Efficient RoBERTa Fine-Tuning for AGNEWS Classification via LoRA

Sudharshan Ramesh, sr7431 —— Mohammed Hamid, mh7483
Github: [link]

Abstract

This project, part of the NYU Deep Learning Course, investigates the Parameter-Efficient Fine-Tuning (PEFT) of RoBERTa-base for AGNEWS topic classification using Low-Rank Adaptation (LoRA). The AGNEWS dataset consists of news articles categorized into 4 classes (World, Sports, Business, Sci/Tech). Our primary objective was effective classification under a strict constraint of fewer than 1 million trainable parameters, achieved using the Hugging Face PEFT library. By implementing RoBERTa-base with a highly optimized LoRA configuration (rank $r = 3$, targeting `query` and `dense` attention projections), we attained a final submission accuracy of 86.4%. This work details the LoRA setup, training methodology, iterative tuning process, and evaluation, demonstrating significant efficiency gains.

Overview

In this project, we finetuned an efficient text classification model for the AGNEWS dataset using the RoBERTa architecture combined with LoRA (Hu et al. 2022) , ensuring the number of trainable parameters remains under 1 million. The AGNEWS dataset contains over 120,000 news articles, providing a substantial benchmark for classifying text into 4 distinct topic categories (World, Sports, Business, Sci/Tech).

The RoBERTa model (Liu et al. 2019), while powerful, typically requires fine-tuning over 125 million parameters for `RoBERTa-base`. Standard fine-tuning updates all parameters, demanding significant computational resources. To overcome this, we utilized LoRA, a PEFT technique implemented in the Hugging Face 'peft' library (PEFT contributors 2022). LoRA freezes the vast majority of pre-trained weights and introduces a small number of trainable low-rank matrices into specific layers, enabling effective adaptation with minimal parameter overhead. Our focus is on optimizing the LoRA configuration for `RoBERTa-base` to maximize classification performance on AGNEWS within the stringent 1 million trainable parameter budget, exploring very low ranks and specific target modules.

Methodology

We aimed to configure LoRA for the `RoBERTa-base` model efficiently for AGNEWS classification, targeting high accuracy while keeping trainable parameters below 1 million. The methodology involved base model selection, specific LoRA configuration design using the 'peft' library, tailored training pipeline setup using 'transformers' (Wolf et al. 2020), and iterative tuning.

Architecture Design (LoRA Configuration)

We utilized the pre-trained `roberta-base` model accessed via the Hugging Face Transformers library as our foundation. Its 125 million parameters remained frozen during LoRA training.

1. **Parameter-Efficient Fine-Tuning (PEFT) via 'peft' Library:** LoRA was implemented using 'LoraConfig' and 'get_peft_model' from the 'peft' library.
2. **LoRA Configuration ('LoraConfig'):** A highly parameter-efficient configuration was finalized after experimentation:
   - **Rank ('r'):** Set to a very low rank $r = 3$.
   - **Scaling ('lora_alpha'):** Set to `lora_alpha` $= 6$, maintaining the $\alpha = 2r$ ratio.
   - **Target Modules ('target_modules'):** Adapters were applied to the `["query", "dense"]` projection matrices within the self-attention blocks. Targeting the attention output (`dense`) instead of `value` or `key` was found effective in this low-rank setting.
   - **Bias ('bias'):** Set to `"none"`. Only the LoRA matrices were trainable.
   - **Dropout ('lora_dropout'):** A dropout rate of 0.1 was applied to the LoRA layers.
   - **Task Type:** Explicitly set to `TaskType.SEQ_CLS` for sequence classification.
3. **Parameter Count Verification:** This specific configuration ($r = 3$, targets=`["query", "dense"]`, bias=`"none"`) resulted in 985,348 (0.98M) trainable parameters, verified using `model.print_trainable_parameters()`. This is substantially below the 1 million limit, constituting

less than 0.1% (0.7843%) of the base model's total parameters.

