Fine-tuning vs. In-context Learning in Large Language Models: A Formal Language Learning Perspective

Anonymous ACL submission

Abstract

Large language models (LLMs) operate in two learning modes: *fine-tuning* (FT) and *in-context learning* (ICL). We ask which mode exhibits greater language proficiency, and whether their inductive biases in pattern recognition differ. We propose *three desiderata* for the comparison: **(D1)** a precise specification of the learning task, **(D2)** an equal resource allocation to FT and ICL, and **(D3)** a comparable evaluation metric to find the better mode. Several prior studies attempted to compare FT and ICL without satisfying all three desiderata, resulting in mixed and inconclusive results. To satisfy these desiderata, we propose a *formal language learning* task, where syntactic pattern recognition is the main focus. We also introduce a *discriminative test* for language proficiency, enabling direct comparison of FT and ICL.

Empirically, we find that (a) FT has greater language proficiency than ICL on in-distribution generalization, but both perform equally well on out-of-distribution generalization. (b) Their inductive bias, measured as the correlation of string generation, is usually similar, but similarity decreases with better language learning. (c) Unlike FT, ICL performance differs substantially across models of varying sizes and families, and becomes sensitive to tokens used in the languages. Thus, our controlled setup reveals subtle behavior of FT and ICL, which is difficult to capture in natural language datasets.

1 Introduction

Large language models (LLMs) have two principal learning modes: *fine-tuning* (FT) (Kaplan et al., 2020) and *in-context learning* (ICL) (Brown et al., 2020), to learn a new language, e.g., adapting to new domains. FT simulates a closed-book exam, where LLMs learn by updating model parameters. ICL simulates an open-book exam, where LLMs learn from in-context examples without any parameter update. Both learning modes are applied in various natural language processing (NLP) tasks, such as text summarization (Radford et al., 2019), question-answering (Yang et al., 2018), etc. Therefore, it is a natural question to ask which mode is **more language proficient** or more effective in learning a new language, i.e., which mode recognizes patterns in the language better. A related question is whether their *inductive bias* in learning is similar or different, i.e., whether they have similar (implicit or explicit) assumptions about recognizing patterns in the language (Mitchell, 1980). Answering such questions objectively has significant implications for the future deployment of LLMs in various tasks (Raiaan et al., 2024).

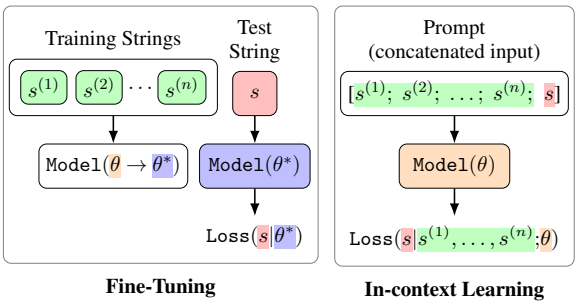


Figure 1: Fine-tuning (left) and in-context learning (right) are two learning modes of an LLM. On formal language learning, the task is to generate unseen strings from the language through syntactic pattern recognition (desideratum **D1**). Under an equal setting (**D2**), fine-tuning updates parameters ($\theta \rightarrow \theta^*$) based on training strings and generates a test string with a cross-entropy loss. In-context learning takes a concatenated input prompt, where training strings are the prefix to generate the test string. A comparable evaluation metric is thus needed, since both input prompts and parameters of the models are different between learning modes (**D3**).

Our Contributions. *What are the set of principles for comparing language proficiency of* FT *and* ICL*?* The question is relevant for any scientific study to compare two related processes (see motivation in Figure 1). Our key contribution is the

introduction of three-fold desiderata, as explained below, for comparing FT and ICL, and a controlled experimental framework realizing these desiderata. Several prior studies attempted to compare FT and ICL without satisfying all three desiderata, resulting in mixed and inconclusive results. Specifically, the closest to our work is Mosbach et al. (2023), who partially satisfy desideratum **D1** and **D2**, but fail to satisfy **D3** (details in Section 2).

D1. *Specification of the Learning Task: Syntax-focused Learning with Zero-prompting.* We compare FT and ICL on learning a *probabilistic formal language*, which is a distribution of strings accepted by a probabilistic grammar (Manning, 2003; Chater and Manning, 2006). The task is to generate new strings from the language by learning syntactic patterns from training strings (Section 3).

There are advantages of comparing FT and ICL on formal languages: (a) they contain syntax only, which is not ambiguous like semantics – the main focus of prior studies (Mosbach et al., 2023). The syntactic pattern recognition evaluates the native auto-regressive next token generation ability of LLMs. (b) Formal languages provide full control over the data distribution, and facilitate a precise differentiation between in-distribution and out-of-distribution languages via language distance, needed to evaluate generalization on in- and out-of-distribution tasks. (c) They are synthetic in nature and avoid data contamination (Xu et al., 2024). Satisfying all these aspects is hard for publicly available natural language datasets.

A practical challenge is *communicating the task* to the LLMs, since different LLMs may not understand the prompt instruction similarly (Wu et al., 2025). To avoid the subjectivity in designing prompt instructions, we consider a *zero-prompting* setup: the LLM only sees training strings, and we evaluate how well a new string is generated by the LLM, without any explicit instruction.

D2. *Allocation of Equal Resources.* A fair comparison requires allocating an equal resource to FT and ICL. We provide the same training and test data to both learning modes of the same LLM, consistent with Mosbach et al. (2023). In addition, FT and ICL have disjoint hyperparameters, such as batch-size, learning rates, and fine-tuning epochs in FT versus repetitions of examples, temperature in inference in ICL. We propose to compare the best performance of FT and ICL over respective hyperparameter settings, which is loosely performed by Mosbach et al. (2023); Yin et al. (2024).

D3. *Comparable Evaluation Metric.* There are two potential tests for language proficiency of LLMs: generative and discriminative tests – the latter is proposed by us. The generative test focuses on strings in the language and computes their probability of generation. We argue that *the generation probability is not comparable across learning modes and models*. The discriminative test, however, checks whether strings in the language are generated with higher probability than strings outside the language, i.e., whether a classification is possible between in-language and out-language strings based on their generation probability. We claim that *the classification score produced by the discriminative test is comparable between* FT *and* ICL, unlike the generative test (Section 4).

Experimental Results. We experiment with 18 open-source LLMs from 6 model families and multiple formal languages, and reach the following conclusions: (a) Different LLMs converge to optimal FT performance, while their ICL ability varies substantially. Model size becomes a factor for improved performance in ICL but not in FT. (b) On in-distribution generalization, where *training and test languages are the same*, FT dominates ICL except in some LLMs where ICL is close to FT. On out-of-distribution generalization where *training and test languages differ*, both learning modes perform equally, and generalizes to closer out-of-distribution languages only. (c) The inductive bias, measured by the correlation of output generation probability of FT and ICL, is often similar but not equal. Similarity in biases decreases when language learning of either mode improves with higher training data. (d) FT is robust across languages, measured by changing underlying grammar rules or tokens. However, ICL performance is largely impacted by the actual tokens used in the language.

Finally, we explicate the issues of testing LLMs with natural language datasets, such as data contamination and ill-defined in-distribution and out-of-distribution tasks, in Appendix D. Instead, we reiterate the need for synthetic formal languages for a rigorous scientific study on the capabilities of LLMs. Along the way, we position the paper as a stepping stone to motivate future research.

2 Motivation and Related Work

Here, we review related work and motivate why a comprehensive study comparing FT and ICL in a controlled setup is necessary.

Independent Studies on FT and ICL. Several works independently investigate FT (Kaplan et al., 2020; Zhang et al., 2024; Hu et al., 2024) and ICL in LLMs (Shen et al., 2023; Reddy, 2023; Pan et al., 2023; Chen et al., 2025), and relate learning performance with model size, training data, etc. Our work uses synthetic data and thus differs with most works performed on NLP datasets, where pre-training can affect FT and ICL performance.

Benchmarks. NLP datasets (Rajpurkar et al., 2016; Kwiatkowski et al., 2019) provide high-level descriptions of learning tasks, where in-distribution and out-of-distribution tasks are less well-defined. Even within in-distribution tasks, we argue that there is no formal guarantee of coherence between training and test examples – unlike a formal language, where all examples belong to the same language. Also, public datasets may result in data contamination providing an unfair advantage to some LLMs (Dominguez-Olmedo et al., 2024). We find both issues on MNLI dataset (Williams et al., 2018), as previously studied by Mosbach et al. (2023) on comparing FT and ICL, where our result contradicts their findings: On out-of-distribution tasks, FT and ICL perform equally well on formal languages, but FT is better than ICL on MNLI dataset (Appendix D). The contradiction highlights the need for a well-defined learning task (desideratum **D1**).

Comparison of FT and ICL. The comparison between FT and ICL has mixed conclusions, often due to violating desideratum **D2**. Several studies conclude that FT outperforms ICL (Brown et al., 2020; Mosbach et al., 2023; Liu et al., 2022b; Lester et al., 2021; Bhatia et al., 2023; Asai et al., 2024). However, the conclusions are based on using different model sizes, unequal number of examples, and high variance across runs. Other studies find ICL better than FT (Yin et al., 2024; Bertsch et al., 2024; Kaneko et al., 2025; Soudani et al., 2024; Awadalla et al., 2022), which usually execute suboptimal FT (e.g., 1 epoch), giving ICL an advantage. To our knowledge, no prior work considers a comparable evaluation metric aligned with desideratum **D3**, which we introduce in Section 4. Furthermore, our comparison of the inductive biases of FT and ICL represents a novel contribution.

Formal Languages in LLM Research. Owing to their greater controllability, formal languages have been widely used to investigate the NLP capabilities of LLMs (Jumelet and Zuidema, 2023), including their inductive biases in language learning (Papadimitriou and Jurafsky, 2023; White and Cotterell, 2021; Hopkins, 2022). Leveraging formal languages as a testbed, prior studies have compared the representational capacity of LLMs with various sequence-based models (Shi et al., 2022; Chi et al., 2023; Bhattamishra et al., 2020; Merrill, 2023; Strobl et al., 2023a; Hahn, 2020), and analyzed the classes of formal languages that LLMs can learn (Delétang et al., 2022; Hahn and Rofin, 2024; Cotterell et al., 2018; Mielke et al., 2019; Borenstein et al., 2024). Notably, LLMs have been shown to learn hierarchical and probabilistic formal languages that mirror the recursive structure of natural language (Allen-Zhu and Li, 2023; Murty et al., 2022; Liu et al., 2022a).

To our knowledge, no prior work has employed formal languages to compare the language proficiency and inductive biases of different LLM learning modes – this forms the central contribution of our work. Extend related work is in Appendix A.

3 Experimental Framework

In this section, we discuss our experimental framework by introducing formal languages and how we teach the language to the LLM in FT and ICL.

Formal Languages. Following Allen-Zhu and Li (2023), we use *probabilistic formal languages*, particularly the class generated by hierarchical probabilistic context free grammars (HPCFGs), as the objects of LLM learning (desideratum **D1**) - HPCFGs have the recursive structure of natural languages. Formally, a probabilistic formal language L is defined on a set of tokens or alphabet $\mathbf{T}$, and specifies a probability distribution P_L over strings, $P_L : \mathbf{T}^* \to [0, 1]$, where $\mathbf{T}^*$ is the set of all strings. A string s is *in-language* w.r.t. L if $P_L(s) > 0$, and *out-language* if $P_L(s) = 0$. $\mathbf{T}$ is a proper subset of the vocabulary $\mathbf{V}$ of all tokens of the LLM.

Construction of Out-language Strings. We quantify the *degree of incorrectness* of an out-language string as a *distance* from the language under investigation, which we utilize in the discriminative test in Section 4. We generate grammatically incorrect strings in two ways: (a) *Incorrect by edit*: We edit in-language strings to create out-language strings (through the addition, deletion and replacement of tokens at random positions), where edit distance is the number of edits made to the in-language string. (b) *Incorrect by randomization*: We sample random strings over the language's alphabet set, retaining only the distribution of string lengths from the language. On average, such random strings have a

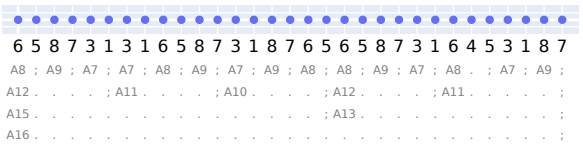


Figure 2: A string from language L_1, generated by a hierarchical grammar. The grammar contains non-terminal A's, alphabet (or terminals) $\mathbf{T} = \{1, 2, \ldots, 9\}$, and hierarchical production rules. For example, the rule '$A16 \rightarrow A15\ A13$' indicates that non-terminal $A16$ is expanded to $A15$ followed by $A13$, and so on, until reaching alphabet $\mathbf{T}$ (formal definition in Appendix B).

very high edit distance from the language. In both cases, we ensure non-membership of out-language strings via a grammar parser.

Languages. We consider six languages, denoted by $\{L_i\}_{i=1}^{6}$, based on a combination of two distinct HPCFGs, and three distinct alphabet sets (details in Section 5). For each language, we sample non-overlapping training ($n_{\text{train}} \in \{1, 2, 4, \ldots, 1024\}$) and test strings ($n_{\text{test}} = 1024$), following the distribution in a given language (desideratum **D2**). Figure 2 illustrates a representative string from L_1. Additional details on formal languages, respective grammars, the sampling process, and length distributions of generated strings are in Appendix B.

Teaching the Language to an LLM. To teach a language L to an LLM, we sample strings from L and feed them to the LLM via `FT` or `ICL`. `FT` is generally performed for a fixed number of epochs, denoted by $m = 50$, where in each epoch the LLM iterates over the strings while minimizing a loss function, such as cross-entropy loss. Formally, consider a dataset of n strings $D \triangleq \{s^{(j)}\}_{j=1}^{n}$ sampled from the language, $D \sim L$. For a given string s and its token s_i at the i-th position, let $P_M(s_i|s_{[1,i-1]})$ be the probability that the LLM M assigns to the token s_i given the prefix tokens $s_{[1,i-1]}$. The cross-entropy loss of the LLM on D is the per-token negative log probability at every token position of all strings in D, $\texttt{loss}_M(D) \triangleq -\frac{1}{n}\sum_{s\in D}\frac{1}{|s|}\sum_{i=1}^{|s|}\log P_M(s_i \mid s_{[1,i-1]})$.

In `ICL`, we provide the same strings in D as in-context examples. Specifically, `ICL` takes a set of ordered examples $\langle s^{(1)}, \cdots, s^{(n)}\rangle$ as a prefix for a test string s. The `ICL` examples are concatenated using separators, e.g., semicolons, leading to a prompt $s^{(1)}[\texttt{sep}]\cdots s^{(n)}[\texttt{sep}]s$. Similar to epochs in `FT`, we consider repeating examples in `ICL` a fixed number of times, $m \in \{1, 2, 4, 8, 16\}$. In both `FT` and `ICL`, we find and compare language proficiency at the optimal epoch or repetition m^*, satisfying desideratum **D2**.

We study 18 open-source LLMs from 6 model families: Mistral (Jiang et al., 2023), Llama (Touvron et al., 2023a,b; Dubey et al., 2024), Qwen (Yang et al., 2024), Gemma (Team et al., 2024a,b), Pythia (Biderman et al., 2023), and Opt (Zhang et al., 2022), ranging from 0.5B to 13B parameters. Each experiment is repeated three times by randomly sampling training strings with different seeds. Additional details on hyperparameters are provided in Appendix B.

4 The Test for Language Proficiency

We teach a formal language to an LLM via `FT` or `ICL`. A fundamental question that arises here is: *what does it mean for an LLM to be better or more proficient in a language?* Below, we discuss a generative test and a newly proposed *discriminative test* – the latter compares the proficiency of different modes directly and fairly (desideratum **D3**).

The Generative Test. While learning a language, the generative test evaluates how well unseen test strings from the language are generated by an LLM – higher the generation performance, better the language proficiency. This is a straightforward metric, and is adopted widely in the literature (Kallini et al., 2024; Jumelet and Zuidema, 2023; Bhattamishra et al., 2020; Wang, 2021; Akyürek et al., 2024).

Formally, consider two LLMs M and M' and a target language L. M and M' can be two learning modes of the same LLM as well. Using the generative test, M is more language proficient in L than M', if M generates strings in L with higher probability or lower loss than M', formally, $\texttt{loss}_M(L) < \texttt{loss}_{M'}(L)$.

Issues with the Generative Test. Two reasons hinder a direct comparison `FT` and `ICL` using a generative test. (i) Absolute probability (or loss, perplexity) is incomparable across LLMs: generation probability is impacted by pre-training setup, vocabulary, model parameters, random initialization, etc. As a result, different LLMs optimally trained on the same language do not guarantee language generation with the same probability. (ii) `FT` and `ICL` result in different input prompts and require comparing the same LLM with different parameters (Figure 1). The compounding factors make comparison impossible – if `FT` and `ICL` generate a string with different probability, we cannot decide whether the difference is due to different input

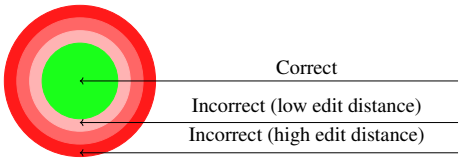
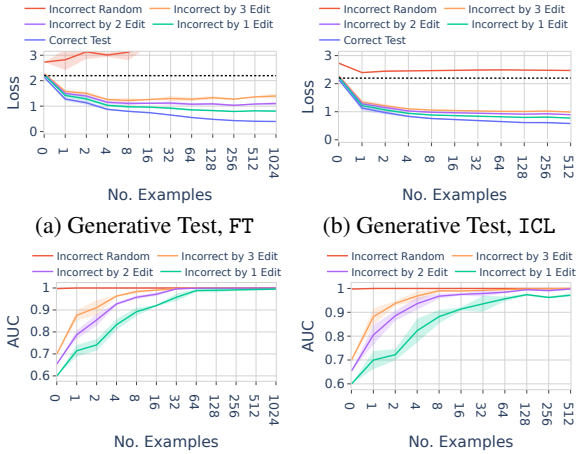


Figure 3: We visualize the set of all strings in a hierarchy, where the inner green circle denotes grammatically correct in-language strings, and the outer red circle denotes grammatically incorrect out-language strings. The generative test focuses on generation performance within the green circle, while the discriminative test focuses on comparative generation performance between green and red (specially at low edit distance) circles.



(a) Generative Test, FT (b) Generative Test, ICL

(c) Discriminative Test, FT (d) Discriminative Test, ICL

Figure 4: Language proficiency of Mistral-7B on language L_1, while varying the number of examples in both learning modes.

prompts or model parameters or both. To overcome these issues, we propose a discriminative test, which considers strings outside the language.

The Discriminative Test. The key intuition behind the discriminative test is: *if an LLM learned a language, it should generate strings in the language with higher probability than strings outside the language*. Thus, the discriminative test attempts to classify in-language and out-language strings based on their generation probability, where the success of classification is an implication of language proficiency. As shown in Figure 3, the test can be stricter by picking *close* out-language strings (according to some distance metric like edit distance) to in-language strings and checking if they can still be identified as out-language.

Formally, let $\mathsf{T}(L)$ denote out-language strings, constructed by editing or transforming strings in L and ensuring that they are not in L. Consider a binary (linear) classifier, where input is the generation probability of strings in $L \cup \mathsf{T}(L)$ by an LLM, and the classification task is to determine their membership. Let $\texttt{auc}_M(L, \mathsf{T}(L)) \in [0, 1]$ be the AUC (area under the receiver operating characteristic curve) of the classifier using model M; the higher the value the better. Thus, LLM M is more language proficient in L than M', if $\texttt{auc}_M(L, \mathsf{T}(L)) > \texttt{auc}_{M'}(L, \mathsf{T}(L))$.

Claim 1. *For a given language, the discriminative test is comparable between two learning modes of an LLM and across LLMs, unlike a generative test.*

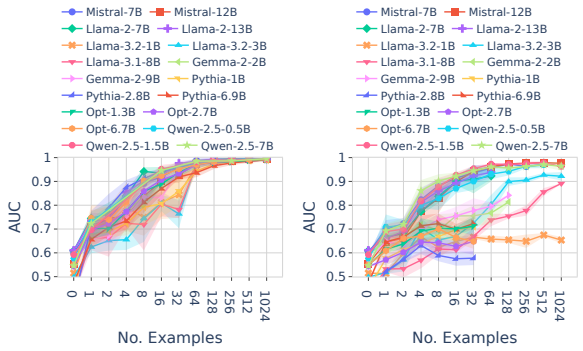
To support our claim, the discriminative test asks the same LLM or learning mode (i.e., equal parameters) to generate in-language and out-language strings, where all strings undergo the same prompt formatting. Thus, the derived classification score is comparable across learning modes and LLMs (we defer details in Appendix F).

