DOES FINE-TUNING STILL BEAT IN-CONTEXT LEARNING UNDER LORA?

Anonymous authors
Paper under double-blind review

ABSTRACT

Does the advantage of fine-tuning (FT) over in-context learning (ICL) on synthetic formal languages persist under parameter-efficient adaptation? We compare full FT, LoRA and 4-bit QLoRA with single-pass ICL on Qwen2.5 base models from 0.5B to 3B parameters, using matched sets of 32 training strings and a published discriminative ROC-AUC evaluation. Under the published protocol, which selects each run's best epoch using test scores, all tested fine-tuning methods achieve higher mean AUC than ICL at every evaluated model size. However, at 3B, this ranking reverses when the same runs are evaluated after 50 epochs. To test whether the advantage survives validation-based checkpoint selection, we repeat the 3B comparison across four languages, choosing epochs on fresh validation strings and evaluating on separate held-out test strings. Full FT, LoRA and QLoRA achieve mean test AUCs of 0.942, 0.921 and 0.919, respectively, versus 0.873 for ICL; only the full-FT advantage remains statistically significant after correction for multiple comparisons. Selecting epochs on the test strings instead raises mean AUC only slightly, while all three fine-tuning methods again fall below ICL on average at epoch 50. Under the published protocol, adapter-versus-full-FT comparisons also depend on the adapter learning rate. Thus, in this 32-example setting at 3B, validation-based stopping preserves full FT's advantage over single-pass ICL, with weaker evidence for the adapters, whereas training for 50 epochs reverses the comparison.

1 INTRODUCTION

Fine-tuning and in-context learning are two standard ways to adapt pretrained language models. Comparing them on natural-language tasks is difficult: benchmark items may occur in pretraining data (Xu et al., 2024), and task boundaries can be unclear. Ghosh et al. (2026) address these problems with synthetic context-free languages. Known grammars give exact membership labels, and newly sampled strings are unlikely to overlap with pretraining data. They find that fine-tuning outperforms in-context learning on in-distribution strings.

Their study updates every model parameter and leaves open whether parameter-efficient methods "may lead to different conclusions" (Ghosh et al., 2026). We test this with LoRA (Hu et al., 2021) and 4-bit QLoRA (Dettmers et al., 2023). The answer is not assured: LoRA can learn less than full fine-tuning on natural-language corpora (Biderman et al., 2024).

We reuse the released grammar generators and evaluation, give methods the same training strings, and first reproduce the original baselines. We compare full FT, ICL, LoRA at ranks 8, 64 and 256, and QLoRA across three model sizes and up to four languages. We then repeat the 3B comparison with checkpoints chosen on fresh validation strings. This tests how far the published result extends to low-rank adaptation and when the method ranking changes.

- **The best-epoch result extends to LoRA and QLoRA.** Under the reference protocol, every adaptation method has higher mean AUC than ICL at every tested scale (Section 4.2).
- **Training duration changes the comparison.** At 3B, full FT and LoRA retain higher mean AUC than ICL with validation-based checkpoint selection. All three adaptation methods fall below ICL at epoch 50 on both the original and new strings (Sections 4.3–4.4).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

- **Adapter-versus-full comparisons depend on the training setup.** Full FT leads the adapters by about two AUC points at the development learning rate on fresh strings, but these differences do not survive multiplicity correction. The rate selected on a separate grammar did not transfer, and changing the software and hardware has a comparable effect on full FT (Sections 4.5 and 5).

2 Related work

Fine-tuning versus in-context learning. ICL adapts a frozen model using prompt examples (Brown et al., 2020). Comparisons with fine-tuning give mixed results. Liu et al. (2022) find parameter-efficient fine-tuning more accurate and cheaper than ICL; Mosbach et al. (2023) find similar generalization at matched model sizes. Other work finds ICL competitive, especially with many prompt examples (Bertsch et al., 2024; Yin et al., 2024). Results depend on how much information prompts provide (Le Scao & Rush, 2021), whether models recognize a familiar task or learn from examples (Pan et al., 2023), model scale (Wei et al., 2023), and example order and format (Zhao et al., 2021). Ghosh et al. (2026) find a full-FT advantage on formal languages. We test whether that specific result extends to LoRA and QLoRA under the original and validation-selected evaluations.

Parameter-efficient adaptation. Adapters (Houlsby et al., 2019) and prompt tuning (Lester et al., 2021) train a small set of added parameters. LoRA trains low-rank weight updates (Hu et al., 2021); QLoRA adds these to a 4-bit base model (Dettmers et al., 2023). Their performance relative to full fine-tuning depends on the data, model size and method (Zhang et al., 2024; Srinivasan et al., 2024). LoRA can learn less while forgetting less on large instruction-tuning corpora (Biderman et al., 2024). We study the much smaller setting of 32 training strings.

Formal languages. Exact membership labels and controlled structure make formal languages useful for studying model capacity (Bhattamishra et al., 2020; Delétang et al., 2023; Strobl et al., 2023; Liu et al., 2023) and inductive biases (White & Cotterell, 2021; Papadimitriou & Jurafsky, 2023; Kallini et al., 2024). Prior work studies transformers learning hierarchical probabilistic grammars (Allen-Zhu & Li, 2023; Murty et al., 2023) and learning formal languages in context (Akyürek et al., 2024). Newly generated strings also reduce the contamination concerns of natural-language benchmarks (Xu et al., 2024; Dominguez-Olmedo et al., 2025).

