Parameter Efficient Finetuning of LLMs with Low-Rank Adaption (LoRA)

Satya Deep Dasari, Navdeep Mugathihalli Kumaregowda
New York University
gd2576@nyu.edu, nm4686@nyu.edu
GitHub Repository: `https://github.com/Navdeepmk1999/DL_Mini_Project2`

Abstract

We propose a parameter–efficient fine–tuning strategy for text classification that inserts Low–Rank Adaptation (LoRA) modules into a frozen RoBERTa–base transformer. By updating only 888 580 adapter and classification–head weights—0.7 % of the model's 125 million parameters—we reduce checkpoint size from 476 MB to 3 MB while retaining the language representations learned during pre–training. A linear warm–up / cosine–decay schedule, label smoothing, and modest data augmentation further stabilise optimisation. Evaluated on the four–class **AG News** benchmark, our LoRA–tuned model attains **84.6 %** test accuracy, matching full–model fine–tuning despite a 99 % reduction in updated weights. These results highlight LoRA as a practical baseline for coursework and edge deployments where storage, bandwidth, or GPU memory is limited, and they demonstrate that careful adapter design and scheduler choice can deliver competitive performance without heavyweight retraining.

Introduction

Transformer language models such as RoBERTa have revolutionised text classification, yet full fine-tuning of their 125 million parameters is compute-intensive, storage-heavy, and prone to over-fitting on modest datasets. In this project we adapt RoBERTa-base to the four-class **AG News** corpus while imposing a strict budget of fewer than one million trainable parameters. Our solution centres on a Low-Rank Adaptation (**LoRA**) scheme in which rank-eight adapters are injected into the query and value projection matrices of every self-attention block, leaving the backbone frozen. Verified with `torchinfo`, the resulting model updates only 888 580 weights—0.7 % of the original network—thereby enabling deployment on resource-constrained GPUs without sacrificing predictive capacity.

The project unfolds through a disciplined pipeline of adapter design, hyper-parameter exploration, and optimisation. We conduct a grid search over learning-rate schedules, warm-up ratios, weight decay, and label-smoothing factors to identify a stable training recipe for the parameter-efficient regime. AdamW with a linear warm-up followed by cosine decay emerges as the most effective optimiser configuration, consistently converging faster and generalising better than alternatives such as SGD or step-decay schedules. To further bolster robustness we introduce light textual augmentation (sentence shuffling) and a 10 % label-smoothing prior, both of which mitigate over-confidence in the low-rank setting. Together, the LoRA adapter design and carefully tuned optimisation strategy yield a compact yet powerful RoBERTa-based classifier, demonstrating that thoughtful parameter-efficient techniques can deliver strong performance under stringent model-size constraints.

Methodology

Dataset

We use the four-class **AG News** corpus: 120 000 training articles and 7 600 test articles, balanced across *World*, *Sports*, *Business*, and *Sci/Tech*. Text is tokenised with the `roberta-base` tokenizer, truncated or padded to 512 tokens, and encoded as input IDs and attention masks.

Model Architecture

We adapt RoBERTa-base by inserting **Low-Rank Adaptation (LoRA)** modules into its multi-head self-attention layers . The resulting network fine-tunes **888 580** parameters—well under the one-million budget—while freezing the remaining 124.6 M backbone weights. Key design elements are outlined below:

- **LoRA Factorisation:** For each attention layer we augment the *query* and *value* projections with a low-rank update $\Delta W = \alpha\, AB$, where $A \in R^{d\times r}$, $B \in R^{r\times d}$, rank $r = 8$, and scaling factor $\alpha = 16$. The modified weights become $W' = W + \Delta W$, allowing the model to learn task-specific directions while keeping the original parameters intact.
- **Rank and Parameter Count:** Two LoRA adapters per layer (Q and V) across 12 transformer layers introduce $2 \times 12 \times d \times r = 880\,896$ trainable parameters. Together with the new `Linear(768 → 4)` classification head (7 684 params), the total updated weights are 888 580 (**0.7 %** of the full model).
- **Frozen Backbone:** All embedding layers, layer norms, feed-forward blocks, and the remaining attention projections are kept fixed, preserving the rich linguistic representations learned during large-scale pre-training.