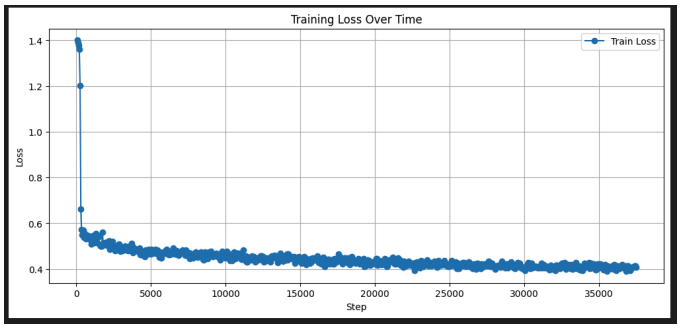
Data Augmentation

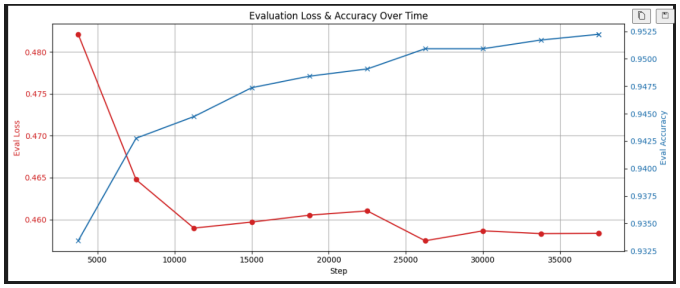
No explicit text data augmentation techniques were used beyond the standard tokenization performed by the `roberta-base` tokenizer on the original AGNEWS training dataset.

Training Strategy

The Hugging Face 'Trainer' API facilitated the training process with the following optimized settings:

1. **Optimizer:** The AdamW optimizer (Loshchilov and Hutter 2017) was used (default optimizer for Hugging Face Transformers) with a weight decay (`weight_decay`) value of 0.01.
2. **Learning Rate Scheduler:** A cosine learning rate scheduler (`lr_scheduler_type="cosine"`) was used with a relatively high peak learning rate (`learning_rate`) of 3×10^{-4}. A warmup phase constituting 6% of training steps (`warmup_ratio=0.06`) was employed.
3. **Loss Function:** Standard Cross-Entropy Loss suited for multi-class classification was used. Label smoothing with a factor of 0.1 (`label_smoothing_factor=0.1`) was applied.
4. **Batch Size and Gradient Accumulation:** Per-device training batch size of 16 (`per_device_train_batch_size=16`) with gradient accumulation over 2 steps (`gradient_accumulation_steps=2`), yielding an effective batch size of 32. Evaluation batch size (`per_device_eval_batch_size`) was 64.
5. **Mixed Precision Training:** Automatic mixed precision (`fp16=True`) was utilized.
6. **Training and Evaluation Loop:** Training ran for a maximum of 10 epochs (`num_train_epochs=10`). Evaluation (`evaluation_strategy="epoch"`) and checkpoint saving (`save_strategy="epoch"`) occurred after each epoch. The best model checkpoint based on validation accuracy (`metric_for_best_model="accuracy"`) was loaded at the end (`load_best_model_at_end=True`). An 'EarlyStoppingCallback' with a patience of 3 epochs was active.





Hyperparameter Tuning Process

We performed hyperparameter tuning using a manual, iterative approach rather than automated frameworks. To efficiently explore the parameter space under the project constraints, we adopted **a two-stage strategy**:

1. **Exploratory Short Runs:** Initial experiments with different combinations of key parameters were conducted for only **3 epochs** each. This allowed for rapid assessment of various settings and identification of promising configurations based on early validation performance trends. The main parameters explored in this stage included:
   - LoRA Rank ('r'): Testing values like 3, 8, 10, 16 while respecting the parameter limit.
   - LoRA Target Modules (`target_modules`): Comparing combinations such as `["query", "value"]`, `["query", "value", "key"]`, and `["query", "dense"]`.
   - Learning Rate (`learning_rate`): Evaluating different rates like 3×10^{-5}, 5×10^{-5}, and 3×10^{-4}.
   - LR Scheduler Type (`lr_scheduler_type`): Comparing different decay strategies (e.g., linear vs. cosine).
2. **Selection and Final Training:** The configuration demonstrating the most promising learning behavior and validation accuracy within the initial 3-epoch runs was selected.
3. **Full Training Run:** This selected configuration (which was ultimately $r = 3$, targets=`["query", "dense"]`, LR=3×10^{-4}, cosine scheduler) was then used for the final, complete training run over a maximum of **10 epochs**. This run incorporated early stopping and saved the best checkpoint based on validation accuracy, leading to the final reported results.

This two-stage approach allowed for efficient exploration of the hyperparameter space while ensuring the final model was trained thoroughly using the best-identified settings.