Training duration and evaluation. With few examples, fine-tuning is sensitive to random seeds, data order and training duration (Dodge et al., 2020; Mosbach et al., 2021; Zhang et al., 2021). Early stopping on held-out data is a standard response (Prechelt, 1998); our 3B results show this sensitivity in a setting with exact membership labels. Selecting models on their reporting data gives optimistic estimates (Cawley & Talbot, 2010; Varma & Simon, 2006) and is a form of data leakage (Kapoor & Narayanan, 2023). Comparisons must also account for run-to-run variation (Bouthillier et al., 2021). We therefore distinguish the reference protocol's test-selected best epochs from validation-selected checkpoints.

3 Setup

Languages. We use the generators released by Ghosh et al. (2026). G_α is the hierarchical probabilistic context-free grammar called `cfg3b` with disjoint terminals, following Allen-Zhu & Li (2023); G_β is a second hierarchical grammar from the same code. Digit and Latin-letter alphabets give four languages. The original study uses two grammars and three alphabets, but its third alphabet (under-trained tokens) has no Qwen token list in the code. The code also does not identify the paper's grammars, so we cannot claim that our languages are a subset of theirs. For each language, we sample 10,000 strings with a fixed seed and split them equally into training and test pools. Datasets used across machines were verified byte-identical.

Discriminative test. The released test scores each string by its mean per-token negative log-likelihood (NLL). Positives are held-out in-language strings; negatives are out-of-language strings one edit from a test string. The generator produces ten negatives per positive. Evaluation uses the released default: the same 128 positives and 128 negatives for every run on a language. A one-feature

logistic regression is fitted on a stratified half of these strings. ROC-AUC is then computed on the remaining 64 positives and 64 negatives.

Models and methods. We use Qwen2.5 base models with 0.5B, 1.5B and 3B parameters (Qwen Team, 2024). ICL receives k=32 training strings in its prompt; each fine-tuning method trains on the same strings. Full FT follows the released recipe: 50 epochs, learning rate 5×10^{-5}, batch size 8, and a linear schedule with 5% warmup. LoRA adapts all attention and MLP projections at ranks $r \in \{8, 64, 256\}$, with α=$2r$ and no dropout. QLoRA uses a 4-bit NF4 base model, rank 64, and the rate selected for LoRA at that rank, without separate tuning. Our ICL baseline presents the examples once. The reference protocol tunes their repetition count per configuration. We tried two repetitions in one reproduction run: AUC fell from 0.937 to 0.744, after which we fixed one pass. Our baseline is therefore less optimized in this respect. Each main configuration uses three seeds to select training strings, matching the original study's three repetitions.

Reported epoch. The reference protocol reports the highest test AUC over 51 checkpoints: the initial model and each of 50 training epochs. It thus chooses the epoch on the same strings used to report performance. The median selected epoch is 10. We call this "best-epoch AUC" and also report "epoch-50 AUC" (Section 4.3).

Validation-selected rerun. We repeated the 3B comparison across four languages and three seeds for full FT, both adapters at the development rate, and ICL. The protocol was written before these runs. For each language, we drew fresh validation and test sets, each with 128 positives and 128 negatives. We excluded all previously scored strings and all positives whose perturbations had supplied a scored negative. Each new negative is a parser-verified one-edit perturbation of its own positive; no positive and its perturbations span the two sets. Training strings, recipes and padded input length match the original runs. We stored per-string scores for both sets at every epoch. Epochs were selected using only validation scores (epochs 1–50, earliest on ties), and the selections were committed before computing any test AUC. The primary result is test AUC at that epoch, using a classifier fitted on validation strings. Test curves and test-selected peaks were analysed afterwards. As pre-specified, primary tests are unadjusted exact Wilcoxon tests.

LoRA learning rates. We sweep rates for each model and rank on a development language, using one seed and requiring the optimum to lie inside the range (Appendix A). Development languages differ from that model's test languages: G_β digits for 0.5B and 1.5B, and a separate grammar sharing no test-language production rules for 3B. Initially, 3B adapters on G_α digits used 10^{-3}, selected at 1.5B. When we added the other three test languages, including G_β, we swept rates on the separate 3B development grammar. It selected 3×10^{-4}, and we reran the adapters. This rate scored worse than 10^{-3} on G_α digits, so we also tested 10^{-3} on the other three languages. That comparison followed inspection of test results and is reported only as a sensitivity analysis.

Compute and reporting. All paired comparisons use the same software and hardware setup: PyTorch 2.8 and two NVIDIA RTX PRO 6000 GPUs, with one GPU per run. We use PyTorch (Paszke et al., 2019), Transformers (Wolf et al., 2020) and PEFT (Mangrulkar et al., 2022) for training, and scikit-learn (Pedregosa et al., 2011) for classification and AUC. Of 252 stored runs, 208 involve training; 6 additional archived runs support the learning-rate sensitivity analysis. We report paired-t confidence intervals and unadjusted two-sided exact Wilcoxon signed-rank tests (Wilcoxon, 1945) over language-and-seed pairs. For the rerun's five primary comparisons, we also report a post-hoc Holm (Holm, 1979) correction (Appendix C). Pairs share grammar structures, alphabets and training-string seeds; neither analysis accounts for this dependence or establishes generalization across independently drawn grammars.

4 Results

4.1 Reproducing the reference baselines

We first reproduced the baselines on Qwen2.5-0.5B and G_α digits. With 1,024 training strings, full FT reached 0.994 best-epoch AUC at epoch 21, consistent with the original study's near-ceiling

Table 1: AUC at each run's best test epoch (mean ± std with 32 examples, on the two G_α languages that every model size covers). The std pools variation across languages and seeds. Epochs are selected on the reported test strings (Section 4.3). Adapters at 3B use the rate selected on the development grammar. "–" marks configurations not run.

