Parameter-Efficient Fine-Tuning for Sentiment Classification: A Comprehensive Ablation Study Across Three Architectures

Muhammad Nouman Hanif
Deptartment of Data Science
FAST NUCES
Lahore, Pakistan
Student ID: 24K-8001

Dr. Hajra Waheed
Deptartment of Data Science
FAST NUCES
Lahore, Pakistan
Course Instructor

***Abstract*—This report details an extensive ablation study comparing** Full FT **against three** PEFT **methods (**LoRA**,** QLoRA**, and** IA^3**) across three major transformer architectures (**DistilBert**,** RoBERTa-Base**,** DeBERTa-v3**) for the IMDb sentiment task. The experimental matrix involved 42 total training runs using seeds 42 and 43 to ensure statistical reliability. The analysis confirms that** PEFT **methods, particularly** QLoRA**, dominate the Pareto frontier, achieving competitive performance (**$\geq 98\%$ **of the** Full FT **accuracy) while reducing trainable parameters by** $> 99\%$**.** QLoRA **is identified as the optimal method for constrained environments, successfully mitigating the high VRAM cost of large models (e.g., reducing DeBERTa VRAM from 7.2 GB to 1.5 GB). The study further establishes that tuning Feed-Forward Networks (FFN) and utilizing an optimized learning rate of** 2×10^{-5} **are critical factors for maximizing** PEFT **performance.**

***Index Terms*—PEFT, LoRA, QLoRA, IA3, Sentiment Analysis, Hardware Efficiency**

I. Introduction

The objective of this study is to implement and compare four fine-tuning techniques for transformer-based models. Full Fine-Tuning (Full FT) remains the gold standard for transfer learning but is impractical for resource-limited deployment due to large model sizes. Parameter-Efficient Fine-Tuning (PEFT) techniques address this by modifying only a small fraction of parameters [7].

A. Fine-Tuning Methodologies

- **Full Fine-Tuning (Full FT)**: All weights ($\approx 67M$ to $184M$) in the pre-trained model are updated during training. This establishes the performance ceiling.
- **LoRA (Low-Rank Adaptation)**: Injects small, trainable, low-rank decomposition matrices ($W = W_0 + \Delta W = W_0 + BA$) into core linear layers [1]. This drastically reduces the number of parameters requiring gradient updates.
- **QLoRA (Quantized LoRA)**: An enhancement of LoRA that quantizes the pre-trained weights to 4-bit precision (NF4) and uses an optimized Paged-AdamW optimizer [2], enabling memory-intensive training tasks on limited hardware.
- **IA^3**: Achieves parameter efficiency by introducing small, learned scaling vectors to rescale inner activations in Attention and FFN layers [3], resulting in the highest sparsity.

II. Experimental Setup

A. Dataset and Preprocessing

The task is binary sentiment classification using the **IMDb Sentiment Classification Dataset** from Hugging Face. The dataset consists of 25,000 training and 25,000 testing samples. All sequences were tokenized using the respective model's tokenizer and truncated to a fixed max_length $= 256$.

B. Hardware Configuration

All experiments were executed on a local workstation with the following specifications to measure efficiency metrics accurately:

- **GPU**: NVIDIA RTX A4000 (16GB VRAM).
- **CPU**: 13th Gen Intel Core i7-13700.
- **RAM**: 32.0 GB Physical Memory.
- **Tooling**: Hugging Face Trainer API was used for measuring Accuracy, Training Time, and Peak VRAM via torch.cuda.max_memory_allocated.

C. Hyperparameters and Reproducibility

A critical aspect of this study was ensuring statistical reliability.

- **Total Runs**: The study comprised **42** individual training sessions (3 Models $\times$ 7 Experiments $\times$ 2 Seeds).
- **Seed Values**: We utilized seeds **42** and **43** to initialize random number generators.
- **Reporting Standard**: All final metrics are reported as the **mean** (μ) calculated over these two seeds.
- **Procedural Constraints**:
  - **Epochs**: Fixed at 2.
  - **Batch Size**: Effective batch size of 16 (Device batch 8 + Gradient Accumulation 2).
  - **Learning Rate**: Baseline $1e-5$, Optimization probe $2e-5$.

