Fine-tuning RoBERTa with Low-Rank Adaptation for News Classification

Team Members:
Neha Patil - np2998
Ruchi Jha - rj2807
Satvik Upadhyay – su2250

Abstract

This paper explores parameter-efficient fine-tuning of a pre-trained RoBERTa model for news classification using Low-Rank Adaptation (LoRA). On the AG News dataset, we optimized LoRA configuration and training settings to stay under 1 million trainable parameters. Our approach achieved 92.71% validation accuracy, with 84.73% and 83.75% on the public and private test sets, respectively. The results highlight the effectiveness of LoRA for high-performance text classification under tight parameter budgets.

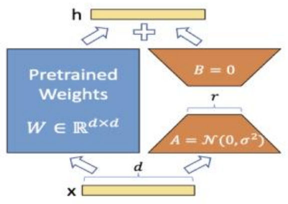

h
Pretrained Weights
$W \in \mathbb{R}^{d \times d}$
$B = 0$
r
$A = \mathcal{N}(0, \sigma^2)$
d
x

Figure 1: Our reparametrization. We only train A and B.

Figure 1: Block diagram of LoRA

The figure above highlights the structure of a LoRa model.
Github Link

Methodology

We fine-tuned a RoBERTa-based transformer model for news classification on the AG News dataset under a 1 million trainable parameter constraint using Low-Rank Adaptation (LoRA). The dataset contains 120,000 training and 7,600 test samples evenly split across four classes: World, Sports, Business, and Science/Technology.

Our preprocessing included:
- **Text normalization:** replacing newline characters with spaces and stripping whitespace for consistent formatting
- **HTML entity replacement**: converting common entities like " and & to their character equivalents
- **Tokenization**: using RoBERTa's tokenizer with a maximum sequence length of 128, which was sufficient for most articles while maintaining computational efficiency
- **Padding**: set to "max_length" to ensure uniform input dimensions across each batch

To stay within the parameter budget, we applied LoRA with rank 4 and scaling factor 16 to the attention query, value, and output dense layers. We used HuggingFace's datasets, transformers, and PEFT libraries for implementation.

Model training involved:
- Loading and preprocessing the AG News dataset
- Tokenizing inputs using the roberta-base tokenizer with max_length set to 128
- Integrating LoRA modules using the PEFT library
- Tuning hyperparameters including batch size, learning rate, dropout, number of epochs, optimizer, and learning rate scheduler to optimize performance under the constraint.

Model Architecture

Our goal was to fine-tune a RoBERTa-base model (125M parameters) on the AG News dataset using Low-Rank Adaptation (LoRA), while staying under a strict constraint of 1 million trainable parameters. The final configuration resulted in 999,172 trainable parameters, just 0.795% of the full model, achieving a strong balance between performance and efficiency.

We initially explored a low-rank (r=2), high-alpha (α=32) LoRA setup targeting only the query and value layers. While parameter-efficient, this configuration yielded only ~82% accuracy due to limited capacity and overly strong perturbations. We iteratively refined our architecture and training procedure as follows:

Final Model Architecture:

- Base Model: RoBERTa-base with 12 transformer layers, 768 hidden dimensions, and 12 attention heads
- Task: Sequence classification with 4 output labels
- LoRA Configuration:
  - Rank (r): 4
  - Alpha (α): 16
  - Target modules: query, value, output.dense
  - LoRA dropout: 0.05
  - Bias: none

• Tokenizer: roberta-base with add_prefix_space=True, max sequence length 128

This LoRA setup allowed expressive adaptation through low-rank matrices while maintaining a small parameter footprint. The rank of 4 was chosen as a sweet spot—higher than initial trials to improve capacity, but not too large to exceed the budget. The α=16 scaling factor applied an effective magnitude of 4 to the LoRA updates (α/r), compensating for the restricted rank.

Focusing LoRA only on attention and output projection layers proved effective, as these modules are critical for capturing task-relevant patterns. The dropout of 0.05 added regularization, reducing overfitting.

Training Strategy and Improvements:

We tuned several hyperparameters to optimize performance:

• Learning rate: tested from 1e-5 to 5e-5
• Batch sizes: 4, 8, and 16
• Schedulers: linear decay, cosine with restarts
• Label smoothing: 0.05
• Gradient accumulation: 4 steps

These experiments highlighted the importance of learning rate and batch size for convergence. Schedulers and label smoothing further improved generalization. With the final configuration, the model reached 92.71% validation accuracy in just 3 epochs. Trends in loss suggest 5–6 epochs may further improve performance, especially for underrepresented classes.

