

The Efficiency Threshold: Few-Shot Prompting vs. LoRA

Stanford CS224N Custom Project

Shreyas C S

Department of Computer Science
Stanford University
shreyas9@stanford.edu

Daniel Grossman

Department of Computer Science
Stanford University
daniel26@stanford.edu

Abi Lopez

Department of Computer Science
Stanford University
abilopez@stanford.edu

Abstract

LLM adaptation is typically addressed via in-context learning (ICL) or parameter-efficient fine-tuning (PEFT). The "Context-Weight Tradeoff" is investigated to determine the precise data threshold where PEFT outperforms ICL under real-world latency constraints. Results indicate that reasoning tasks benefit from ICL's prompt-based demonstrations, whereas structured classification tasks benefit from LoRA's parameter adaptation. Furthermore, ICL is limited by context windows, whereas LoRA shifts computational costs to training, maintaining stable inference efficiency. The choice between methods is dictated by task characteristics and deployment constraints.

1 Key Information to include

- Mentor: Advit Deepak
- External Collaborators/Sharing project: No

2 Introduction

Large language models (LLMs) are adapted using prompt-based methods such as zero-shot prompting and in-context learning (ICL), as well as parameter-efficient fine-tuning (PEFT) methods like LoRA and ReFT. ICL utilizes task demonstrations in prompts, increasing inference-time computation without training. Conversely, LoRA updates a small parameter set during training, allowing for shorter prompts and lower inference costs. These tradeoffs are critical for real-world deployments constrained by latency and memory. The performance and efficiency tradeoffs between zero-shot prompting, ICL, PEFT, and dynamic prompt optimization via GEPA are evaluated across five datasets. Task structure dictates the optimal method; reasoning tasks favor ICL, while classification favors LoRA. Additionally, context window limits constrain ICL, while LoRA shifts costs to training.

3 Related Work

Model adaptation is achieved through prompting or parameter updates. ICL [1] allows few-shot adaptation without task-specific training. Alternatively, PEFT methods like LoRA [2] freeze pre-trained weights and introduce trainable low-rank updates, minimizing training parameters. Previous comparisons indicate PEFT can outperform ICL in accuracy and efficiency. This research extends

prior work [3] by empirically evaluating the context-weight tradeoff between ICL and LoRA as available examples increase under matched experimental conditions.

4 Approach

Zero-Shot analysis was executed using vLLM for batched generation, with greedy (temperature=0, top_p=1.0) decoding applied to ensure deterministic outputs. ICL and LoRA are evaluated across labeled data budgets $N \in \{16, 32, 64, 128, 256\}$. For ICL, predictions are generated by conditioning the pretrained model on N labeled examples provided directly in the prompt. For LoRA, trainable low-rank matrices are added to transformer attention and projection layers, utilizing the N examples for training rather than inference. Additionally, a GEPA pipeline dynamically refines prompt instructions based on error reflections. PeFT and ReFT architectures are also fine-tuned, with ReFT interventions constrained to a low-rank dimension of $r = 4$ at the fourth-to-last hidden layer.

4.1 Zero-Shot Baseline

As a baseline, we evaluate each pretrained model in a zero-shot setting, where predictions are generated directly from task instructions without any labeled demonstrations or fine-tuning. This baseline establishes the $N = 0$ point of the context-weight tradeoff curve and reflects the capability of the pretrained model prior to task-specific adaptation.

4.2 In-Context Learning

In-context learning performs a task by conditioning the model on labeled demonstrations included directly in the prompt. For each evaluation input, the prompt contains N stratified labeled examples followed by the test input. For the four classification tasks, prompts instruct the model to output exactly one label from a predefined label set. For GSM8K, prompts use chain-of-thought (CoT) formatting and require the final answer to appear after the delimiter #####. Thus, ICL uses the labeled examples only at inference time through prompt construction.

4.3 LoRA Fine-Tuning

Low-Rank Adaptation (LoRA) [2] is implemented using the Hugging Face PEFT library on causal language models. LoRA freezes the pretrained model weights and learns low-rank updates in transformer attention and feed-forward layers. In our implementation, adapters use rank $r = 8$, $\alpha = 16$, and dropout 0.05, with the same labeled data budgets as the corresponding ICL conditions.

For a pretrained weight matrix W_0 , LoRA parameterizes the update as

$$\Delta W = \frac{\alpha}{r} BA,$$

where $B \in \mathbb{R}^{d \times r}$ and $A \in \mathbb{R}^{r \times k}$. The modified forward pass is

$$h = W_0 x + \Delta W x.$$

4.4 GEPA

A GEPA optimization pipeline is implemented to dynamically refine prompt instructions. An initial baseline instruction is provided to the model. Predictions are then generated for a subset of 16 training examples. A reflection prompt is triggered by any incorrect predictions, wherein the generated error and expected output are supplied back to the model. A new, optimized instruction is generated from this reflection and utilized for the final evaluation set.

4.5 PeFT Variation

We additionally evaluate a LoRA-based variation for sequence classification using `PeftModelForSequenceClassification` and `LlamaForSequenceClassification` architectures. Stratified sampling is applied to group the datasets by label and sample a fixed number of examples per class to simulate a few-shot learning environment. The models are subsequently fine-tuned for 10 epochs using a standard Trainer API.

4.6 ReFT

Representation Fine-Tuning (ReFT) is implemented utilizing the `pyrefit` library [4]. Rather than updating model weights, interventions are specifically targeted at the output of the fourth-to-last hidden layer of the base models. A `LoreftIntervention` is applied with a strictly constrained low-rank dimension of $r = 4$, establishing a highly parameter-efficient adaptation bottleneck.

5 Experiments

5.1 Data

Zero-Shot analysis occurred across all 3 models with various splits and sizes dependent on benchmark summarized by Table 1 and evaluated on the entire set.

