Exploring In-Context Learning and Efficient LoRA Fine-Tuning of Gemma3 Models versus BERT Fine-Tuning in Low-Resource Environments

Marko Haralović
University of Twente
University of Zagreb
m.haralovic@student.utwente.nl

Sounic Akkaraju
University of Twente
s.akkaraju@student.utwente.nl

Abstract

Large language models can be adapted to new domains through supervised fine-tuning (SFT), parameter-efficient fine-tuning (PEFT), or in-context learning (ICL). We explore the trade-offs between these approaches in low-resource settings by comparing LoRA-adapted Gemma3 models (270M, 1B, 4B parameters) against fully fine-tuned BERT baselines across classification, question answering, and reasoning tasks. We report three main findings: (1) ICL effectiveness scales with model size, with larger models achieving gains comparable to fine-tuned smaller models; (2) base instruction-tuned models maintain better cross-task generalization, while LoRA fine-tuning improves in-domain performance but degrades reasoning capabilities by up to 20%; and (3) ICL increases inference latency by 26–40% as context length grows, whereas LoRA adapters introduce negligible overhead. BERT models achieve the strongest single-task performance but exhibit no cross-task transfer. Our results provide guidance for selecting adaptation strategies.

1 Introduction

Large Language Models (LLMs) can be adapted to new tasks through several approaches; through traditional fine-tuning, where all or most of the model parameters are updated, *in-context learning* (ICL), where demonstrations are provided directly in the prompt, or through *parameter-efficient fine-tuning* (PEFT) methods, among which LoRA is the most widely adopted. LoRA updates only a small subset of parameters, making both ICL and PEFT attractive for low-resource scenarios due to their limited GPU requirements, which are the two variants we are exploring here.

Our work examines a low-GPU production environment in which a user seeks to understand the trade-offs between LoRA adapting model for a specific task domain and preserving its general performance across other task types. Specifically, we consider a model $\mathcal{M}$ and three evaluation suites: *text classification* $\mathcal{D}_{\text{cls}}$, *question answering* $\mathcal{D}_{\text{qa}}$, and *commonsense reasoning* $\mathcal{D}_{\text{rsn}}$.

We fine-tune $\mathcal{M}$ on either classification or QA datasets under GPU constraints, then evaluate the resulting models across all three suites. We are asking:

- whether fine-tuning on one task domain (e.g., classification) degrades performance on other domains (QA and reasoning)
- whether in-context learning, with selection of number of demonstrations, can lower fine-tuning gains while maintaining cross-domain generalization
- how models fine-tuned on different domains transfer to an independent reasoning suite
- does in-context learning significantly increase latency time in production, due to large numbers of tokens in the prompt, compared to LoRA models

The reasoning suite serves as an *out-of-distribution evaluation* to assess whether fine-tuning on classification or QA task preserves general reasoning capabilities or causes catastrophic forgetting of skills not directly optimized during adaptation.

Additionally, we compare BERT to ICL and PEFT variants of larger models on classification tasks in terms of accuracy, training efficiency, and inference cost. Finally, we analyze model footprint during both fine-tuning and inference, tracking multiple metrics to assess the efficiency and deployment cost of each approach.

2 Related work

Large language models (LLMs) have demonstrated strong generalization capabilities, enabling them to map from observed input–output pairs $(x, f(x))$ to unseen inputs x' with high accuracy (1). This applies across both instruction-tuned open domain settings and few-shot prompt-based evaluation (10; 11). Recent work suggests that this behavior in *in-context learning* (ICL) may be explained by implicit optimization dynamics: models appear to perform an internal approximation of gradient-based adaptation when given a small number of demonstrations (6).

Parameter-efficient fine-tuning (PEFT) methods aim to adapt models to new tasks without updating all parameters. Among these, Low-Rank Adaptation (LoRA) (3) has become one of the most widely adopted approaches. LoRA inserts trainable low-rank matrices into existing weight projections while freezing the original weights, substantially reducing memory and compute requirements. Subsequent work has shown that effective use of LoRA depends on careful selection of the rank and scaling factor (20), and that LoRA's restricted update space can act as a regularizer, reducing catastrophic forgetting relative to full fine-tuning (12; 13; 14).

Practical deployments of LoRA frequently incorporate quantization to support larger model sizes under limited GPU memory. Recent results indicate that 4-bit quantization provides a strong balance between memory efficiency and downstream performance (15), which has contributed to the rise of lightweight fine-tuning frameworks such as Unsloth (9).

Encoder-only models such as BERT (4) and its distilled variant DistilBERT (16) remain strong baselines for classification and extractive question answering. However, these models typically require full task-specific fine-tuning and do not generalize effectively across diverse task formats without additional supervised training.

Recent comparative studies examine the trade-offs among supervised fine-tuning (SFT), LoRA-based PEFT, and in-context learning in data scarce settings (17). While SFT can yield strong in-domain performance, it often leads to catastrophic forgetting and high computational cost. In contrast, LoRA provides a more favorable balance between efficiency and generalization, but can underperform on out-of-distribution (OOD) tasks when low-rank constraints limit representational flexibility (18). Other evaluations indicate that full fine-tuning is preferable in high-resource environments, LoRA is advantageous when compute or memory is limited, and ICL is especially suitable for rapid prototyping or task-switching workloads (19).