Method	Qwen2.5-0.5B	Qwen2.5-1.5B	Qwen2.5-3B
ICL	0.837±0.098	0.871±0.100	0.923±0.034
LoRA r=8	0.977±0.012	0.983±0.006	–
LoRA r=64	0.974±0.012	0.979±0.013	0.954±0.012
LoRA r=256	0.984±0.010	–	–
QLoRA r=64	0.963±0.043	–	0.951±0.005
Full FT	0.969±0.014	0.965±0.015	0.956±0.012

result. ICL with 32 examples reached 0.937; repeating them twice lowered AUC to 0.744. This is a qualitative check because our languages and evaluation size differ (Section 3). These runs used a different software and hardware setup and are reported separately (Section 5).

4.2 Under the reference protocol, the result extends to LoRA and QLoRA

On the two languages tested at every size, all adaptation methods have higher mean best-epoch AUC than ICL (Table 1; Appendix F). At 0.5B, full FT beats ICL in 6/6 pairs ($p = 0.031$, the minimum with six pairs), and LoRA r=8 in 6/6. At 1.5B, the counts are 5/6 for full FT and 6/6 for LoRA r=8. As ICL improves with model size, full FT's margin falls from 0.132 at 0.5B to 0.094 at 1.5B and 0.033 at 3B.

Extending 3B to all four languages (Table 4, twelve pairs) gives a full-FT margin of $+0.085$ (95% CI $[+0.024, +0.146]$; 11/12 pairs; $p < 0.001$). Every adapter at every rate also exceeds ICL on average, from $+0.058$ to $+0.096$, though not uniformly: QLoRA at the development rate beats ICL in 7/12 pairs ($p = 0.077$), and on G_β with digits both adapters at that rate score below ICL (Appendix B). The margin is largest where ICL is weakest: 3B ICL reaches 0.674 on G_β with Latin letters against 0.952 on G_α with digits (Figure 2), so per-language results matter and a single-language study would have given a different impression. Within this protocol, replacing full fine-tuning with LoRA or QLoRA does not overturn the published finding.

4.3 The reported epoch decides the comparison

With 32 training strings, the difference between best-epoch and epoch-50 AUC (Table 2) is between 0.077 and 0.220 for every trained method and size. The per-epoch curves (Figure 1, and Figure 4 for the original strings) show the shape: AUC peaks within roughly ten epochs and then declines as the run continues to the fixed 50-epoch budget.

At 3B, reading the same runs at epoch 50 reverses the ordering. Across the four languages, full FT falls to 0.740, LoRA to 0.741 and QLoRA to 0.747, against 0.848 for ICL: full FT now trails ICL by 0.107 (full FT ahead in 3/12 pairs, $p = 0.012$), LoRA by 0.107 ($p = 0.007$) and QLoRA by 0.100 ($p = 0.021$).

This reversal combines two possible effects: performance declining with further training, and the advantage of selecting a checkpoint on the test set, which ICL cannot use. The contrast alone cannot separate them. We therefore repeat the comparison with validation-selected epochs and fresh test strings.

4.4 Selecting the epoch on held-out strings

On the held-out test strings, at each run's validation-selected epoch, full FT scores 0.942, LoRA 0.921 and QLoRA 0.919, against 0.873 for ICL (Table 3). Full FT exceeds ICL by $+0.070$ (95% CI $[+0.018, +0.122]$; 12/12 pairs; unadjusted $p < 0.001$). LoRA exceeds ICL by $+0.048$ ($[+0.003, +0.093]$; $p = 0.042$). QLoRA's estimated gain, $+0.047$, is close to LoRA's but more uncertain ($[-0.008, +0.101]$; $p = 0.129$); the contrast between a nominally significant and a non-significant test is not evidence that QLoRA gives no benefit. As a sensitivity check that we did not pre-specify, a

Table 2: AUC at the best test epoch versus at epoch 50 (means). The last column is their difference; it mixes decline after the peak with the advantage of choosing the epoch on the reported strings. ICL has no epoch axis, so its two columns coincide. The n=32 rows at 0.5B and 1.5B cover the two G_α languages; the n=1024 rows cover G_α with digits.

Setting	Method	Best-epoch AUC	Epoch-50 AUC	Best − epoch 50
0.5B, n=32	ICL	0.837	0.837	–
0.5B, n=32	Full FT	0.969	0.752	+0.217
0.5B, n=32	LoRA r=8	0.977	0.807	+0.170
0.5B, n=32	LoRA r=64	0.974	0.754	+0.220
0.5B, n=32	QLoRA r=64	0.963	0.802	+0.160
1.5B, n=32	ICL	0.871	0.871	–
1.5B, n=32	Full FT	0.965	0.748	+0.217
1.5B, n=32	LoRA r=8	0.983	0.905	+0.077
1.5B, n=32	LoRA r=64	0.979	0.767	+0.212
3B, n=32, 4 lang.	ICL	0.848	0.848	–
3B, n=32, 4 lang.	Full FT	0.933	0.740	+0.192
3B, n=32, 4 lang.	LoRA r=64	0.906	0.741	+0.165
3B, n=32, 4 lang.	QLoRA r=64	0.913	0.747	+0.165
0.5B, n=1024	Full FT	0.990	0.833	+0.157
0.5B, n=1024	LoRA r=8	0.993	0.988	+0.005
0.5B, n=1024	LoRA r=64	0.996	0.988	+0.008
1.5B, n=1024	Full FT	0.984	0.832	+0.153
1.5B, n=1024	LoRA r=8	0.982	0.842	+0.140
1.5B, n=1024	LoRA r=64	0.991	0.978	+0.013

Table 3: Validation-selected rerun at 3B (four languages, three seeds, 12 runs per method). Each trained run's epoch is chosen from validation scores; test columns use held-out strings no earlier run scored. "Test, best epoch" chooses the epoch on the test strings themselves (the reference protocol applied to the new strings). Adapters use rank 64 at the development learning rate.