For ICL, LoRA, and GEPA, we evaluate on five datasets spanning classification and reasoning: SuperGLUE RTE [5] (textual entailment), BoolQ [6] (yes/no question answering), Financial PhraseBank [7] (financial sentiment classification), the RAFT ADE corpus [8] (biomedical sentence classification), and GSM8K [9] (grade-school math reasoning). For ReFT, evaluation additionally includes Dolly15k alongside GSM8K, RTE, Financial PhraseBank, and RAFT.

For ICL, LoRA, GEPA, PeFT, ReFT, training is conducted across dynamically scaled and stratified sample sizes of $N \in \{16, 32, 64, 128, 256, 512\}$. Model evaluation is standardized on subsets ranging from 200 to 250 samples per dataset to ensure consistent throughput measurements.

5.2 Evaluation method

For datasets with explicit answer labels, accuracy is computed via exact match after strict string normalization, which includes lowercasing and punctuation removal. For classification tasks, generated outputs are mapped to valid label sets; unmapped outputs are treated as incorrect. For the GSM8K dataset, numeric answers are isolated using the “####” delimiter or regular expressions, with formatting characters removed. IFBench evaluation additionally incorporates strict and loose scoring at both prompt and instruction levels. For Zero-Shot, statistical stability is validated by subsampling evaluation sets at $N \in \{16, 32, 64, 128, 256\}$ using 200 bootstrap draws (Figure 8).

Computational efficiency is quantified across multiple dimensions for every model-dataset- N configuration. Latency is measured as the average wall-clock time per evaluation example: $\text{Latency} = \frac{\text{total generation time}}{\# \text{ evaluation examples}}$. Throughput is calculated as tokens processed per second for both total and generated tokens. Time per output token (TPOT) is defined as $\text{TPOT} = \frac{\text{total generation time}}{\# \text{ generated tokens}}$. Additionally, peak GPU memory utilization (VRAM) is tracked during training and inference utilizing native profiling capabilities, and prompt token counts are monitored.

5.3 Experimental details

Zero-Shot and Few-Shot In-Context Learning (ICL) evaluations were executed in a serverless Modal environment utilizing NVIDIA H100 GPUs, bfloat16 precision, and TensorFloat-32 (TF32). Dynamic batch sizing was employed to optimize hardware utilization and prevent memory errors, progressively scaling down as the number of demonstrations (N) increased.

In a separate set of ICL and LoRA experiments, models were evaluated on NVIDIA B200 GPUs using 4-bit NF4 quantization. We evaluate five pretrained causal language models: Llama-3-8B, Llama-3-8B-Instruct, Qwen3-4B, Qwen3-4B-Instruct-2507, and Qwen3-8B. Data sampling used a fixed seed of 42, and inputs were truncated to 2048 tokens. ICL used a batch size of 16 with deterministic decoding and task-specific generation limits (8 tokens for classification tasks and 256 for GSM8K). LoRA fine-tuning ran for 100 steps via the HuggingFace Trainer with adapters of rank $r = 8$ and scaling factor $\alpha = 16$.

GEPA computations were executed using eager attention and bfloat16 precision, restricting text generation to a maximum of 40 new tokens per instance. PeFT and ReFT methodologies were similarly evaluated across varied sample sizes; PeFT training spanned 10 epochs, while ReFT configurations were trained for 25 epochs leveraging TF32 optimizations. Strict token generation boundaries were consistently enforced during inference across all evaluated configurations.

5.4 Results

The complete 3×6 zero-shot accuracy grid is presented in Table 2 and Figure 9. The highest mean accuracy is achieved by Qwen3-8B, while Qwen3-4B follows closely despite possessing half the parameters. Results are categorized into three tiers. Tier 1 consists of tasks largely solved at $N = 0$. Tier 2 is characterized by extreme subtask variance, and Tier 3 represents tasks where most constraints fail to be followed. Throughput (Figure 10) is observed to vary by over two orders of magnitude. The highest throughput is yielded by GSM8K, while the slowest is recorded for Financial PhraseBank. The fastest model across benchmarks is consistently shown to be Qwen3-4B.

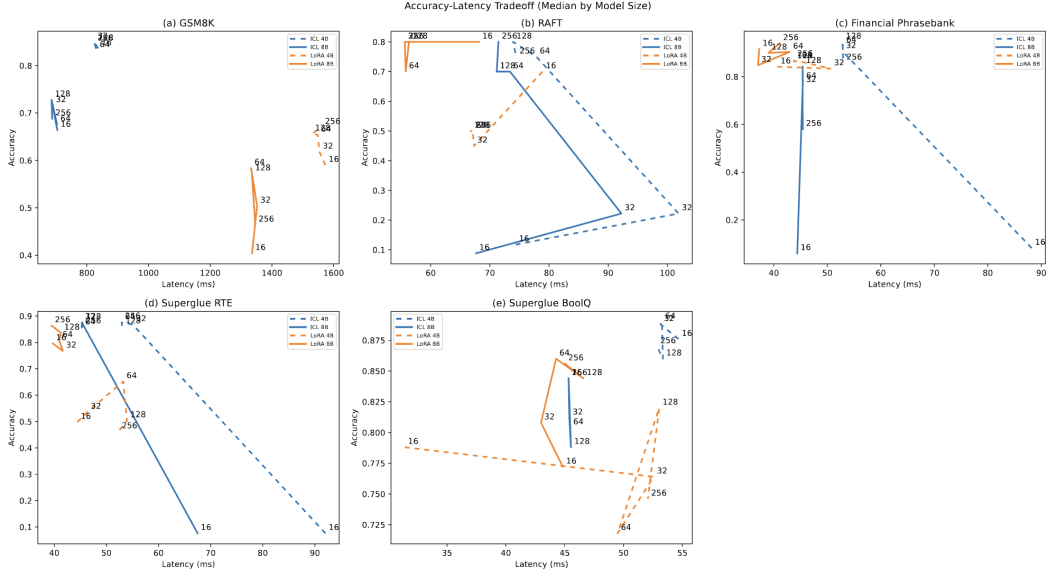


Figure 1: Accuracy–latency tradeoffs for ICL and LoRA across datasets and training example counts N , averaged across six models. Points closer to the upper-left indicate more efficient configurations.