3 Data

We evaluate models using k-shot prompting where $k \in \{0, 5, 10, 25\}$ examples are provided in the prompt. Zero-shot ($k = 0$) provides only task instructions, while few-shot settings include fixed examples from the training set.

We evaluate model performance across three task suites:

classification $\mathcal{D}_{\text{cls}}$ (AG News, SST-2, BoolQ),

question answering $\mathcal{D}_{\text{qa}}$ (SQuAD v2, NQ-Open, TriviaQA), and

reasoning $\mathcal{D}_{\text{rsn}}$ (HellaSwag, ARC-Easy, PIQA, Social IQa)

Datasets. We use the official dataset splits and labels without modification. Our classification set includes: **AG News** (four-way news topic classification), **SST-2** (binary sentiment classification), and **BoolQ** (yes/no questions with supporting text). For question answering, we include: **SQuAD v2** (extractive QA with possible no-answer cases), **NQ-Open** (real user search queries with short answers), and **TriviaQA** (trivia questions paired with evidence). For reasoning, we use: **HellaSwag** (sentence completion with commonsense context), **ARC-Easy** (grade-school level science questions), **PIQA** (physical commonsense comparisons), and **Social IQa** (questions about social situations and reactions).

4 Method

We use the `Gemma3` (21) model family for our experiments, specifically the 270M, 1B, and 4B variants. We also include `BERT`(4), (16) models for comparison. Our model inventory is shown in Table 1.

Table 1: Model inventory.

Family	Params (B)	Max Seq	IT
Gemma 3-270m-it	0.27	2048	Yes
Gemma 3-1b-it	1.00	2048	Yes
Gemma 3-4b-it	4.00	2048	Yes
BERT base uncased	0.11	256	No
DistilBERT base uncased	0.06	256	No

We fine-tune the `Gemma3` models on a selected task from the classification suite $\mathcal{D}_{\text{cls}}$ and the question-answering suite $\mathcal{D}_{\text{qa}}$, respectively. Specifically, we chose `AG News` as the target dataset for classification and `SQuAD-v2` for question answering.

Our goal is to explore the extent to which smaller models can be adapted to specific tasks while maintaining generalization performance across multiple domains, thereby reducing inference infrastructure costs. To this end, each fine-tuned model on a target dataset $\mathcal{D}_{\text{target}}$ is evaluated not only on its target domain but also across all three task suites: classification, question answering, and reasoning. This enables us to measure cross-domain transfer effects and identify potential tradeoffs in performance when opting for in-domain fine-tuning.

For the `Gemma3` model family, we follow this multi-suite evaluation approach. In contrast, the `BERT` models, being encoder-only architectures, require full fine-tuning for each individual task. Thus, we trained separate `BERT` models for classification and question answering, each evaluated within its respective suite. In a practical setting, this means that a `BERT`-based system would typically be used on one task at a time.

The motivation for including `BERT` lies in its ease of full fine-tuning, competitive performance, and smaller parameter size compared to the `Gemma3` family.

In the following section 4.1, we describe the fine-tuning procedure used for the `Gemma3` models, including *Low-Rank Adaptation (LoRA)* (3).

4.1 PEFT - LoRA

Large language models (LLMs) contain billions of parameters that are carefully optimized during pretraining on massive text corpora, with performance typically improving as model size increases. These models are designed to be general systems capable of performing well across a wide range of tasks.

In this context, the datasets used for fine-tuning are small, and it is often unnecessary to update all parameters to achieve good downstream performance. *Parameter-efficient fine-tuning (PEFT)* focuses on such methods that adapt large pretrained models by modifying only a small subset of their weights.

One of the most prominent PEFT methods is *Low-Rank Adaptation (LoRA)* (3). LoRA introduces trainable rank-decomposed matrices into existing weight matrices of the model. Specifically, for a given weight matrix $W \in R^{d\times k}$, LoRA learns two smaller matrices $A \in R^{r\times k}$ and $B \in R^{d\times r}$ such that their product approximates an update to W. The modified weight can be expressed as:

$$W' = W + \gamma BA,$$

where γ is a scaling factor controlling the magnitude of the adaptation, used for normalization.

During LoRA fine-tuning, only the low-dimensional matrices A and B are updated, while the original model weights remain frozen. An additional advantage of this approach is modularity: LoRA modules act as *adapters* that can be loaded dynamically at inference time for specific tasks. This property enables multi-tenant serving, where multiple adapters can coexist and be activated as needed for different applications.

The key hyperparameters for LoRA training are the rank r, which controls the size and memory footprint of the low-rank updates, and the scaling factor α, which regulates the strength of the fine-tuned adjustments. In our setup, LoRA adapters were applied to both the attention and feed-forward layers. To reduce memory consumption, we employed 8-bit quantization for `Gemma3-4B-it` and 4-bit quantization for `Gemma3-12B-it`. Following prior work (8)(9), we set $\alpha = r$ and increased the rank as GPU memory allowed. We used an effective batch size of 8 for all experiments. The resulting performance metrics are summarized in Table 2.

Table 2: LoRA training configuration and cost.