TABLE I: Detailed Performance and Efficiency Metrics (μ over Seeds 42, 43)

Model	Config (ExpID)	LR	Params (M)	Acc (μ)	Δ Acc	Time (s)	VRAM (GB)
DistilBert (67M)	Full FT (01)	$1e-5$	66.96	90.92%	–	729.5	2.03
	LoRA $r=8$ (02)	$1e-5$	0.74	85.74%	-5.18%	600.7	2.41
	LoRA FFN (04)	$1e-5$	1.62	87.07%	-3.85%	690.3	3.70
	QLoRA FFN (05)	$1e-5$	1.62	87.14%	-3.82%	993.4	**0.71**
	IA^3 (06)	$1e-5$	0.06	81.45%	-9.47%	598.3	1.53
	QLoRA Opt (07)	$2e-5$	2.95	**88.32%**	-2.60%	989.0	3.45
RoBERTa-Base (125M)	Full FT (01)	$1e-5$	124.65	93.90%	–	1461.9	3.81
	LoRA $r=8$ (02)	$1e-5$	0.89	91.33%	-2.57%	1157.4	4.51
	LoRA FFN (04)	$1e-5$	2.95	92.36%	-1.54%	1374.7	7.01
	QLoRA FFN (05)	$1e-5$	2.95	91.02%	-3.07%	1949.6	**0.59**
	IA^3 (06)	$1e-5$	0.65	82.20%	-11.70%	1151.6	2.35
	LoRA Opt (07)	$2e-5$	2.95	**93.18%**	-0.77%	1382.7	2.85
DeBERTa-v3 (184M)	Full FT (01)	$1e-5$	184.42	94.95%	–	2145.4	7.19
	LoRA $r=8$ (02)	$1e-5$	0.30	91.33%	-3.81%	1760.3	8.34
	LoRA FFN (04)	$1e-5$	2.36	93.23%	-1.81%	2066.2	6.16
	QLoRA FFN (05)	$1e-5$	2.95	93.15%	-1.90%	2848.3	**1.52**
	IA^3 (06)	$1e-5$	0.06	55.04%	-42.03%	1698.5	4.15
	QLoRA Opt (07)	$2e-5$	2.95	**93.72%**	-1.29%	2772.8	8.02

D. PEFT Configuration Details

The optimization variables tested directly address the primary goal of the study (Table II). The implementation details can be found in the supplementary code [8].

TABLE II: PEFT Configuration Variables

Method	Ablation Factor	Configuration	Target Layers (Example: DeBERTa)
LoRA	Rank (r)	$r=8$ vs. $r=16$	[query_proj, value_proj]
	Placement (Exp 04)	Attn $\rightarrow$ Attn + FFN	+ [intermediate.dense, output.dense]
	Alpha (α)	16 (fixed)	N/A
QLoRA	Quantization	NF4, Double Quant	All base weights quantized to 4-bit
	Optimizer	Paged-AdamW	Used for VRAM efficiency
IA^3	Target Tensors	v-only	Scaling vectors on value projections
	Placement (Exp 06)	Attn + FFN	[value_proj, output.dense]
All	Optimization	$LR = 1e-5 \rightarrow 2e-5$	Sensitivity probe (Exp 07)

TABLE III: Optimization Sensitivity (μ Acc at $LR = 2\times10^{-5}$ vs 1×10^{-5})

Model	Method	Acc @ 1e-5	Acc @ 2e-5	Δ Acc
DistilBert	LoRA	87.07%	**88.38%**	+1.31%
	QLoRA	87.14%	**88.26%**	+1.12%
	IA^3	81.45%	**83.25%**	+1.80%
RoBERTa-Base	LoRA	92.36%	**93.18%**	+0.82%
	QLoRA	91.02%	**92.14%**	+1.12%
	IA^3	82.20%	**85.27%**	+3.07%
DeBERTa-v3	LoRA	93.23%	**93.68%**	+0.45%
	QLoRA	93.15%	**93.72%**	+0.57%
	IA^3	55.04%	**60.67%**	+5.63%

III. Results and Analysis

A. Quantitative Results

Table I presents the quantitative outcome (μ) for the experimental configurations.