Demonstration of Language-Proficiency Test. In Figure 4, we demonstrate the language proficiency of an LLM w.r.t. generative test (loss) in the top row and discriminative test (AUC) in the bottom row, for both FT and ICL.

Observation 1. Generative test alone is misleading. In Figure 4a and 4b, with increasing examples, the loss decreases (or probability increases) on in-language test strings, as well as strings that are close but outside the language. Therefore, *the generative test alone is insufficient in determining language proficiency on the target language*.

Observation 2. Discriminative test score is correlated with training size and distance of out-language strings. In Figures 4c and 4d, the AUC of the discriminator increases with examples, i.e., the LLM becomes increasingly proficient in the language, by not only generating strings from the language with lower loss, but also discriminating them from strings outside the language. Also, AUC is correlated with the degree of incorrectness of non-grammatical out-language strings; higher the incorrectness, higher the AUC. *Importantly, AUC between* FT *and* ICL *is comparable under an equal setting of examples and degree of incorrectness.*

In the next section, we apply the discriminative test to compare FT and ICL, and report the AUC of discriminating in-language test strings from out-language strings at edit distance 1, resulting in the most difficult discriminative test.



(a) FT (b) ICL

Figure 5: FT and ICL across different LLMs while learning language L_1. Different LLMs demonstrate similar FT performance, but their ICL ability varies.

ICL ability (AUC range)	Model
Good (≥ 0.75)	Qwen-2.5-7B, Mistral-7B, Qwen-2.5-1.5B, Llama-2-13B, Qwen-2.5-0.5B, Llama-2-7B, Mistral-12B
Moderate (≥ 0.6)	Gemma-2-2B, Gemma-2-9B, Pythia-6.9B, Opt-1.3B, Opt-6.7B, Pythia-1B, Llama-3.2-3B, Opt-2.7B, Llama-3.2-1B
Poor (< 0.6)	Llama-3.1-8B, Pythia-2.8B

Table 1: ICL ability of LLMs on language L_1 with up to 32 examples, based on discrimination AUC. In each group, LLMs are sorted in descending ICL ability.

5 Fine-tuning vs. In-context Learning

We study the language proficiency of FT and ICL in LLMs on learning syntactic patterns from formal languages. Specifically, we aim to answer the following research questions to analyze the subtle differences between FT and ICL.[1]

RQ1. When evaluating FT and ICL independently on a given language, how language proficient are different LLMs of varying sizes and families?
RQ2. Which learning mode is more language proficient when evaluated jointly on in-distribution and out-of-distribution generalization?
RQ3. Do FT and ICL result in similar inductive bias while learning a formal language?
RQ4. How robust FT and ICL performance are to changes in languages?

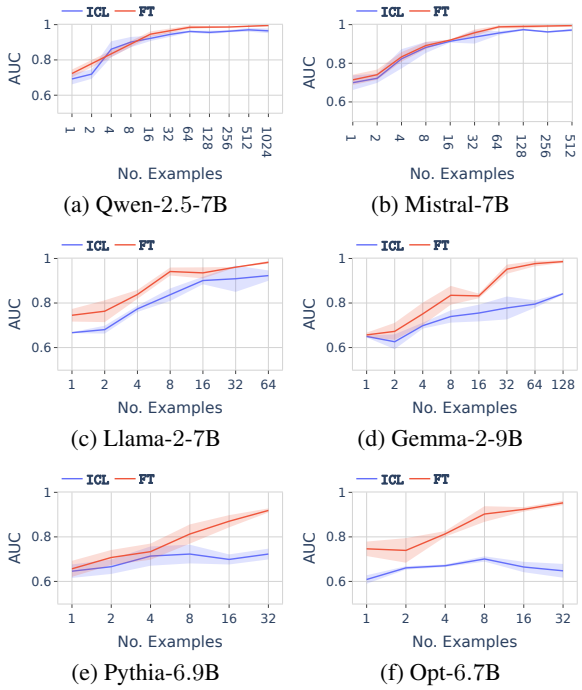
Answer to RQ1: Different LLMs attain a similar and near optimal language proficiency under FT, but their ICL ability varies substantially. In Figure 5, we report the AUC of FT and ICL across LLMs and example sizes while learning language L_1. In both modes, AUC increases with examples, indicating better learning.

Fine-tuning. During FT, all models across families and parameter sizes eventually converge to the optimal AUC (> 0.99) after sufficient training examples, such as 512. Across example sizes $\{1, 16, 64, 256, 1024\}$, the average AUC of FT is almost similar across models: Llama-2 (0.93) > Qwen (0.92) > Mistral (0.91) > Opt (0.91) > Gemma (0.90) > Pythia (0.90) > Llama-3 (0.88), where the respective AUC is inside the parenthesis. Only in few families (e.g., Opt), the largest model achieves the highest AUC. In addition, a more proficient family often achieves its best language proficiency in an earlier epoch. For example, the median epoch is 7.5 for Llama-2, 12 for Opt, and 37 for Llama-3. *Therefore, different LLMs, regardless of sizes and families, may achieve similar language proficiency under FT on a tailored task like formal language learning.*

In-context Learning. In ICL, the AUC varies substantially within a model family and across model families. First we observe that different LLMs have variable context length, restricting them to process different number of ICL examples. To compare all models fairly, we limit our analysis to 32 ICL examples, which all models can fit in their context. We find the following order of ICL ability of LLM families: Qwen (0.78) > Mistral (0.78) > Llama-2 (0.77) > Gemma (0.69) > Opt (0.64) > Pythia (0.61) > Llama-3 (0.59). Due to variable performance, we propose a ranking of ICL ability of LLMs in Table 1. Importantly, within a family, ICL ability does not always correlate with model size (Mistral 7B > Mistral-12B) or model versions (Llama-2-7B > Llama-3.1-8B). Only in some families such as Qwen, Pythia, and Llama-2, the largest model is better in ICL. Unlike FT, repeating ICL examples more than once worsens ICL performance: repeating examples takes up context space, and it is thus better to sample examples from the language distribution without repetition. *To conclude, ICL ability is more variable across LLMs, compared to FT.*

Answer to RQ2: On in-distribution language generalization, FT dominates ICL in most LLMs; only in a subset of LLMs, ICL is close to FT. On out-of-distribution generalization, both FT and ICL perform similarly, and generalize well to nearby languages only. In Figure 6, we compare

[1] Additional results including evaluation on NLP datasets, utilization test of full ICL context by LLMs, and detailed implications of research questions are in the Appendix C.

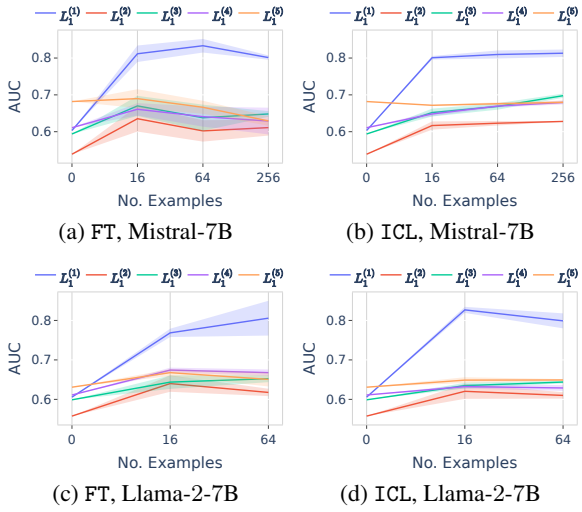


(a) Qwen-2.5-7B (b) Mistral-7B

(c) Llama-2-7B (d) Gemma-2-9B

(e) Pythia-6.9B (f) Opt-6.7B

Figure 6: In-distribution generalization of FT vs. ICL on L_1 in comparable $\approx$ 7B parameter size LLMs. FT usually dominates ICL , except in Qwen-2.5-7B, Mistral-7B and Llama-2-7B, where ICL is close to FT.

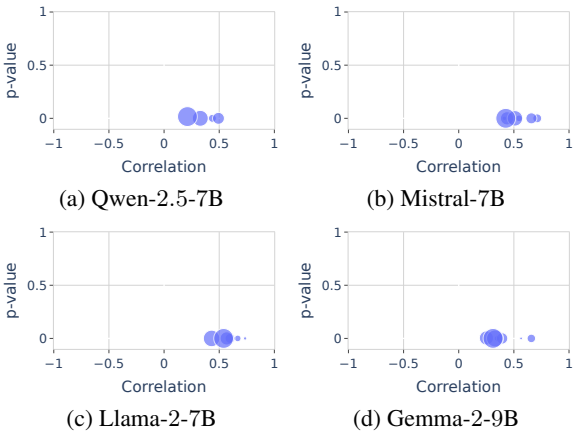


(a) FT, Mistral-7B (b) ICL, Mistral-7B

(c) FT, Llama-2-7B (d) ICL, Llama-2-7B

Figure 7: Out-of-distribution generalization of FT and ICL on increasing distant languages, where both modes perform almost equally. L_1 as the base learned language, and generalization is performed on $L_1^{(\ell)}$, by changing ℓ rules in the grammar of L_1. $L_1^{(\ell)}$ contains all changed rules in $L_1^{(\ell-1)}$. Therefore, $\texttt{dist}(L_1, L_1^{(\ell-1)}) \leq \texttt{dist}(L_1, L_1^{(\ell)})$, where $2 \leq \ell \leq 5$ (see Eq. (1)).

FT and ICL of an LLM on in-distribution language generalization. In most LLMs, FT dominates ICL, and the performance difference is profound when considering more examples. However, in a subset



(a) Qwen-2.5-7B (b) Mistral-7B

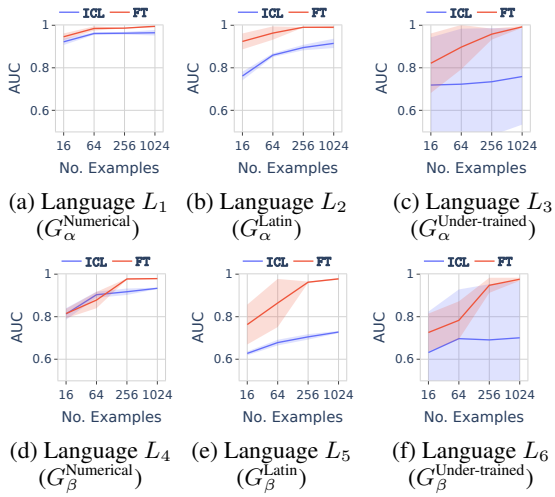
(c) Llama-2-7B (d) Gemma-2-9B

Figure 8: Inductive bias of ICL and FT, computed as the Pearson correlation of generation loss of FT and ICL on identical test strings. Correlation, despite being positive, tends to decrease with higher examples (larger markers).

of models, such as Mistral-7B, Qwen-2.5-7B, and Llama-2-7B, ICL is close to FT – these are usually models with good ICL ability in Table 1. *Therefore,* FT *is more language proficient than* ICL *on in-distribution language generalization.*

In Figure 7, we compare FT and ICL of an LLM on out-of-distribution language generalization, where the LLM first learns the language L_1 in FT or ICL, and then we evaluate on five other languages $\{L_1^{(1)}, \ldots, L_1^{(5)}\}$ of increasing distances from L_1. We emphasize that formal languages offer a systematic distance computation between two languages (i.e., out-of-distribution tasks), unlike any natural language (Appendix D). Surprisingly, both modes perform similarly on out-of-distribution languages, and only perform well on the near distant language $L_1^{(1)}$. *Therefore, the superiority of* FT *over* ICL *on in-distribution generalization is not transferable to out-of-distribution generalization.*

Answer to RQ3: The inductive bias of FT and ICL is often similar, but not equal. Similarity decreases with training examples. To compare inductive bias of FT and ICL in recognizing syntactic patterns in formal languages, we do not focus on how each mode operates internally, but focus on their output correlation in generating the same set of strings. Thus, if correlation is high, inductive bias is similar. In Figure 8, we report the Pearson correlation of generation loss of FT and ICL, where correlation is often positive (< 0.8). More importantly, correlation generally decreases with more training examples, which allows each mode to learn the language better. *To summarize, the*



(a) Language L_1 ($G_\alpha^{\text{Numerical}}$) (b) Language L_2 (G_α^{Latin}) (c) Language L_3 ($G_\alpha^{\text{Under-trained}}$)

(d) Language L_4 ($G_\beta^{\text{Numerical}}$) (e) Language L_5 (G_β^{Latin}) (f) Language L_6 ($G_\beta^{\text{Under-trained}}$)

Figure 9: Robustness of language proficiency of FT and ICL in Qwen-2.5-7B while varying languages in two ways: changing the grammar rules (rows) and changing the alphabet tokens (columns). The underlying grammar for a language is inside the parenthesis. Compared to FT, ICL is sensitive to the tokens used in the language.

inductive bias of FT *and* ICL *is often similar, but similarity decreases when each mode learns the language better with more training examples.*

Answer to RQ4: FT is more robust to changes in languages than ICL. In Figure 9, we study the robustness of FT and ICL on different languages, by changing the underlying grammar rules: G_α and G_β, and the alphabet: numerical, Latin, and under-trained tokens. FT is better than ICL in all languages, consistent with results in in-distribution generalization (Figure 6). Importantly, tokens used in the language introduce variability in results despite having the same grammar rules (across columns), and the variability is more pronounced in ICL than FT. For example, when considering under-trained tokens i.e., tokens barely seen in pre-training (Land and Bartolo, 2024), ICL performance is the worst. *Therefore, for a robust performance,* FT *is preferred over* ICL.

In Appendix D, we extend FT vs. ICL comparison **beyond formal languages to natural languages**, as studied by Mosbach et al. (2023), where our result complies with in-distribution generalization, but not with out-of-distribution generalization. To this end, we identify issues such as data contamination and a poor differentiation between in-distribution and out-of-distribution tasks, factors that we carefully avoid in formal languages.

Key Implications. We reach following implications from our study (details in Appendix G):

- FT is better than ICL if the test and training languages are the same. ICL is however preferred on out-of-distribution languages, where general language understanding of the model is retained as parameters are not updated.
- FT and ICL are likely to recognize patterns similarly as long as the target language is learned less or few examples are given. With more examples, the inductive bias of FT and ICL usually differs.
- Within a family, higher model size can lead to better ICL, but not necessarily better FT. Among model families, Qwen, Mistral, Llama-2, etc. are better in both modes.
- Unlike FT, ICL is token-sensitive. Since models in the same family may have different pre-training recipes impacting the same tokens differently, we may expect variability in ICL within a family (Mistral-7B $>$ Mistral-12B, Llama-2-7B $>$ Llama-3.1-8B). But, if the target language contains less common tokens, FT is preferred.
- The discriminative test is applicable beyond formal languages, where grammatical errors w.r.t. a learned language are identified. If LLM M generates a language L with lower probability than LLM M', but the AUC of discrimination by M is higher than M', then M is more proficient in L, challenging conventional wisdom. Here, M' is less proficient as it generates both L and nearby out-of-language $\mathsf{T}(L)$ with similar probability.

6 Conclusion

We study language proficiency and inductive bias of FT and ICL – two principal learning modes in LLMs. We propose three desiderata for a fair comparison, which prior studies overlook. Subsequently, we consider the task of learning formal languages, and propose a comparable discriminative test for language proficiency.

Our controlled experimental framework leads to important findings: FT is better than ICL on in-distribution generalization, but both perform equally on out-of-distribution generalization. Their inductive bias is similar, but similarity decreases as both modes learn the language better with more training examples. Unlike FT, ICL performance is more sensitive to tokens in the language, even with the same underlying grammar rules. Many of our results on synthetic formal languages are hard to obtain in ill-constructed natural language datasets. Therefore, we emphasize the need for a formal language benchmark for studying LLMs.

Limitations

Our objective in the paper is to systematically compare the language proficiency and inductive bias of fine-tuning (FT) and in-context learning (ICL) – two principle learning modes in which an LLM adapts to a new domain. To achieve this objective, we choose the task of learning syntactic patterns in a formal language, which has several advantages. Furthermore, we propose a discriminative test for evaluating the language proficiency of LLMs. While our research approach is to be careful about all choices we make, we also highlight following limitations of the current work that demand further study.

Formal languages are limited to context-free languages. The paper initially focuses on context-free languages, which mimic the recursive structure of natural languages. However, we highlight the need for further study to confirm our findings in other classes of formal languages, such as regular and context-sensitive languages.

The study is limited to $\leq$ 13B parameter size models. Our goal is to apply FT and ICL of LLMs of an equal parameter size. Since FT is more compute intensive, we limit our experiments to a maximum of 13B parameter size models. Moreover, we do not perform an extensive hyperparameter search in FT, such as batch size, learning rate, etc. Rather, we find the optimal epoch for each FT run and compare it with the optimal repetition of examples in ICL. Furthermore, we restrict experiments to full fine-tuning, while acknowledging that several parameter-efficient fine-tuning methods exist and may result in a different conclusion.

Larger models ($>$ 13B) may have better in-context learning performance. Does it invalidate our results? Since ICL is inferior to FT on in-distribution performance, a natural question is whether considering larger models would further improve ICL. While we expect ICL to improve with size, so does FT, keeping our initial findings consistent.

We find variable ICL performance across LLMs. How can we explain this? To explain the variability of ICL performance, we conduct two studies: (a) determine if existing LLMs utilize their full ICL context (see Appendix E), and (b) identify the sensitivity of ICL on tokens used in our experiments (see **RQ4** in Section 5). The former result shows that a subset of LLMs reach their ICL limit and can not further improve from additional examples, while the rest cannot reach their ICL limit. The latter result shows that the tokens used for experimentation have a large impact on ICL performance, and the same set of tokens are possible to be pre-trained with different extent across LLMs. While these results are important, we highlight the need to study model-specific ICL performance as a future work to find a more informed explanation.

Ethics Statement

This research investigates how different learning modes of large language models (LLMs), namely fine-tuning (FT) and in-context learning (ICL), compare in their language proficiency and inductive bias. Our experiments involve controlled and synthetically generated formal languages with no human subject involvement or use of private data. As such, the research study does not present immediate ethical risks from the data collection or model training processes. The scientific results of this study have profound implications in choosing the right mode of learning for LLMs in real-world applications.

References

Ekin Akyürek, Bailin Wang, Yoon Kim, and Jacob Andreas. 2024. In-context language learning: Arhitectures and algorithms. *arXiv preprint arXiv:2401.12973*.

Zeyuan Allen-Zhu and Yuanzhi Li. 2023. Physics of language models: Part 1, learning hierarchical language structures. *ArXiv e-prints, abs/2305.13673, May*.

Akari Asai, Sneha Kudugunta, Xinyan Yu, Terra Blevins, Hila Gonen, Machel Reid, Yulia Tsvetkov, Sebastian Ruder, and Hannaneh Hajishirzi. 2024. BUFFET: Benchmarking large language models for few-shot cross-lingual transfer. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1771–1800, Mexico City, Mexico. Association for Computational Linguistics.

Anas Awadalla, Mitchell Wortsman, Gabriel Ilharco, Sewon Min, Ian Magnusson, Hannaneh Hajishirzi, and Ludwig Schmidt. 2022. Exploring the landscape of distributional robustness for question answering models. *arXiv preprint arXiv:2210.12517*.

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

Kush Bhatia, Avanika Narayan, Christopher M De Sa, and Christopher Ré. 2023. Tart: A plug-and-play transformer module for task-agnostic reasoning. *Advances in Neural Information Processing Systems*, 36:9751–9788.

Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. 2020. On the ability and limitations of transformers to recognize formal languages. *arXiv preprint arXiv:2009.11264*.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, and 1 others. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Nadav Borenstein, Anej Svete, Robin Chan, Josef Valvoda, Franz Nowak, Isabelle Augenstein, Eleanor Chodroff, and Ryan Cotterell. 2024. What languages are easy to language-model? a perspective from learning probabilistic regular languages. *arXiv preprint arXiv:2406.04289*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Nick Chater and Christopher D Manning. 2006. Probabilistic models of language processing and acquisition. *Trends in cognitive sciences*, 10(7):335–344.

Wentong Chen, Yankai Lin, ZhenHao Zhou, HongYun Huang, YanTao Jia, Zhao Cao, and Ji-Rong Wen. 2025. ICLEval: Evaluating in-context learning ability of large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 10398–10422, Abu Dhabi, UAE. Association for Computational Linguistics.