In the ICL vs LoRA evaluations, accuracy generally improves with increasing N for both methods, though trends vary by dataset and model size. For GSM8K, ICL consistently outperforms LoRA for both 4B and 8B models, with relatively stable performance as N increases. In contrast, LoRA often matches or exceeds ICL on classification tasks such as RAFT and Financial PhraseBank, particularly for larger models. For SuperGLUE RTE and BoolQ, both methods converge to similar accuracies once $N \geq 32$. Overall, ICL shows stronger performance on the reasoning task (GSM8K), while LoRA is more competitive on classification datasets.(Figure 11)

Latency remains relatively stable as N increases for both methods across datasets. For GSM8K, ICL exhibits substantially lower latency than LoRA, while for classification tasks such as RAFT, Financial PhraseBank, and SuperGLUE RTE, LoRA generally achieves lower latency. BoolQ shows minimal differences between methods, with both maintaining similar latency across model sizes. Overall, LoRA tends to provide lower latency on classification datasets, whereas ICL is more efficient on the reasoning task. (Figure 12)

The accuracy–latency tradeoff varies by task and model size. For GSM8K, ICL consistently occupies a more favorable region with both higher accuracy and lower latency than LoRA. In contrast, for RAFT and Financial PhraseBank, LoRA achieves comparable or higher accuracy while maintaining lower latency, particularly for larger models. For SuperGLUE RTE and BoolQ, both methods achieve similar accuracy levels, with differences primarily reflected in latency rather than predictive performance. Overall, ICL provides the strongest tradeoff on the reasoning task, while LoRA tends to offer a more efficient accuracy–latency balance on classification datasets.(Figure 1)

Across all datasets and model sizes, ICL consistently achieves higher throughput and lower time per output token than LoRA. For 4B models, ICL reaches roughly 150–338 tok/s compared to about 106–156 tok/s for LoRA, with TPOT values between 3.24–6.68 ms versus 6.54–10.42 ms. A similar

pattern holds for 8B models, where ICL attains 168–406 tok/s while LoRA remains between 115–165 tok/s. In contrast, LoRA consistently reduces evaluation-time memory usage. Peak VRAM decreases from roughly 5.5–5.8% to 2.5–4.5% for 4B models and from about 7.5–8.0% to 5.1–7.2% for 8B models. Overall, ICL provides faster inference, while LoRA reduces GPU memory requirements. (Table 3)

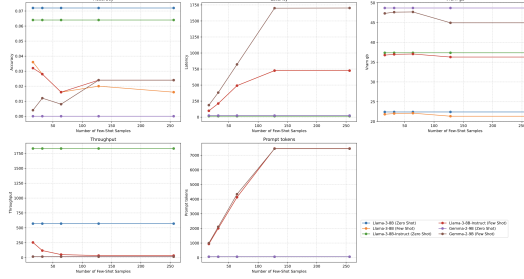


Figure 2: Context weight tradeoff between Zero-Shot and Few-Shots on GSM8K dataset

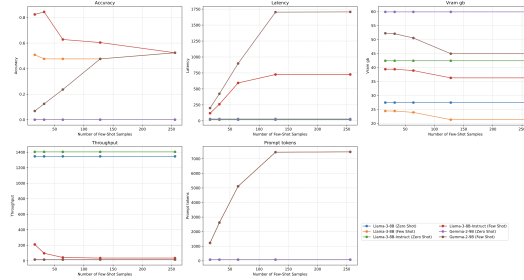


Figure 3: Context weight tradeoff between Zero-Shot and Few-Shots on SuperGLUE dataset

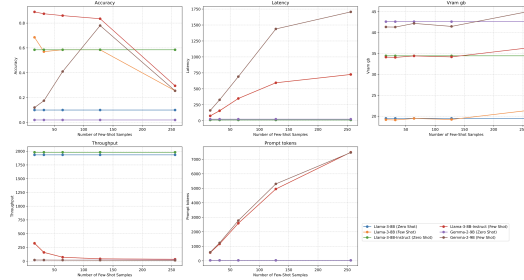


Figure 4: Context weight tradeoff between Zero-Shot and Few-Shots on FINBank dataset

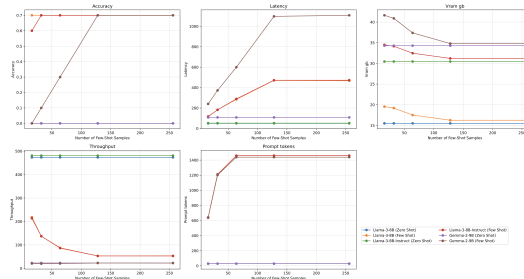


Figure 5: Context weight tradeoff between Zero-Shot and Few-Shots on RAFT dataset

Following findings are based on the additional Zero-Shot and Few-Shot ICL evaluations (Figures - 2, 3, 4 & 5): Highly variable predictive performance is observed as N is scaled. Significant improvements are noted for Gemma-2-9B on specific tasks (Figures - 3 & 4), while severe degradation is experienced by Llama-3-8B-Instruct (Figure - 5). For complex reasoning, accuracy is seen to remain uniformly negligible (Figure - 2). A fundamental tradeoff between context size and inference speed is demonstrated. A strict linear increase in latency is observed as examples are added, while zero-shot baselines are maintained at low latency. Linear growth in prompt tokens is tracked until maximum context lengths are saturated (Figures - 3 & 5). Furthermore, GPU memory allocation is observed to remain stable regardless of N .