Method	Val. AUC	Test AUC	Test, epoch 50	Test, best epoch	Median epoch
ICL	0.875	0.873	–	–	–
Full FT	0.954	0.942	0.756	0.944	5
LoRA r=64	0.932	0.921	0.759	0.921	5.5
QLoRA r=64	0.935	0.919	0.763	0.925	5.5

Holm correction over the five primary comparisons leaves full FT versus ICL at $p = 0.002$ and raises LoRA versus ICL to 0.107 (Appendix C); neither the correction nor the tests account for pairs sharing grammar structures. Validation selects early epochs (median 5 for full FT and 5.5 for both adapters), where every curve in Figure 1 peaks. Per language (Appendix D), ICL again trails most on G_β with Latin letters, and on G_β with digits it scores above both adapters.

Selecting epochs on the new test set instead raises mean test AUC by 0.002 for full FT, 0.001 for LoRA and 0.005 for QLoRA; one QLoRA run gains 0.038 (Appendix C). This compares two selection rules on a finite test set; it does not isolate population selection bias. In a separate, unplanned check, we evaluated reruns at the original runs' reported epochs. The 35 of 36 reruns whose training-string scores matched up to that epoch (Section 5) reach 0.938, 0.906 and 0.916 for full FT, LoRA and QLoRA on new test strings, versus 0.933, 0.907 and 0.913 originally. These values also differ in evaluation design: the original AUC uses 64 positives and 64 negatives after fitting on the other half, with differently sampled negatives. Their difference therefore cannot be attributed to selection bias alone.

At epoch 50, full FT, LoRA and QLoRA differ from ICL on the new strings by -0.117, -0.114 and -0.110, respectively ($p = 0.007$, 0.007 and 0.007). Thus, performance also declines on strings not used to choose epochs. Full FT's advantage, and the less certain LoRA advantage, remain with validation-based stopping but disappear at the reference training budget.

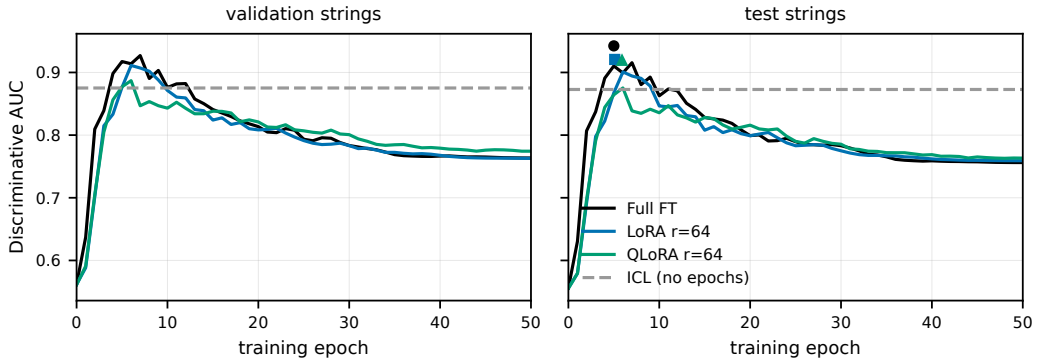


Figure 1: Mean AUC per epoch on the validation (left) and held-out test strings (right) in the 3B rerun. Markers give mean test AUC at each run's validation-selected epoch, drawn at the median selected epoch (LoRA and QLoRA shifted by 0.4 epoch so both are visible). The averaged curves peak lower than the markers because runs peak at different epochs.

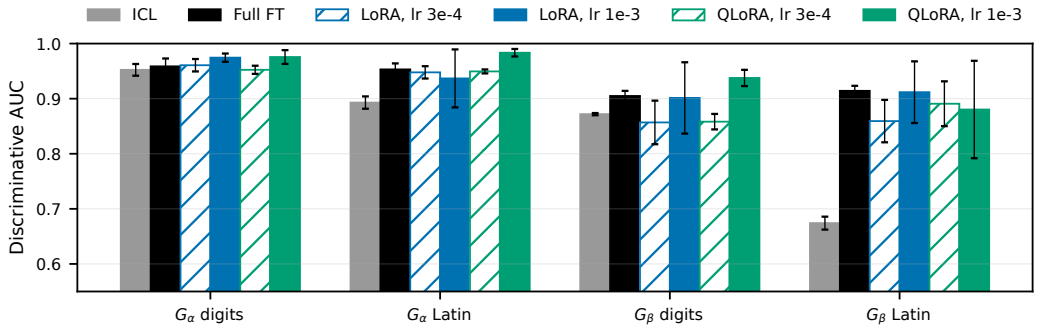


Figure 2: Qwen2.5-3B, 32 examples, AUC at the best test epoch per language (mean $\pm$ std over three seeds). Hatched bars use the LoRA rate selected on the development grammar; solid bars use 10^{-3}.

4.5 LoRA versus full fine-tuning is not a stable ranking

At 0.5B and 1.5B, LoRA $r{=}8$ has a slightly higher mean than full FT ($+0.008$ and $+0.018$), with no statistically detectable difference in this sample ($p = 0.156$ and 0.062).

At 3B, the comparison changes with the LoRA learning rate. With the rate selected on the development grammar ($3{\times}10^{-4}$), full FT leads LoRA by 0.027 AUC (LoRA ahead in 3/12 pairs, $p = 0.012$). The validation-selected rerun, which uses this rate, agrees on fresh strings: full FT leads LoRA by 0.022 (unadjusted $p = 0.034$) and QLoRA by 0.023 ($p = 0.027$), though neither survives a Holm correction ($p = 0.107$).