By freezing all pre-trained weights and only training LoRA modules, we drastically reduced memory and computation costs while preserving the expressive power of RoBERTa.

Our training pipeline was designed to maximize performance while adhering to the 1 million parameter constraint. We used HuggingFace's transformers and PEFT libraries, training on a GPU-enabled Kaggle environment (GPU T4 x2) with mixed precision to optimize both speed and memory usage.

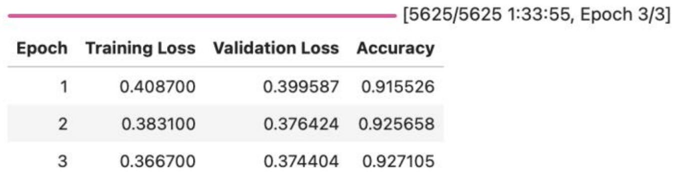
Training Configuration:
• Framework: HuggingFace Transformers + PEFT (LoRA)
• Epochs: 3
• Batch Size: 8 (training), 64 (evaluation)
• Optimizer: AdamW with betas (0.9, 0.999), ε = 1e-8
• Scheduler: Cosine with restarts
• Warmup Ratio: 0.15
• Weight Decay: 0.01
• Label Smoothing: 0.05
• Mixed Precision: BF16 (bfloat16)
• Evaluation Steps: 500
• Early Stopping: Enabled by saving the best model based on validation loss

Optimization Strategy:

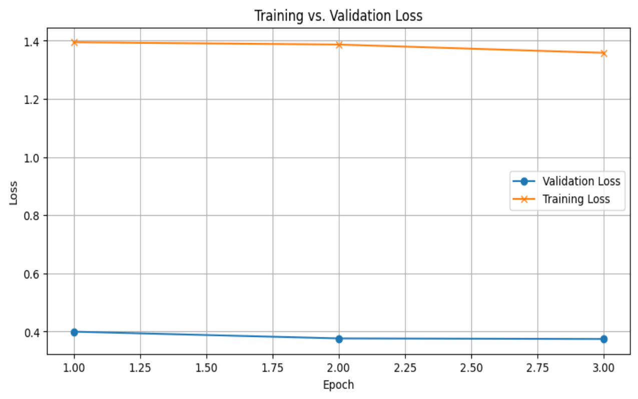
• Cosine Scheduler with Restarts: Helps escape local minima and accelerates convergence
• Warmup Phase (15%): Smoothly ramps up the learning rate to prevent unstable updates during initial steps
• Label Smoothing (0.05): Reduces overconfidence in predictions, improving generalization
• Gradient Accumulation (4 steps): Simulates a larger batch size without exceeding memory limits
• Mixed Precision Training (BF16): Accelerates computation while maintaining numerical stability

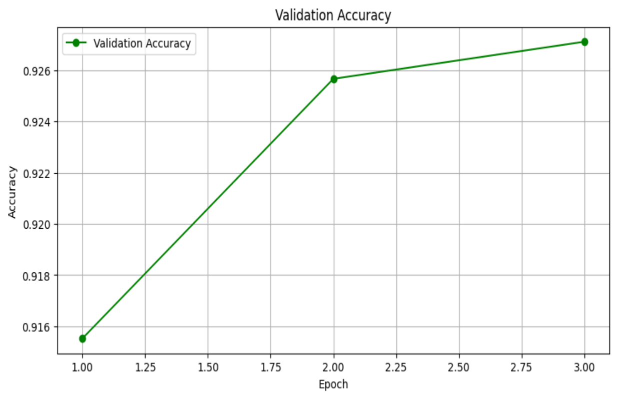
Evaluation and Model Selection

During training, we evaluated the model after each epoch on the validation set, using accuracy as the primary metric. The training process showed consistent improvement across epochs:


[5625/5625 1:33:55, Epoch 3/3]

Epoch	Training Loss	Validation Loss	Accuracy
1	0.408700	0.399587	0.915526
2	0.383100	0.376424	0.925658
3	0.366700	0.374404	0.927105





Results

Final Accuracy Metrics:
• Validation Accuracy: 92.71%
• Kaggle Public Test Score: 84.73%
• Kaggle Private Test Score: 83.75%

These results demonstrate the effectiveness of Low-Rank Adaptation (LoRA) in achieving strong performance on a constrained budget of under 1 million trainable parameters.