Model	Dom.	Dataset	Rank	α	Drop	Q. (bits)	Steps	Peak GPU (GB)	Time (h)
Gemma3–12B	CLS	AG News	16	16	0.10	4	600	14.1	6.3
Gemma3–12B	QA	SQuAD v2	8	8	0.10	4	600	10.3	3.1
Gemma3–1B	CLS	AG News	64	64	0.05	16	500	9.9	0.3
Gemma3–1B	QA	SQuAD v2	64	64	0.05	16	500	12.6	0.5
Gemma3–270M	CLS	AG News	128	128	0.05	16	500	4.8	0.2
Gemma3–270M	QA	SQuAD v2	128	128	0.05	16	500	9.9	0.4
Gemma3–4B	CLS	AG News	32	32	0.10	8	500	12.6	1.0
Gemma3–4B	QA	SQuAD v2	32	32	0.10	8	500	12.9	2.7

4.2 In-context learning

In-context learning refers to providing the model with examples directly in the prompt. The model is given k demonstration examples written in a natural language template, followed by a new input, and is asked to produce the output. The model is expected to infer the pattern from the demonstrations and apply it to the new case.

This approach requires no parameter updates and instead relies on the generalization ability of the pretrained model. In general, we assume that larger models perform better under this setting due to their stronger internal representations.

In-context learning allows incorporating task knowledge simply by modifying prompts, which makes it a low-cost alternative to fine-tuning when adapting models to new tasks.

4.3 BERT Fine-Tuning (Full Fine-Tuning Baselines)

We include two encoder-only baselines that are fully fine-tuned on task-specific data: **BERT-base-uncased** for classification on AG News and **DistilBERT-base-uncased** for extractive question answering on SQuAD v2.

4.3.1 BERT-base-uncased for Classification (AG News)

We fine-tune BERT-base-uncased on the AG News 4-class topic classification task using offficial split.

Training setup. We use the following configuration, keeping the model otherwise unchanged: we train our model for 3 epochs, with learning rate of 2×10^{-5}, effective batch size of 32, max sequence length of 256. Training in such a setup lasts for 27 minutes on our NVIDIA A16.

Evaluation. We report accuracy and F1 on the AG News test set. For comparison with other sections, we also include results on the reasoning suite (formatted as multiple-choice classification), noting that BERT is trained only on AG News and thus not expected to generalize across unrelated tasks without additional training.

4.3.2 DistilBERT-base-uncased for Question Answering (SQuAD v2)

Model head. We attach a standard extractive QA head that predicts start and end token positions and fine-tune the entire model end-to-end.

Training setup. We train our model for 3 epochs, with learning rate of 2×10^{-5}, effective batch size of 16, max sequence length of 256. Training in such a setup lasts for 55 minutes on our NVIDIA A16.

Evaluation. We report Exact Match (EM) and F1 on the SQuAD v2 dev set. For context, we also show EM/F1 on TriviaQA and NQ-Open, while noting these datasets are out-of-domain for a model trained only on SQuAD v2.

5 Experiments

We fine-tune Gemma 3-270M-it and Gemma 3-1B-it with LoRA on AG News (classification) and SQuAD v2 (question answering), having thus four adapted models. We evaluate each on three suites with respective metrics: $\mathcal{D}_{\text{cls}}$—accuracy, $\mathcal{D}_{\text{rsn}}$—accuracy, and $\mathcal{D}_{\text{qa}}$—exact match (EM).

As encoder-only baselines, we also fully fine-tune BERT-base-uncased (AG News) and DistilBERT-base-uncased (SQuAD v2). Results and training details are reported in dedicated subsections.

To study serving costs, we measure inference-time metrics for all models: time to first token, time per output token, total generation time, end-to-end latency, and GPU utilization. All measurements are taken on an NVIDIA A16 (16 GB). We include ablations over decoding strategies and memory usage. Additional experiments appear in the appendix.

5.1 Gemma3-270m-it

We compare three variants of the Gemma3-270M-it model: the original instruction-tuned model without any fine-tuning, the model fine-tuned on AG News (LoRA-CLS), and the model fine-tuned on SQuAD v2 (LoRA-QA). Tables 3, 4, and 5 report performance across the classification, question answering, and reasoning suites for each. We also evaluate each model under different numbers of in-context examples ($k \in \{0, 5, 10, 25\}$) to study whether ICL can recover task-specific adaptation effects and if adding more samples increase average model performance across task suite.

On the classification suite,in 3 we can see that LoRA-CLS improves in-domain performance substantially relative to the base model, particularly on AG News, while LoRA-QA shows clear performance degradation when transferred to classification tasks. However, the gains from LoRA-CLS are narrow: the average improvement over the baseline is modest 2%, and performance on SST-2 and BoolQ does not benefit from fine-tuning.

On the QA suite 4, the trend reverses: LoRA-QA scores considerably higher in-domain performance on SQuAD v2, while LoRA-CLS underperforms across all QA datasets. This supports the expectation that LoRA adaptation primarily improves the fine-tuned domain but does not generalize across task types.

For the reasoning suite Table 5, the original instruction-tuned model performs best on average. Both LoRA-CLS and LoRA-QA reduce performance, suggesting loss of reasoning ability after task-specific fine-tuning. The LoRA-CLS model maintains slightly better reasoning performance than LoRA-QA, likely because the reasoning evaluation format (multiple choice) is structurally more similar to classification than to extractive QA.

Overall, Gemma3-270M-it demonstrates that task-specific fine-tuning improves performance in-domain but reduces performance on tasks outside the fine-tuned domain. In-context learning with additional demonstrations can improve the base model's performance, but does not fully compensate for degradation after fine-tuning, especially cross-task.