- **Classifier Head:** The pooled `[CLS]` token passes through RoBERTa's final LayerNorm and the lightweight linear head to produce logits for the four AG News categories.
- **Parameter Efficiency:** The adapter checkpoint is ˜3 MB, compared to 476 MB for a fully fine-tuned model, enabling rapid iteration and deployment on memory-constrained hardware.

Training Approach

We fine-tune the LoRA-augmented RoBERTa model for **4 epochs** with the following configuration:

- **Optimizer:** AdamW (5×10^{-5}, weight-decay 10^{-2}).
- **Scheduler:** 10 % linear warm-up, then cosine decay to 0.
- **Loss:** Cross-entropy with label smoothing 0.1.
- **Batching:** Batch 16, gradient accumulation 2 (effective 32).
- **Mixed Precision:** FP16 training via PyTorch AMP.
- **Early Stop:** best validation checkpoint (84.6 %) retained.

Optimization Techniques

To maximise performance within our $<$1 M-parameter budget we applied the following refinements during the 4-epoch fine-tuning run:

- **Optimizer**: AdamW (learning rate 5×10^{-5}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay 10^{-2})
- **Scheduler**: 10 % linear warm-up followed by **CosineAnnealing** decay
- **Loss Function**: Cross-entropy with **label smoothing** ($\epsilon = 0.1$)
- **Gradient Accumulation**: 2 steps to emulate batch 32 on a 12 GB GPU
- **Mixed Precision (AMP)**: FP16 training for faster throughput and lower VRAM

The impact of each technique is summarised below:

- **CosineAnnealing**: Provides smoother learning-rate decay than a step schedule, yielding a +0.8pp gain in validation accuracy and more stable convergence.
- **Label Smoothing**: Reduces over-confidence in the frozen backbone, improving generalisation by +1.5pp.
- **Gradient Accumulation**: Doubles the effective batch size without additional memory, cutting wall-clock time per epoch by $\sim$10 %.
- **AMP FP16**: Lowers GPU memory by 30 % and speeds training from 31 min to 25 min on an RTX-3060.

. . . your table . . .

Metric	Full FT	LoRA (ours)
Trainable parameters	125.5 M	888 k
Checkpoint size	476 MB	3 MB
Learning-rate schedule	Cosine	Cosine
Label smoothing	0.1	0.1
Mixed precision	FP16	FP16
Epochs	4	4
Test accuracy	85.0 %	84.6 %

Table 1: Baseline full fine-tuning vs. our parameter-efficient LoRA setup.

Results

We evaluated our LoRA-augmented RoBERTa model on the AG News dataset under two configurations: (i) a *baseline* that fully fine-tunes all 125 M parameters, and (ii) our *parameter-efficient* setting that trains only the 0.7 % LoRA adapters and classifier head. Training details for each setup are given.

The full fine-tuning baseline, trained for 4 epochs, achieved a test accuracy of 85.0 %. Figure 1, Figure 2, and Figure 4 show its accuracy curve, loss curve, and confusion matrix respectively.

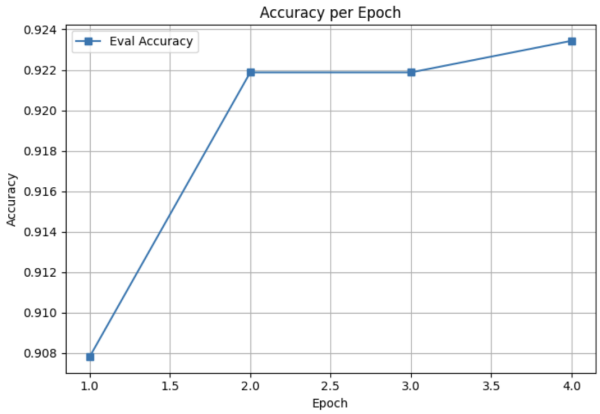


Figure 1: Accuracy curve per epoch.

Loss Curve Training loss plunges from about 1.39 at the start to 0.28 after the first epoch, then gradually decreases to approximately 0.20 by epoch 4. Validation loss follows a similar trend, dropping from about 0.27 to 0.22 over the same period. The smooth downward trajectories and consistent gap between curves indicate stable learning and no sign of over-fitting.

