Assessing the Limits of In-Context Learning beyond Functions using Partially Ordered Relation

Debanjan Dutta, Faizanuddin Ansari, Swagatam Das
Electronics and Communication Sciences Unit
Indian Statistical Institute, Kolkata
{debanjandutta_r, faizanuddin_r, swagatam.das}@isical.ac.in

Abstract

Generating rational and generally accurate responses to tasks, often accompanied by example demonstrations, highlights Large Language Model's (LLM's) remarkable In-Context Learning (ICL) capabilities without requiring updates to the model's parameter space. Despite having an ongoing exploration focused on the inference from a document-level concept, its behavior in learning well-defined functions or relations in context needs careful investigation. In this article, we present the performance of ICL on partially ordered relation by introducing the notion of inductively increasing complexity in prompts. In most cases, the saturated performance of the chosen metric indicates that while ICL offers some benefits, its effectiveness remains constrained as we increase the complexity in the prompts, even in the presence of sufficient demonstrative examples. The behavior is evident from our empirical findings and has further been theoretically justified in terms of its implicit optimization process. The code is available here.[1]

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in in-context learning (ICL) (Brown et al., 2020), where tasks are learned from few-shot demonstrations without parameter updates. While ICL has spurred extensive research dealing with applications spanning language understanding, code generation, and mathematical reasoning (Ahn et al., 2023; Cheng et al., 2024), its ability to reason about relational structures—particularly partially ordered sets (posets)—remains unexplored. Posets are fundamental to mathematics and real-world hierarchies. For example, genealogical relations such as one in the Mahabharata (Bedobyas) – `Bashistha was the ancestor of Shaktri, Shaktri was the ancestor of Parasar, Parasar was the ancestor of Krishno-dwaipayan Bedobyas` – is an example of linear order. Their mathematical breadth is further illustrated by a relation say *inclusion* ($\subseteq$). Zorn's lemma (Zorn, 1935), one of the widely used results to prove several fundamental theorems in algebra, finds a proof involving posets. This exhibitory relevance of posets, thus demands investigation on reasoning about transitivity, antisymmetry, and dependencies beyond function-like mappings.

Existing works on ICL have primarily explored linear functions (Garg et al., 2022), regular languages (Akyürek et al., 2024), and discrete-valued functions (Bhattamishra et al., 2024). While these studies reveal insights into model adaptability, they overlook the challenges posed by relational reasoning, where outputs for a single image, unlike functions, may involve multiple pre-images. Along this line of study on exploring ICL's behavior on algorithms (Veličković et al., 2022), functions, and even long-context NLP tasks (Bertsch et al., 2025), investigating their behavior on structured mathematical tasks remains understudied. This research gap is critical for posets, which provide a rigorous framework for evaluating reasoning in hierarchical scenarios. Yet, it is unclear whether LLMs can infer such relations through ICL, especially as task complexity scales. These challenges form the primary motivation for our research.

By leveraging posets, we propose a novel evaluation framework using k-shot c-complex prompts, where k denotes the number of demonstrations and c quantifies complexity through incremental extensions of Hasse diagrams (see Fig. 1). Our experiments span different open-source LLMs and models from GPT family, evaluating their ability to infer relations on posets: the *less than* $(\mathbb{N}, <)$ and the *divisibility* $(\mathbb{N}, |)$. The exponential growth in vertices of the Hasse diagram of the poset $(2^S, \subseteq)$, for any finite set S has prevented us from includ-

[1] https://github.com/DDuttaGit/ICLonPartialOrder

Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 900–918
December 20-24, 2025

ing them in the study. We further fine-tune models (e.g., BERT, RoBERTa) to evaluate performance when enriched with data-specific knowledge.

Key contributions are as follows: (1) We, for the first time, investigate the **performance saturation** of ICL on posets when evaluated under the proposed framework (Section 3) involving k-shot c-complex prompts. Through a series of extensive experiments (Section 4), we present that while LLMs can partially encode posets during ICL, their reasoning is constrained by architectural limitations, particularly in capturing transitivity and anti-symmetry. (2) We have theoretically justified this bounded approximation capacity of ICL in terms of its implicit optimization process (Section 5). This becomes evident through the analysis of LLMs' output when viewed as *task vectors* (Section 6).

2 Revisiting the ICL Landscape

Recent studies have established a strong theoretical basis for ICL by analyzing its connection to gradient-based optimization. Building on Ahn et al. (2023)'s work on linear Transformers performing in-context gradient descent, Cheng et al. (2024) demonstrated similar behavior in non-linear Transformers under specific activation conditions. Furthermore, Dai et al. (2023) empirically validated the duality between Transformer attention and gradient descent, showing in-context updates mimic implicit fine-tuning, particularly when leveraging momentum-based attention.

In the context of analyzing ICL on functions from discrete space: Bhattamishra et al. (2024) found frozen-weight GPT-2 models showed non-trivial performance on discrete functions like nearest neighbor. Akyürek et al. (2024) highlighted in-context (regular) language learning excels via implicit implementation of n-gram statistics. Despite having theoretical universal simulation power (Qiu et al., 2025; Furuya et al., 2025), Bertsch et al. (2025) noted diminishing gains in long context ICL performance in traditional NLP tasks, suggesting fundamental limitations in contextual inference. Moreover, Akyürek et al. (2023) and Guo et al. (2024) demonstrated that ICL effectiveness is highly dependent on the properties of the underlying function, prompting the question of how LLMs perform when dealing with relations and partial functions rather than well-structured functions. By analyzing poset, we extend ICL research into relational reasoning, filling up a crucial gap in understanding how language models infer dependencies beyond standard function learning.

