Text Classification with RoBERTa and LoRA: A Parameter-Efficient Fine-Tuning Strategy

Sai Navyanth Penumaka[1], Karthik Sunkari[2], Geethika Rao Gouravelli[3]

sp8138@nyu.edu, ks7929@nyu.edu, gg2879@nyu.edu

New York University

Code: Github

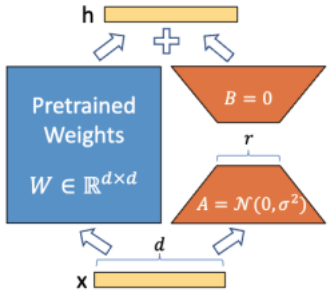
Abstract

In this project, we built a RoBERTa-based text classification model using Low-Rank Adaptation (LoRA) to fine-tune efficiently under a strict parameter limit. The task was to classify news articles from the AG News dataset while keeping the number of trainable parameters below one million. To achieve this, we froze the base model and trained only small LoRA adapters in the query and value attention projections. Along the way, we explored adding deeper classification heads, applying synonym-based data augmentation, and experimenting with regularization techniques like Monte Carlo Dropout and Early Stopping. Our final model reached a test accuracy of 85.63%, with only 667,396 trainable parameters, indicating that LoRA is an effective tool for fine-tuning large models when computing resources are limited.

Introduction

Large language models (LLMs) such as RoBERTa have demonstrated strong performance across a wide range of NLP tasks. However, fine-tuning these models end-to-end can be resource-intensive, especially in settings where compute or memory is limited. This has led to growing interest in parameter-efficient fine-tuning (PEFT) methods that allow for effective adaptation without updating the full model.

One such method is Low-Rank Adaptation (LoRA), which introduces small trainable matrices into the attention layers of transformers. These low-rank adapters are trained while the base model remains frozen, significantly reducing the number of trainable parameters and overall training cost without a major drop in performance.



LoRA Schematic

The pretrained weights $W \in R^{d \times d}$ are frozen. Only the low-rank matrices A and B are trained, with $B = 0$ and $A \sim N(0, \sigma^2)$. There are two main advantages to this approach. First, by training fewer parameters, LoRA makes gradient updates lighter and faster, which is very helpful when you're working with limited hardware. Second, because the base model stays untouched, it's easy to reuse or swap in different adapters depending on the task. The key hyperparameters in LoRA are the rank r of the low-rank update (which controls its capacity) and the scaling factor α, which determines how strongly the update influences the original weights.

In this project, we apply LoRA to a text classification task using the AG News dataset, which consists of four categories: World, Sports, Business, and Sci/Tech. RoBERTa is used as the base encoder, and we train only the LoRA adapters on top. We also explore several variations, including the addition of a feedforward classifier head, synonym-based data augmentation, and regularization techniques such as Monte Carlo Dropout and Early Stopping.

Our objective is to identify configurations that balance accuracy and efficiency, ultimately building a model that performs competitively while remaining under the 1 million trainable parameter limit.

Methodology

1. Dataset and Preprocessing:

We used the AG News dataset, which contains approximately 120K training and 7.6K test samples across four categories: World, Sports, Business, and Sci/Tech. Each sample includes a short news headline and description.

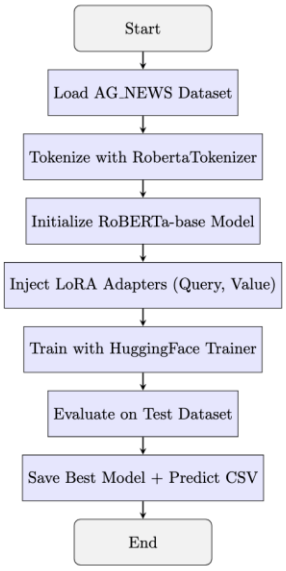
All text was tokenized using the 'RoBERTaTokenizer' with truncation and padding applied to a fixed maximum length of 64 tokens. Labels were mapped to integer indices for use with cross-entropy loss. We also experimented with synonym-based augmentation using the nlpaug library. Augmented samples were concatenated with the original training set, but this was disabled in the final run due to negligible gains under the parameter budget.

2. Data Augmentation:

To evaluate the effect of textual augmentation, we applied synonym substitution using the 'nlpaug' library with WordNet as the source. This technique was used to generate paraphrased versions of training samples, which were then combined with the original data to form an augmented dataset. While this increased dataset diversity, the overall impact on accuracy was minimal. To avoid introducing additional noise, augmentation was disabled in the final configuration.