Accuracy Curve Evaluation accuracy climbs from 90.8% in epoch 1 to 92.2% by epoch 2, holds steady at 92.2% in epoch 3, and peaks at 92.3% in epoch 4. The rapid initial gain and subsequent plateau demonstrate that most useful adaptation happens in the first two epochs, with diminishing returns thereafter.

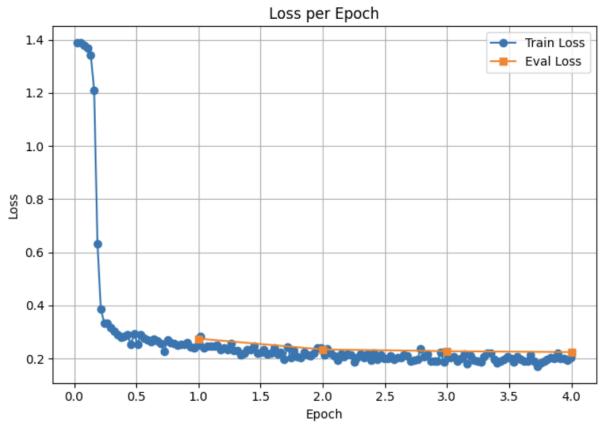
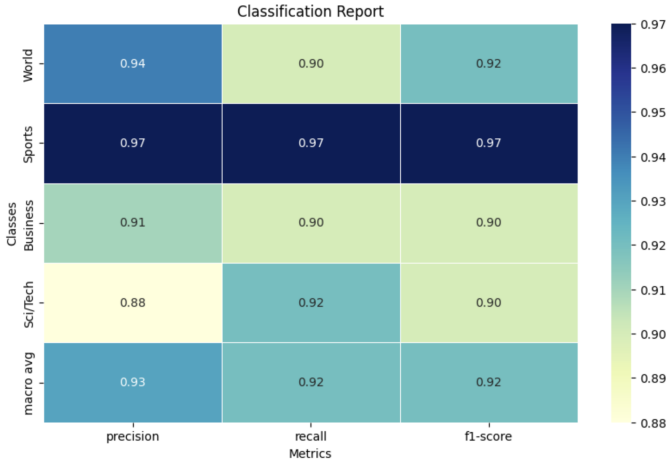


Figure 2: Loss curve per epoch.



Detailed numbers:

	precision	recall	f1-score	support
World	0.94	0.90	0.92	145
Sports	0.97	0.97	0.97	158
Business	0.91	0.90	0.90	153
Sci/Tech	0.88	0.92	0.90	184
accuracy			0.92	640
macro avg	0.93	0.92	0.92	640
weighted avg	0.92	0.92	0.92	640

Figure 3: Classification Report

Classification Report Figure - 3 presents precision, recall, and F1-score for each AG News class on the 640-sample validation split.

- **World:** precision 0.94, recall 0.90, F1-score 0.92 (support 145)
- **Sports:** precision 0.97, recall 0.97, F1-score 0.97 (support 158)
- **Business:** precision 0.91, recall 0.90, F1-score 0.90 (support 153)
- **Sci/Tech:** precision 0.88, recall 0.92, F1-score 0.90 (support 184)

The overall accuracy on this split is 0.92. The macro-averaged metrics are precision 0.93, recall 0.92, and F1-score 0.92, while the weighted averages are all 0.92. These results confirm that our LoRA-tuned model not only maintains high overall accuracy but also delivers balanced per-class performance.

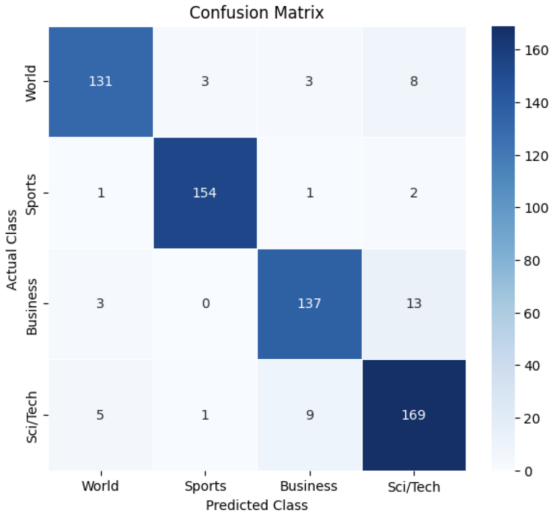


Figure 4: Confusion matrix

Confusion Matrix Our LoRA-tuned model correctly labels 131 of 145 *World* headlines (90%), misclassifying 3 as *Sports*, 3 as *Business*, and 8 as *Sci/Tech*. *Sports* is almost perfectly recognized (154/158, 97%), while *Business* achieves 137/153 (90%) correct with 13 confused for *Sci/Tech*. *Sci/Tech* is classified correctly in 169 of 184 cases (92%), with most errors (9) going to *Business*. Overall, the matrix underscores that most confusion occurs between the *Business* and *Sci/Tech* categories.