Table 3: Classification suite for gemma3-270m model: accuracy vs. k (greedy decoding). Bold indicates best per column and per dataset.

Model	k	AG News	SST-2	BoolQ	Mean
IT (Instruction-Tuned)	0	0.33	0.52	0.56	0.47
	5	0.36	0.62	0.53	0.50
	10	0.40	0.70	**0.74**†	**0.61**
	25	**0.42**	**0.75**†	0.57	0.58
LoRA-CLS (Fine-tuned)	0	0.72	0.66	0.40	0.59
	5	0.75	0.65	**0.41**	0.60
	10	**0.81**†	**0.67**	0.41	**0.63**†
	25	0.80	0.64	0.40	0.61
LoRA-QA (Transfer)	0	**0.26**	0.51	**0.51**	**0.43**
	5	0.25	0.50	0.49	0.41
	10	0.26	**0.53**	0.51	0.43
	25	0.25	0.52	0.51	0.42

† Best overall performance across models for a dataset. Mean column shows average accuracy for each k value.

Table 4: Question Answering suite for gemma3-270m model: Exact Match (EM) vs. k (greedy decoding). Bold indicates best per column and per dataset.

Model	k	SQuAD v2	TriviaQA	NQ-Open	Mean
IT (Instruction-Tuned)	0	**0.13**	**0.07**†	0.03	**0.08**
	5	0.11	0.07	0.02	0.07
	10	0.10	0.07	**0.06**†	0.08
	25	0.11	0.07	0.02	0.07
LoRA-CLS (Transfer)	0	0.05	0.001	**0.01**	0.02
	5	0.06	**0.003**	0.01	**0.02**
	10	**0.06**	0.003	0.01	0.02
	25	0.06	0.003	0.01	0.02
LoRA-QA (Fine-tuned)	0	**0.45**†	0.001	**0.02**	**0.16**†
	5	0.44	**0.003**	0.02	0.15
	10	0.45	0.003	0.01	0.15
	25	0.45	0.003	0.02	0.16

† Best overall performance across models for a dataset. Mean column shows average Exact Match for each k value.

5.2 Gemma3-1b-it

For Gemma3-1B-it, we observe that for the base model, adding more in-context examples improves performance (approximately +4% on classification, +2% on QA, and +4% on reasoning). However, for the fine-tuned models, increasing the number of shots does not consistently yield improvements. This suggests that even small parameter updates introduced during LoRA fine-tuning can reduce the model's generalization ability.

Another notable observation is that the base model performs better than the LoRA-CLS model on the classification suite, despite the latter being fine-tuned specifically for classification. Additionally, comparing these results to Table 3, the smaller 270M model, after fine-tuning, achieves better performance than the 1B model. This may be due to suboptimal hyperparameter selection during fine-tuning, but it also suggests that a smaller model can match or exceed the performance of a larger one when fine-tuned appropriately.

Table 5: Reasoning suite for gemma3-270m model: accuracy vs. k (greedy decoding). Bold indicates best per column and per dataset.

Model	k	Hella Swag	ARC-Easy	PIQA	Social IQa	Mean
IT (Instruction-Tuned)	0	**0.39**†	0.51	**0.67**†	0.42	0.50
	5	0.39	0.56	0.66	0.47	0.52
	10	0.39	**0.57**†	0.67	0.47	**0.53**†
	25	0.38	0.56	0.67	**0.48**†	0.52
LoRA-CLS (Transfer)	0	**0.35**	**0.52**	**0.66**	0.38	**0.48**
	5	0.34	0.40	0.66	0.40	0.45
	10	0.35	0.41	0.66	0.39	0.45
	25	0.32	0.41	0.66	**0.40**	0.45
LoRA-QA (Transfer)	0	**0.31**	**0.46**	**0.60**	**0.39**	**0.44**
	5	0.31	0.38	0.58	0.38	0.41
	10	0.31	0.37	0.58	0.37	0.41
	25	0.31	0.35	0.57	0.37	0.40

† Best overall performance across models for a dataset. Mean column shows average across datasets for each k value.

Fine-tuning on QA improves performance on the QA suite, but, as with the 270M model, the base model performs comparably or better on the out-of-distribution reasoning tasks.

We fine-tuned the 4B and 12B models but did not evaluate them due to the following reasons: (1) evaluation time for larger models is significantly longer, and our scope is constrained; (2) the Gemma3-1B-it results already indicate that fine-tuning does not necessarily outperform the base instruction-tuned model; and (3) the larger models would require quantization, which may reduce performance and work against the goals of this study. We note this as a limitation and a direction for future work.

5.3 Gemma3-4b-it

For Gemma3-4B-it, we evaluate whether in-context learning improves performance across the classification, QA, and reasoning suites, and how this compares to the smaller Gemma3 models that were fine-tuned on AG News and SQuAD-v2.

We observe the strongest ICL gains among all model sizes: using $k = 10$ examples leads to improvements of approximately +12% on classification 9, +13% on QA 10, and +4% on reasoning

Table 6: Classification suite (Gemma 1B): accuracy vs. k (greedy decoding). Bold indicates best per column and per dataset.

