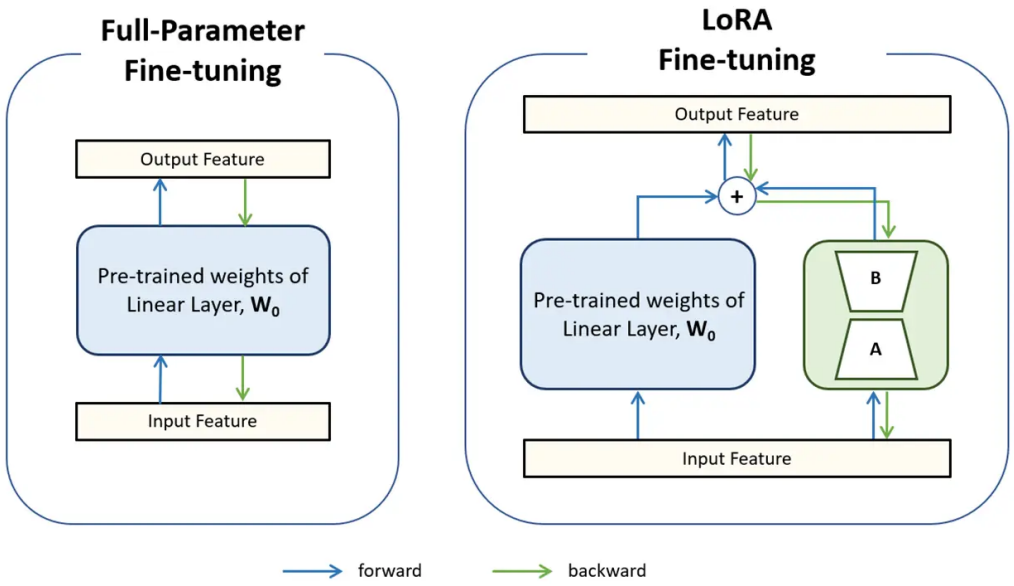

Figure 2: Comparison between Full-Parameter Fine-tuning (left) and LoRA Fine-tuning (right). In full fine-tuning, all weights in the pre-trained model are updated. In contrast, LoRA freezes the original weights W_0 and introduces trainable low-rank matrices A and B to capture task-specific adaptation, thereby reducing the number of trainable parameters.

and evaluation metrics (every 50 steps), allowing us to track the model's learning curve.

As visualized in Figure 3, the training loss rapidly decreased during the initial phase of training, dropping from above 1.4 to below 0.3 by step 300. Validation loss showed a similar trend, stabilizing around step 600. Simultaneously, validation accuracy surged from 28% to over 90% early in training and remained steady afterward. These dynamics indicate that the model quickly learned to classify news topics with high accuracy and avoided overfitting due to the lightweight LoRA architecture.

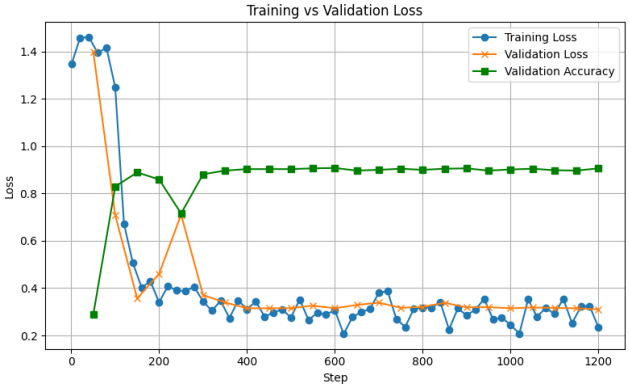

Figure 3: Training vs Validation Loss and Accuracy

Confusion Matrix

To further assess the model's classification performance, we compute a confusion matrix on the validation set (shown in Figure 4). The matrix reveals strong class-wise performance. The model classified most samples correctly across all four categories: **World**, **Sports**, **Business**, and **Sci/Tech**.