Under the reference protocol, raising the adapter rate to 10^{-3} reduces the full-FT lead over LoRA to 0.002, with no statistically detectable difference ($p = 0.733$), and QLoRA moves from below full FT to above it (-0.020 to $+0.011$, 11/12 pairs). The higher rate scores higher on the test languages on average for both adapters (LoRA $+0.025$, 11/12 pairs; QLoRA $+0.031$, 11/12), although on the development grammar the lower rate scored 0.984 against 0.929.

Because 10^{-3} was preferred after inspecting test results, we report it as a sensitivity analysis, not a recommended 3B recipe. Its effect is as large as the method differences: the selected rate can make full FT lead, remove a detectable difference, or put QLoRA ahead. We therefore make no general directional claim about LoRA versus full FT.

Rank. Higher rank does not consistently improve best-epoch AUC: $r{=}256$ has the highest mean at 0.5B, yet $r{=}8$ is at least as good as $r{=}64$ at both 0.5B and 1.5B (Table 1).

Table 4: Qwen2.5-3B across four languages, AUC at the best test epoch, paired on language and seed (12 pairs). Intervals are paired-t; p from unadjusted two-sided exact Wilcoxon tests.

Comparison (A vs B)	Pairs	ΔAUC	95% CI	A wins	p
Full FT vs ICL	12	$+0.085$	$[+0.024, +0.146]$	11/12	<0.001
LoRA (3e-4) vs ICL	12	$+0.058$	$[+0.005, +0.112]$	9/12	0.034
LoRA (1e-3) vs ICL	12	$+0.083$	$[+0.019, +0.148]$	10/12	0.009
QLoRA (3e-4) vs ICL	12	$+0.065$	$[+0.003, +0.126]$	7/12	0.077
QLoRA (1e-3) vs ICL	12	$+0.096$	$[+0.045, +0.148]$	12/12	<0.001
LoRA (3e-4) vs Full FT	12	-0.027	$[-0.051, -0.002]$	3/12	0.012
LoRA (1e-3) vs Full FT	12	-0.002	$[-0.031, +0.028]$	8/12	0.733
QLoRA (3e-4) vs Full FT	12	-0.020	$[-0.038, -0.002]$	4/12	0.064
QLoRA (1e-3) vs Full FT	12	$+0.011$	$[-0.019, +0.042]$	11/12	0.034
LoRA (1e-3) vs LoRA (3e-4)	12	$+0.025$	$[+0.001, +0.049]$	11/12	0.027
QLoRA (1e-3) vs QLoRA (3e-4)	12	$+0.031$	$[-0.008, +0.071]$	11/12	0.034

More training data. With 1,024 training strings, best-epoch AUC lies between 0.965 and 0.999 for all fine-tuning methods and the differences between them are not detectable with three seeds (Appendix E). The late decline differs by method here (Table 2): full FT loses 0.157 at 0.5B and 0.153 at 1.5B between its best epoch and epoch 50, whereas LoRA r=64 loses 0.008 and 0.013, so at this data size the methods differ in their epoch-50 means even where their best-epoch values are indistinguishable.

5 CONTROLS

Differences of one to three AUC points require careful controls. The following checks affected our interpretation.

Compute stack. With data, code and seed fixed, full FT scored 0.975 on a laptop GPU with PyTorch 2.6 and 0.987 on the main setup, a difference of 0.012; LoRA changed by 0.002. On the main setup, four repeats gave identical best-epoch full-FT AUC, supporting a setup effect for this configuration. We therefore pair runs only within one setup and report the reproduction separately. Training is nevertheless not exactly deterministic: AUC agrees through epoch 14 and then diverges, with epoch-50 values from 0.762 to 0.792. This variation is much smaller than the epoch-50 gap to ICL.

Replay agreement. The rerun was intended to change only evaluation strings. The released code pads all strings to the longest loaded string, so changing evaluation data could also change training inputs. We fixed the padded length and compared per-string NLL on the 32 training strings at every epoch. All 48 reruns match at epoch 0 within stored-score rounding; 46 match throughout, while two LoRA runs diverge from epochs 2 and 11. This agrees with the nondeterminism above. Score agreement supports similar training trajectories, but does not prove identical weights. The primary comparison uses only the new runs and does not depend on replay agreement.

Silent batch-size changes. The released code halves the batch size after a GPU memory error and continues without logging the change. We disabled this fallback and checked that optimizer step counts match batch size 8. All 208 training runs pass.

Evaluation errors. Including padding tokens when grouping scores by string reduced a converged model's AUC to chance. We now remove padding and reject scoring when the model's special tokens cannot be identified. A training crash also left a truncated score file that was initially marked complete without a score. Runs missing edit-distance scores now raise an error and are rerun.

Learning-rate selection. A sweep ending at its best rate can leave LoRA under-tuned. At 0.5B and r=8, extending our range beyond 10^{-3} (AUC 0.884) found 3×10^{-3} (0.977). We also replaced a candidate development grammar that shared every production rule with a test grammar with one sharing none.

6 LIMITATIONS

Validation-based selection was tested only at 3B, with 32 training strings and the development learning rate. Other sizes, the 1,024-string setting and the higher adapter rate use only the reference protocol; we did not measure the effect of test-based selection there. ICL remains single-pass: adaptation gets a validation-tuned stopping point, but ICL gets no comparable tuning of repetition count. Each validation and test set has only 128 positives and 128 negatives per language, making epoch selection noisy. Test scores were stored at every epoch; the holdout relies on committing selections before computing test AUC, rather than withholding the score files. Only full FT versus ICL remains significant at 0.05 after Holm correction. The rerun protocol preceded its runs but followed the original results; the old-checkpoint comparison was not pre-specified.