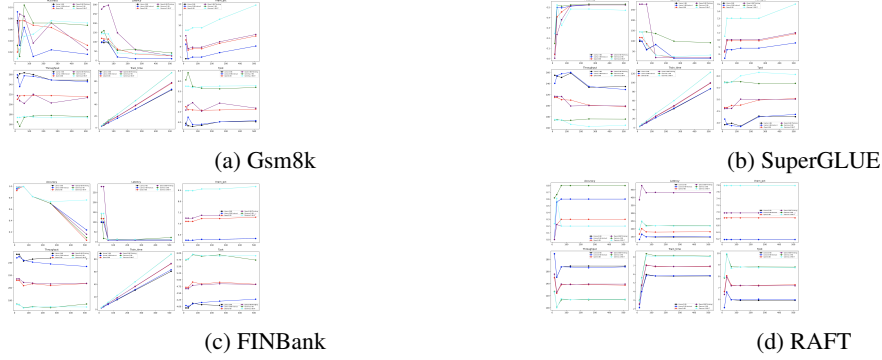


Figure 6: ReFT - Evaluation Metrics

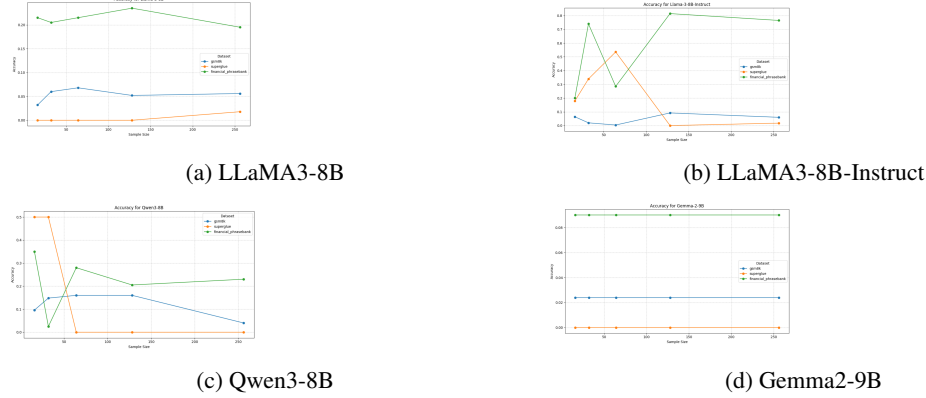


Figure 7: GEPA - Accuracy across N samples

For LoRA-adapted models, increased accuracy from 0.4200 at $N = 16$ to 0.6150 at $N = 256$ is observed alongside strictly constant inference latency, for Gemma-2-9B (Figure - 16d). Memory constraints are consistently dominated by the base model size.

High volatility across different sample sizes is indicated by the GEPA optimization results. Uniformly low accuracy (peak of 16.0%) is observed on GSM8K (Figures - 7c & 7b), while completely static accuracy is exhibited by Gemma-2-9B (Figure - 7d).

Significant performance variance is recorded under ReFT adaptation, with peak accuracy levels between 0.6 and 0.8 (Figure - 6d). Optimal accuracy is often recorded at lower sample sizes (Figure - 6c), and training time is observed to scale strictly linearly.

For PeFT, accuracy scaling is achieved by both Meta-Llama-3-8B and Qwen3-8B as N increases across multiple evaluated datasets (Figure - 15a).

6 Analysis

The observed tradeoff suggests that adaptation strategy should be chosen based on task structure rather than accuracy alone. ICL appears especially well suited to reasoning problems such as GSM8K, where explicit demonstrations can shape multi-step generation at inference time. In contrast, classification tasks benefit less from carrying examples in the prompt, since the required input-output mapping is comparatively compact and can often be captured effectively through parameter updates. This helps explain why LoRA remains competitive, and sometimes preferable, on several classification benchmarks despite its lower token-generation efficiency.

The efficiency results also reveal that “better” depends on the deployment constraint being optimized. ICL consistently generates output faster, but this advantage comes with higher evaluation-time memory usage because demonstrations must remain in the prompt. LoRA shifts this cost into training, yielding slower generation but lower inference-time VRAM. Taken together, the results indicate that the context–weight tradeoff is fundamentally a systems tradeoff: ICL is attractive when reasoning quality and fast generation matter most, whereas LoRA is more favorable when memory efficiency and reusable task adaptation are the primary concerns.

ICL scaling is also constrained by the context window. As the number of demonstrations N increases, prompts grow until the maximum sequence length is reached, after which earlier examples are truncated. This truncation occurred frequently in large- N runs, meaning additional demonstrations were sometimes partially excluded from the model input. LoRA avoids this limitation during inference because training examples are incorporated into adapter parameters rather than the prompt. Consequently, a key limitation of ICL is its reliance on finite context windows, which restricts how effectively the method can scale with increasing numbers of demonstrations.

A severe bottleneck is created by GEPA’s reliance on inherent instruction-following capabilities, leading to erratic performance in base models due to difficulty in generating corrective instructions. Static accuracy in some models is hypothesized to result from instruction parsing failures, while a higher capacity for error-driven reflections is demonstrated by Instruct-tuned variants.

For PeFT, significant decay in training loss and improved accuracy are observed as N increases. A higher capacity to fit small training sets is shown by Qwen3-8B compared to Meta-Llama-3-8B, though this fitting capability is accompanied by slightly higher memory consumption and inference latency. Extreme memory efficiency is demonstrated by ReFT, though the limited representational capacity of a rank-4 intervention is hypothesized to be insufficient for open-ended tasks. Furthermore, potential overfitting or representational collapse is suggested by accuracy degradation at higher values of N on select classification tasks.

A tradeoff between prompt-based and parameter-based adaptation is highlighted overall. Rapid adaptation without training is enabled by ICL, but it is constrained by context limits; conversely, stable inference costs are maintained by LoRA despite an upfront training requirement. Ultimately, reasoning tasks are better suited for prompt demonstrations, whereas structured tasks benefit from parameter adaptation.

7 Conclusion

This work evaluates multiple adaptation strategies for large language models, including zero-shot prompting, in-context learning (ICL), parameter-efficient fine-tuning with LoRA, dynamic prompt optimization via GEPA, and Representation Fine-Tuning (ReFT). Across multiple reasoning and classification tasks, the results show that performance depends strongly on task structure and computational constraints rather than a single universally superior method. Reasoning tasks such as GSM8K benefit most from ICL’s prompt-based demonstrations, which consistently yield higher accuracy and favorable accuracy–latency tradeoffs. In contrast, classification tasks such as RAFT and Financial PhraseBank exhibit smaller performance differences, where LoRA often achieves comparable accuracy while maintaining lower latency. These findings suggest that explicit demonstrations are particularly effective for reasoning problems, whereas parameter updates can efficiently capture simpler classification mappings.