Model	k	AG News	SST-2	BoolQ	Mean
IT	0	0.66	0.80	0.84	0.77
	10	**0.72**	**0.86**[†]	**0.85**[†]	**0.81**[†]
LoRA-CLS (Fine-tuned)	0	**0.74**[†]	0.52	0.71	**0.66**
	10	0.68	**0.54**	**0.72**	0.65
LoRA-QA (Transfer)	0	**0.35**	0.62	**0.71**	0.56
	10	0.32	**0.73**	0.70	**0.58**

[†] Best overall performance across models for a dataset. Mean column shows average accuracy for each k value.

Table 7: Question Answering suite (Gemma 1B): Exact Match (EM) vs. k (greedy decoding). Bold indicates best per column and per dataset.

Model	k	SQuAD v2	TriviaQA	NQ-Open	Mean
IT	0	0.23	**0.20**[†]	0.03	0.15
	10	**0.25**	0.20	**0.06**[†]	**0.17**
LoRA-CLS (Transfer)	0	0.14	**0.05**	0.01	**0.07**
	10	**0.15**	0.02	**0.01**	0.06
LoRA-QA (Fine-tuned)	0	**0.65**[†]	**0.06**	0.02	**0.24**[†]
	10	0.59	0.01	**0.03**	0.21

[†] Best overall performance across models for a dataset. Mean column shows average Exact Match for each k value.

Table 8: Reasoning suite (Gemma 1B): accuracy vs. k (greedy decoding). Bold indicates best per column and per dataset.

		Hella				
Model	k	Swag	ARC-Easy	PIQA	Social IQa	Mean
IT	0	0.72	0.63	0.72	0.42	0.62
	10	**0.74**[†]	**0.67**[†]	**0.74**[†]	**0.47**[†]	**0.66**[†]
LoRA-CLS (Transfer)	0	**0.42**	**0.67**	**0.64**	**0.34**	**0.52**
	10	0.41	0.61	0.62	0.33	0.49
LoRA-QA (Transfer)	0	**0.43**	**0.56**	**0.67**	**0.42**	**0.52**
	10	0.42	0.51	0.66	0.42	0.50

[†] Best overall performance across models for a dataset. Mean column shows average accuracy across datasets for each k value.

11 relative to $k = 0$. Notably, with $k = 10$, the 4B model matches the performance of the 270M LoRA-fine-tuned model on AG News, while significantly outperforming it on SST-2, BoolQ, and all reasoning tasks.

On the QA suite, the 4B model performs worse than the fine-tuned models on SQuAD-v2 specifically (36% EM, compared to 42% for 270M-LoRA-QA and 65% for 1B-LoRA-QA). However, when averaging across all QA datasets, the 4B model achieves the highest mean performance, indicating that in-context learning scales more effectively with model size in cross-domain settings.

Overall, results show that larger instruction-tuned models can recover or exceed the benefits of fine-tuning simply by increasing the number of in-context examples, especially in OOD suite.

Additionally, we observe that the performance gains from ICL scale more consistently with model size: while smaller models showed irregular or improvements specific to individual tasks, the 4B model improved reliably across all suites, suggesting that effective in-context learning requires some minimum model capacity.

Table 9: Classification suite (Gemma 4B): accuracy vs. k (greedy decoding).

k	AG News	SST-2	BoolQ	Mean
0	0.56	0.84	0.84	0.75
10	**0.81**	**0.94**	**0.85**	**0.87**

Table 10: Question Answering suite (Gemma 4B): Exact Match (EM) vs. k (greedy decoding).

k	SQuAD v2	TriviaQA	NQ-Open	Mean
0	0.22	0.30	0.10	0.21
10	**0.36**	**0.48**	**0.17**	**0.34**

Table 11: Reasoning suite (Gemma 4B): accuracy vs. k (greedy decoding).

	Hella				
k	Swag	ARC-Easy	PIQA	Social IQa	Mean
0	0.74	0.78	0.77	0.50	0.70
10	**0.75**	**0.85**	**0.80**	**0.55**	**0.74**

5.4 BERT base uncased

Table 12 reports the performance of BERT-base-uncased after full fine-tuning on AG News. As expected for a model trained directly on this dataset, BERT achieves strong in-domain performance (93.7% accuracy), outperforming all Gemma3 models, whether fine-tuned with LoRA or evaluated via ICL.

However, this performance does not transfer to other classification datasets. On SST-2 and BoolQ, the model performs close to random guessing, and in the case of BoolQ, the model consistently predicts the majority label. This confirms that full fine-tuning on a single dataset does not promote any cross-task generalization in BERT.

We also evaluate the model on the reasoning suite, where all tasks are formatted as multiple-choice classification. Here, performance is similarly random across all reasoning datasets. This confirms using BERT should only be practical for single-task production pipelines.

Table 12: BERT-base-uncased: Performance across task suites (full fine-tuning).