Prediction Distribution (8,000 Test Samples):
• Class 0 (World): 1,552 samples (19.4%)
• Class 1 (Sports): 2,000 samples (25.0%)
• Class 2 (Business): 1,827 samples (22.8%)
• Class 3 (Science/Tech): 2,621 samples (32.8%)

The distribution shows a mild skew toward Science/Technology predictions, which may reflect either a learned bias in the model or characteristics of the test set itself.

Performance Insights:
Despite achieving **92.71%** validation accuracy, the test scores (83.75% private, 84.73% public) reveal a performance gap that may be attributed to mild overfitting or a distribution shift between training/validation and test data. Nevertheless, the model performed robustly under tight constraints.

Key Findings:

1. LoRA Enables Scalable Fine-Tuning:
Our results confirm that LoRA can match or exceed the performance of larger fine-tuned models using <1% of the original parameter count, making it ideal for resource-constrained environments.

2. Targeted Parameter Adaptation Is Effective:
Applying LoRA only to the attention query, value, and output projection layers was sufficient for strong performance, suggesting that selective adaptation is often enough for downstream classification tasks.

3. Hyperparameters Matter Greatly:
The combination of cosine scheduling, gradient accumulation, warmup, and label smoothing played a crucial role in achieving high accuracy. This highlights the need to focus not only on model architecture but also on training dynamics.

4. Balanced Regularization Yields Strong Generalization:
Regularization techniques like dropout (0.05) and weight decay (0.01) helped prevent overfitting, contributing to the model's generalization on unseen data.

Limitations and Future Work

While our LoRA-based fine-tuning approach achieved strong results, there remain several areas for improvement and exploration in future work.

1. Sequence Length Optimization:
Our fixed maximum sequence length of 128 tokens may have truncated longer articles, potentially omitting useful context. Future work could explore dynamic sequence lengths, hierarchical encoding, or more sophisticated pooling strategies to retain more information from longer texts.

2. Expanded LoRA Configuration Search:
Although our configuration (rank=4, α=16) performed well, a more extensive per-layer search across different ranks and scaling factors could unlock further gains. Techniques like layer-wise LoRA customization or rank tuning could better match the representational needs of each transformer block.

3. Training Schedule and Strategy Enhancements:
We limited training to 3 epochs. Extending this to 5–6 epochs or using early stopping could help the model learn more nuanced class features, particularly for minority classes. Gradient-based adaptive epoch scheduling could also be investigated.

4. Data Preprocessing and Augmentation:
While our preprocessing pipeline was effective, further improvements—such as advanced text normalization, handling special characters, or removing noise—may help. Lightweight augmentation techniques like EDA (Easy Data Augmentation), synonym replacement, or back-translation could improve robustness and address prediction skew.

5. Class Imbalance and Prediction Skew:
Our model predicted Class 3 (Science/Tech) more frequently than others. This skew may be mitigated using better class balancing, re-weighting strategies, or synthetic oversampling for underrepresented classes.

6. Regularization and Generalization Techniques:
Although dropout (0.05) and weight decay (0.01) were effective, more advanced strategies like DropPath, stochastic depth, or knowledge distillation from a larger teacher model could enhance generalization while keeping the parameter count low.

7. Model Ensembling and Test-Time Strategies:
Combining predictions from multiple LoRA-adapted models with varying configurations or applying test-time augmentation could improve robustness and reduce variance.

Lessons Learnt:

• Smaller LoRA ranks caused underfitting, while larger α values destabilized training
• A limited number of epochs constrained performance on complex or rare examples
• Careful balance between parameter efficiency and expressiveness is crucial
• Training dynamics, especially learning rate scheduling and warmup, significantly impact outcomes

By addressing these limitations and incorporating the proposed improvements, future iterations of this work could push accuracy even higher while continuing to respect strict resource constraints.

Conclusion

Despite strict constraints, our final model was robust, highly efficient, and adaptable, demonstrating the power of LoRA for parameter efficient fine tuning. By carefully configuring LoRA adaptations and optimizing the training pipeline, we achieved strong performance on the AG News dataset while keeping trainable parameters under 1 million.

Our results underscore the effectiveness of selective parameter updates, thoughtful regularization, and well-tuned training dynamics in enabling high performance text classification with minimal resources. As transformer models continue to grow, techniques like LoRA will be essential for extending the reach of state-of-the-art NLP to resource constrained settings and real-world applications.

References

For this model to work, we took some helps from LLMs like ChatGPT. But the underlying model was provided, and the key tweaks and methodologies were all our work.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2021). LoRA: Low-Rank Adaptation of Large Language Models. arXiv preprint arXiv:2106.09685.

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level Convolutional Networks for Text Classification. Advances in Neural Information Processing Systems, 28.