We test one model family up to 3B and four languages that may differ from those in the reference paper. Three seeds give six pairs at 0.5B and 1.5B (minimum two-sided Wilcoxon $p = 0.031$) and twelve at 3B. These share grammars and alphabets, so they are not independent grammar samples. Learning-rate sweeps use one seed; full FT retains the reference rate, and QLoRA inherits LoRA's rate, so full FT is the less tuned method. The rate-transfer finding comes from one model size and development grammar, and the software/hardware comparison covers only the repeated configurations.

7 CONCLUSION

Under the published best-epoch protocol, all tested adapters retain higher mean AUC than ICL at each evaluated model size. At 3B, choosing epochs on fresh validation strings preserves full FT's advantage on held-out test strings, gives weaker evidence for LoRA, and leaves QLoRA's similar estimated gain uncertain. Training duration matters: with 32 training strings, performance peaks early, and all three adaptation methods fall below ICL at epoch 50 at 3B, on both the original and new strings. Learning rate and software/hardware changes can move AUC by as much as the one-to-three-point differences between adapters and full FT. Comparisons should choose stopping points on held-out data, report learning-rate selection, use a consistent software and hardware setup, and verify training batch sizes.

REPRODUCIBILITY STATEMENT

The anonymized supplement is also available at `https://anonymous.4open.science/r/icl-vs-peft-formal-languages-E0FB`. It includes code, configs, tests, the rerun protocol and split manifest, and all run records (configuration, metrics, per-epoch AUC and software versions). It also contains per-string scores at every rerun epoch and compact exports of the original 3B scores, with raw-file checksums. Scripts generate all reported numbers, tables and figures; a fresh archive extraction reproduced them identically. A separate verifier recomputes the rerun's selected epochs and test metrics and checks file hashes. The archive reproduces the analysis; retraining requires GPUs and the released implementation at the pinned commit, as model checkpoints were not saved. Regenerated datasets are checked against stored checksums. Appendix G lists hardware, software and changes to the released implementation.

AI USE STATEMENT

AI assistants were used for drafting, editing, and LaTeX formatting support. All research design, experiments, analysis, and claims are the authors' own, and the authors reviewed and verified all content.

ETHICS STATEMENT

This work uses synthetic formal languages generated by released grammar code and publicly released pretrained models. It involves no human subjects, personal data or crowd work, and we see no direct route to harmful applications.

REFERENCES

Ekin Akyürek, Bailin Wang, Yoon Kim, and Jacob Andreas. In-context language learning: Architectures and algorithms. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 1, learning hierarchical language structures. *arXiv preprint arXiv:2305.13673*, 2023.

Amanda Bertsch, Maor Ivgi, Uri Alon, Jonathan Berant, Matthew R. Gormley, and Graham Neubig. In-context learning with long-context models: An in-depth exploration. In *First Workshop on Long-Context Foundation Models at ICML 2024*, 2024.

Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. On the ability and limitations of transformers to recognize formal languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020.

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John P. Cunningham. LoRA learns less and forgets less. *Transactions on Machine Learning Research*, 2024. arXiv:2405.09673.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Naz Sepah, Edward Raff, Kanika Madan, Vikram Voleti, et al. Accounting for variance in machine learning benchmarks. In *Proceedings of Machine Learning and Systems*, volume 3, 2021.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020.

Gavin C. Cawley and Nicola L. C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(70):2079–2107, 2010.

Grégoire Delétang, Anian Ruoss, Jordi Grau-Moya, Tim Genewein, Li Kevin Wenliang, Elliot Catt, Chris Cundy, Marcus Hutter, Shane Legg, Joel Veness, et al. Neural networks and the Chomsky hierarchy. In *International Conference on Learning Representations*, 2023.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. *arXiv preprint arXiv:2305.14314*, 2023.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*, 2020.

Ricardo Dominguez-Olmedo, Florian E. Dorner, and Moritz Hardt. Training on the test task confounds evaluation and emergence. In *International Conference on Learning Representations*, 2025.

Bishwamittra Ghosh, Soumi Das, Till Speicher, Qinyuan Wu, Mohammad Aflah Khan, Deepak Garg, Krishna P. Gummadi, and Evimaria Terzi. Fine-tuning vs. in-context learning in large language models: A formal language learning perspective. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026. arXiv:2604.23267.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*, 2019. arXiv:1902.00751.

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. *arXiv preprint arXiv:2106.09685*, 2021.

Julie Kallini, Isabel Papadimitriou, Richard Futrell, Kyle Mahowald, and Christopher Potts. Mission: Impossible language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024.

Sayash Kapoor and Arvind Narayanan. Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*, 4(9), 2023. arXiv:2207.07048.

Teven Le Scao and Alexander M. Rush. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2627–2636, 2021.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

Bingbin Liu, Jordan T. Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. In *International Conference on Learning Representations*, 2023.

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. *arXiv preprint arXiv:2205.05638*, 2022.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. PEFT: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`, 2022.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. On the stability of fine-tuning BERT: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations*, 2021.

Marius Mosbach, Tiago Pimentel, Shauli Ravfogel, Dietrich Klakow, and Yanai Elazar. Few-shot fine-tuning vs. in-context learning: A fair comparison and evaluation. In *Findings of the Association for Computational Linguistics: ACL 2023*, 2023. arXiv:2305.16938.

Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher D. Manning. Characterizing intrinsic compositionality in transformers with tree projections. In *International Conference on Learning Representations*, 2023.

Jane Pan, Tianyu Gao, Howard Chen, and Danqi Chen. What in-context learning "Learns" in-context: Disentangling task recognition and task learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 8298–8319, 2023.