System-level efficiency measurements further highlight a practical tradeoff between the two primary adaptation approaches. ICL consistently generates tokens faster, achieving higher throughput and

lower time per output token, but this advantage comes with increased evaluation-time memory usage because demonstrations must remain in the prompt. LoRA instead shifts computational cost to training, resulting in slower generation but substantially reduced inference-time VRAM. In practice, this means ICL is well suited to scenarios prioritizing fast generation and reasoning performance, while LoRA offers a more memory-efficient alternative for scalable deployment.

Distinct behaviors are exhibited by other adaptation techniques. While sequence classification is effectively improved by PEFT with limited data ($N \leq 256$), instability was proven by dynamic prompt optimization via the GEPA pipeline. Prompt instructions are dynamically refined by the GEPA pipeline based on error reflections triggered by incorrect predictions, but a struggle to consistently correct logic errors is faced by base models. Furthermore, stronger loss convergence is indicated to be offered by Qwen3-8B, whereas better VRAM and throughput efficiency are provided by Meta-Llama-3-8B. Finally, substantial efficiency gains over traditional methods are provided by Representation Fine-Tuning (ReFT), but high task-specific volatility is demonstrated. These ReFT interventions were strictly targeted at the output of the fourth-to-last hidden layer with a constrained low-rank dimension of $r = 4$.

Overall, these findings highlight the broader context–weight tradeoff in adapting large language models. In practice, the most effective strategy depends not only on predictive performance but also on the computational constraints and task structure of the deployment setting.

Prompt Engineering (ICL): Best suited for complex reasoning tasks requiring multi-step logic and for scenarios prioritizing fast generation (higher throughput and lower inference latency).

Post-Training (LoRA): Best suited for structured classification tasks, memory-constrained deployments (lower evaluation-time VRAM), and environments with limited context windows where demonstration truncation must be avoided.

Future Work: Future work should examine whether the relative strengths of ICL and PEFT depend not only on task category but also on evaluation structure. In particular, the collapse of LoRA performance on IFBench suggests that instruction-following and constraint-based evaluations may require different adaptation strategies, potentially including RL-based optimization. Hybrid strategies combining contextual prompting and parameter-efficient adaptation could also be explored, along with how the context–weight tradeoff evolves as model sizes and context windows continue to expand.

Team contributions

Shreyas: A critical cloud-based pipeline was engineered to evaluate context-weight tradeoffs across advanced LLM paradigms, including LoRA, ReFT via `pyreft`, GEPA, Zero-Shot and Few-Shots. Robust data preprocessing and stratified sampling were implemented for six diverse datasets. Crucial LoRA training for five causal models was orchestrated. Furthermore, an extensive inference loop was constructed to benchmark accuracy, latency, and VRAM metrics, culminating in automated scaling plots and comprehensive documentation and consolidating the source code in the GitHub repository that ensured overall project success.

Abi Lopez: I designed and implemented the experimental infrastructure used to run and evaluate the ICL and LoRA experiments across six models, five datasets, and multiple dataset splits. This included data preprocessing, prompt construction, model evaluation, and instrumentation for measuring latency, throughput, and VRAM used in the reported results. I also contributed substantially to organizing, writing, and editing both the milestone and final project reports.

Daniel Grossman: Evaluated Zero-Shot baselines across all datasets for all models. Ran external additional analysis of IFBench LoRA (distilling Qwen3.5-9B into Llama3-8B after finding that LoRA consistently dropped accuracy to 0 across all models). Created shared google sheets used to track progress of entire project. Contributed to writing and editing all submitted reports. Introduced vLLM to project as well.

References

- [1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 1877–1901, 2020.
- [2] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.
- [3] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.
- [4] Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. ReFT: Representation Finetuning for Large Language Models. *arXiv preprint arXiv:2404.03592*, 2024. Software available at <https://github.com/stanfordnlp/pyreft>.
- [5] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, et al. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- [6] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 2924–2936. Association for Computational Linguistics, 2019.
- [7] Pekka Malo, Ankur Sinha, Pekka Korhonen, Jyrki Wallenius, and Pyry Takala. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 2014.
- [8] Neel Alex, Eli Lifland, Lewis Tunstall, Abhishek Thakur, et al. RAFT: A real-world few-shot text classification benchmark. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2021.
- [9] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- [10] Jeffrey Zhou, Tianyi Lu, Swaroop Mishra, et al. IFBench: Instruction following evaluation for large language models. *arXiv preprint arXiv:2305.14782*, 2023.

A Appendix (optional)

Table 1: Benchmark summary for Zero-Shot evaluation.

Benchmark	Split	Size	Task Type
Financial PhraseBank [7]	test	1,000	3-class sentiment
GSM8K [9]	test	1,319	Math reasoning
SuperGLUE BoolQ [5]	validation	3,270	Yes/No QA
SuperGLUE RTE [5]	validation	277	2-class entailment
RAFT (10 subtasks) [8]	test	150–5,000	Multi-domain classification
IFBench [10]	test	300	Instruction following

Table 2: Zero-Shot accuracy (%) across all model-benchmark configurations.