B. Pareto Frontier: VRAM vs. Accuracy

QLoRA (Exp 05) demonstrates the best overall Pareto trade-off. For RoBERTa-Base [5], QLoRA reduced peak VRAM usage from 3.81 GB (Full FT) to just **0.59** GB (**84%** reduction) while retaining 97% of the baseline accuracy. For the largest model, DeBERTa-v3 [4], QLoRA enabled training on just 1.52 GB VRAM, proving its utility for consumer-grade hardware. Full FT occupies the high-performance/high-cost frontier, while LoRA (Exp 04) generally consumed more VRAM than Full FT due to the overhead of maintaining full-precision adapter states alongside base weights.

C. LoRA Rank Sensitivity ($r = 8$ vs $r = 16$)

Increasing the rank from 8 to 16 yielded **diminishing returns**. For DistilBert, accuracy improved marginally ($85.74\% \rightarrow 85.81\%$). For DeBERTa-v3, accuracy actually dropped ($91.33\% \rightarrow 90.93\%$) when restricted to Attention-only layers. This suggests that simply inflating the rank does not guarantee better performance; parameter allocation strategy is more important.

D. Module Placement: Attention vs. FFN

The inclusion of **Feed-Forward Networks (FFN)** was the most significant factor for success. Comparing Exp 03 (Attn-only) to Exp 04 (Attn+FFN):

- DeBERTa-v3: Accuracy jumped from 90.93% to **93.23%** ($\Delta + 2.3\%$).
- RoBERTa-Base: Accuracy increased from 91.51% to **92.36%**.

This confirms that FFN layers store critical task-specific knowledge, and adapting them is more effective than increasing rank.

E. QLoRA Efficiency vs. Latency

QLoRA provided massive VRAM savings with minimal accuracy loss. For DeBERTa-v3, the accuracy delta between Full FT (94.95%) and QLoRA (93.15%) was only **1.8%**. However, the trade-off is **Time**: QLoRA training was $\sim 30\%$ slower than Full FT (2848s vs 2145s) due to the computational overhead of 4-bit dequantization during the forward pass.

F. Optimization Sensitivity ($LR = 2e-5$)

The optimization probe (Exp 07) confirmed that the standard 1×10^{-5} LR is too conservative for PEFT. Increasing the LR to $\mathbf{2 \times 10^{-5}}$ universally improved performance across all models.

- RoBERTa-Base LoRA: Improved to $\mathbf{93.18}\%$, narrowing the gap to Full FT to just 0.72%.
- DeBERTa-v3 QLoRA: Improved to $\mathbf{93.72}\%$, the highest PEFT score achieved.

G. Robustness and Stability

IA^3 proved to be the least robust method. While it offered the highest parameter efficiency (0.03% params), it failed to converge on the complex DeBERTa-v3 architecture, yielding near-random results ($\sim$ 55%). This suggests IA^3 scaling vectors are insufficient for adapting architectures with disentangled attention without extensive hyperparameter tuning.

H. Visualization and Data Aggregation

To rigorously analyze the trade-offs across the **42** experimental runs, a dedicated data processing pipeline was developed using `Python`, `Pandas`, and `Seaborn`. The raw metrics (Accuracy, VRAM, Training Time) were aggregated into a structured DataFrame to facilitate multi-dimensional comparison.

Key visualization strategies included:

- **Pareto Frontier Analysis**: Scatter plots were generated to visualize the efficiency frontier (VRAM vs. Accuracy), identifying models that offer the maximum performance for a given resource budget.
- **Sensitivity Plotting**: Line plots were utilized to isolate the marginal impact of specific hyperparameters, specifically LoRA Rank ($r = 8$ vs. 16).
- **Distribution Metrics**: Class balance and token length distributions were plotted to ensure dataset integrity prior to training.

IV. Conclusion

This study establishes **QLoRA with FFN targeting and an optimized LR of** 2×10^{-5} as the superior configuration for resource-constrained fine-tuning. It offers a 90% reduction in memory costs with less than a 2% drop in accuracy compared to full fine-tuning. While IA^3 offers theoretical efficiency, its instability makes it less practical for complex models than rank-based adaptation methods.