Isabel Papadimitriou and Dan Jurafsky. Injecting structural hints: Using language models to study inductive biases in language learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, et al. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830, 2011.

Lutz Prechelt. Early stopping – but when? In *Neural Networks: Tricks of the Trade*, volume 1524 of *Lecture Notes in Computer Science*, pp. 55–69. Springer, 1998.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Krishna Prasad Varadarajan Srinivasan, Prasanth Gumpena, Madhusudhana Yattapu, and Vishal H. Brahmbhatt. Comparative analysis of different efficient fine tuning methods of large language models (LLMs) in low-resource setting. *arXiv preprint arXiv:2405.13181*, 2024.

Lena Strobl, William Merrill, Gail Weiss, David Chiang, and Dana Angluin. Transformers as recognizers of formal languages: A survey on expressivity. *arXiv preprint arXiv:2311.00208*, 2023.

Sudhir Varma and Richard Simon. Bias in error estimation when using cross-validation for model selection. *BMC Bioinformatics*, 7:91, 2006.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, et al. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*, 2023.

Jennifer C. White and Ryan Cotterell. Examining the inductive bias of neural language models with artificial languages. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 2021.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020.

Cheng Xu, Shuhao Guan, Derek Greene, and M-Tahar Kechadi. Benchmark data contamination of large language models: A survey. *arXiv preprint arXiv:2406.04244*, 2024.

Qingyu Yin, Xuzheng He, Chak Tou Leong, Fan Wang, Yanzhao Yan, Xiaoyu Shen, and Qiang Zhang. Deeper insights without updates: The power of in-context learning over fine-tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 4138–4151, 2024.

Biao Zhang, Zhongtao Liu, Colin Cherry, and Orhan Firat. When scaling meets LLM finetuning: The effect of data, model and finetuning method. In *International Conference on Learning Representations*, 2024.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. Revisiting few-sample BERT fine-tuning. In *International Conference on Learning Representations*, 2021.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pp. 12697–12706, 2021.

A LoRA LEARNING-RATE SWEEPS

Rates were selected on the development language for each model and rank, with 32 training strings and a single seed. The selected rate (bold) lies inside the swept range in every case.

Table 5: LoRA learning-rate sweeps on each model's development language (one seed, 32 training strings; AUC at the best test epoch of the development language). The selected rate is in bold.

Model	Rank	Dev language	lr: AUC (best in bold)
Qwen2.5-0.5B	8	G_β digits	1e-04: 0.782, 3e-04: 0.863, 1e-03: 0.884, **3e-03: 0.977**, 1e-02: 0.604
Qwen2.5-0.5B	64	G_β digits	1e-04: 0.879, 3e-04: 0.878, **1e-03: 0.917**, 2e-03: 0.620, 3e-03: 0.585
Qwen2.5-0.5B	256	G_β digits	1e-04: 0.877, **3e-04: 0.928**, 1e-03: 0.597
Qwen2.5-1.5B	8	G_β digits	1e-03: 0.820, **3e-03: 0.922**, 1e-02: 0.611
Qwen2.5-1.5B	64	G_β digits	3e-04: 0.804, **1e-03: 0.909**, 3e-03: 0.590
Qwen2.5-3B	64	generated HPCFG	3e-05: 0.900, 1e-04: 0.962, **3e-04: 0.984**, 1e-03: 0.929, 3e-03: 0.615

B PER-LANGUAGE RESULTS AT 3B

AUC at the best test epoch, mean over three seeds.

Table 6: Qwen2.5-3B under the reference protocol: AUC at the best test epoch per language, mean over three seeds, for both adapter learning rates.

			LoRA r=64		QLoRA r=64	
Language	ICL	Full FT	lr 3e-4	lr 1e-3	lr 3e-4	lr 1e-3
G_α, digits	0.952	0.959	0.961	0.974	0.952	0.976
G_α, Latin	0.893	0.953	0.948	0.937	0.949	0.983
G_β, digits	0.872	0.905	0.857	0.901	0.858	0.938
G_β, Latin	0.674	0.914	0.859	0.912	0.891	0.880

C RERUN: MULTIPLICITY AND TEST-BASED EPOCH CHOICE

Table 7 adds a Holm correction to the five primary comparisons; the pre-specified tests are unadjusted. Table 8 reports the difference between test AUC at the test-selected and validation-selected epochs: mean, minimum and maximum over 12 runs per method, and the number of positive differences. The difference is zero when the selected epochs have equal test AUC.

Table 7: Validation-selected rerun: the five primary comparisons (12 pairs each) with unadjusted and Holm-adjusted exact Wilcoxon p-values.

Comparison	Pairs	ΔAUC	p (unadjusted)	p (Holm)
Full FT vs ICL	12	$+0.070$	<0.001	0.002
LoRA vs ICL	12	$+0.048$	0.042	0.107
QLoRA vs ICL	12	$+0.047$	0.129	0.129
LoRA vs Full FT	12	-0.022	0.034	0.107
QLoRA vs Full FT	12	-0.023	0.027	0.107

Table 8: Validation-selected rerun: test-selected minus validation-selected test AUC, over 12 runs per method.

Method	Mean	Min	Max	Runs with a gain
Full FT	0.002	0.000	0.013	3/12
LoRA r=64	0.001	0.000	0.004	4/12
QLoRA r=64	0.005	0.000	0.038	4/12

D PER-LANGUAGE RESULTS OF THE VALIDATION-SELECTED RERUN

Held-out test AUC at each run's validation-selected epoch, mean over three seeds (3B, rank-64 adapters at the development learning rate).

Table 9: Validation-selected rerun: held-out test AUC per language at each run's validation-selected epoch, mean over three seeds.