Benchmark	Llama-3 8B	Qwen3 4B	Qwen3 8B
Financial PhraseBank	90.6	90.0	96.3
GSM8K	77.3	85.3	87.6
SuperGLUE BoolQ	84.6	85.4	86.8
SuperGLUE RTE	78.3	86.6	85.9
RAFT (mean of 10)	36.8	39.6	40.8
IFBench (loose prompt)	28.7	33.3	30.0
Mean	66.0	70.0	71.2

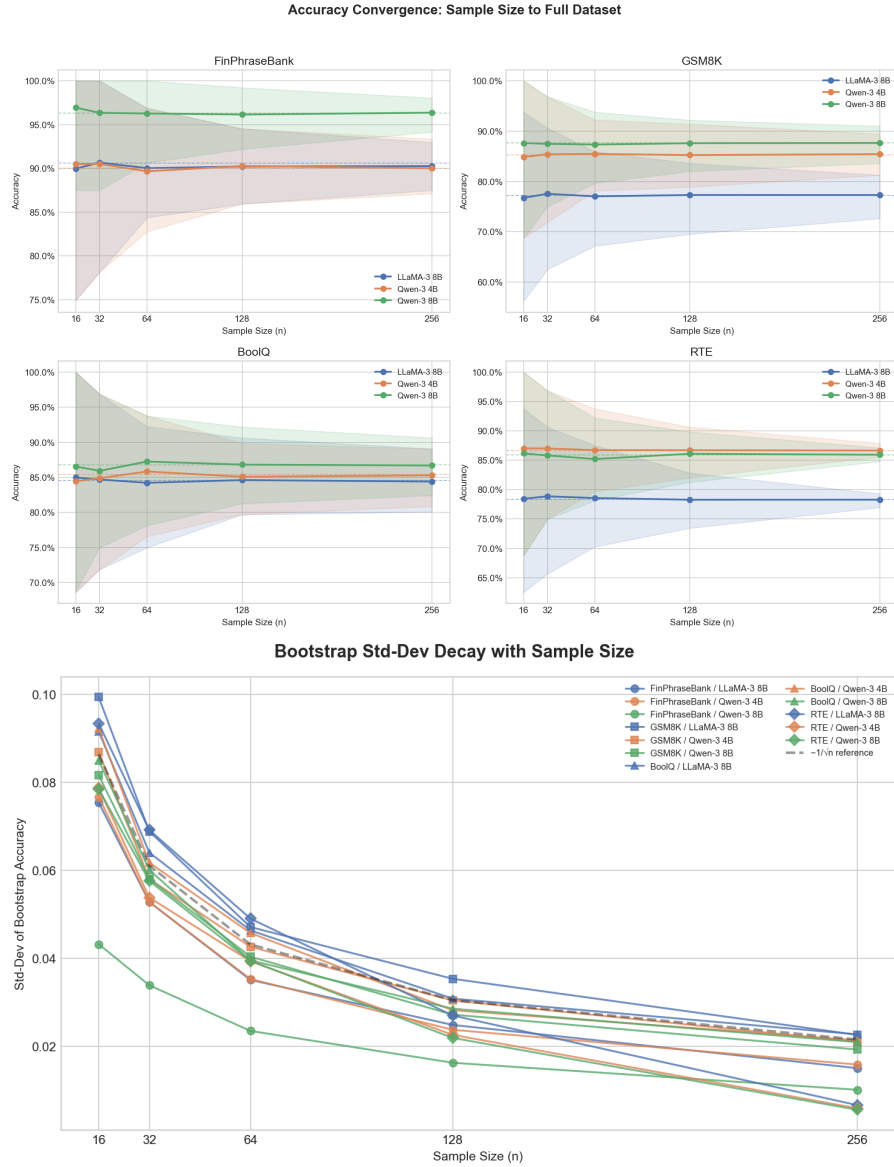


Figure 8: Top: accuracy convergence with sample size. Bottom: bootstrap std-dev decay follows $1/\sqrt{n}$.

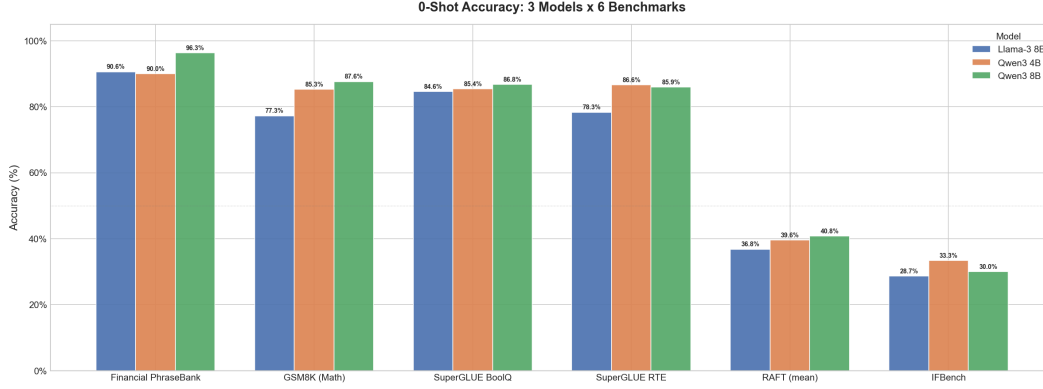


Figure 9: Zero-Shot Accuracy Comparison

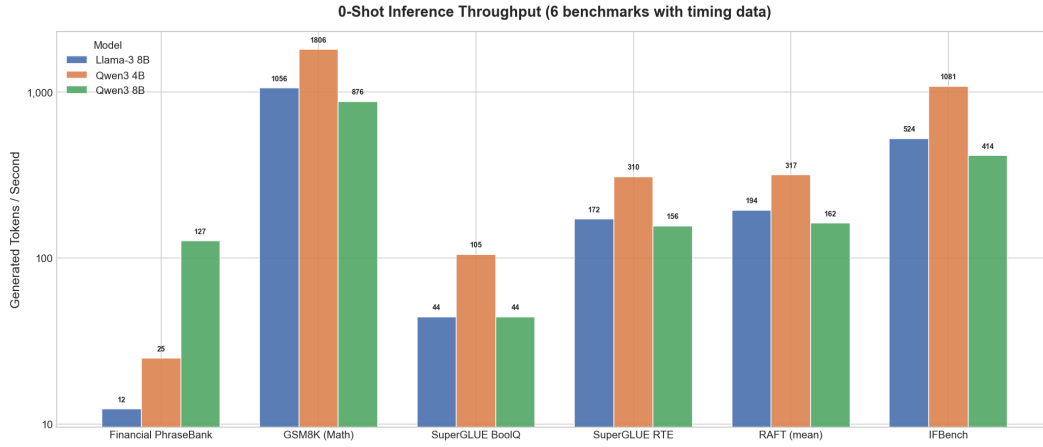


Figure 10: Zero-Shot inference throughput (log scale).