Ta-Chung Chi, Ting-Han Fan, Alexander I Rudnicky, and Peter J Ramadge. 2023. Transformer working memory enables regular language reasoning and natural language length extrapolation. *arXiv preprint arXiv:2305.03796*.

Noam Chomsky. 1956. Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124.

Michael Collins. 2013. Probabilistic context-free grammars (pcfgs). *Lecture Notes*.

Ryan Cotterell, Sabrina J Mielke, Jason Eisner, and Brian Roark. 2018. Are all languages equally hard to language-model? *arXiv preprint arXiv:1806.03743*.

Colin de la Higuera, James Scicluna, and Mark-Jan Nederhof. 2014. On the computation of distances for probabilistic context-free grammars. *arXiv preprint arXiv:1407.1513*.

Grégoire Delétang, Anian Ruoss, Jordi Grau-Moya, Tim Genewein, Li Kevin Wenliang, Elliot Catt, Chris Cundy, Marcus Hutter, Shane Legg, Joel Veness, and 1 others. 2022. Neural networks and the chomsky hierarchy. *arXiv preprint arXiv:2207.02098*.

Ricardo Dominguez-Olmedo, Florian E Dorner, and Moritz Hardt. 2024. Training on the test task confounds evaluation and emergence. *arXiv preprint arXiv:2407.07890*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Michael Hahn. 2020. Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics*, 8:156–171.

Michael Hahn and Mark Rofin. 2024. Why are sensitive functions hard for transformers? *arXiv preprint arXiv:2402.09963*.

Mark Hopkins. 2022. Towards more natural artificial languages. In *Proceedings of the 26th Conference on Computational Natural Language Learning (CoNLL)*, pages 85–94.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Shengding Hu, Yuge Tu, Xu Han, Ganqu Cui, Chaoqun He, Weilin Zhao, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Xinrong Zhang, Zhen Leng Thai, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, and 5 others. 2024. MiniCPM: Unveiling the potential of small language models with scalable training strategies. In *First Conference on Language Modeling*.

Thomas F Icard. 2020. Calibrating generative models: The probabilistic chomsky–schützenberger hierarchy. *Journal of Mathematical Psychology*, 95:102308.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Jaap Jumelet and Willem Zuidema. 2023. Transparency at the source: Evaluating and interpreting language models with access to the true distribution. *arXiv preprint arXiv:2310.14840*.

Julie Kallini, Isabel Papadimitriou, Richard Futrell, Kyle Mahowald, and Christopher Potts. 2024. Mission: Impossible language models. *arXiv preprint arXiv:2401.06416*.

Masahiro Kaneko, Danushka Bollegala, and Timothy Baldwin. 2025. The gaps between fine tuning and in-context learning in bias evaluation and debiasing. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2758–2764.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*.

Sander Land and Max Bartolo. 2024. Fishing for magikarp: Automatically detecting under-trained tokens in large language models. *arXiv preprint arXiv:2405.05417*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Ziqian Lin and Kangwook Lee. 2024. Dual operating modes of in-context learning. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*.

Bingbin Liu, Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. 2022a. Transformers learn shortcuts to automata. *arXiv preprint arXiv:2210.10749*.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022b. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965.

Christopher D Manning. 2003. Probabilistic syntax. *Probabilistic linguistics*, 289341.

William Merrill. 2023. Formal languages and the nlp black box. In *International Conference on Developments in Language Theory*, pages 1–8. Springer.

Sabrina J Mielke, Ryan Cotterell, Kyle Gorman, Brian Roark, and Jason Eisner. 2019. What kind of language is hard to language-model? *arXiv preprint arXiv:1906.04726*.

Tom M Mitchell. 1980. The need for biases in learning generalizations.

Marius Mosbach, Tiago Pimentel, Shauli Ravfogel, Dietrich Klakow, and Yanai Elazar. 2023. Few-shot fine-tuning vs. in-context learning: A fair comparison and evaluation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12284–12314, Toronto, Canada. Association for Computational Linguistics.

Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher D Manning. 2022. Characterizing intrinsic compositionality in transformers with tree projections. *arXiv preprint arXiv:2211.01288*.

Michael Oliver and Guan Wang. 2024. Crafting efficient fine-tuning strategies for large language models. *arXiv preprint arXiv:2407.13906*.

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

Isabel Papadimitriou and Dan Jurafsky. 2023. Injecting structural hints: Using language models to study inductive biases in language learning. *arXiv preprint arXiv:2304.13060*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Mohaimenul Azam Khan Raiaan, Md Saddam Hossain Mukta, Kaniz Fatema, Nur Mohammad Fahad, Sadman Sakib, Most Marufatul Jannat Mim, Jubaer Ahmad, Mohammed Eunus Ali, and Sami Azam. 2024. A review on large language models: Architectures, applications, taxonomies, open issues and challenges. *IEEE access*, 12:26839–26874.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Shauli Ravfogel, Yoav Goldberg, and Tal Linzen. 2019. Studying the inductive biases of rnns with synthetic variations of natural languages. *arXiv preprint arXiv:1903.06400*.

Gautam Reddy. 2023. The mechanistic basis of data dependence and abrupt learning in an in-context classification task. In *The Twelfth International Conference on Learning Representations*.

Lingfeng Shen, Aayush Mishra, and Daniel Khashabi. 2023. Do pretrained transformers really learn in-context by gradient descent? *arXiv preprint arXiv:2310.08540*.

Hui Shi, Sicun Gao, Yuandong Tian, Xinyun Chen, and Jishen Zhao. 2022. Learning bounded context-free-grammar via lstm and the transformer: difference and

the explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 8267–8276.

Heydar Soudani, Evangelos Kanoulas, and Faegheh Hasibi. 2024. Fine tuning vs. retrieval augmented generation for less popular knowledge. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 12–22.

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

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

Lena Strobl, William Merrill, Gail Weiss, David Chiang, and Dana Angluin. 2023b. Transformers as Recognizers of Formal Languages: A Survey on Expressivity. *arXiv preprint*. ArXiv:2311.00208 [cs].

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, and 89 others. 2024a. Gemma: Open models based on gemini research and technology. *Preprint*, arXiv:2403.08295.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, and 179 others. 2024b. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Shunjie Wang. 2021. *Evaluating transformer's ability to learn mildly context-sensitive languages*. University of Washington.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, and 1 others. 2023. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*.

Jennifer C White and Ryan Cotterell. 2021. Examining the inductive bias of neural language models with artificial languages. *arXiv preprint arXiv:2106.01044*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Qinyuan Wu, Mohammad Aflah Khan, Soumi Das, Vedant Nanda, Bishwamittra Ghosh, Camila Kolling, Till Speicher, Laurent Bindschaedler, Krishna Gummadi, and Evimaria Terzi. 2025. Towards reliable latent knowledge estimation in llms: Zero-prompt many-shot based factual knowledge extraction. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, pages 754–763.

Cheng Xu, Shuhao Guan, Derek Greene, M Kechadi, and 1 others. 2024. Benchmark data contamination of large language models: A survey. *arXiv preprint arXiv:2406.04244*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

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

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

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, and 1 others. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

A Extended Related Work

A.1 Learning Modes in LLM: Fine-tuning and In-context Learning

We discuss existing studies that independently investigate fine-tuning and in-context learning, followed by their direct comparison.

Fine-tuning: A number of works (Kaplan et al., 2020; Zhang et al., 2024; Hu et al., 2024; Srinivasan et al., 2024; Oliver and Wang, 2024; Hu et al., 2022) study the effects of fine-tuning or its variants with respect to model scaling, where larger fine-tuned models with less amount of data are better in performance than smaller models with the same amount of data, leading to compute-efficient training. Our experiments on synthetic formal languages do not demonstrate such a pattern, possibly because we are allowing all models of different sizes to reach their optimal fine-tuning performance, where there is no tangible benefit of being a large model.

In-context learning: Given a set of examples as demonstrations, ICL allows LLMs to extract patterns without updating model parameters. Several studies attempt to explain how learning is achieved in ICL, by comparing it to gradient descent (Shen et al., 2023), in-weights learning (Reddy, 2023), and in a controlled setting of learning simple and complex boolean functions (Bhattamishra et al., 2020). Recently, Pan et al. (2023); Lin and Lee (2024) explore the dual characteristics of ICL: (i) task learning, where the test examples are unseen during pre-training and (ii) task recognition/retrieval, where test examples are seen during the pre-training, and LLMs are asked to retrieve them using a different prompt. In a separate line, Wei et al. (2023) study the relationship between ICL and model scale, where overriding semantic priors like flipping labels improves in performance with larger models. In-contrast, Chen et al. (2025) observe that ICL ability does not linearly correlate with model size. Our study finds that in the majority of model families, model size improves ICL, while in few families, a medium sized model is better in ICL.

Fine-tuning versus In-context learning. Several works study a comparison between FT and ICL, where results are inconclusive. Brown et al. (2020); Mosbach et al. (2023); Liu et al. (2022b); Lester et al. (2021); Bhatia et al. (2023); Asai et al. (2024) share the consensus that FT is better than ICL. However, this observation is made under unequal conditions, violating desideratum **D2**: (a) using incomparable models (Liu et al., 2022b), (b) unequal number of examples (Brown et al., 2020; Liu et al., 2022b), and (c) observing high variance across different choices of examples (Asai et al., 2024).

Another group of works led by Yin et al. (2024); Bertsch et al. (2024); Kaneko et al. (2025); Soudani et al. (2024); Awadalla et al. (2022) find that ICL is better than FT. To our best knowledge, none of these works fine-tune the models to their optimal point, e.g., in a naive way, Yin et al. (2024) fine-tune for 1 epoch, and Awadalla et al. (2022) fine-tune for 10 epochs. The inconsistencies in experimental designs motivate us to agree on desideratum **D2**, where different modes of learning are given a fair comparison under an equal allocation of resources.

A.2 Formal Languages and LLMs

Many prior works have studied formal languages in the context of LLMs. There are two broader questions that most studies have asked, which differ from our goal of comparing FT and ICL.

What is the relative representation capability of LLMs compared to other sequences models, or more specifically, what classes of languages are learnable by an LLM? LLMs with a Transformer architecture may have a different representation capability than other neural language models (LMs) like LSTMs and RNNs. We refer to a recent survey discussing the expressiveness of LLMs as a language recognizer (Strobl et al., 2023b). Towards comparing representation capability, Shi et al. (2022) find that both LSTM and Transformer network can simulate CFL with bounded recursion having a similar representation power. However, LSTM has a disadvantage that it fails to decompose the latent representation space unlike a transformer. (Bhattamishra et al., 2020) observe a clear contrast between the performance of Transformers and LSTMs on regular languages. They find that in comparison with LSTMs, Transformers achieve limited performance on languages involving periodicity, modular counting, and even simpler star-free variants of Dyck-1 languages. Delétang et al. (2022) explore how neural network models used for program induction relate to the idealized computational models defined by the Chomsky hierarchy (Chomsky, 1956). They find that neural language models are hard to place on the standard Chomsky hierarchy. Several works criticize their setup, since they consider a language transduction task (mapping one language

to another), which is different from the language recognition task (Icard, 2020). (Borenstein et al., 2024) consider learning strings from deterministic and probabilistic finite state automata. They empirically test the learnability as function of various complexity parameters of the language and the hidden state size of the Transformer and RNN. In a different line of work, (Akyürek et al., 2024) evaluate neural LM's abilities to learn regular languages in ICL. Rather than learning one particular distribution from the training dataset, they infer the generating mechanism using ICL. Similar to (Delétang et al., 2022), they find that RNNs are better suited to modeling formal languages than Transformers. Kallini et al. (2024) construct a continuum of languages that differ in their hardness to learn and show that GPT-2, a variant of LLM, has difficulty in learning the carefully constructed impossible languages, compared to English.

While most of the works in this line capture the expressiveness of LLMs and its differing representation ability with other sequence models, one fundamental criticism we find is the evaluation metrics they consider. As elaborated in Section 4, they are focusing on testing how well an LLM learn the grammar rules or automata state, without utilizing the natural generation capability of the LLMs in generating strings from inside and outside the language. In contrast to their evaluation criteria, ours is more tailored towards how LLMs operate and become proficient in a language.

Does an LLM learn from a given distribution, if so how? Several studies utilize the controlled data generation of formal languages to study different NLP aspects of the LLM. Formal languages, particularly the one derived from context free grammars, can imitate the rich recursive structure of natural languages. Therefore, many studies focus on teaching the LLM strings from a formal language and explain how LLMs might learn them (Allen-Zhu and Li, 2023; Murty et al., 2022; Liu et al., 2022a). In another line, Jumelet and Zuidema (2023) study if causal and masked LLMs capture the true underlying patterns if trained on a true distribution. They find that causal LLMs approximate the theoretically optimal perplexity of the PCFG more closely than masked LLMs. Along that direction, several studies consider the known distribution to analyze the impact of topological features of a language (Cotterell et al., 2018; Mielke et al., 2019; Ravfogel et al., 2019; Mielke et al., 2019; Papadimitriou and Jurafsky, 2023; White and Cotterell, 2021). Several studies propose to augment additional component to LLMs to enable them learning certain class of languages with ease. For example, Chi et al. (2023) propose to add working memory, such as weight sharing, adaptive-Depth, and sliding-dilated attention to GPT model to enable it to learn parity function, which hard for an LLM to learn (Hahn and Rofin, 2024).

In contrast to this line of work, our focus is to apply formal languages to study different modes of learning in LLMs: FT and ICL, which, to our best knowledge, is novel.

B Extended Experimental Setup

All experiments are conducted in compute clusters with Python as the programming language (version 3.10), where we use 8x Nvidia H100 94GB NVL GPUs and 2x AMD EPYC 9554 CPU @ 3.1 GHz, 2x64 cores, and 24x 96GB RAM. FT is performed with a batch size of 8 and a linear learning rate scheduler with a warm-up ratio of 0.05. We fix the learning rate for Qwen, Gemma, and Llama-3 families as 5×10^{-5}, Mistral, Opt, and Llama-2 families as 5×10^{-6}, and Pythia family as 10^{-5}.

Below, we provide details of the formal languages used in our experiments, along with their formal definitions. Intuitively, we carefully design languages to show the robustness of our results by changing the grammar rules and token types of the language.

Formal Languages and Grammars. Throughout our experiments, we provide the LLM strings sampled from a probabilistic formal language. Underneath, a probabilistic formal language is represented by a *probabilistic formal grammars*, or simply *grammars* (Collins, 2013). Specifically, a grammar consists of two sets of symbols called the *non-terminals* and *terminals*, a set of rules to rewrite strings over these symbols that contain at least one nonterminal – also called the *production rules*, and a probability distribution over the production rules. Formally, a probabilistic formal grammar, is defined as a quintuple.

$$G \triangleq (\mathbf{N}, \mathbf{T}, \mathbf{R}, S, \mathbf{P})$$

where $\mathbf{N}$ is the set of non-terminals, $\mathbf{T}$ is the set of terminals (equivalently, tokens), $\mathbf{R}$ is the set of production rules, $S \in \mathbf{N}$ is the start non-terminal, and $\mathbf{P}$ is the set of probabilities on production rules.

Formal languages are divided into well-known classes based on the *complexity* of the language

$S \to A16$ [1]
$A16 \to A15\ A13$ [0.50]
$A16 \to A13\ A15\ A14$ [0.50]
$A13 \to A11\ A12$ [0.50]
$A13 \to A12\ A11$ [0.50]
$A14 \to A11\ A10\ A12$ [0.50]
$A14 \to A10\ A11\ A12$ [0.50]
$A15 \to A12\ A11\ A10$ [0.50]
$A15 \to A11\ A12\ A10$ [0.50]
$A10 \to A7\ A9\ A8$ [0.50]
$A10 \to A9\ A8\ A7$ [0.50]
$A11 \to A8\ A7\ A9$ [0.50]
$A11 \to A7\ A8\ A9$ [0.50]
$A12 \to A8\ A9\ A7$ [0.50]
$A12 \to A9\ A7\ A8$ [0.50]
$A7 \to 3\ 1$ [0.50]
$A7 \to 1\ 2\ 3$ [0.50]
$A8 \to 6\ 5$ [0.50]
$A8 \to 6\ 4\ 5$ [0.50]
$A9 \to 9\ 8\ 7$ [0.50]
$A9 \to 8\ 7$ [0.50]

$S \to A16$ [1]
$A16 \to A15\ A13$ [0.50]
$A16 \to A13\ A15\ A14$ [0.50]
$A13 \to A11\ A12$ [0.50]
$A13 \to A12\ A11$ [0.50]
$A14 \to A11\ A10\ A12$ [0.50]
$A14 \to A10\ A11\ A12$ [0.50]
$A15 \to A12\ A11\ A10$ [0.50]
$A15 \to A11\ A12\ A10$ [0.50]
$A10 \to A7\ A9\ A8$ [0.50]
$A10 \to A9\ A8\ A7$ [0.50]
$A11 \to A8\ A7\ A9$ [0.50]
$A11 \to A7\ A8\ A9$ [0.50]
$A12 \to A8\ A9\ A7$ [0.50]
$A12 \to A9\ A7\ A8$ [0.50]
$A7 \to c\ a$ [0.50]
$A7 \to a\ b\ c$ [0.50]
$A8 \to f\ e$ [0.50]
$A8 \to f\ d\ e$ [0.50]
$A9 \to i\ h\ g$ [0.50]
$A9 \to h\ g$ [0.50]

Figure 10: Production rules of $G_\alpha^{\text{Numerical}}$ (left) and G_α^{Latin} (right).

membership problem, i.e., the *complexity* of the grammars needed to generate them (Chomsky, 1956). In this paper, we use one class of grammars, namely, hierarchical probabilistic context-free grammars (HPCFGs) (Allen-Zhu and Li, 2023). Specifically, our experiments are based on teaching LLMs languages represented by HPCFGs. We use HPCFGs because they are simple syntactically and can represent languages that are structurally similar to natural languages (Allen-Zhu and Li, 2023; Shi et al., 2022).

Description of Grammars and Identified Languages. In our experiments, we consider two generic structure for the considered grammars, one adapted from (Allen-Zhu and Li, 2023), namely G_α, and another is proposed by us, namely G_β. We propose variant of these grammars by considering different alphabet sets.

In Figure 10, in the first generic structure G_α, each grammar has $\mathbf{N} = \{S, A7, A8, \ldots, A16\}$ and $\mathbf{T} = \{1, 2, 3, \ldots, 9\}$. The grammar has four levels of hierarchy: the non-terminals from top to bottom levels are $\{A16\}$, $\{A13, A14, A15\}$, $\{A10, A11, A12\}$, and $\{A7, A8, A9\}$, followed by terminals $\{1, 2, 3, \ldots, 9\}$. Since the terminals are derived from numerical characters, we call this grammar $G_\alpha^{\text{Numerical}}$; and if the terminals are derived from Latin characters, we call this grammar G_α^{Latin}, respectively. Each non-terminal (except the start non-terminal) has two expansion rules, consisting of non-terminals from the immediate lower level. Further, the expansion rules are probabilistic, where the sum of probabilities of all expansion rules from a given non-terminal is 1.