Tuning Challenges and Observations

Based on the iterative tuning process (summarized in Table 1), we observed the following:

1. **Target Module Impact:** Different combinations of target modules were explored, including targeting only `query` ($r = 22$), standard attention components

(`["query", "value"]` with $r = 11$, and `["query", "value", "key"]` with $r = 7$), and the less common `["query", "dense"]` ($r = 3$). The configuration targeting `["query", "dense"]` yielded the best validation performance (94.7%) in initial 3-epoch tests.

2. **Rank vs. Target Interplay:** Achieving optimal results involved balancing rank and target modules. The very low rank $r = 3$ configuration, when paired specifically with the `["query", "dense"]` targets, slightly outperformed configurations using higher ranks but different target modules (e.g., $r = 7$ or $r = 11$) in the initial 3-epoch exploration phase.
3. **Parameter Limit Constraint:** All promising configurations tested were designed to approach, but not exceed, the 1 million trainable parameter limit, settling around 0.98-0.99M parameters. This highlights the challenge of maximizing adaptation capacity right up to the strict budget boundary.
4. **Learning Rate:** A relatively high learning rate (3×10^{-4}) remained effective for the final configuration's 0.98M trainable parameters.

Lessons Learned

1. **High Efficiency Fine-Tuning Achieved:** Strong classification performance (86.4% test accuracy) was achieved by fine-tuning a significantly reduced number of parameters (985k, 0.78% of the base model), demonstrating the effectiveness of the Low-Rank Adaptation (LoRA) technique for resource-constrained scenarios.
2. **Configuration Interplay:** Optimal LoRA performance depends heavily on the interplay between rank, target modules (`target_modules`), and training hyperparameters like learning rate (`learning_rate`). Unconventional configurations (e.g., very low rank with specific targets) can be surprisingly effective, as seen with our final $r = 3$ model.
3. **PEFT Library Effectiveness:** The 'peft' library provides a robust interface for implementing and experimenting with LoRA.
4. **Budget-Performance Trade-off:** Excellent results are achievable within strict parameter budgets, requiring careful exploration and balancing of LoRA parameters to maximize performance near the constraint boundary, as evidenced by our tested configurations utilizing 0.98M parameters.

Results and Discussion

The final optimized model, `RoBERTa-base` fine-tuned on AGNEWS using LoRA ($r = 3$, $\alpha = 6$, targeting `["query", "dense"]`) with only **0.98M trainable parameters**, achieved a **final submission test accuracy of 86.4%**.

The peak accuracy observed on the test set during training reached **95.2%.** The training process converged effectively using AdamW (LR=3×10^{-4}), a cosine scheduler with warmup, label smoothing, and an effective batch size of 32. The best model checkpoint based on validation accuracy was loaded for generating the final test result.

The 86.4% test accuracy is a strong result given the extremely low number of trainable parameters ($< 1\%$ of `RoBERTa-base`), demonstrating LoRA's efficiency. The higher learning rate paired with the very low rank proved crucial. The performance gap between validation (95.2%) and test (86.4%) suggests that while the model learned validation patterns well, generalization to unseen test data was more challenging.

Table 1 summarizes key experimental results. Future work could explore different PEFT methods, advanced regularization, or data-centric approaches.

Table 1: Comparison of Different LoRA Configurations on AGNEWS

Model Config	Trainable Params	epochs	Test Acc (%)
RoBERTa + LoRA (r=11, q,v)	0.99 M	3	94.5
RoBERTa + LoRA (r=7, q,v,k)	0.98 M	3	94.6
RoBERTa + LoRA (r=22, q)	0.99 M	3	93.8
RoBERTa + LoRA (r=3, q,dense)	0.98 M	3	94.7
RoBERTa + LoRA (r=3, q,dense)	**0.98 M**	10	**95.2**

References

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations (ICLR)*.

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*.

Loshchilov, I.; and Hutter, F. 2017. Decoupled Weight Decay Regularization. *arXiv preprint arXiv:1711.05101*.

PEFT contributors. 2022. PEFT: Parameter-Efficient Fine-Tuning of Foundation Models. `https://github.com/huggingface/peft`.

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; Davison, J.; Shleifer, S.; von Platen, P.; Ma, C.; Jernite, Y.; Plu, J.; Xu, C.; Le Scao, T.; Gugger, S.; Drame, M.; Lhoest, Q.; and Rush, A. M. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 38–45. Online: Association for Computational Linguistics.