Dataset	Method	Throughput	4B		8B		
			TPOT	VRAM (%)	Throughput	TPOT	VRAM (%)
GSM8K	ICL	308.66	3.24	5.77	336.25	3.05	7.48
GSM8K	LoRA	123.10	8.51	3.41	142.46	7.61	5.93
RAFT	ICL	230.42	6.64	5.79	245.88	6.46	8.04
RAFT	LoRA	110.83	9.14	2.47	115.24	9.44	5.08
Financial PhraseBank	ICL	223.19	5.68	5.49	310.32	5.02	7.64
Financial PhraseBank	LoRA	150.26	7.36	2.66	156.58	7.61	5.23
SuperGLUE RTE	ICL	337.77	5.54	5.73	405.79	4.93	7.93
SuperGLUE RTE	LoRA	155.62	6.54	2.94	165.42	6.20	5.51
SuperGLUE BoolQ	ICL	149.72	6.68	5.77	167.87	5.99	7.98
SuperGLUE BoolQ	LoRA	106.11	10.42	4.47	132.44	9.29	7.21

Table 3: Inference efficiency metrics across datasets and model sizes. Throughput measures generated tokens per second, TPOT measures time per output token (ms), and VRAM indicates peak evaluation-time GPU memory usage.

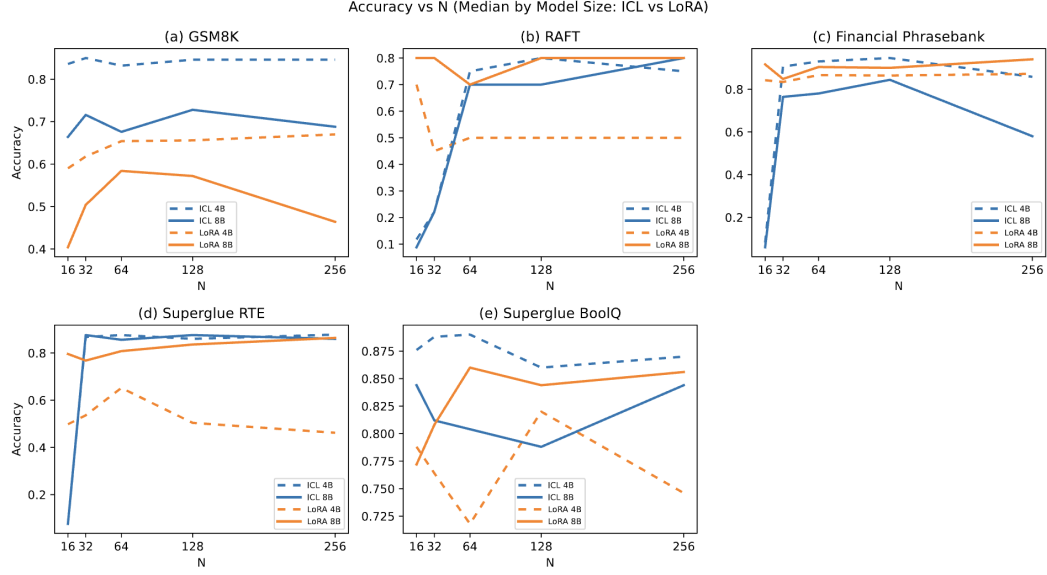


Figure 11: Accuracy vs. number of training examples N for ICL and LoRA across datasets, averaged across six models.

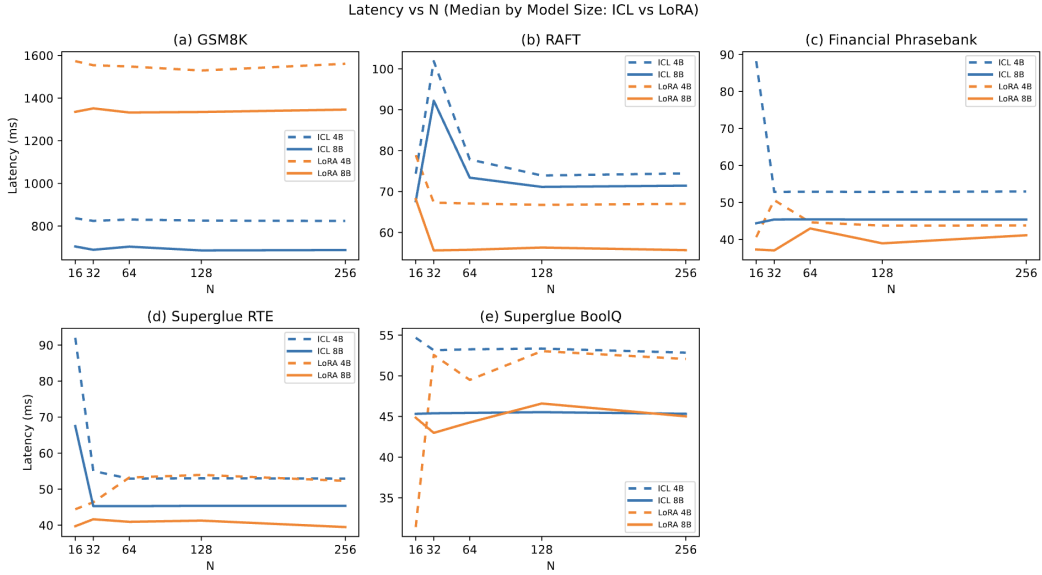


Figure 12: Latency vs. number of training examples N for ICL and LoRA across datasets, averaged across six models.