3. Model Architecture and LoRA Configuration:

Our model is built on the roberta-base architecture, fine-tuned for sequence classification using Parameter-Efficient Fine-Tuning (PEFT) with Low-Rank Adaptation (LoRA). Rather than updating all model weights, we freeze the base encoder and inject trainable low-rank adapters into specific attention components—specifically the query and value projections.



Model Architecture

We use a LoRA rank of $r=3$, a scaling factor $\alpha=6$, and a dropout rate of 0.05. The classification head remains lightweight and unchanged from the default Hugging Face implementation. In one configuration, we also experimented with a deeper feedforward classifier, but the added complexity led to a parameter count exceeding our 1M limit.

Training was conducted using Hugging Face's Trainer API, with early stopping (patience = 5), linear learning rate scheduling, and standard evaluation metrics. The final model was trained for 1 epoch with a batch size of 32 and learning rate of 5×10^{-6}. All settings were controlled via a centralized configuration module for reproducibility.

The total number of trainable parameters in the final model was **667,396**, approximately 0.5% of the full roberta-base model, comfortably within the competition constraint.

4. Training Configuration:

We trained the model for a single epoch using a batch size of 32 and a learning rate of 5×10^{-6}, optimized with AdamW. A linear learning rate scheduler with warmup was used to gradually reduce the learning rate, which helped ensure smooth convergence. Early stopping was enabled with a patience of 5 to avoid overfitting, especially given the low number of trainable parameters. Evaluation was conducted every 500 steps, and the best-performing checkpoint was selected based on validation accuracy.

Parameter	Value
Model Architecture	roberta-base
LoRA Rank (r)	3
LoRA Alpha (α)	6
LoRA Dropout	0.05
Max Sequence Length	64
Train Batch Size	32
Epochs	1
Learning Rate	5×10^{-6}
Trainable Parameters	667,396
Frozen Base Model	Yes

Key training configuration and hyperparameters.

5. Experimentation journey:

Throughout the project, we iteratively tested various design choices to meet the performance goals while staying under the 1 million parameter limit:

- LoRA configuration: We experimented with multiple combinations of rank and scaling factor. A rank of 3 and α of 6 offered the best balance between expressiveness and size.
- Classification head: We initially tested a deeper feedforward classification head (FNN), which showed minor accuracy improvements. However, the added parameters pushed us over the limit, so we reverted to the simpler default head.
- Freezing strategy: We tried partially unfreezing upper layers of RoBERTa, but this didn't lead to meaningful gains and increased trainable parameters significantly. Freezing the entire base model and updating only LoRA adapters proved more effective.
- Regularization techniques: We applied dropout in both the adapter modules and classifier layers. Additionally, we tested Monte Carlo Dropout at inference time, which had a small but noticeable impact on prediction consistency.
- Data augmentation: We explored synonym-based augmentation using nlpaug, but it introduced variability without clear gains, which is likely due to the short and concise nature of AG News samples. It was excluded in the final training pipeline.

This experimentation cycle helped us refine a training configuration that was simple, stable, and efficient which helped in giving strong results under strict resource constraints.

Results and Findings:

1. Final Accuracy and Evaluation

Our final model achieved a test accuracy of 85.63% and validation accuracy of 88.10% on the AG News classification task. These results were obtained after training for a single epoch using early stopping with a patience of 5. Evaluation was performed using Hugging Face's Trainer API with accuracy as the primary metric.

Dataset	Accuracy
Validation	88.10%
Test	85.63%

Final performance of the LoRA-fine-tuned RoBERTa model.

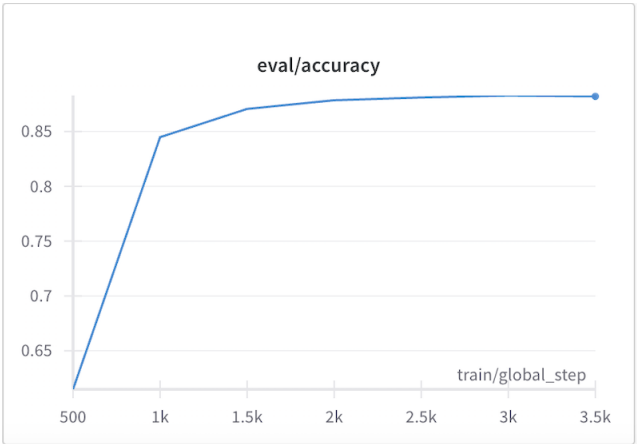
2. Parameter Budget and LoRA Efficiency

One of the key constraints of the competition was to remain under 1 million trainable parameters. Our LoRA-based setup introduced only 667,396 trainable parameters by modifying the query and value projections in RoBERTa's attention layers, while keeping the rest of the model frozen. This approach allowed us to train efficiently while maintaining strong performance.