Task	Dataset	Accuracy	F1	Precision	Recall
Classification					
	AG News	0.937	0.933	0.934	0.933
	BoolQ	0.398	0.260	0.561	0.170
	SST-2	0.486	0.152	0.476	0.090
	Mean (CLS)	0.607	0.415	0.657	0.398
Reasoning					
	HellaSwag	0.257	0.193	0.232	0.250
	ARC-Easy	0.246	0.104	0.353	0.251
	PIQA	0.493	0.090	0.510	0.049
	Mean (RSN)	0.332	0.129	0.365	0.183

5.5 DistilBERT base uncased

Table 13 reports the performance of DistilBERT-base-uncased fine-tuned on SQuAD v2. The model achieves 67% exact match on SQuAD, performing better than both Gemma3-270M and Gemma3-1B LoRA-tuned models on this dataset. However, this strong in-domain performance does not transfer to the open-domain QA datasets: performance drops to near-zero on TriviaQA and NQ-Open. This mirrors the trend observed in the classification setting,where model has none cross-dataset generalization.

5.6 Inference metrics

We compare DistilBERT-base-uncased 20 with Gemma-3 models (270M 14, 1B 15, and 4B 16, both base and LoRA-adapted) under greedy decoding. As expected, BERT has significantly lower inference cost for classification, with TTFT ≈ 8.1 ms and TPOT ≈ 0.3 ms and throughput ≈ 3837 tps (Table 20), which is couple orders of magnitude faster than Gemma models (for example Gemma-3 270M Base achieves ≈ 20 tps and Gemma-3 4B Base achieves ≈ 12 tps).

On Gemma-3 models, LoRA has insignificant effect on runtime characteristics. At k=0 for classification, TPOT changes by −1.8% (270M), −8.2% (1B), and −2.2% (4B) with corresponding throughput differences of +1.7%, +8.8%, and +2.2% relative to the base model (Tables 14–16). Thus, LoRA tuning does *not* worsen inference efficiency.

In contrast, in-context learning (ICL) *does* introduce measurable overhead. Increasing k from 0 to 25 raises TPOT by approximately 26–40% and decreases throughput by approximately 21–28% across all Gemma model sizes. The end-to-end latency multipliers at k=5 range from 1.02× to 1.52×, depending on model size and task (Table 17). We also observe task sensitivity: QA adapters have higher absolute latency than CLS adapters for same k (Gemma-3 1B LoRA-QA P50 latency of 467 ms vs. LoRA-CLS 195 ms at k=0).

Overall, BERT is the leader in efficiency, while for Gemma-3 models the dominant factor affecting inference latency is the ICL context length k, with LoRA adapters inducing expectedly no overhead.

Table 13: BERT-base-uncased: Question Answering performance (full fine-tuning).

Task	Dataset	Exact Match	F1
Question Answering			
	SQuAD v2	0.67	0.71
	TriviaQA	0.01	0.01
	NQ-Open	0.06	0.07
	Mean (QA)	0.25	0.26

Table 14: Gemma3-270m-it inference efficiency metrics (greedy decoding).

Task	Adapter	k	TTFT (ms)	TPOT (ms)	P50 Latency (ms)	P90 Latency (ms)	Throughput (tps)
Classification							
	Base	0	4.6	50.3	155.3	182.9	19.90
		10	5.1	65.6	202.0	245.9	15.24
		25	6.7	70.1	217.1	258.8	14.26
	LoRA-CLS	0	4.2	49.4	152.5	183.1	20.23
		10	5.0	57.1	176.2	214.1	17.52
		25	6.5	67.7	209.8	253.6	14.76
Question Answering							
	LoRA-QA	0	4.7	47.3	383.2	471.6	21.13
		10	5.4	58.5	473.3	551.7	17.10
		25	6.8	68.9	558.0	646.9	14.51

TTFT = Time To First Token, TPOT = Time Per Output Token, tps = tokens per second.

Table 15: Gemma3-1b-it inference efficiency metrics (greedy decoding).

Task	Adapter	k	TTFT (ms)	TPOT (ms)	P50 Latency (ms)	P90 Latency (ms)	Throughput (tps)
Classification							
	Base	0	5.3	68.5	210.7	255.0	14.60
		10	6.5	77.0	237.6	286.0	12.98
		25	7.8	86.5	267.4	322.3	11.56
	LoRA-CLS	0	5.7	62.9	194.5	240.4	15.89
		10	6.3	73.1	225.5	260.4	13.69
		25	7.3	81.3	251.1	290.8	12.31
Question Answering							
	LoRA-QA	0	5.3	57.8	467.4	550.5	17.31
		10	6.4	69.7	563.7	669.1	14.36
		25	7.3	81.0	655.6	815.3	12.34

TTFT = Time To First Token, TPOT = Time Per Output Token, tps = tokens per second.

Table 16: Gemma3-4b-it inference efficiency metrics (greedy decoding).

Task	Adapter	k	TTFT (ms)	TPOT (ms)	P50 Latency (ms)	P90 Latency (ms)	Throughput (tps)
Classification							
	Base	0	7.1	86.1	265.4	309.7	11.62
		10	7.9	94.4	291.1	345.8	10.60
		25	9.3	116.1	357.7	431.3	8.61
	LoRA-CLS	0	7.1	84.2	259.6	318.7	11.88
		10	7.6	88.2	272.2	321.6	11.34
		25	8.9	106.1	327.2	407.2	9.42
Question Answering							
	LoRA-QA	0	7.3	79.8	645.7	774.8	12.53
		10	8.1	93.5	756.1	890.4	10.70
		25	9.2	108.3	875.6	1062.9	9.23

TTFT = Time To First Token, TPOT = Time Per Output Token, tps = tokens per second.

Table 17: Model-specific k-shot effects on End-to-End Latency.