In Figure 11, the second generic structure G_β is inspired by bridging two HPCFGs together, and simulating a long range dependencies within the generated strings. Specifically, the two sub-grammars at $B4$ and sub-grammar at $E4$ are connected by non-terminal $C1_i$; and $E4$ ends with $T1_j$. Long range dependencies are communicated

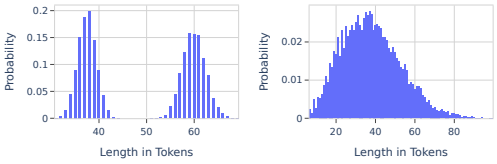
$S \to S5$ [1]
$S5 \to B4\ C1_1\ E4\ T1_1$ [0.25]
$S5 \to B4\ C1_2\ E4\ T1_2$ [0.25]
$S5 \to B4\ C1_3\ E4\ T1_3$ [0.25]
$S5 \to B4\ C1_4\ E4\ T1_4$ [0.25]
$B4 \to B3$ [0.3333]
$B4 \to B3\ B3\ B3$ [0.3333]
$B4 \to B3\ B3$ [0.3333]
$B3 \to B2$ [0.3333]
$B3 \to B2$ [0.3333]
$B3 \to B2\ B2$ [0.3333]
$B2 \to B1$ [0.3333]
$B2 \to B1$ [0.3333]
$B2 \to B1\ B1\ B1$ [0.3333]
$B1 \to 2\ 9\ 3$ [0.3333]
$B1 \to 9\ 6\ 1$ [0.3333]
$B1 \to 1\ 8\ 6\ 2$ [0.3333]
$E4 \to E3$ [0.3333]
$E4 \to E3\ E3$ [0.3333]
$E4 \to E3\ E3\ E3$ [0.3333]
$E3 \to E2$ [0.3333]
$E3 \to E2\ E2$ [0.3333]
$E3 \to E2$ [0.3333]
$E2 \to E1\ E1$ [0.3333]
$E2 \to E1$ [0.3333]
$E2 \to E1\ E1\ E1$ [0.3333]
$E1 \to 5\ 6$ [0.3333]
$E1 \to 1\ 8\ 6\ 6$ [0.3333]
$E1 \to 1\ 5\ 1\ 5\ 5\ 9$ [0.3333]
$T1_1 \to 1$ [1]
$T1_2 \to 2$ [1]
$T1_3 \to 3$ [1]
$T1_4 \to 4$ [1]
$C1_1 \to 5$ [1]
$C1_2 \to 6$ [1]
$C1_3 \to 7$ [1]
$C1_4 \to 8$ [1]
$C1_5 \to 9$ [1]

$S \to S5$ [1]
$S5 \to B4\ C1_1\ E4\ T1_1$ [0.25]
$S5 \to B4\ C1_2\ E4\ T1_2$ [0.25]
$S5 \to B4\ C1_3\ E4\ T1_3$ [0.25]
$S5 \to B4\ C1_4\ E4\ T1_4$ [0.25]
$B4 \to B3$ [0.3333]
$B4 \to B3\ B3\ B3$ [0.3333]
$B4 \to B3\ B3$ [0.3333]
$B3 \to B2$ [0.3333]
$B3 \to B2$ [0.3333]
$B3 \to B2\ B2$ [0.3333]
$B2 \to B1$ [0.3333]
$B2 \to B1$ [0.3333]
$B2 \to B1\ B1\ B1$ [0.3333]
$B1 \to b\ i\ c$ [0.3333]
$B1 \to i\ f\ a$ [0.3333]
$B1 \to a\ h\ f\ b$ [0.3333]
$E4 \to E3$ [0.3333]
$E4 \to E3\ E3$ [0.3333]
$E4 \to E3\ E3\ E3$ [0.3333]
$E3 \to E2$ [0.3333]
$E3 \to E2\ E2$ [0.3333]
$E3 \to E2$ [0.3333]
$E2 \to E1\ E1$ [0.3333]
$E2 \to E1$ [0.3333]
$E2 \to E1\ E1\ E1$ [0.3333]
$E1 \to e\ f$ [0.3333]
$E1 \to a\ h\ f\ f$ [0.3333]
$E1 \to a\ e\ a\ e\ e\ i$ [0.3333]
$T1_1 \to a$ [1]
$T1_2 \to b$ [1]
$T1_3 \to c$ [1]
$T1_4 \to d$ [1]
$C1_1 \to e$ [1]
$C1_2 \to f$ [1]
$C1_3 \to g$ [1]
$C1_4 \to h$ [1]
$C1_5 \to i$ [1]

Figure 11: Production rules of $G_\alpha^{\text{Numerical}}$ (left) and G_α^{Latin} (right).

Table 2: Notations of grammars and identified languages.

Grammar	Identified Language
$G_{\alpha}^{\text{Numerical}}$	L_1
$G_{\alpha}^{\text{Latin}}$	L_2
$G_{\alpha}^{\text{Under-trained-tokens}}$	L_3
$G_{\beta}^{\text{Numerical}}$	L_4
G_{β}^{Latin}	L_5
$G_{\beta}^{\text{Under-trained-tokens}}$	L_6



(a) L_1 (also L_2, L_3) (b) L_4 (also L_5, L_6)

Figure 12: Length distribution of considered probabilistic languages, based on 10000 sampled strings per language.

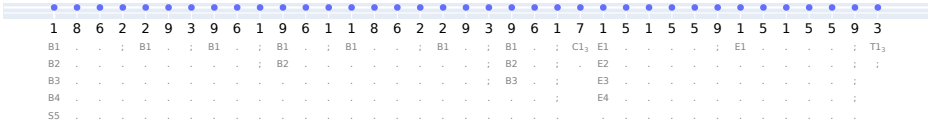
through $C1_i$ and $T1_j$, by enforcing $i = j$ at each expansion of $S5$.

Table 2 shows the mapping of notations between grammars and identified languages. Figure 12 denotes the length distribution of generated strings from different languages. Figure 13 demonstrates how hierarchical non-terminals are applied in different positions in the representative strings.

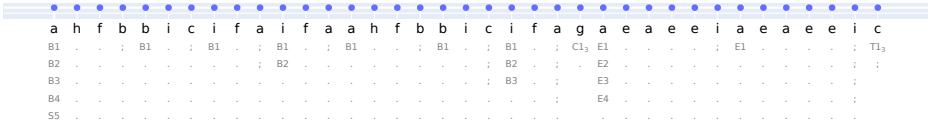
Sampling Strings from a Formal Language. Given a language L generated by a HPCFG, we first need to obtain *training* samples, i.e., set of i.i.d. samples of strings *in-language* L. To *sample a string from the language*, we start from a special string in the grammar containing a single, distinguished nonterminal called the "start" or "root" symbol, and apply the production rules to rewrite the string repeatedly. If several rules can be used to rewrite the string at any stage, we sample one such rule from the probability distribution over the rules and apply it. We stop when we obtain a string containing terminals only. This string is a sample drawn from the language. We can repeat this process to draw any number of i.i.d. samples from the language.

(a) Language L_1 (Grammar $G_{\alpha}^{\text{Numerical}}$)

(b) Language L_2 (Grammar $G_{\alpha}^{\text{Latin}}$)



(c) Language L_4 (Grammar $G_{\beta}^{\text{Numerical}}$)



(d) Language L_5 (Grammar G_{β}^{Latin})

Figure 13: Representative strings from different languages, annotated with non-terminals applied in different positions by the respective hierarchical grammar.

Distance Between Languages. In probabilistic languages, a well-known approach to compute language distance is to compare the distribution of strings generated by both languages (de la Higuera et al., 2014). In our implementation, we choose a simplified distance metric based on L_2-norm.

$$\texttt{dist}_{L_2}(L_1, L_2) = \sqrt{\Sigma_{s \in \mathbf{T}^*}(P_{L_1}(s) - P_{L_2}(s))^2} \tag{1}$$

While distance metrics have their nuances, our goal is to systematically modify the original language, more specifically the underlying grammar, such that we can intuitively interpret language distance, irrespective of the distance metric used.

For simulating out-of-distribution generalization of learning modes, we modify the base grammar $G_\alpha^{\text{Numerical}}$ or G as a short notation, in the following way: We construct the five grammars $\{G^{(1)}, \ldots, G^{(5)}\}$, by perturbing ℓ production rules of G, such that $G^{(\ell)}$ contains all perturbed production rules in $G^{(\ell-1)}$. The order in which rule-perturbation is applied is the following:

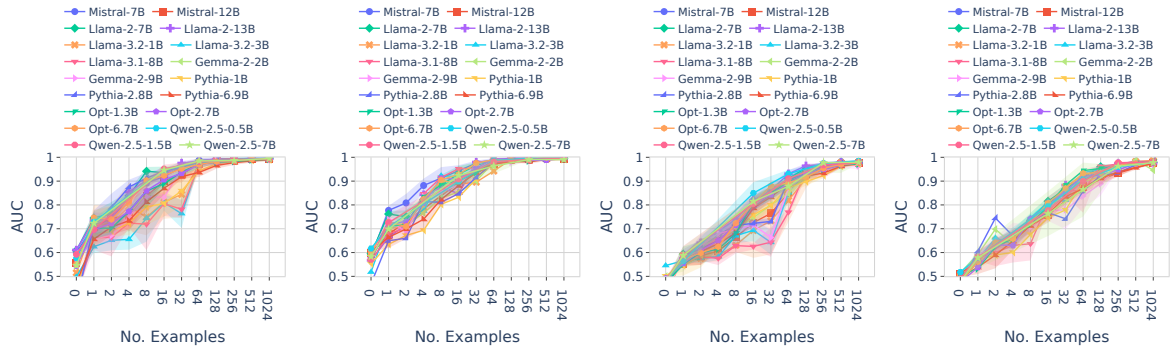
$$A7^{(1)} \rightarrow 1\ 3\ 2\ [0.50]$$
$$A8^{(2)} \rightarrow 5\ 6\ [0.50]$$
$$A9^{(3)} \rightarrow 8\ 7\ 9\ [0.50]$$
$$A7^{(4)} \rightarrow 3\ 1\ [0.50]$$
$$A9^{(5)} \rightarrow 8\ 7\ [0.50]$$

Intuitively, $G^{(1)}$ contains perturbed rule $\{A7^{(1)}\}$, $G^{(2)}$ contains perturbed rule $\{A7^{(1)}, A8^{(2)}\}$, and so on. Finally, each grammar $G^{(\ell)}$ identifies a language $L^{(\ell)}$ in Figure 7.

C Additional Experimental Results

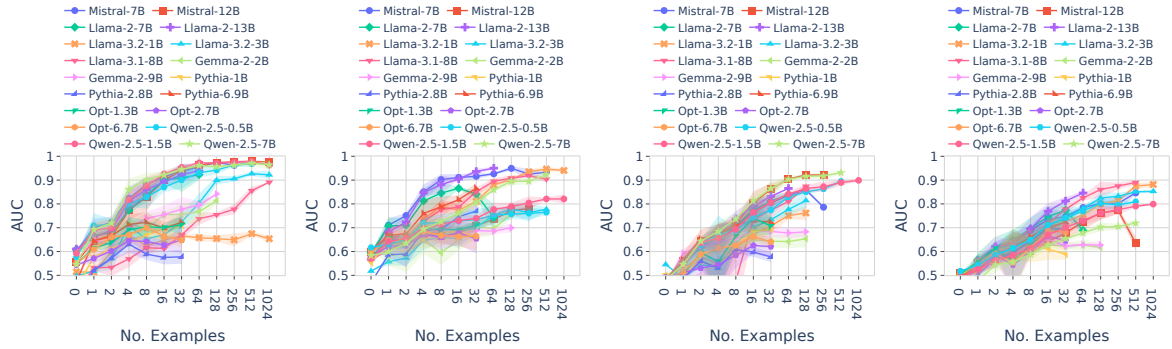
In the following, we outline additional experimental results.

- Independent evaluation of `FT` and `ICL` on different languages across datasets in Figure 14, 15.
- Intra-family `FT` and `ICL` performance in Figure 16, 17.
- Robustness of `FT` and `ICL` of individual models across languages in Figure 18, 19 20, 21.
- Inductive bias of LLMs across languages in Figure 22, 23, 24, 25.
- Out-of-distribution generalization on languages of different distances in Figure 26.
- `FT` vs. `ICL` on natural language datasets in Appendix D.
- Evaluating whether LLMs utilize their full `ICL` contexts in Appendix E.
- Generative vs. discriminative tests for determining language proficiency in Appendix F.
- Detailed implications of the study in Appendix G.



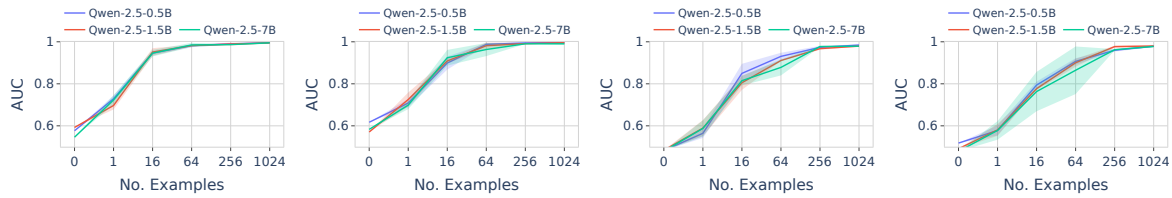
(a) Language L_1 (b) Language L_2 (c) Language L_4 (d) Language L_5

Figure 14: Optimal fine-tuning performance in all models across different languages.

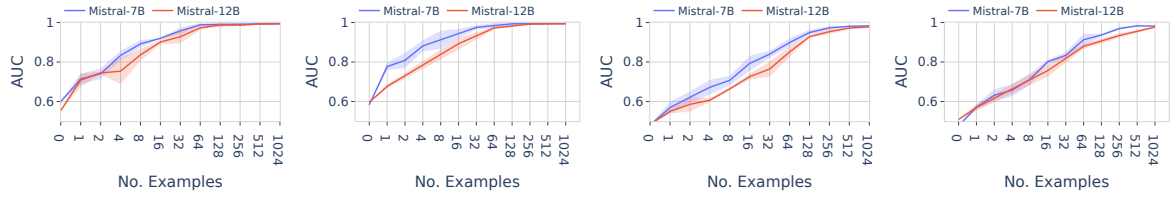


(a) Language L_1 (b) Language L_2 (c) Language L_4 (d) Language L_5

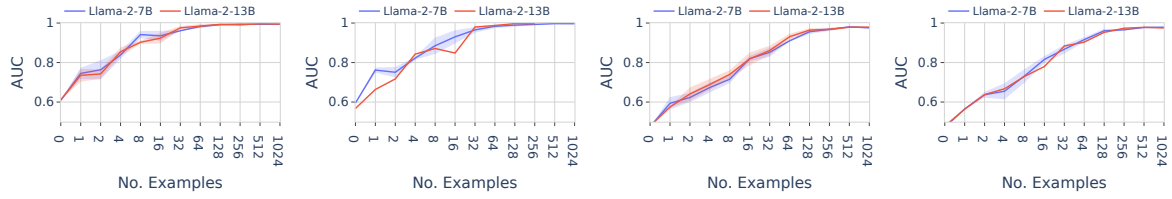
Figure 15: In-context learning performance of all models across different languages



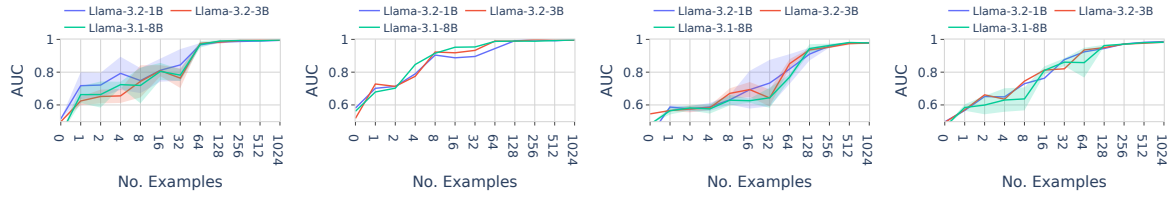
Qwen, Language L_1 | Qwen, Language L_2 | Qwen, Language L_4 | Qwen, Language L_5



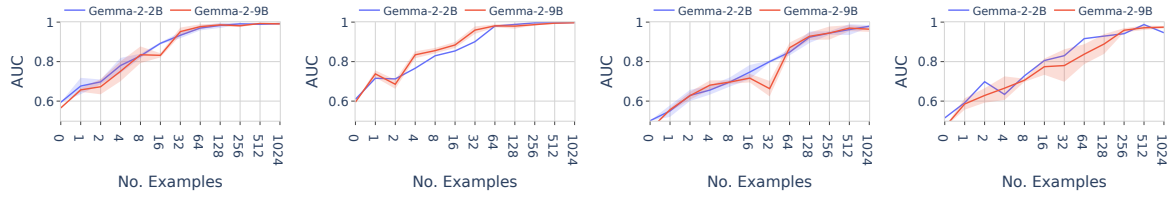
Mistral, Language L_1 | Mistral, Language L_2 | Mistral, Language L_4 | Mistral, Language L_5



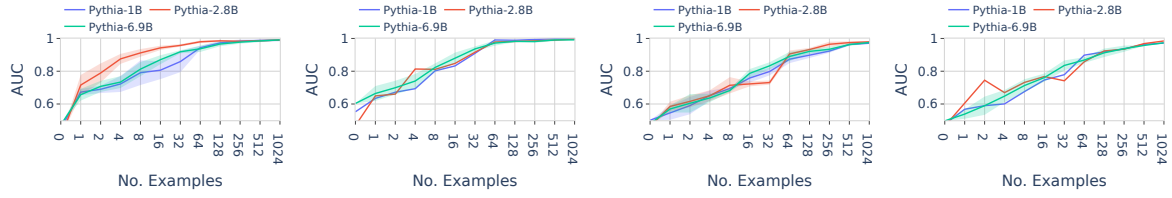
Llama-2, Language L_1 | Llama-2, Language L_2 | Llama-2, Language L_4 | Llama-2, Language L_5



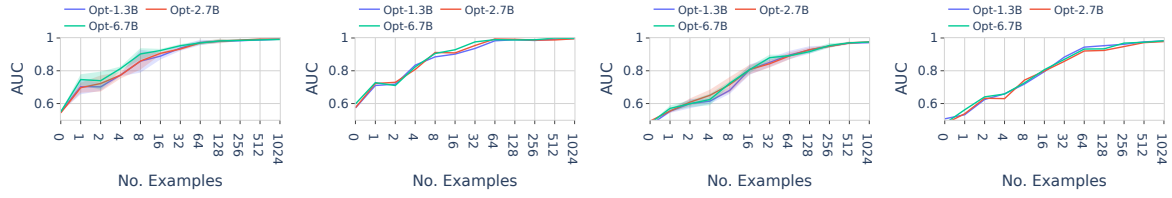
Llama-3, Language L_1 | Llama-3, Language L_2 | Llama-3, Language L_4 | Llama-3, Language L_5



Gemma-, Language L_1 | Gemma, Language L_2 | Gemma, Language L_4 | Gemma, Language L_5

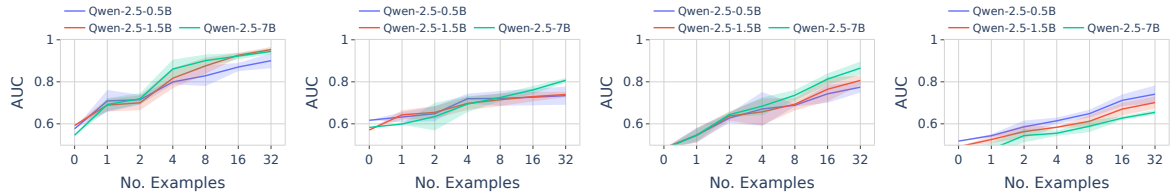


Pythia, Language L_1 | Pythia, Language L_2 | Pythia, Language L_4 | Pythia, Language L_5

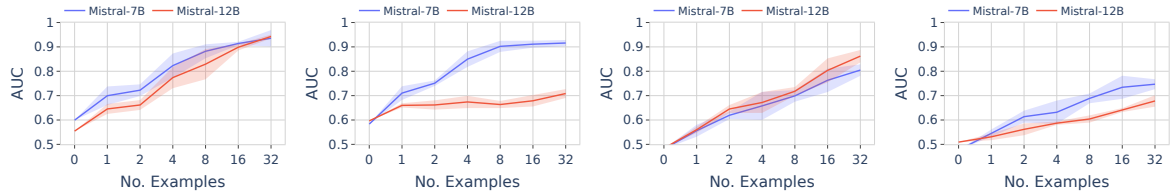


Opt, Language L_1 | Opt, Language L_2 | Opt, Language L_4 | Opt, Language L_5

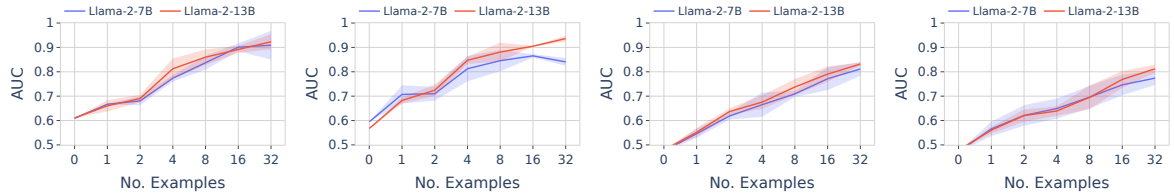
Figure 16: Intra-family FT performance.