Notably, the model achieved near-perfect classification for the *Sports* class, with only 4 misclassifications. The *Sci/Tech* class also exhibited high performance with a minimal number of false predictions. The majority of misclassifications occurred between the *World* and *Business* categories, likely due to overlapping geopolitical-economic content. Despite these few confusions, the overall matrix indicates balanced and robust learning.

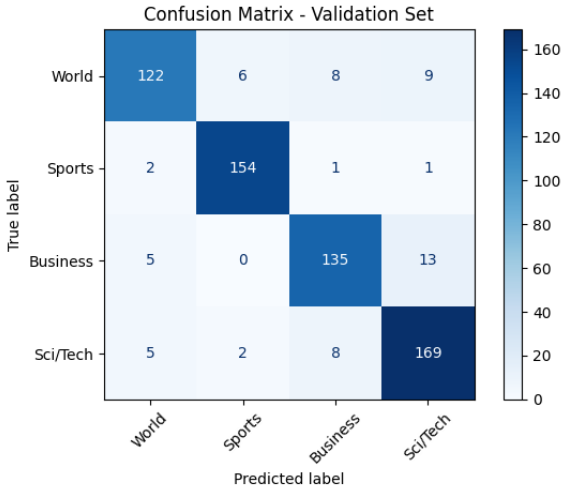

Figure 4: Confusion Matrix on Validation Set

Test Inference and Submission

To evaluate the model on completely unseen data, we ran inference on a hidden test set of 8,000 unlabeled samples provided as part of the Kaggle submission pipeline. The test set was preprocessed using the same tokenizer and pipeline as the training data. Predictions were generated in batches and saved to a CSV file in the required submission format.

A sample of the output is shown in Table 1. The full predictions were successfully uploaded to Kaggle for leaderboard evaluation.

ID	Predicted Label
0	3 (Sci/Tech)
1	0 (World)
2	0 (World)
3	3 (Sci/Tech)
4	1 (Sports)

Table 1: Sample Predictions on Test Set

Final Performance

Below is a concise summary of the model's final performance metrics and compliance with the project constraints:

- **Validation Accuracy:** 90.63% on the held-out 640-sample validation set.
- **Trainable Parameters:** 704,260 trainable parameters, constituting only 0.56% of the full 125M parameter model.
- **Model Architecture:** RoBERTa-base with LoRA adapters injected into attention submodules (query, key, value).
- **LoRA Configuration:** Rank $r = 2$, scaling factor $\alpha = 8$, dropout = 0.05.
- **Training Epochs:** 1 full epoch with a batch size of 16 and SGD optimizer (learning rate = 0.1).
- **Training Time:** Completed within the 1200 step limit.
- **Loss Trends:** Training and validation loss steadily decreased; validation accuracy remained stable above 90%.
- **Class-wise Performance:** Confusion matrix shows strong per-class performance with minimal confusion between similar categories.
- **Test Inference:** Predictions generated for 8000 test samples and saved as a CSV for Kaggle submission.
- **Hardware Used:** Fine-tuning performed on NVIDIA RTX 4070 Laptop GPU via Kaggle's GPU runtime.

References

Gang, Y.; Shun, J.; and Qing, M. 2025. Smarter Fine-Tuning: How LoRA Enhances Large Language Models.

OpenAI. 2023. ChatGPT: An AI Language Model. https://openai.com/chatgpt. Accessed: 2025-04-21.

Shun, J.; and Zheng, C. ???? Revolutionizing Large Model Fine-Tuning: The Role of LoRA in Parameter-Efficient Adaptation. Authorea Preprints. Accessed: 2025-04-21.

Tiwary, A.; Sarkar, S. D.; Singh, A. P.; Agarwal, P. K.; Burman, S.; and Poddar, R. 2025. Fine-Tuning Vision Transformer Using LoRA for Image Classification. In *2025 8th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech)*, 1–4. IEEE.

Ye, C.; and Shi, X. 2024. Optimizing News Topic Classification with Instructional Fine-Tuning of Chatglm3. In *Proceedings of the 3rd International Conference on Computer, Artificial Intelligence and Control Engineering*, 573–577.