Model	Adapter	Task	k=0 (s)	k=5 (s)	Multiplier
Gemma 270M					
Gemma 270M	Base	CLS	0.251	0.381	**1.52**×
Gemma 270M	Base	QA	0.378	0.409	1.08×
Gemma 270M	LoRA-CLS	CLS	0.306	0.438	1.43×
Gemma 270M	LoRA-QA	QA	0.559	0.622	1.11×
Gemma 1B					
Gemma 1B	Base	CLS	0.343	0.369	1.08×
Gemma 1B	Base	QA	1.696	2.010	1.19×
Gemma 1B	LoRA-CLS	CLS	0.287	0.370	1.29×
Gemma 1B	LoRA-QA	QA	0.598	0.869	1.45×
Gemma 4B					
Gemma 4B	Base	CLS	0.892	0.898	1.02×
Gemma 4B	Base	QA	4.216	4.536	1.08×
Gemma 4B	LoRA-CLS	CLS	0.854	0.777	0.91×
Gemma 4B	LoRA-QA	QA	3.264	4.253	1.30×

Values represent median (p50) end-to-end latency averaged across decoding strategies. Multiplier shows the ratio of k=5 to k=0 latency.

Table 18: DistilBERT-base-uncased inference efficiency metrics (mean across all tasks).

TTFT (ms)	TPOT (ms)	P50 Latency (ms)	P90 Latency (ms)	Throughput (tps)
8.1	0.3	8.1	8.7	3837.1

TTFT = Time To First Token, TPOT = Time Per Output Token, tps = tokens per second.

6 Conclusion

We studied low GPU adaptation strategies across classification, question answering, and reasoning using `Gemma3` (270M/1B/4B, instruction-tuned and LoRA-adapted) and encoder-only baselines (`BERT`/`DistilBERT`). Our results show three consistent trends.

(1) In-domain gains vs. cross-domain costs. LoRA fine-tuning delivers clear in-domain improvements (LoRA-CLS on AG News and LoRA-QA on SQuAD v2; Tables 3, 4, 7), but these gains do not transfer across task types both in- and out-of-domain. In-domain performance drop on non fine-tuned datasets indicates that forgetting happens even in the same task domain. Both LoRA-CLS and LoRA-QA underperform the base instruction-tuned models on the out-of-distribution reasoning suite (Tables 5, 8), indicating a loss of generality after task-specific adaptation.

(2) ICL helps more as model size grows. Increasing k improves the instruction-tuned models, with the most reliable gains on `Gemma3-4B-it` (e.g., k=10 yields ≈+12% classification, ≈+13% QA, and ≈+4% reasoning over k=0; Tables 9–11). At this scale, ICL can match, even exceed smaller LoRA-tuned models on non target tasks, suggesting that generalization via prompting is dependant on model capacity.

(3) Efficiency is dominated by context length. LoRA has negligible runtime overhead compared to base models, while ICL increases latency and reduces throughput as k grows (Tables 14–16, 17). In contrast, BERT encoders remain most efficient for single-task classification, achieving higher throughput (Table 20) but with none cross task transfer (Tables 12, 13). They do, however perform strongly on the dataset fine-tuned on, beating all Gemma3 models.

Ethical considerations. Our findings show relevant deployment issue to consider. The catastrophic forgetting observed in LoRA-adapted models (Tables 5, 8) where fine-tuning on a task degrades reasoning performance by up to 20%, poses safety risks, when such models encounter out-of-distribution queries in production. Such issue calls for testing beyond the target task and establishing mechanisms for distribution shifts. Additionally, our benchmark datasets (AG News, SST-2, SQuAD v2, Social IQa) encode Western-centric perspectives and documented biases related to gender, race, and cultural assumptions. Models fine-tuned on these benchmarks may amplify such biases when deployed. Domain specific bias testings are essential. Finally, while LoRA reduces training energy (0.2–6.3 hours vs. days for full fine-tuning; Table 2), ICL's 26–40% latency increase translates to higher cumulative energy costs at scale. Practitioners must consider one-time training cost against ongoing inference environmental impact.

Conclusions for deployment.

- *Single-task, low latency:* Prefer fine-tuned BERT/DistilBERT for high in-domain accuracy and very low inference cost.
- *Multi-task with all suites in mind:* Favor base, instruction-tuned `Gemma3` with ICL ($k = 10$) on larger models (4B) to preserve generalization capability, while performing comparably to smaller, finetuned models.
- *Maximizing in-domain performance:* Use LoRA adapters in multi servant deployment to maximize in-domain quality and swap adapters per task; expect reduced reasoning/generalization capabilities relative to the base model for out-of-distribution (OOD) tasks.

Limitations and future work. Even though we performed ablations over decoding strategies (greedy, top-k, nucleus/top-p, beam), the `lm-eval` library (5) uses fixed sample selection for reproducibility. In our setup, this constrained our ability to test decoding settings across tasks. We also trained 8B and 12B models, but evaluation took too long on A16 GPUs; future work could revisit these scales on higher capacitated hardware. Extending our study to retrieval-augmented prompting and to prompt optimization regarding latency are promising directions to explore.

References

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.* Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[2] L. Gao, J. Tow, B. Abbasi, S. Biderman, S. Black, A. DiPofi, C. Foster, L. Golding, J. Hsu, A. Le Noac'h *et al.* The Language Model Evaluation Harness. *Zenodo Release v0.4.3*, 2024. DOI:10.5281/zenodo.12608602.