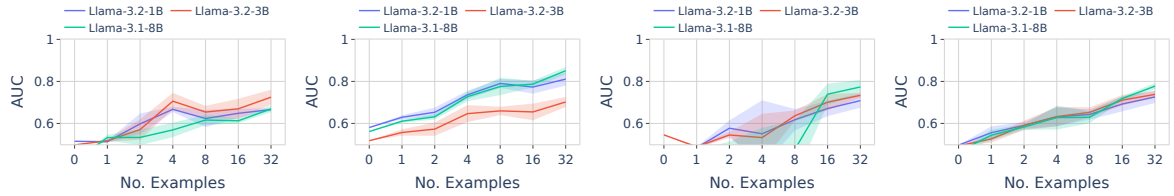
Qwen, Language L_1 | Qwen, Language L_2 | Qwen, Language L_4 | Qwen, Language L_5



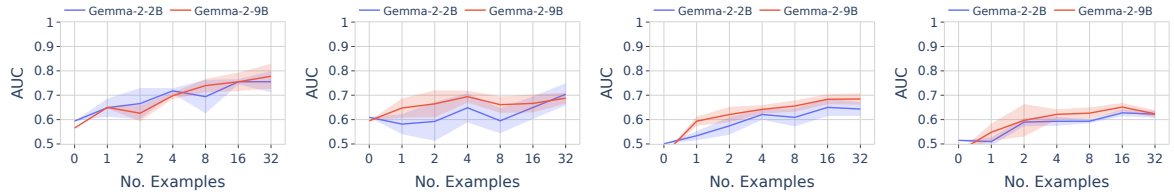
Mistral, Language L_1 | Mistral, Language L_2 | Mistral, Language L_4 | Mistral, Language L_5



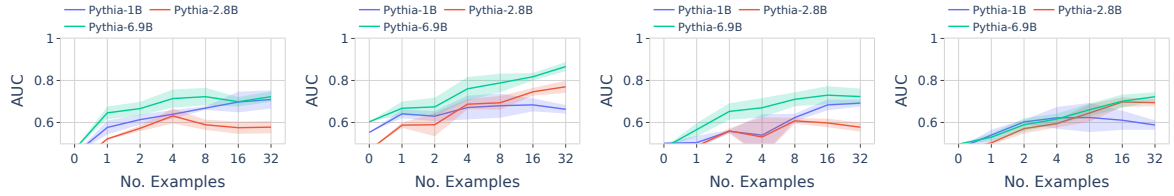
Llama-2, Language L_1 | Llama-2, Language L_2 | Llama-2, Language L_4 | Llama-2, Language L_5



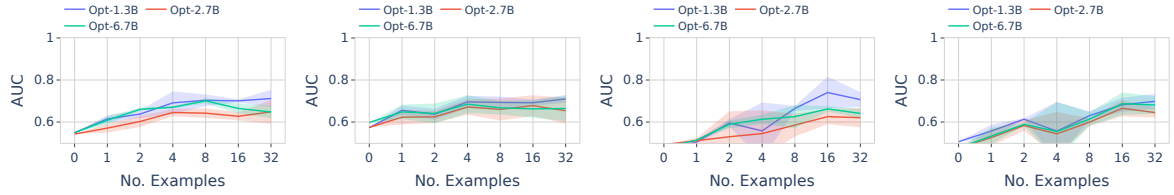
Llama-3, Language L_1 | Llama-3, Language L_2 | Llama-3, Language L_4 | Llama-3, Language L_5



Gemma-, Language L_1 | Gemma, Language L_2 | Gemma, Language L_4 | Gemma, Language L_5

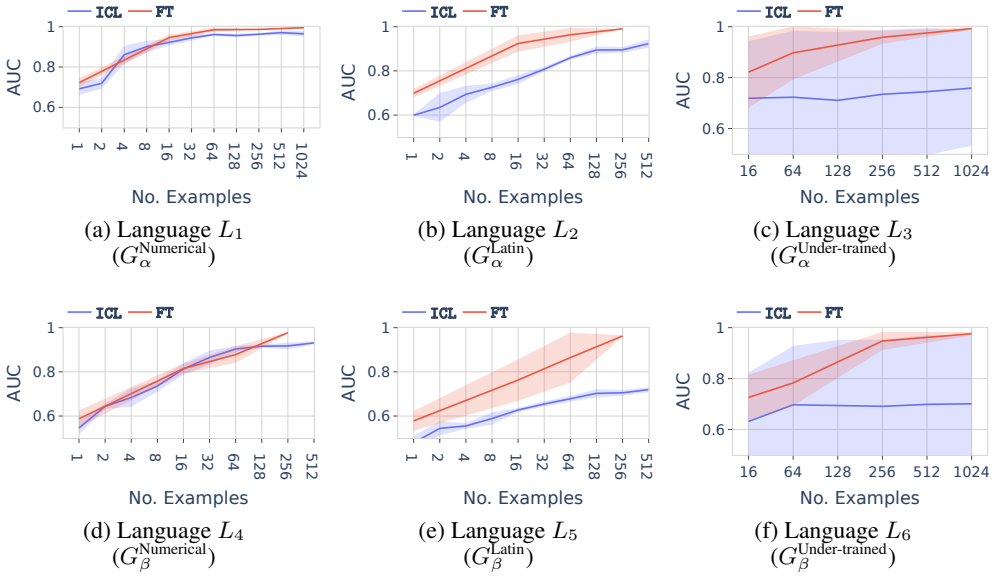


Pythia, Language L_1 | Pythia, Language L_2 | Pythia, Language L_4 | Pythia, Language L_5



Opt, Language L_1 | Opt, Language L_2 | Opt, Language L_4 | Opt, Language L_5

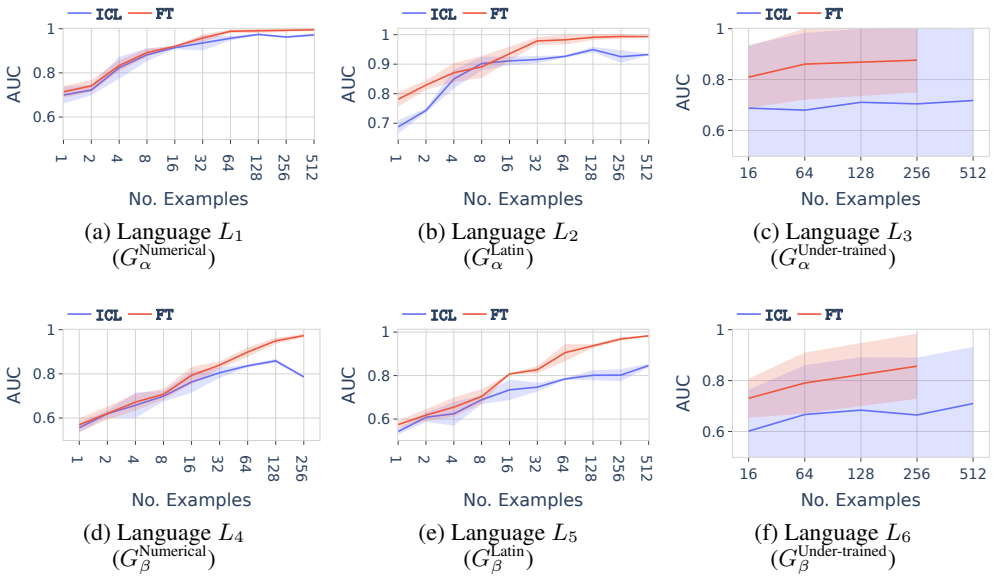
Figure 17: Intra-family ICL performance.



(a) Language L_1 ($G_\alpha^{\text{Numerical}}$) (b) Language L_2 (G_α^{Latin}) (c) Language L_3 ($G_\alpha^{\text{Under-trained}}$)

(d) Language L_4 ($G_\beta^{\text{Numerical}}$) (e) Language L_5 (G_β^{Latin}) (f) Language L_6 ($G_\beta^{\text{Under-trained}}$)

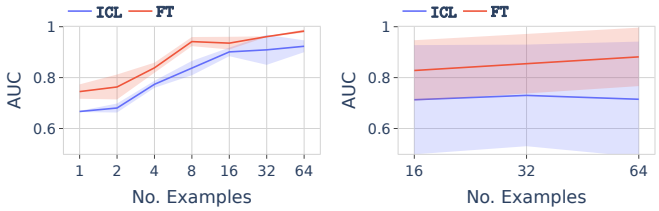
Figure 18: Qwen-2.5-7B: comparison between fine-tuning and in-context learning across different languages



(a) Language L_1 ($G_\alpha^{\text{Numerical}}$) (b) Language L_2 (G_α^{Latin}) (c) Language L_3 ($G_\alpha^{\text{Under-trained}}$)

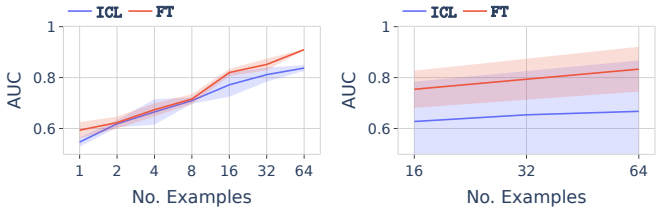
(d) Language L_4 ($G_\beta^{\text{Numerical}}$) (e) Language L_5 (G_β^{Latin}) (f) Language L_6 ($G_\beta^{\text{Under-trained}}$)

Figure 19: Mistral-7B: comparison between fine-tuning and in-context learning across different languages



(a) Language L_1 ($G_\alpha^{\text{Numerical}}$)

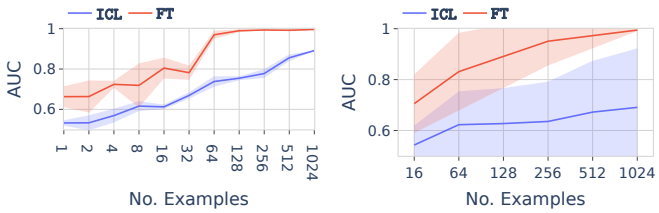
(b) Language L_3 ($G_\alpha^{\text{Under-trained}}$)



(c) Language L_4 ($G_\beta^{\text{Numerical}}$)

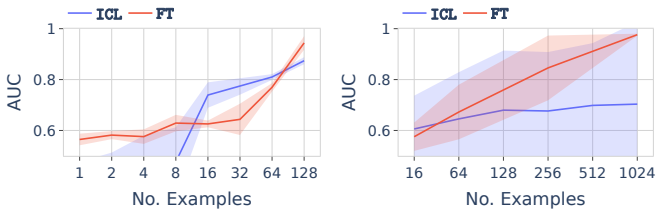
(d) Language L_6 ($G_\beta^{\text{Under-trained}}$)

Figure 20: Llama-2-7B: comparison between fine-tuning and in-context learning across different languages.



(a) Language L_1 ($G_\alpha^{\text{Numerical}}$)

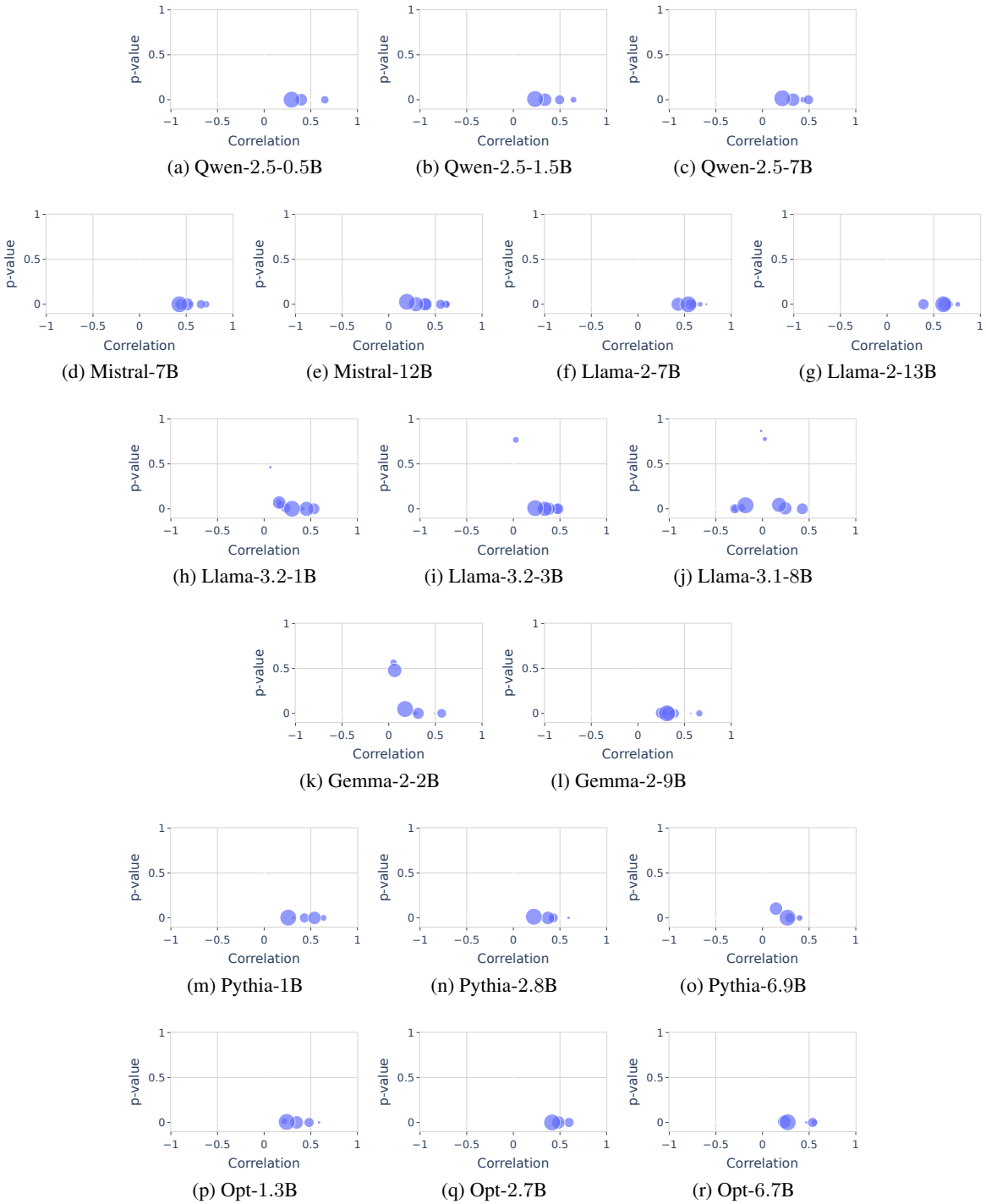
(b) Language L_3 ($G_\alpha^{\text{Under-trained}}$)



(c) Language L_4 ($G_\beta^{\text{Numerical}}$)

(d) Language L_6 ($G_\beta^{\text{Under-trained}}$)

Figure 21: Llama-3.1-8B: comparison between fine-tuning and in-context learning across different languages



(a) Qwen-2.5-0.5B (b) Qwen-2.5-1.5B (c) Qwen-2.5-7B

(d) Mistral-7B (e) Mistral-12B (f) Llama-2-7B (g) Llama-2-13B

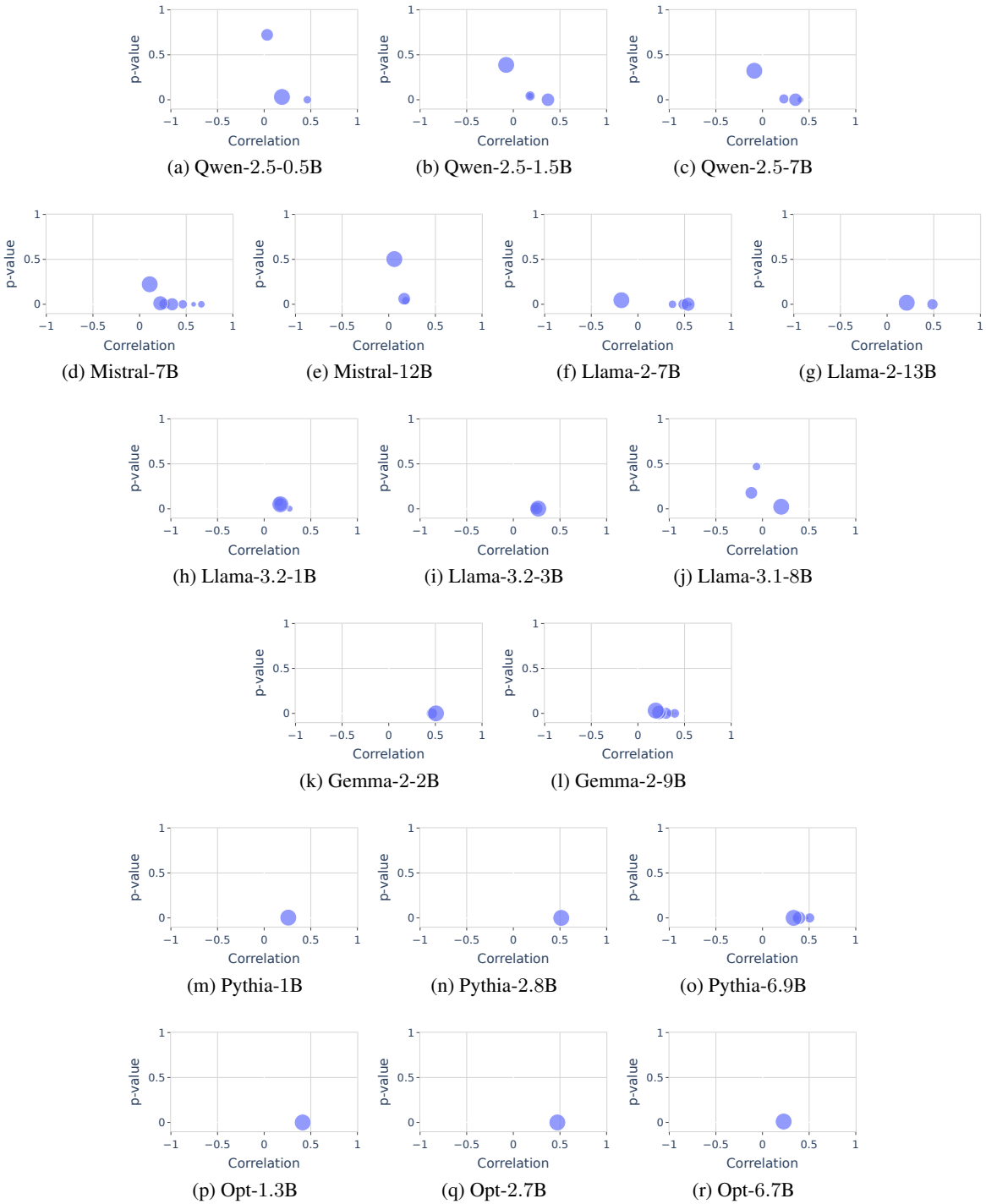
(h) Llama-3.2-1B (i) Llama-3.2-3B (j) Llama-3.1-8B

(k) Gemma-2-2B (l) Gemma-2-9B

(m) Pythia-1B (n) Pythia-2.8B (o) Pythia-6.9B

(p) Opt-1.3B (q) Opt-2.7B (r) Opt-6.7B

Figure 22: Inductive bias of ICL and FT on language L_1, computed as the Pearson correlation of generation loss of FT and ICL on identical test strings. Correlation, despite being positive, tends to decrease with higher examples (larger markers).