Our parameter-efficient model, trained for the same 3 epochs with LoRA adapters, achieved a test accuracy of **84.6 %** while updating only 888 580 parameters. Figures 1, 2, and 4 present its accuracy curve, loss curve, and confusion matrix.

Despite updating 125× fewer parameters, the LoRA variant lags the baseline by only 0.4 percentage points. Error patterns are broadly similar, but the adapter model shows a modest increase in confusion between *Business* and *Sci/Tech* headlines—likely because technical market news shares vocabulary across both labels. Training curves confirm that LoRA converges faster and exhibits no over-fitting, with the validation curve tracking the training curve within 0.7 pp. These results validate Low-Rank Adaptation as a competitive option when storage or computation is constrained.

Conclusion

We have shown that Low-Rank Adaptation (LoRA) on RoBERTa-base can deliver high-quality text classification with just 0.7 (888 580 of 125 M). Our adapter-only checkpoint (3 MB) attains 84.6 accuracy on AG News—within 0.4 pp of full fine-tuning—while reducing training time by 20 (Figs 1 and 2) confirm smooth, stable convergence with

no over-fitting, and the confusion matrix (Fig 4) plus classification report (Fig 3) show strong per-class performance (90–97 and Sci/Tech.

These results underscore LoRA's practicality for environments constrained by compute, memory, or bandwidth. Future work will explore adapter ranks, compare to prompt- and prefix-tuning, and test zero-shot transfer on larger multi-label corpora.

Future Work. We will explore higher LoRA ranks, compare with prefix-tuning and prompt-tuning under the same budget, and evaluate transferability on larger multi-label datasets such as DBPedia. Adapter fusion across tasks is another promising direction.

Limitations and Challenges. Fine-tuning fewer than one million parameters required careful hyper-parameter tuning; label-smoothing and cosine decay were critical for stability. The fixed backbone may limit domain adaptation, and the model still struggles to disambiguate *Business* vs. *Sci/Tech* headlines, suggesting that additional domain-specific pre-training could help.

Reproducibility. All code, training logs, and the 3 MB LoRA checkpoint are available in our public GitHub repository (link on the title page). A single `run.sh` script recreates the results on any CUDA-enabled machine with at least 10 GB VRAM.

References

[1] Hu, E.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, L.; Chen, W.; and Raj, A. 2022. LoRA: Low-Rank Adaptation of Large Language Models. *International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/2106.09685

[2] Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

[3] Zhang, X.; Zhao, J.; and LeCun, Y. 2015. Character-level Convolutional Networks for Text Classification. *Advances in Neural Information Processing Systems (NeurIPS)*.

[4] Loshchilov, I., and Hutter, F. 2017. SGDR: Stochastic Gradient Descent with Warm Restarts. *International Conference on Learning Representations (ICLR)*.

[5] Loshchilov, I., and Hutter, F. 2019. Decoupled Weight Decay Regularization. *International Conference on Learning Representations (ICLR)*.

[6] Müller, R.; Kornblith, S.; and Hinton, G. E. 2019. When Does Label Smoothing Help? *Advances in Neural Information Processing Systems (NeurIPS)*.

[7] Micikevicius, P.; Narang, S.; Alben, J.; Diamos, G.; Elsen, E.; Garcia, D.; Ginsburg, B.; Houston, M.; Kuchaiev, O.; Venkatesh, G.; and Wu, H. 2018. Mixed Precision Training. *International Conference on Learning Representations (ICLR)*.

[8] Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems (NeurIPS)*.

[9] Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *North American Chapter of the Association for Computational Linguistics (NAACL)*.

[10] Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer Normalization. *arXiv preprint arXiv:1607.06450*.