Macro-F1 Convergence (Avg of 3 Models): Sample Size to Full Dataset

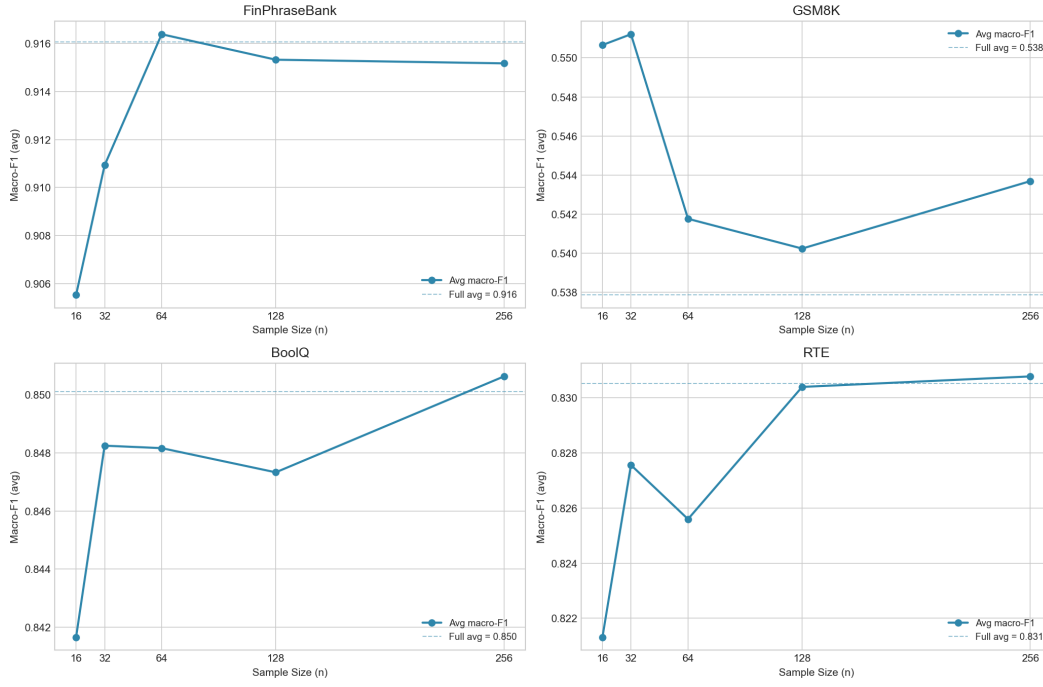


Figure 13: Zero-Shot Macro F1 Convergence Averaged Across Models

Macro-F1 Convergence: Sample Size to Full Dataset

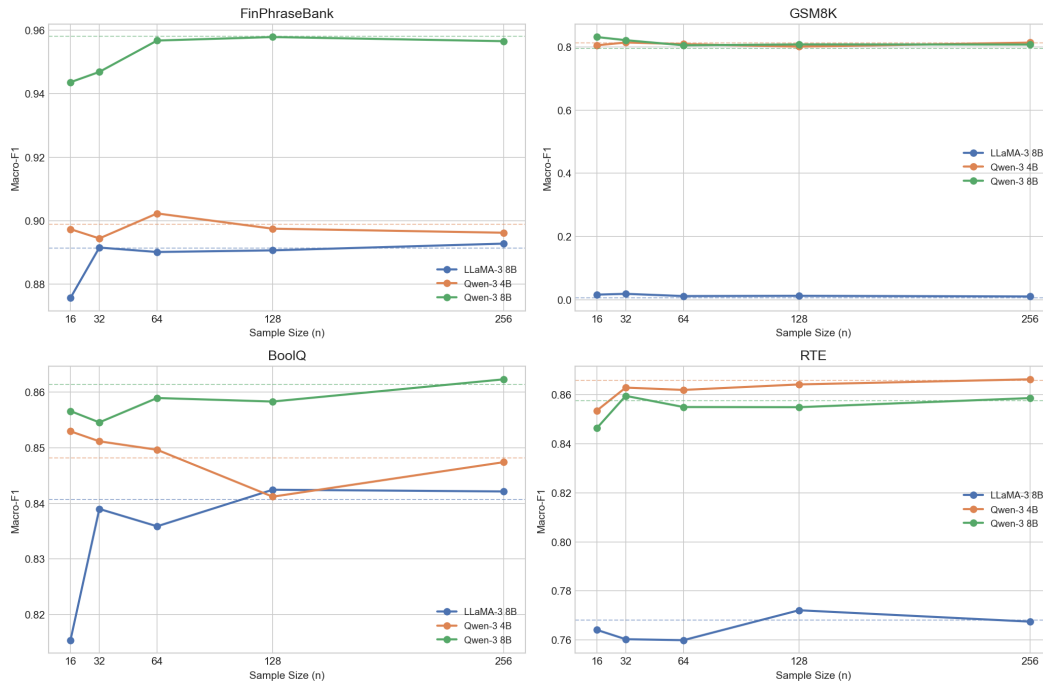
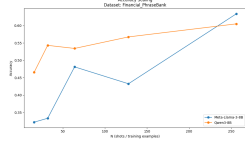
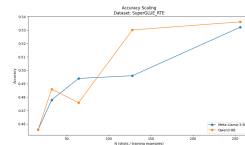


Figure 14: Zero-Shot Macro F1 Convergence by Model

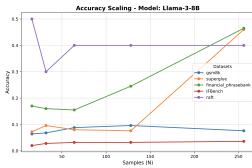


(a) FINBank

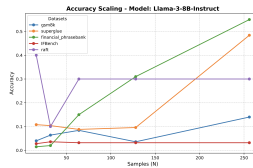


(b) SuperGLUE

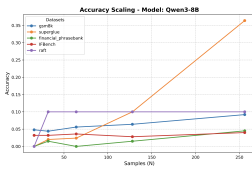
Figure 15: PeFT - Accuracy across N samples



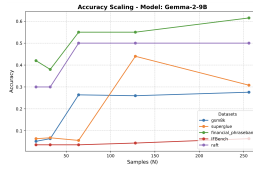
(a) LLaMA3-8B



(b) LLaMA3-8B-Instruct



(c) Qwen3-8B



(d) Gemma2-9B

Figure 16: Fine-tuned LoRA - Accuracy across N samples