(a) Qwen-2.5-0.5B (b) Qwen-2.5-1.5B (c) Qwen-2.5-7B

(d) Mistral-7B (e) Mistral-12B (f) Llama-2-7B (g) Llama-2-13B

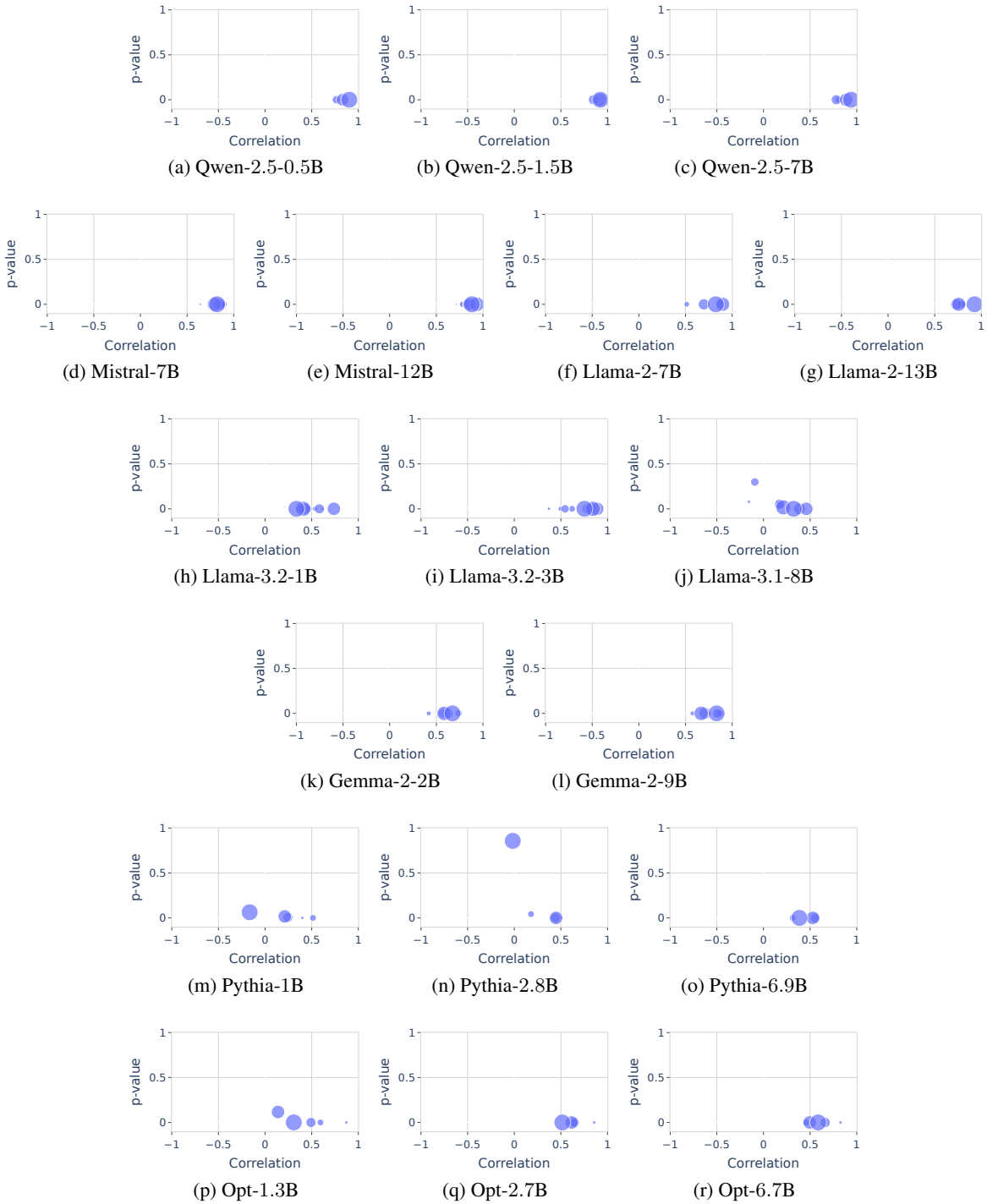
(h) Llama-3.2-1B (i) Llama-3.2-3B (j) Llama-3.1-8B

(k) Gemma-2-2B (l) Gemma-2-9B

(m) Pythia-1B (n) Pythia-2.8B (o) Pythia-6.9B

(p) Opt-1.3B (q) Opt-2.7B (r) Opt-6.7B

Figure 23: Inductive bias of ICL and FT on language L_2, computed as the Pearson correlation of generation loss of FT and ICL on identical test strings. Correlation, despite being positive, tends to decrease with higher examples (larger markers).



(a) Qwen-2.5-0.5B (b) Qwen-2.5-1.5B (c) Qwen-2.5-7B

(d) Mistral-7B (e) Mistral-12B (f) Llama-2-7B (g) Llama-2-13B

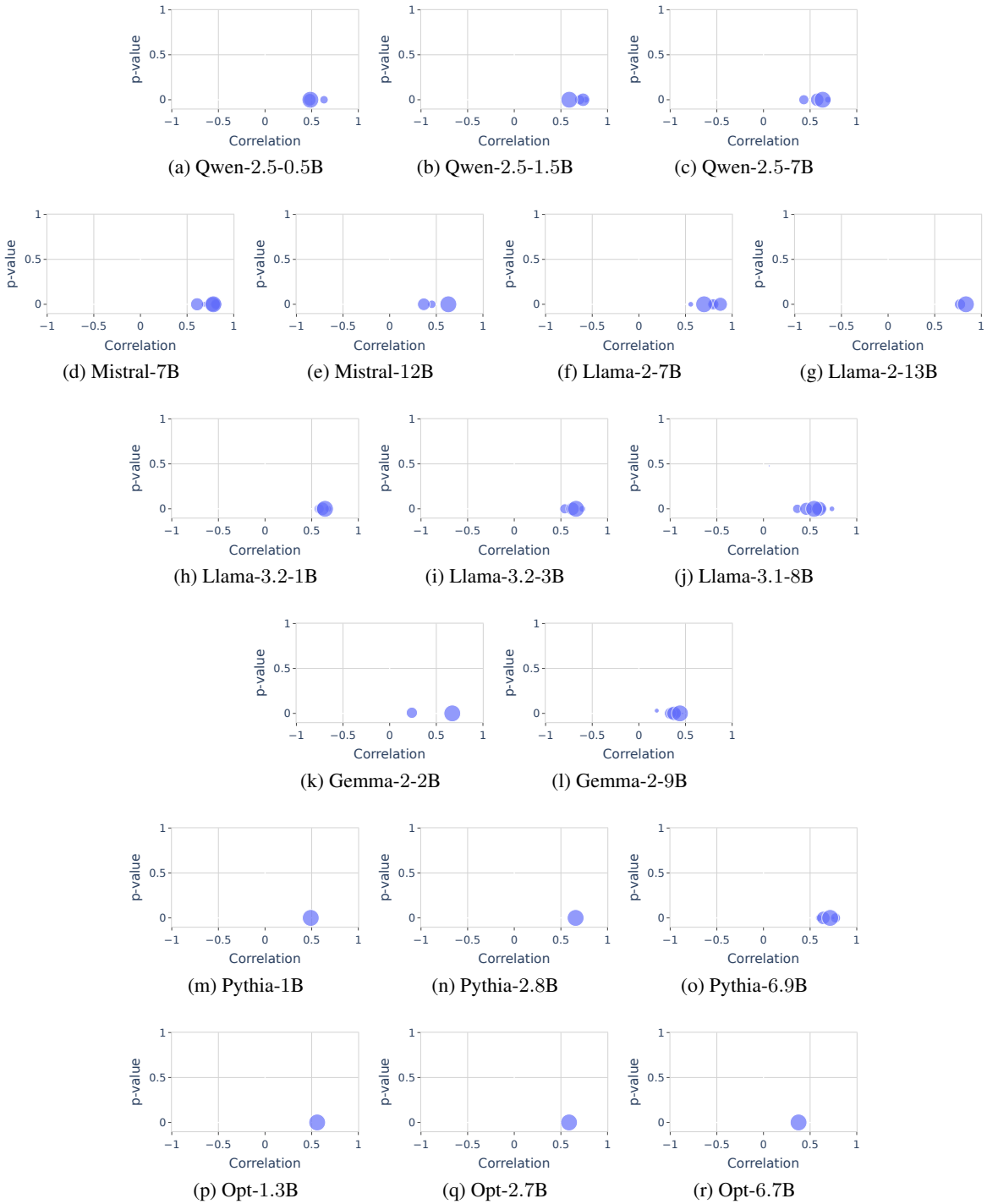
(h) Llama-3.2-1B (i) Llama-3.2-3B (j) Llama-3.1-8B

(k) Gemma-2-2B (l) Gemma-2-9B

(m) Pythia-1B (n) Pythia-2.8B (o) Pythia-6.9B

(p) Opt-1.3B (q) Opt-2.7B (r) Opt-6.7B

Figure 24: Inductive bias of ICL and FT on language L_4, computed as the Pearson correlation of generation loss of FT and ICL on identical test strings. Correlation, despite being positive, tends to decrease with higher examples (larger markers).



(a) Qwen-2.5-0.5B (b) Qwen-2.5-1.5B (c) Qwen-2.5-7B

(d) Mistral-7B (e) Mistral-12B (f) Llama-2-7B (g) Llama-2-13B

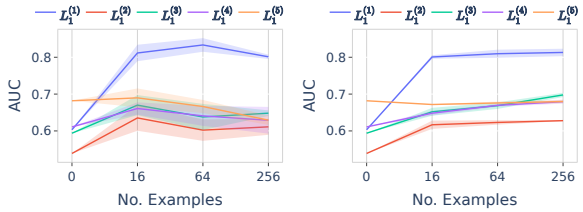
(h) Llama-3.2-1B (i) Llama-3.2-3B (j) Llama-3.1-8B

(k) Gemma-2-2B (l) Gemma-2-9B

(m) Pythia-1B (n) Pythia-2.8B (o) Pythia-6.9B

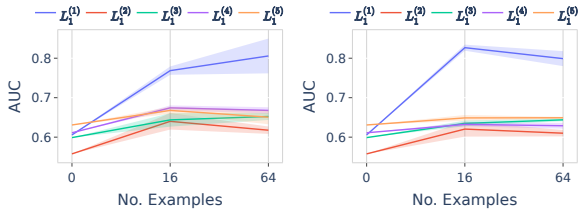
(p) Opt-1.3B (q) Opt-2.7B (r) Opt-6.7B

Figure 25: Inductive bias of ICL and FT on language L_5, computed as the Pearson correlation of generation loss of FT and ICL on identical test strings. Correlation, despite being positive, tends to decrease with higher examples (larger markers).



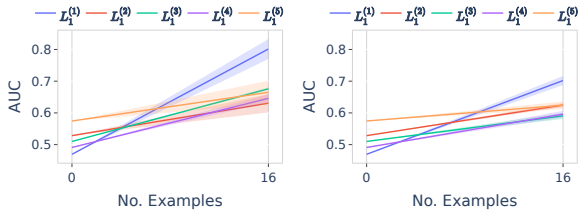
(a) FT, Mistral-7B

(b) ICL, Mistral-7B



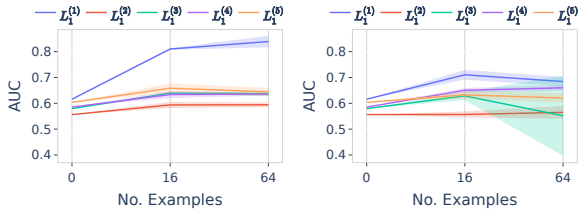
(c) FT, Llama-2-7B

(d) ICL, Llama-2-7B



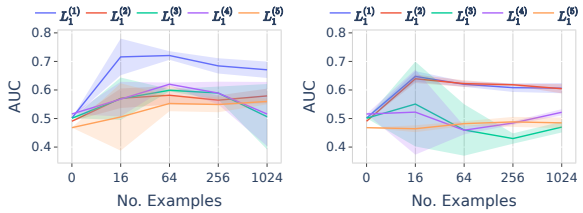
(e) FT, Pythia-6.9B

(f) ICL, Pythia-6.9B



(g) FT, Gemma-2-2B

(h) ICL, Gemma-2-2B



(i) FT, Llama-3.2-1B

(j) ICL, Llama-3.2-1B

Figure 26: Out-of-distribution generalization to languages of increasing distance using FT and ICL. The first and second column denote generative loss, while the third and fourth column denote discriminative performance. We consider L_1 as the base language. We create languages of higher distance, denoted by $L_1^{(\ell)}$, by changing ℓ production rules in the grammar of L_1. $L_1^{(\ell)}$ contains all changed rules in $L_1^{(\ell-1)}$. Hence, $\texttt{dist}(L_1, L_1^{(1)}) \leq \cdots \leq \texttt{dist}(L_1, L_1^{(5)})$ (see Eq. (1))

D Fine-tuning vs. In-context Learning on Natural Language Datasets

We conduct a comparison of FT and ICL on a natural language dataset to observe whether our findings on formal languages generalize to natural language datasets. We consider a natural language inference task on MNLI dataset (Williams et al., 2018), as studied in the related work by Mosbach et al. (2023). The learning objective is to generate the sentiment label given premise and hypothesis (see Table 3).

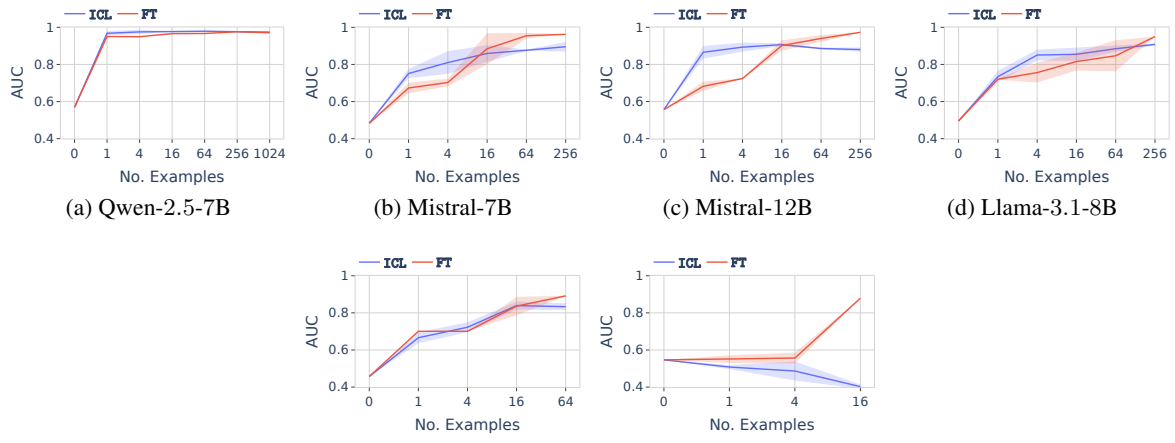
Issues of Data Contamination. In Figure 27, FT surpasses ICL on the MNLI dataset with increasing examples, *which is consistent with our findings on formal languages* in Figure 6. However, Qwen-2.5-7B model performs much better than other models in both learning modes, suggesting the *possibility* of data contamination. For evidence, MNLI dataset is proposed in 2018, which is earlier than the release of Qwen-2.5-7B model in 2024. Therefore, it is difficult to fairly compare different models or their learning modes on publicly available datasets, if a subset of models are possibly trained on the testing dataset (Dominguez-Olmedo et al., 2024). This further strengthens our case that *synthetic formal languages should be adopted widely to critically evaluate the performance of LLMs, where the risk of data contamination is minimum.*

Difficulty in Identifying In-distribution vs. Out-of-distribution Tasks. In Figure 28, we demonstrate in-distribution and out-of-distribution performance side-by-side on the MNLI dataset for both learning modes. The differentiation of tasks is determined by the genre of (premise, hypothesis) pairs. If the genre of the testing pair matches with training pairs, then the task is in-distribution. Otherwise, the task is out-of-distribution. However, we do not observe any difference in the comparison of FT vs. ICL based on tasks – FT is better than ICL in both tasks. This is a contradiction with our findings in formal languages in Figure 7, where the distance between tasks is well-defined, and both FT and ICL perform equally well in the out-of-distribution task. This experiment highlights the ambiguity of specifying learning tasks in natural language datasets, the core theme in desideratum **D1** in Section 1. *Therefore, for an objective comparison, it is important to carefully define in-distribution and out-of-distribution tasks, which is easier in formal languages than natural languages.*

Table 3: Construction of in-language and out-language strings for the MNLI dataset (Williams et al., 2018), where the out-language string differ with in-language string only in the sentiment label. The discriminative test is successful, if the generation probability of the correct label in the in-language string is higher than the incorrect label in the out-language string. Prompt instruction is shown in the below table.

In-language string	Out-language string (edit at label)
Premise: One of our number will carry out your instructions minutely. **Hyopthesis**: A member of my team will execute your orders with immense precision. **Label**: entailment	**Premise**: One of our number will carry out your instructions minutely. **Hyopthesis**: A member of my team will execute your orders with immense precision. **Label**: neutral
Premise: Fun for adults and children. **Hyopthesis**: Fun for only children. **Label**: contradiction	**Premise**: Fun for adults and children. **Hyopthesis**: Fun for only children. **Label**: entailment

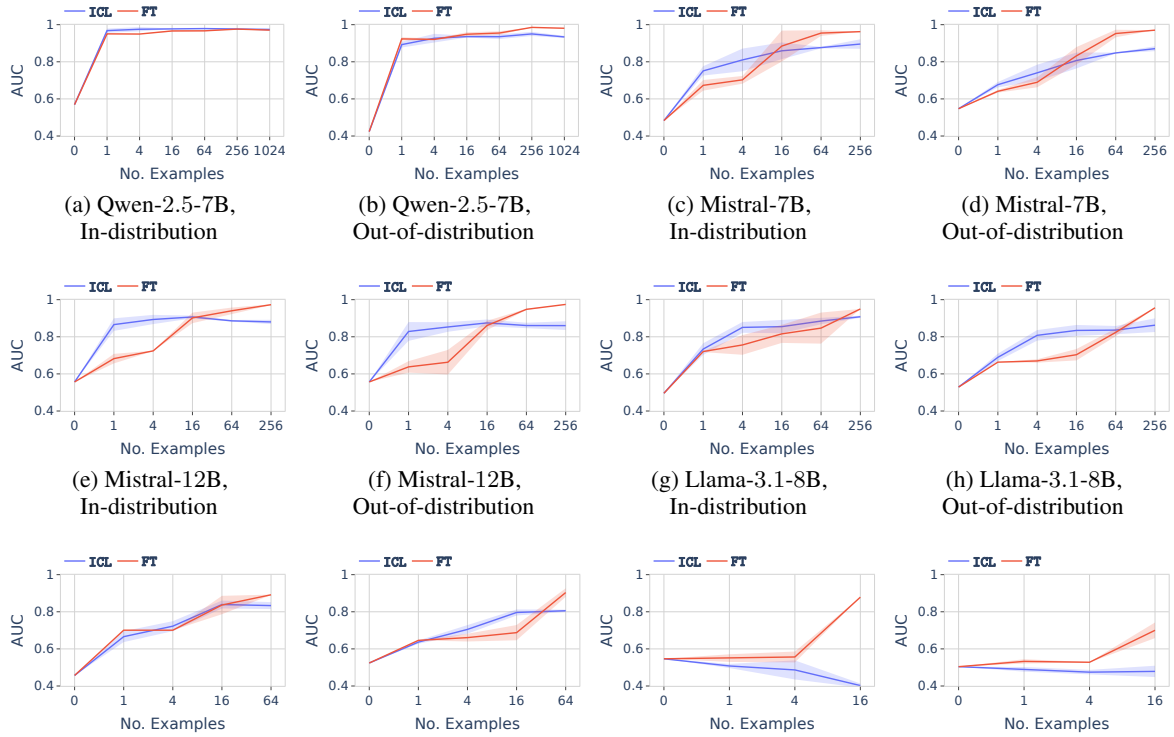
Prompt Instruction (beginning of the prompt)
Provide a classification label for the pair, indicating the relationship between the premise and hypothesis: - entailment : The hypothesis logically follows from the premise. - neutral : The hypothesis is neither entailed nor contradicted by the premise. - contradiction : The hypothesis contradicts the premise.



(a) Qwen-2.5-7B (b) Mistral-7B (c) Mistral-12B (d) Llama-3.1-8B

(e) Llama-2-13B (f) Opt-6.7B

Figure 27: In-distribution generalization of FT and ICL on MNLI dataset, where the learning task is to perform natural language inference by generating the sentiment label {entailment, neutral, contradiction} given premise and hypothesis. On a high level, FT is better than ICL with more examples, consistent with results on formal languages. In a detailed analysis, we observe that different LLMs perform differently given the same problem, indicating the possibility of data contamination in some well performed LLMs, such as Qwen-2.5-7B.



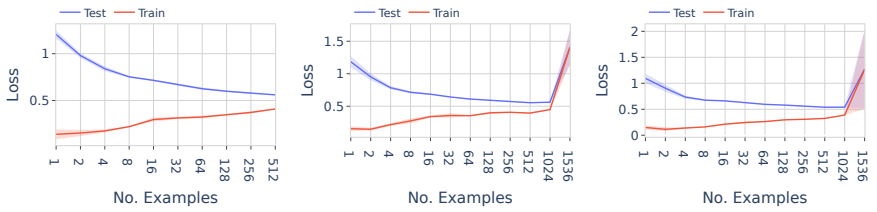
(a) Qwen-2.5-7B, In-distribution
(b) Qwen-2.5-7B, Out-of-distribution
(c) Mistral-7B, In-distribution
(d) Mistral-7B, Out-of-distribution
(e) Mistral-12B, In-distribution
(f) Mistral-12B, Out-of-distribution
(g) Llama-3.1-8B, In-distribution
(h) Llama-3.1-8B, Out-of-distribution
(i) Llama-2-13B, In-distribution
(j) Llama-2-13B, Out-of-distribution
(k) Opt-6.7B, In-distribution
(l) Opt-6.7B, Out-of-distribution

Figure 28: MNLI dataset: In-distribution (inference within the same genre, Column 1 and 3) vs. out-of-distribution (inference across genres, Column 2 and 4) generalization performance0 of FT and ICL, where there is no substantial difference across tasks. This is a fundamental problem in natural language datasets, where the identification of tasks can be ambiguous, and LLMs may not distinguish them. Overall, FT is better than ICL, which contradicts our results on formal languages where FT is only better in in-distribution generalization, but both learning modes are equally performing in out-of-distribution generalization.