Language	ICL	Full FT	LoRA r=64	QLoRA r=64
G_α, digits	0.961	0.974	0.975	0.966
G_α, Latin	0.919	0.969	0.965	0.969
G_β, digits	0.899	0.913	0.877	0.857
G_β, Latin	0.712	0.914	0.866	0.885

E PAIRED STATISTICS AT 0.5B AND 1.5B

Best-epoch AUC paired on language and seed. The n=32 rows cover the two G_α languages (six pairs; smallest attainable two-sided Wilcoxon p is 0.031); the n=1024 rows cover G_α with digits (three pairs, where the smallest attainable p is 0.25).

Table 10: Paired comparisons at 0.5B and 1.5B under the reference protocol (AUC at the best test epoch, paired on language and seed). Intervals are paired-t; p from unadjusted two-sided exact Wilcoxon tests.

Setting	Comparison	Pairs	ΔAUC	95% CI	A wins	p
0.5B, n=32	LoRA r=8 vs ICL	6	+0.140	[+0.043, +0.237]	6/6	0.031
0.5B, n=32	LoRA r=8 vs Full FT	6	+0.008	[−0.004, +0.020]	4/6	0.156
0.5B, n=32	LoRA r=64 vs ICL	6	+0.137	[+0.040, +0.233]	6/6	0.031
0.5B, n=32	LoRA r=64 vs Full FT	6	+0.005	[−0.007, +0.017]	4/6	0.438
0.5B, n=32	LoRA r=256 vs ICL	6	+0.147	[+0.043, +0.251]	6/6	0.031
0.5B, n=32	LoRA r=256 vs Full FT	6	+0.015	[+0.004, +0.026]	6/6	0.031
0.5B, n=32	QLoRA r=64 vs ICL	6	+0.125	[−0.005, +0.256]	5/6	0.094
0.5B, n=32	QLoRA r=64 vs Full FT	6	−0.006	[−0.060, +0.047]	5/6	0.438
0.5B, n=32	Full FT vs ICL	6	+0.132	[+0.032, +0.231]	6/6	0.031
1.5B, n=32	LoRA r=8 vs ICL	6	+0.111	[+0.005, +0.217]	6/6	0.031
1.5B, n=32	LoRA r=8 vs Full FT	6	+0.018	[+0.000, +0.036]	5/6	0.062
1.5B, n=32	LoRA r=64 vs ICL	6	+0.108	[−0.003, +0.219]	5/6	0.062
1.5B, n=32	LoRA r=64 vs Full FT	6	+0.014	[−0.007, +0.036]	4/6	0.312
1.5B, n=32	Full FT vs ICL	6	+0.094	[−0.007, +0.194]	5/6	0.094
0.5B, n=1024	LoRA r=8 vs Full FT	3	+0.003	[−0.004, +0.011]	2/3	0.500
0.5B, n=1024	LoRA r=64 vs Full FT	3	+0.006	[−0.004, +0.016]	3/3	0.250
1.5B, n=1024	LoRA r=8 vs Full FT	3	−0.003	[−0.028, +0.023]	2/3	1.000
1.5B, n=1024	LoRA r=64 vs Full FT	3	+0.006	[−0.020, +0.033]	2/3	0.500

F ADDITIONAL FIGURES

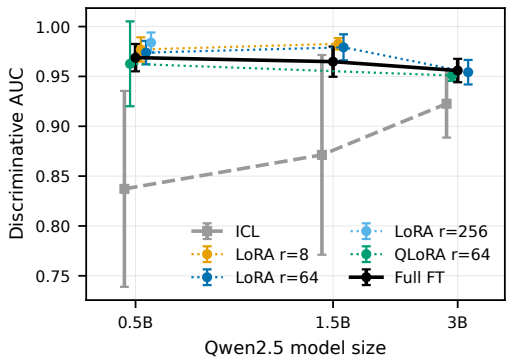


Figure 3: AUC at the best test epoch against model size on the two G_α languages (32 examples; error bars are standard deviations pooled over languages and seeds).

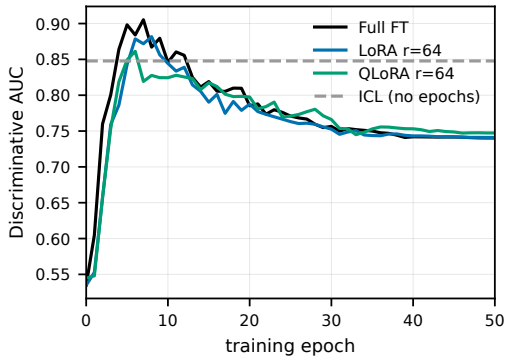


Figure 4: Mean AUC per epoch at 3B across four languages. All three training methods peak early, then decline below the ICL baseline. The corresponding 3B best-epoch entries in Table 2 read off each run's own peak, which is higher than this averaged curve.

G Deviations from the released implementation

- Evaluation on 128 strings per set (the released default) rather than the 1,024 described in the reference paper.
- Evaluation and ICL inference batch size 64 instead of 8, after verifying on both paths that per-epoch AUC is bit-identical at the two batch sizes.
- Automatic batch-size reduction on out-of-memory errors disabled (Section 5).
- Single-process launcher replacing the distributed launcher; one GPU per cell.
- ICL uses a single pass over the 32 examples rather than a per-configuration repetition count.
- Grammar-edit and per-rule evaluation sets are not generated, as the discriminative test does not use them.
- Main results: Python 3.12, PyTorch 2.8 with CUDA 12.8, PEFT 0.21, two NVIDIA RTX PRO 6000 GPUs. Reproduction runs: Python 3.10, PyTorch 2.6, a laptop GPU.