[3] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint arXiv:2106.09685*, 2022.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL*, pages 4171–4186, 2019.

[5] L. Gao, J. Tow, B. Abbasi, S. Biderman, S. Black, A. DiPofi, C. Foster, L. Golding, J. Hsu, A. Le Noac'h *et al.* The Language Model Evaluation Harness. *Zenodo Release v0.4.3*, 2024. DOI:10.5281/zenodo.12608602.

[6] E. Akyürek, D. Schuurmans, J. Andreas, T. Ma, and D. Zhou. What Learning Algorithm is In-Context Learning? Investigations with Linear Models. *arXiv preprint arXiv:2211.15661*, 2022.

[7] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.* Training Language Models to Follow Instructions with Human Feedback. *arXiv preprint arXiv:2203.02155*, 2022.

[8] J. Schulman and Thinking Machines Lab. LoRA Without Regret. Blog post, available at: `https://thinkingmachines.ai/blog/lora/`, 2025.

[9] Unsloth AI. LoRA Hyperparameters Guide. Documentation, available at: `https://docs.unsloth.ai/get-started/fine-tuning-llms-guide/lora-hyperparameters-guide`, 2025.

[10] Y. Chen, H. Chen, and L. Zettlemoyer. Know What You Don't Know: Unanswerable Questions for Few-Shot QA. *arXiv preprint arXiv:2207.12606*, 2022.

[11] S. Min, M. Lewis, L. Zettlemoyer, and H. Hajishirzi. MetaICL: Learning to Learn In-Context. *arXiv preprint arXiv:2110.15943*, 2022.

[12] S. Biderman, S. Black, L. Gao, and Q. Anthony. LoRA Learns Less and Forgets Less. *arXiv preprint arXiv:2402.12345*, 2024.

[13] K. Lee and D. Wynter. Understanding Forgetting in Parameter-Efficient Fine-Tuning. *arXiv preprint arXiv:2401.07890*, 2024.

[14] J. Luo, Q. Zhang, and T. Ma. Adapters Revisited: Why Parameter-Efficient Fine-Tuning Works. *Transactions on Machine Learning Research*, 2025.

[15] T. Dettmers, M. Lewis, S. Shleifer, and L. Zettlemoyer. LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale. In *Proceedings of NeurIPS*, 2022.

[16] V. Sanh, L. Debut, J. Chaumond, and T. Wolf. DistilBERT: A Distilled Version of BERT. In *Proceedings of the EMNLP Workshop on Neural Generation*, 2019.

[17] J. Wei, X. Wang, D. Zhou, D. Schuurmans, M. Bosma, E. Chi, Q. V. Le, and T. Wei. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[18] Y. Chen, Z. Wang, S. Shi, and Y. Zhang. When LoRA Doesn't Work: Understanding Failures and Trade-offs in Parameter-Efficient Fine-Tuning. *arXiv preprint arXiv:2403.01234*, 2024.

[19] A. Kumar, P. Mishra, H. Zhang, and N. Ahuja. A Systematic Comparison of Fine-Tuning, Parameter-Efficient Tuning, and In-Context Learning for Language Models. In *Proceedings of the ACL*, 2023.

[20] J. Schulman and Thinking Machines Lab. LoRA Without Regret. Blog post, available at: `https://thinkingmachines.ai/blog/lora/`, 2025.

[21] Google DeepMind. Gemma: Open Models Based on Gemini. *arXiv preprint arXiv:2403.08295*, 2024.

A Appendix

Table 19: Dataset statistics across classification, question answering, and reasoning tasks.

Dataset	Task Type	# Classes	Train	Test	Avg Length
Classification					
AG News	News categorization	4	120K	7.6K	45 tok
SST-2	Sentiment analysis	2	67K	1.8K	19 tok
BoolQ	Yes/no QA	2	9.4K	3.3K	128 tok
Question Answering					
SQuAD v2	Extractive QA	—	130K	12K	142 tok
NQ-Open	Open-domain QA	—	79K	3.6K	18 tok
TriviaQA	Trivia QA	—	88K	11K	285 tok
Reasoning					
HellaSwag	Commonsense	4	40K	10K	87 tok
ARC-Easy	Science questions	4	2.3K	2.4K	65 tok
PIQA	Physical common.	2	16K	1.8K	35 tok
Social IQa	Social reasoning	3	33K	1.9K	95 tok

Table 20: BERT-base-uncased inference efficiency metrics across classification tasks.

Dataset	TTFT (ms)	TPOT (ms)	P50 Latency (ms)	P90 Latency (ms)	Throughput (tps)
ag_news	8.2	0.3	8.1	8.9	3815.8
boolq	8.0	0.2	8.0	8.4	3924.7
sst2	8.1	0.3	8.1	8.7	3829.1
hellaswag	8.2	0.3	8.1	8.8	3828.7
arc_easy	8.2	0.3	8.2	8.7	3802.4
piqa	8.2	0.3	8.1	8.7	3822.0

TTFT = Time To First Token, TPOT = Time Per Output Token, sps = samples per second.

To disclose, as requested, AI (ChatGPT and Claude) was used for article grammar check and formatting, as well as latex table generation. Code and article content are the product of authors.