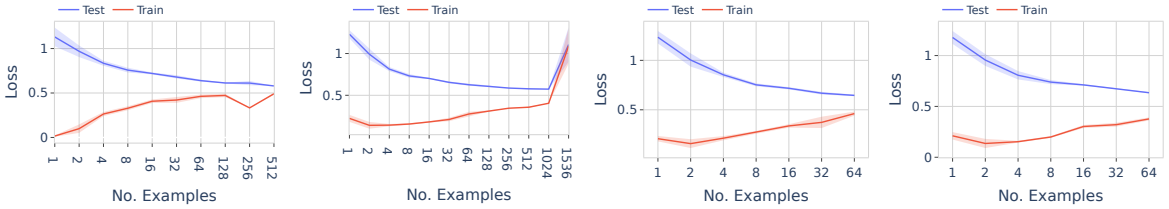
E Testing the Limit of In-context Learning

To find the limit of ICL ability of an LLM, we rely on the convergence of training and test loss in ICL as examples are added. Intuitively, training loss provides a practical *lower bound* of test loss in ICL – an LLM can no longer improve in ICL when both losses converge. To obtain training loss, we first provide ICL examples from the training set and later compute the loss of generating each training example already present in the context.

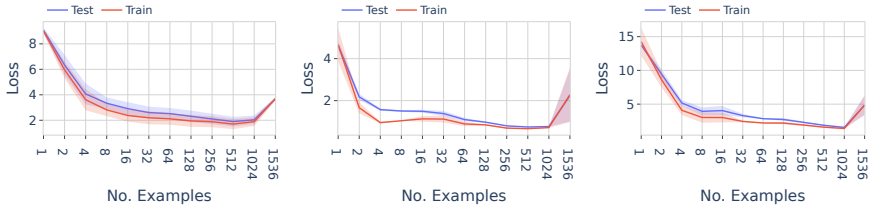
Empirically in all languages, test loss converges to train loss, i.e., ICL *limit is reached* in the majority of LLMs, except in Llama-2 and Opt family. These two families have limited context (4K and 2K tokens, respectively), and there is a gap between losses while exhausting their context length. Moreover, long context LLMs, such as Qwen-2.5-7B and Llama-3.1-8B with 128K context length, cannot further improve from additional examples as both losses converge and later increase near to the limit (see Figure 29). *Therefore, formal language learning enables us to categorize LLMs into two: (a) LLMs that cannot reach the* ICL *limit, and (b) LLMs that reach their* ICL *limit and do not improve with additional examples.*.



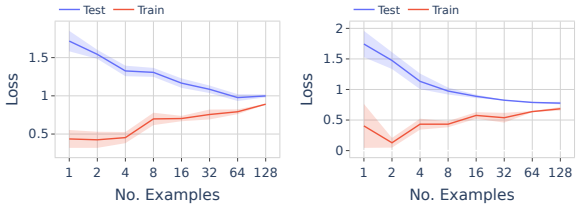
(a) Qwen-2.5-0.5B (b) Qwen-2.5-1.5B (c) Qwen-2.5-7B



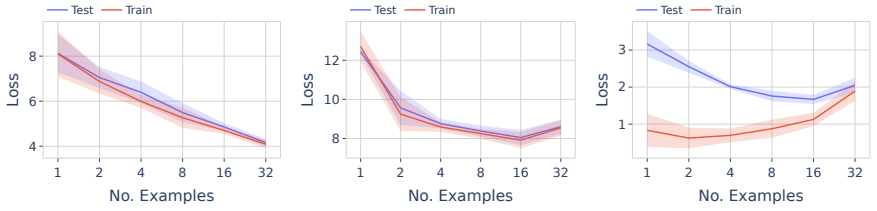
(d) Mistral-7B (e) Mistral-12B (f) Llama-2-7B (g) Llama-2-13B



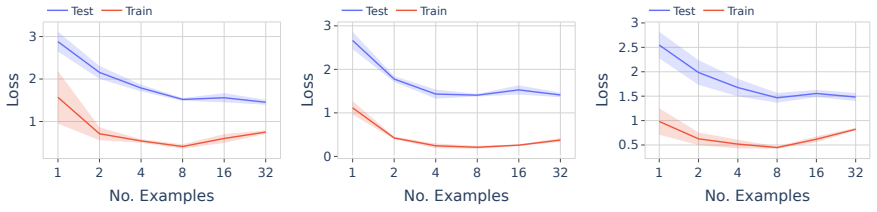
(h) Llama-3.2-1B (i) Llama-3.2-3B (j) Llama-3.1-8B



(k) Gemma-2-2B (l) Gemma-2-9B

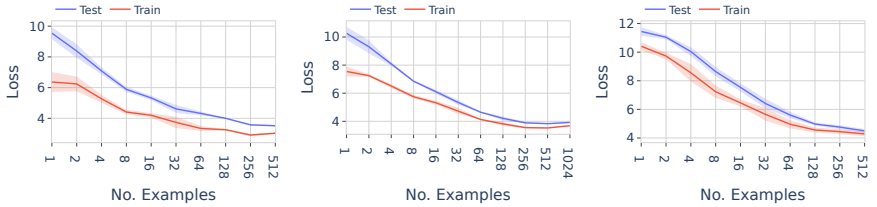


(m) Pythia-1B (n) Pythia-2.8B (o) Pythia-6.9B

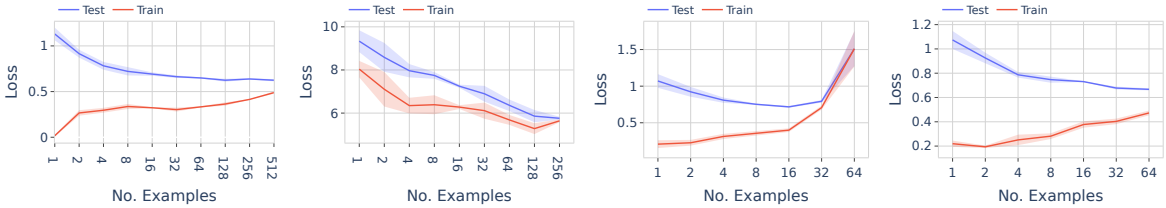


(p) Opt-1.3B (q) Opt-2.7B (r) Opt-6.7B

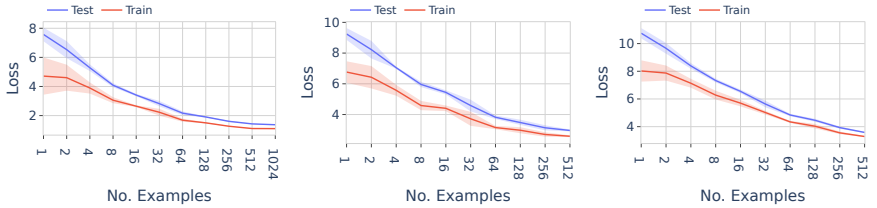
Figure 29: Testing the limit of utilizing ICL context (1536 examples $\approx$ 77K tokens) on language L_1. Training loss provides a lower bound of test loss in ICL. Long context LLMs cannot further improve from additional examples.



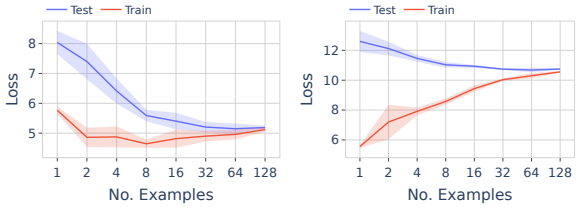
(a) Qwen-2.5-0.5B (b) Qwen-2.5-1.5B (c) Qwen-2.5-7B



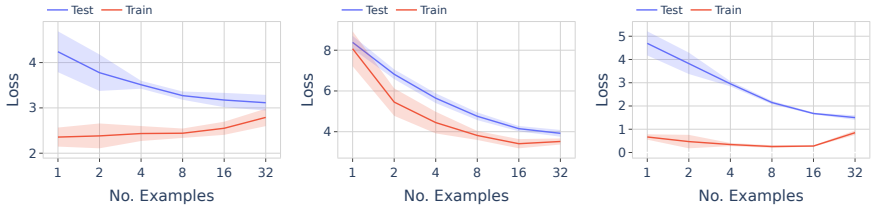
(d) Mistral-7B (e) Mistral-12B (f) Llama-2-7B (g) Llama-2-13B



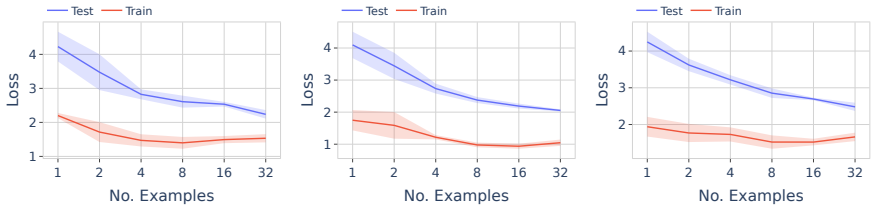
(h) Llama-3.2-1B (i) Llama-3.2-3B (j) Llama-3.1-8B



(k) Gemma-2-2B (l) Gemma-2-9B

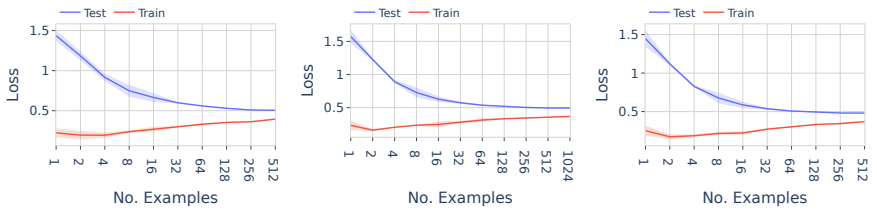


(m) Pythia-1B (n) Pythia-2.8B (o) Pythia-6.9B

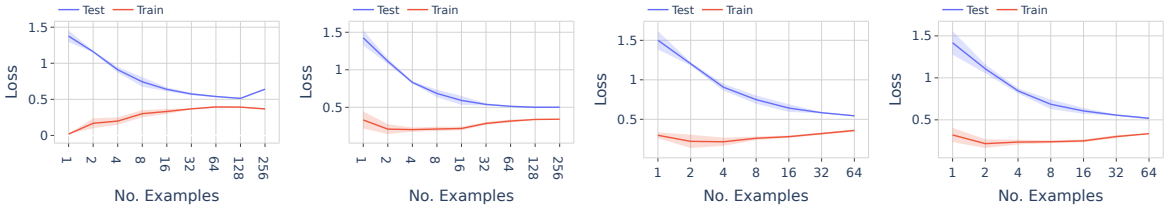


(p) Opt-1.3B (q) Opt-2.7B (r) Opt-6.7B

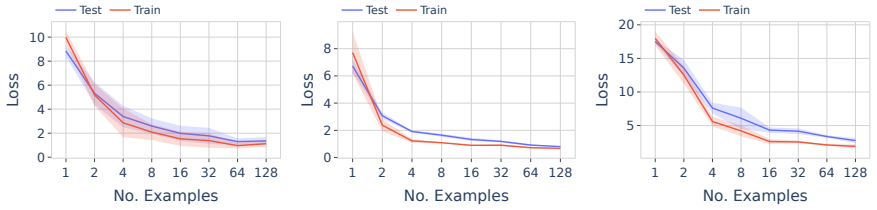
Figure 30: Testing the limit of utilizing ICL context (1536 examples $\approx$ 77K tokens) on language L_2. Training loss provides a lower bound of test loss in ICL. Long context LLMs cannot further improve from additional examples.



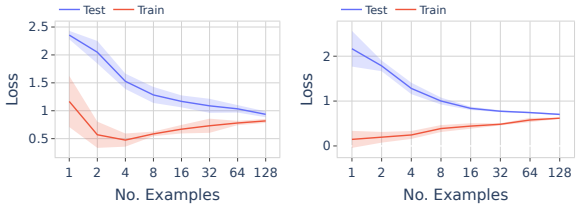
(a) Qwen-2.5-0.5B (b) Qwen-2.5-1.5B (c) Qwen-2.5-7B



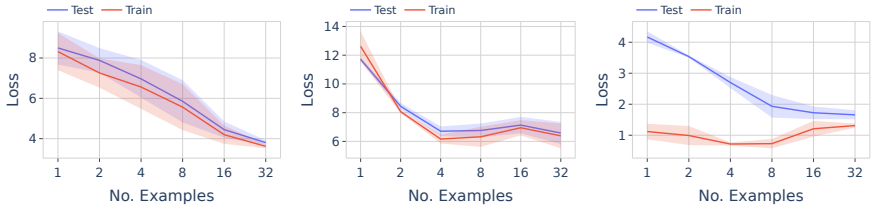
(d) Mistral-7B (e) Mistral-12B (f) Llama-2-7B (g) Llama-2-13B



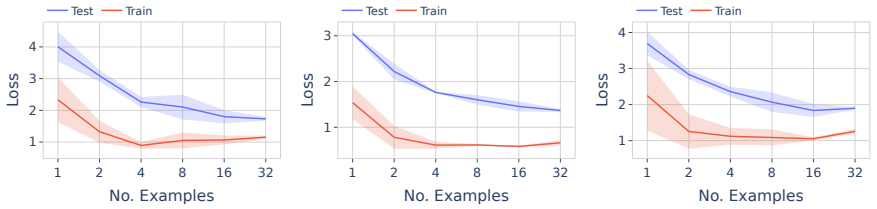
(h) Llama-3.2-1B (i) Llama-3.2-3B (j) Llama-3.1-8B



(k) Gemma-2-2B (l) Gemma-2-9B

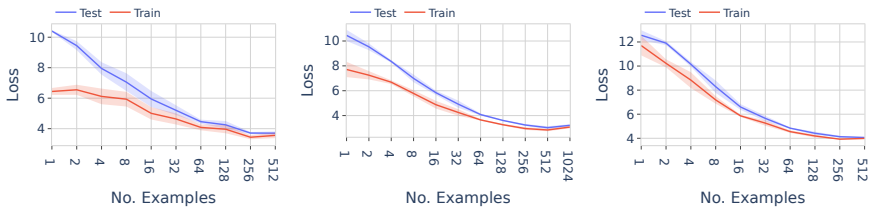


(m) Pythia-1B (n) Pythia-2.8B (o) Pythia-6.9B

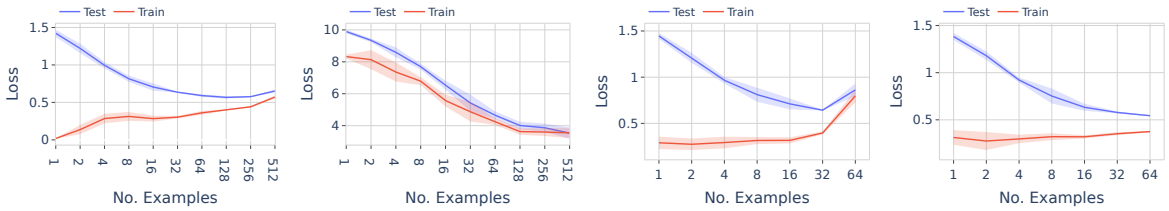


(p) Opt-1.3B (q) Opt-2.7B (r) Opt-6.7B

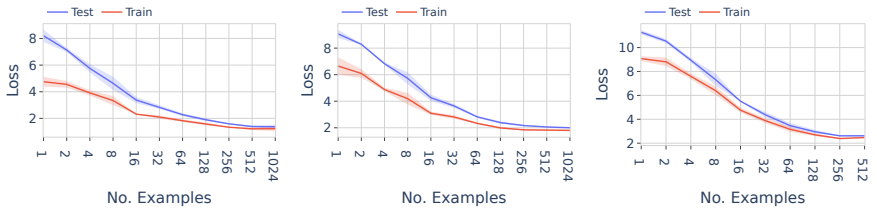
Figure 31: Testing the limit of utilizing ICL context (1536 examples $\approx$ 77K tokens) on language L_4. Training loss provides a lower bound of test loss in ICL. Long context LLMs cannot further improve from additional examples.



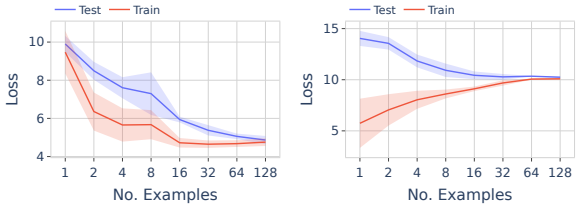
(a) Qwen-2.5-0.5B (b) Qwen-2.5-1.5B (c) Qwen-2.5-7B



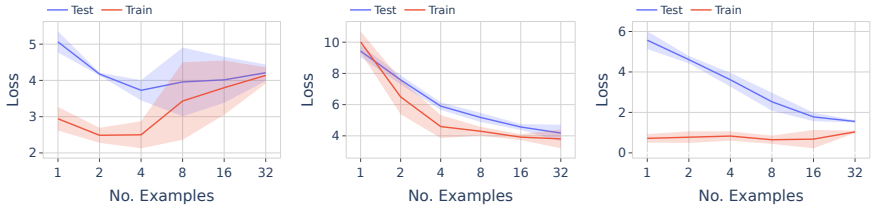
(d) Mistral-7B (e) Mistral-12B (f) Llama-2-7B (g) Llama-2-13B



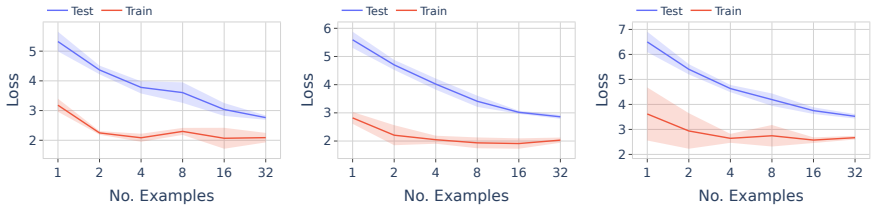
(h) Llama-3.2-1B (i) Llama-3.2-3B (j) Llama-3.1-8B



(k) Gemma-2-2B (l) Gemma-2-9B



(m) Pythia-1B (n) Pythia-2.8B (o) Pythia-6.9B



(p) Opt-1.3B (q) Opt-2.7B (r) Opt-6.7B

Figure 32: Testing the limit of utilizing ICL context (1536 examples $\approx$ 77K tokens) on language L_5. Training loss provides a lower bound of test loss in ICL. Long context LLMs cannot further improve from additional examples.

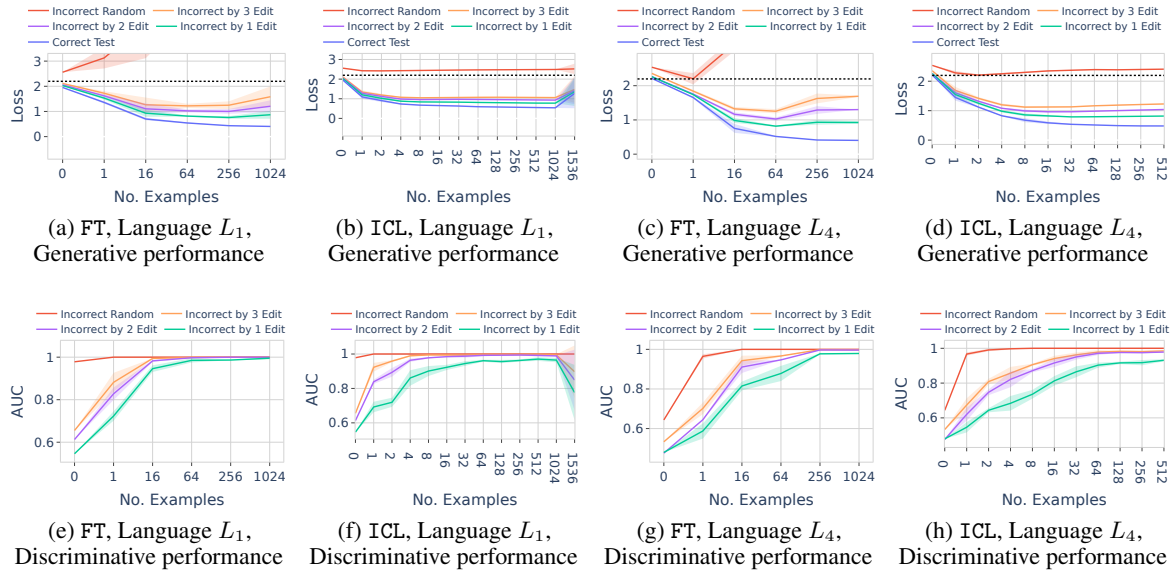
F Discriminative Test

Claim 1**.** *For a given language, the discriminative test is comparable between two learning modes of an LLM and across LLMs, unlike a generative test.*

We elaborate on the arguments of why the discriminative test is comparable between two learning modes of an LLM and across LLMs, but not the generative test. Concisely, the generative probability used in the generative test is not comparable across LLMs, but a classification score based on the differentiating generative probabilities on two different sets of strings is comparable across LLMs – the latter is the discriminative test.