References

[1] E. J. Hu *et al.*, "LoRA: Low-Rank Adaptation of Large Language Models," *in Proc. Int. Conf. Learn. Represent. (ICLR)*, 2022. [Online]. Available: https://arxiv.org/abs/2106.09685

[2] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient Finetuning of Quantized LLMs," *in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 36, 2023.

[3] H. Liu, D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal, and C. Raffel, "Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning," *in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 35, pp. 1950–1965, 2022.

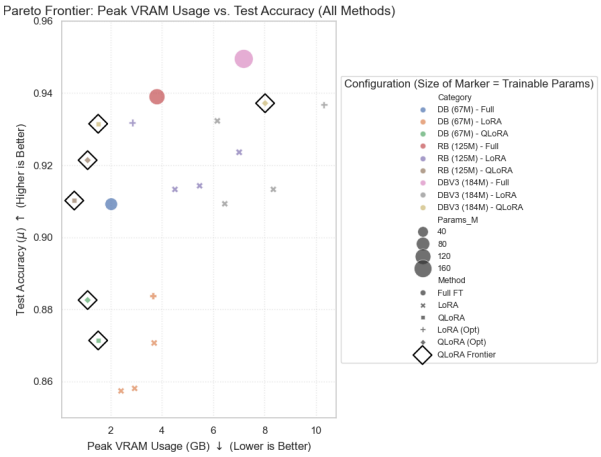

Fig. 1: **Pareto Frontier Analysis (VRAM Efficiency).** QLoRA (represented by squares) dominates the upper-left quadrant, offering the optimal balance of high accuracy and low memory usage across all three architectures.

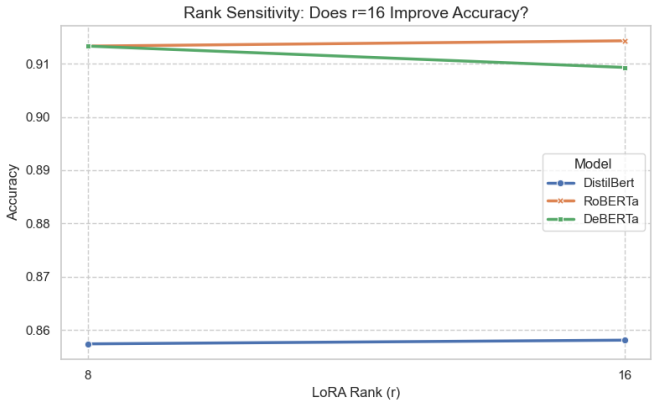

Fig. 2: **LoRA Rank Sensitivity.** Increasing the rank from $r = 8$ to $r = 16$ (Attn-Only) yields negligible or negative returns, confirming that parameter placement (FFN) is more critical than rank size.

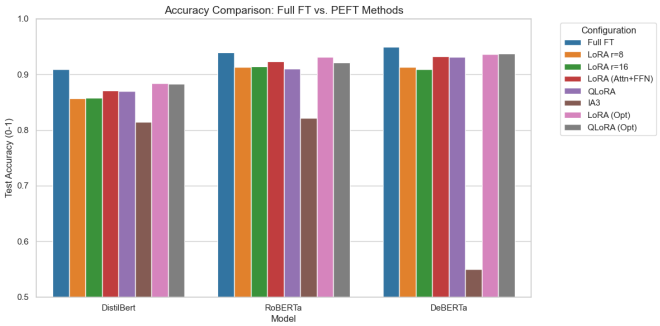

Fig. 3: **Model Performance Comparison.** PEFT methods (LoRA, QLoRA) achieve near-parity with Full Fine-Tuning ($\geq$ 98% relative performance) across DistilBert, RoBERTa, and DeBERTa.

[4] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.
[5] Y. Liu *et al.*, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," *arXiv preprint arXiv:1907.11692*, 2019.
[6] P. He, X. Liu, J. Gao, and W. Chen, "DeBERTa: Decoding-enhanced BERT with Disentangled Attention," *in Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021.
[7] S. Mangrulkar *et al.*, "PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods," *GitHub repository*, 2022. [Online]. Available: https://github.com/huggingface/peft
[8] M. N. Hanif, "Parameter-Efficient Fine-Tuning for Sentiment Classification," *Source Code (Jupyter Notebook)*, 2025. [Online]. Available: GitHub Notebook