3. Training Setup and Evaluation Curves

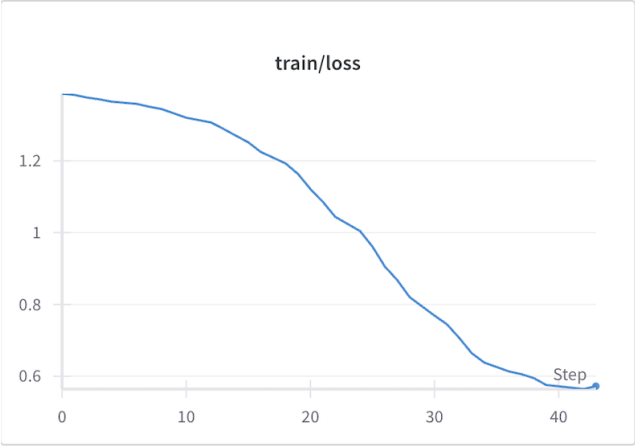
Training was conducted for a single epoch with a batch size of 32 and a learning rate of 5×10 −6, using the AdamW optimizer. We used a linear learning rate scheduler with warmup, and evaluation occurred every 500 steps. Early stopping was enabled to prevent overfitting, and the best model was selected based on validation accuracy.

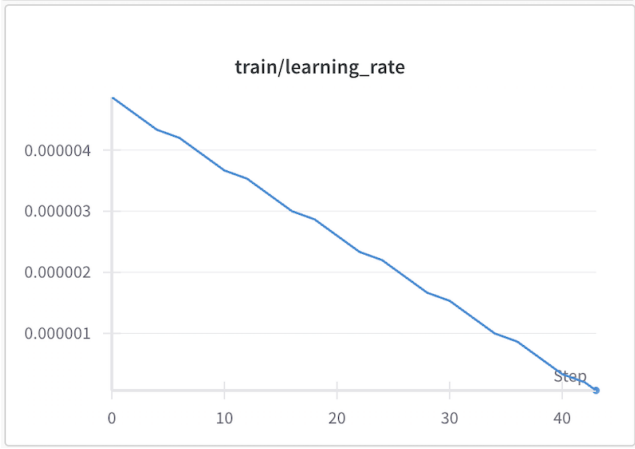
We visualized key training and evaluation metrics to better understand how the model converged. The evaluation accuracy curve shows that most of the gains were achieved early in training, with accuracy quickly rising and stabilizing around **85.6%**—closely matching our final test performance.



On the training side, the loss steadily decreased throughout the single epoch, confirming effective learning. The linear learning rate schedule also behaved as expected, decaying

smoothly toward zero. Together, these trends suggest that the model trained stably and efficiently, despite only updating a small number of adapter parameters.





4. Lessons Learned:

This project provided several key insights:
- LoRA enabled efficient fine-tuning with minimal overhead, updating only ~0.5% of parameters.
- Freezing the base model reduced compute and memory needs, without sacrificing performance.
- Adding deeper classification heads (e.g., FNNs) showed minor gains but often exceeded the parameter budget.
- Monte Carlo Dropout helped smooth predictions and improved generalization in some cases.
- Synonym-based data augmentation was less effective on short text inputs like AG News and occasionally introduced noise.

Conclusion:

Our experiments demonstrate that RoBERTa with LoRA provides an effective and lightweight solution for text classification under resource constraints. By combining adapter-based fine-tuning with thoughtful architecture and training design, we achieved a test accuracy of 85.63% with fewer than 1 million trainable parameters. These results highlight LoRA's potential for low-resource scenarios, offering strong performance without the need for full model fine-tuning.

Citations:

1. *Liu, Y., et al. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv preprint arXiv:1907.11692*
2. *Hu, E. J., et al. (2021). LoRA: Low-Rank Adaptation of Large Language Models. arXiv preprint arXiv:2106.09685*
3. *Lewis, M., et al. (2020). Pretrained Language Models for a Multilingual News Dataset. AG News*
4. *Paszke, A., et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. NeurIPS*
5. *Wolf, T., et al. (2020). Transformers: State-of-the-Art Natural Language Processing. EMNLP*
6. *Acknowledgement: We used chatGPT to refine and structure our code.*