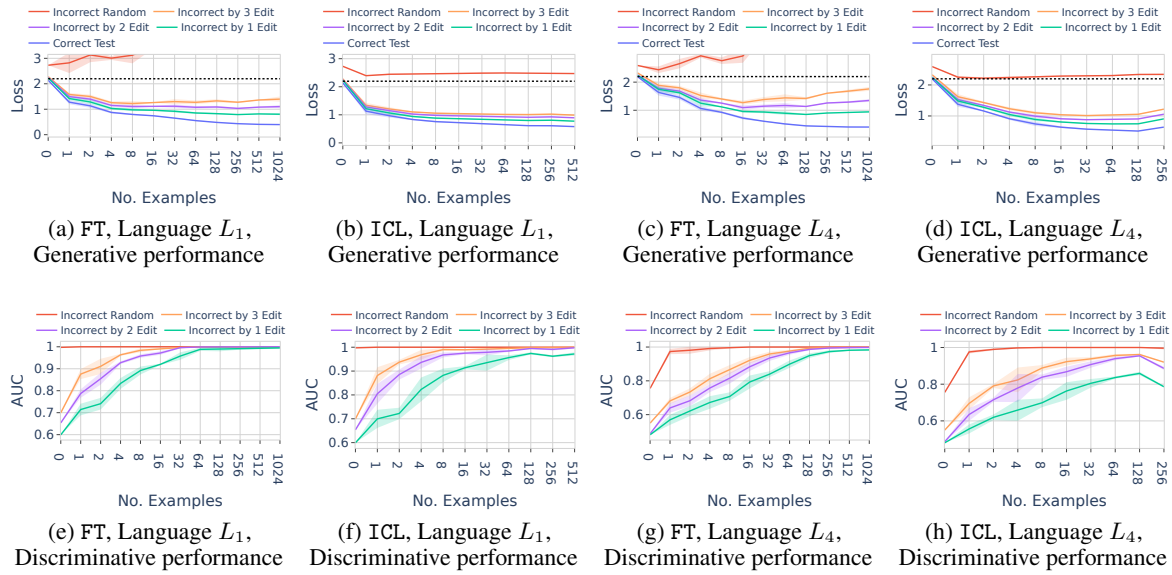
The discriminative test computes a classification score in $[0, 1]$ to determine how well strings in a language are discriminated from strings outside the language, based on generation probability given by the learning mode or the LLM. While computing generation probability, all in-language and out-language strings undergo the same input formatting, and are generated under the same parameters and hyperparameters of the LLM or learning mode. For example, the same concatenated prefix is applied to all strings in `ICL` versus null prefix in `FT` (see Figure 1). Finally, a learning mode or an LLM is more language proficient if the classification score is higher.

Since the learning performance is measured as a classification problem, the discriminative test is comparable, as long as the same set of in-language and out-language strings is used, and the same classification setup is applied.



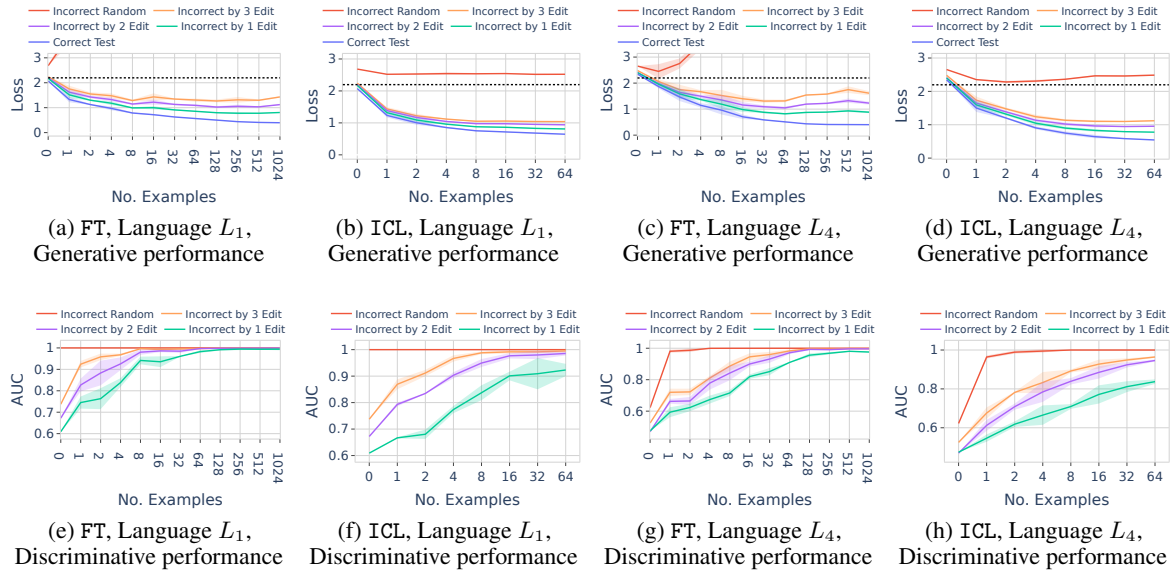
(a) FT, Language L_1, Generative performance
(b) ICL, Language L_1, Generative performance
(c) FT, Language L_4, Generative performance
(d) ICL, Language L_4, Generative performance
(e) FT, Language L_1, Discriminative performance
(f) ICL, Language L_1, Discriminative performance
(g) FT, Language L_4, Discriminative performance
(h) ICL, Language L_4, Discriminative performance

Figure 33: Qwen-2.5-7B: Language proficiency according to generative (first row) and discriminate (second row) tests. First two columns are for language L_1, and the last two columns are for language L_4.



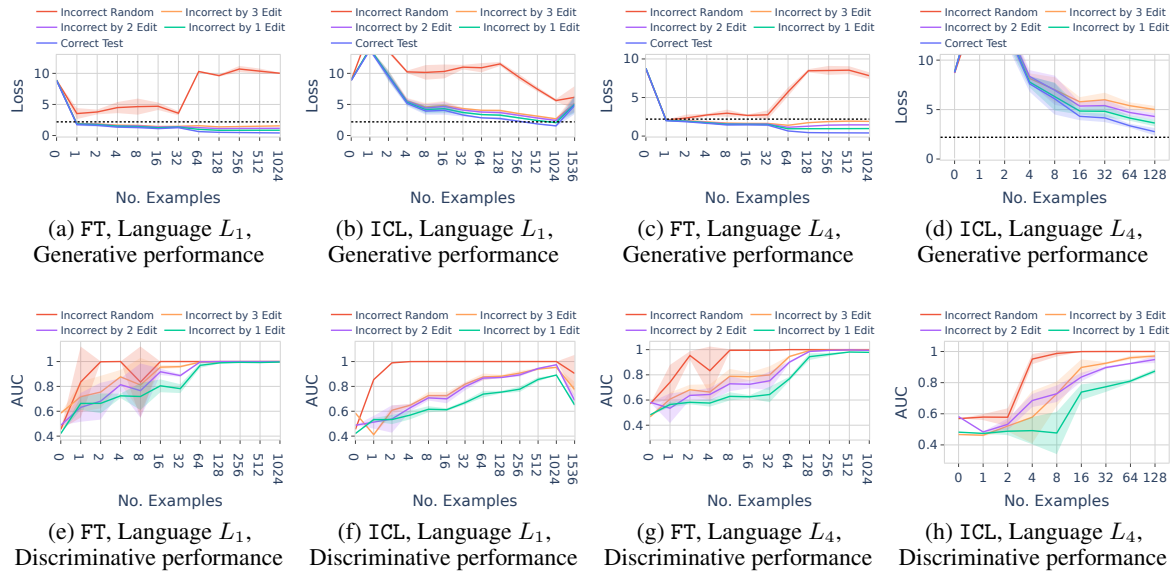
(a) FT, Language L_1, Generative performance
(b) ICL, Language L_1, Generative performance
(c) FT, Language L_4, Generative performance
(d) ICL, Language L_4, Generative performance
(e) FT, Language L_1, Discriminative performance
(f) ICL, Language L_1, Discriminative performance
(g) FT, Language L_4, Discriminative performance
(h) ICL, Language L_4, Discriminative performance

Figure 34: Mistral-7B: Language proficiency according to generative (first row) and discriminate (second row) tests. First two columns are for language L_1, and the last two columns are for language L_4.



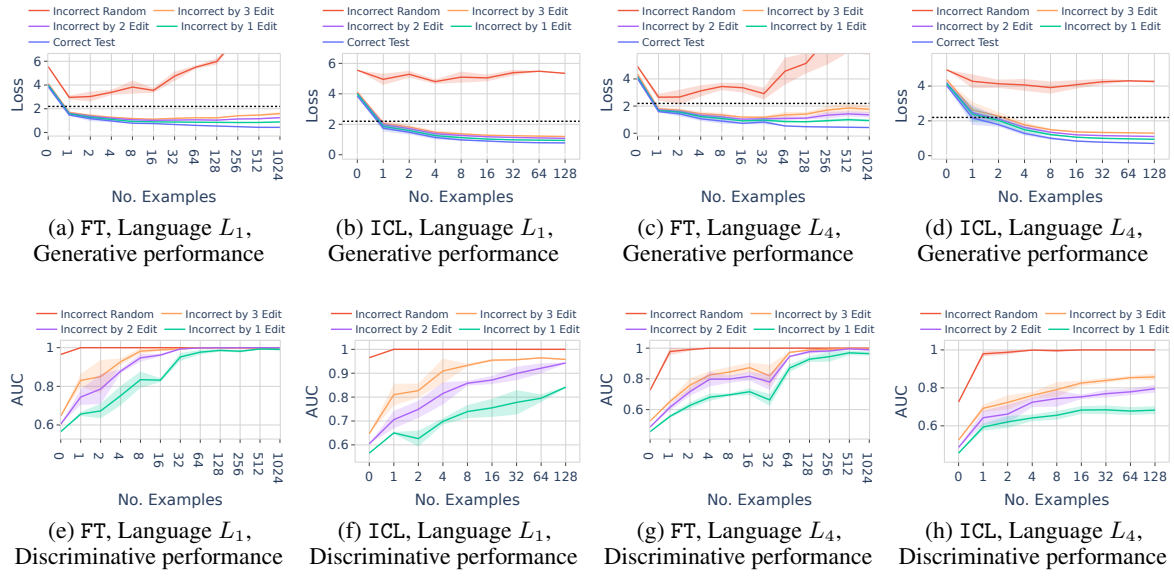
(a) FT, Language L_1, Generative performance
(b) ICL, Language L_1, Generative performance
(c) FT, Language L_4, Generative performance
(d) ICL, Language L_4, Generative performance
(e) FT, Language L_1, Discriminative performance
(f) ICL, Language L_1, Discriminative performance
(g) FT, Language L_4, Discriminative performance
(h) ICL, Language L_4, Discriminative performance

Figure 35: Llama-2-7B: Language proficiency according to generative (first row) and discriminate (second row) tests. First two columns are for language L_1, and the last two columns are for language L_4.



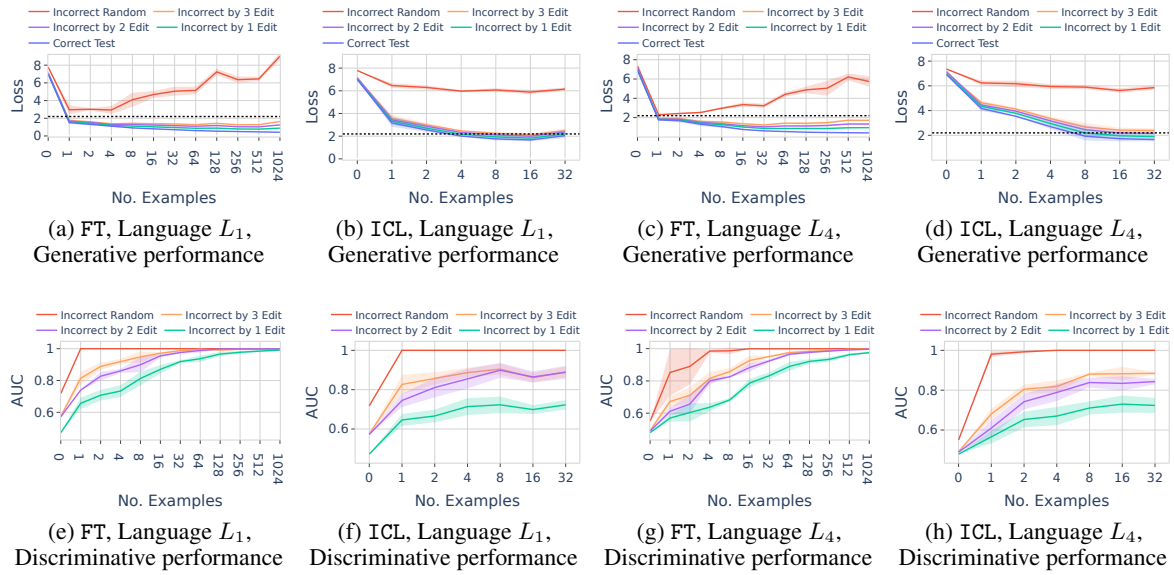
(a) FT, Language L_1, Generative performance
(b) ICL, Language L_1, Generative performance
(c) FT, Language L_4, Generative performance
(d) ICL, Language L_4, Generative performance
(e) FT, Language L_1, Discriminative performance
(f) ICL, Language L_1, Discriminative performance
(g) FT, Language L_4, Discriminative performance
(h) ICL, Language L_4, Discriminative performance

Figure 36: Llama-3.1-8B: Language proficiency according to generative (first row) and discriminate (second row) tests. First two columns are for language L_1, and the last two columns are for language L_4.



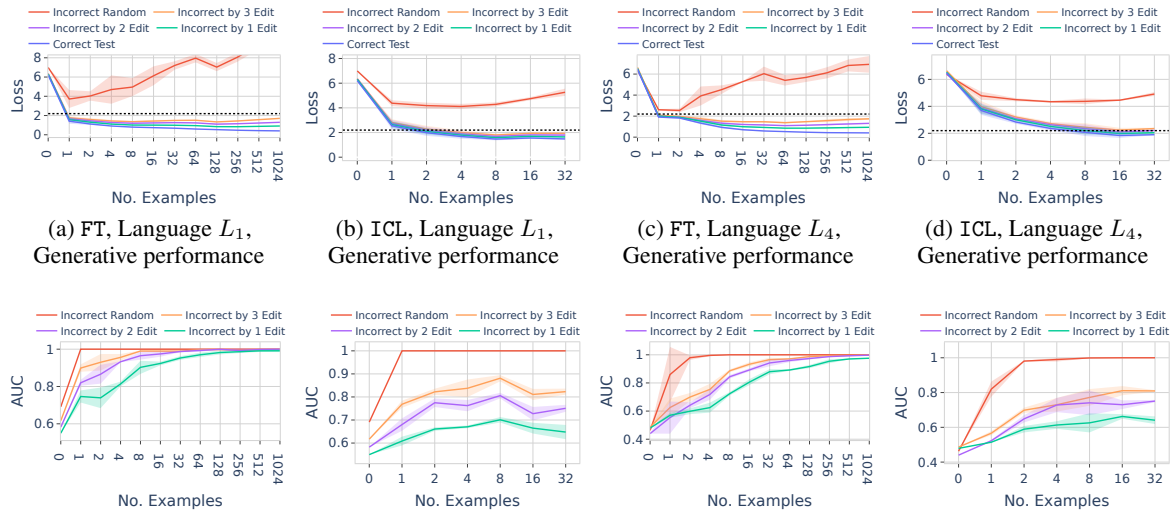
(a) FT, Language L_1, Generative performance
(b) ICL, Language L_1, Generative performance
(c) FT, Language L_4, Generative performance
(d) ICL, Language L_4, Generative performance
(e) FT, Language L_1, Discriminative performance
(f) ICL, Language L_1, Discriminative performance
(g) FT, Language L_4, Discriminative performance
(h) ICL, Language L_4, Discriminative performance

Figure 37: Gemma-2-9B: Language proficiency according to generative (first row) and discriminate (second row) tests. First two columns are for language L_1, and the last two columns are for language L_4.



(a) FT, Language L_1, Generative performance
(b) ICL, Language L_1, Generative performance
(c) FT, Language L_4, Generative performance
(d) ICL, Language L_4, Generative performance
(e) FT, Language L_1, Discriminative performance
(f) ICL, Language L_1, Discriminative performance
(g) FT, Language L_4, Discriminative performance
(h) ICL, Language L_4, Discriminative performance

Figure 38: Pythia-6.9B: Language proficiency according to generative (first row) and discriminate (second row) tests. First two columns are for language L_1, and the last two columns are for language L_4.



(a) FT, Language L_1, Generative performance

(b) ICL, Language L_1, Generative performance

(c) FT, Language L_4, Generative performance

(d) ICL, Language L_4, Generative performance

(e) FT, Language L_1, Discriminative performance

(f) ICL, Language L_1, Discriminative performance

(g) FT, Language L_4, Discriminative performance

(h) ICL, Language L_4, Discriminative performance

Figure 39: Opt-6.7B: Language proficiency according to generative (first row) and discriminate (second row) tests. First two columns are for language L_1, and the last two columns are for language L_4.

G Implications of the Study

We elaborate on the implications of our findings of four research questions in Section 5. We provide our hypothesis for each finding, which may inspire future research.

- In **RQ1**, when learning a language, FT performance converges across LLMs but ICL performance is variable. Our hypothesis is that FT is a direct form of learning, where parameters are explicitly updated. Since all LLMs are fine-tuned optimally, and the considered language is simple and has a hierarchical recursive structure, FT performance converges across LLMs.

  ICL, however, is an indirect form of learning, where the model learns patterns from the context without any parameter update. Hence, ICL performance can be biased by model-specific pre-training, which may differ across LLMs of different sizes and families. As such, ICL performance is variable across LLMs. A more subtle analysis is given below in **RQ4**.

- In **RQ2**, FT is superior to ICL when training and test languages are the same, i.e., in-distribution generalization, but both modes perform equally and only generalize to closer out-languages in out-of-distribution generalization. Therefore, if the test language is different, FT is no longer the better mode, and explicit parameter update in FT does not help. In this case, ICL is a better choice, since the general language understanding of the original model is retained in ICL, compared to FT where the model is explicitly – in the case of out-of-language generalization, unnecessarily – updated.

- In **RQ3**, the inductive bias of FT and ICL are similar, but similarity decreases with more training examples. The similarity of inductive bias is computed as the Pearson correlation of generation loss (or probability) of FT and ICL on identical test strings. Informally, when more examples are provided, the learning mode becomes confident in generating the language, specifically the individual strings from the language. As such, the variance of per-string loss is expected to decrease for a set of strings. We argue that when the range of loss is reduced, FT and ICL differ more on the loss of individual strings, and hence correlation decreases.

- In **RQ4**, ICL is less robust than FT across languages. This is perhaps explained by the hypothesis in **RQ1**, where FT explicitly updates parameters, while ICL does not. Moreover, the sensitivity of ICL performance on actual tokens used in the language suggests a dependency of ICL performance on pre-training, where the same token sets can be trained differently across different LLMs.

- We emphasize the adoption of the discriminative test for evaluating language proficiency in LLMs, across formal and natural languages. The discriminative test ensures that generation of in-language strings is better and even separable from the generation of out-language strings, which is a stronger condition than the generative test.

  For future work on the adoption of the discriminative test, one needs to systematically generate strings outside the language, which we have shown for formal language in Section 3, and an instance of natural language, such as sentiment classification, in Appendix D. Since natural language is less well-defined than formal language, the boundary of in-language and out-language strings may be superficial in natural language, demanding a careful study.