3 Background and Setup for ICL

3.1 Partially Ordered Set & Hasse Diagram

Poset: On a set S, a relation $\preceq$ is said to produce a partial-ordered set or poset $(S, \preceq)$, if it follows:

Reflexivity For every element $s \in S$, $s \preceq s$,

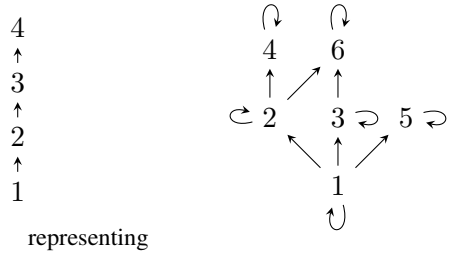
Anti-symmetry For any two elements $s_1, s_2 \in S$, if $s_1 \preceq s_2$ and $s_2 \preceq s_1$, $s_1 = s_2$ and

Transitivity For three elements s_1, s_2 and $s_3 \in S$ if $s_1 \preceq s_2$ and $s_2 \preceq s_3$, $s_1 \preceq s_3$.

A poset, in the absence of the reflexivity property, denoted by $(S, \prec)$, is called a linear order if any two distinct elements $i, j \in S$ are comparable i.e. either $i \prec j$ or $j \prec i$. Our work comprises study on two posets – the *less than* relation on the set of natural numbers $(\mathbb{N}, <)$, representative of any linear order, and the *divides* relation on the set of natural numbers $(\mathbb{N}, |)$, representing any other arbitrary poset.

Observation 1. The linear order on the numerical domain, such as $(\mathbb{N}, <)$ can be viewed as length-dependent lexicography order. Suppose $\Sigma_{\mathbb{N}} = \{1, 2, \ldots, 9\} \cup \{0\}$ denotes the alphabet of $\mathbb{N}$. Then a natural word $x = a_1 a_2 \ldots a_n$ of length n belongs to $\Sigma_{\mathbb{N}}^n$ if and only if $a_i \in \Sigma_{\mathbb{N}}$. Suppose the lexicography order on $\Sigma_{\mathbb{N}}$ is i) the linear order $(0, 1, 2, \ldots, 9)$ and ii) x precedes x' for any natural word $x \in \Sigma_{\mathbb{N}}^i$ and $x' \in \Sigma_{\mathbb{N}}^{i+k}$ where $k > 0$. Then length-dependent lexicography order on $\Sigma_{\mathbb{N}}$ is the linear order $(\mathbb{N}, <)$.

Hasse Diagram: Given a poset $(S, \preceq)$, a Hasse



(a) Hasse representing $(\{1, \ldots, 4\}, <)$ or the linear order $(1, 2, 3, 4)$.

(b) Hasse representing $(\{1, \ldots, 6\}, |)$.

Figure 1: Examples of Hasse diagrams.

diagram (e.g. Fig: 1) is a directed acyclic graph (DAG) $H = (S, E)$ where $E \subset S \times S$ represents the directed edge (a, b) such that $a \preceq b$ and there exists no c such that $a \preceq c \preceq b$ when $a \neq b$ and self-loop in case of $a = b$ depending on reflexivity (or, irreflexivity). The Hasse diagram of the relation *less than* is a directed path. Suppose not, then

$S \rightarrow IET$

$I \rightarrow \Sigma R$

$R \rightarrow Reflexivity\ R' \mid Irreflexivity\ R'$

$R' \rightarrow Anti\text{-}Symmetry\ Transitivity$

$E \rightarrow a \preceq b,\ E \mid a \preceq b$ for $a, b \in \Sigma^+$ and $a \preceq b$

$T \rightarrow DT'$

$D \rightarrow$ Output True or False to the following case(s)

$T' \rightarrow a \preceq b, T' \mid a \preceq b$ for $a, b \in \Sigma^+$.

Figure 2: Grammar for production of a prompt. Illustrative prompts have been provided in Appendix A.

for any three distinct integer a, b_1, and b_2, there exist at least two paths, say from a to b_1 and a to b_2 such that b_1 and b_2 are not connected in either direction. Then, by definition of Hasse b_1 and b_2 are incomparable, contradicting the basic rule that with any two different integers, one is smaller than the other. The Hasse diagram of the relation *divides*, on the other hand, can contain such b_1 and b_2, representing any arbitrary DAG.

3.2 Formalizing k-Shot c-Complex Prompt

Let's revisit the existing definition of ICL (Garg et al., 2022) in relevance to our experimental context. Assuming Σ represents the alphabet in the poset, the grammar producing a prompt P has been provided as Backus–Naur form in Figure 2. A prompt is a concatenation of instruction (I), examples (E), and task (T). The instruction provides information about the alphabet in the poset and definition of anti-symmetry, transitivity, and/or reflexivity depending on the nature of the relation. The task describes the output format and test point(s).

A prompt is generally termed as a k-shot prompt P_k if the non-terminal E has expanded k times. Further, we call P_k *minimal* if the examples in P_k are the pairs $a \preceq b$ such that (a, b) are edges in the respective Hasse diagram. This can be considered aligned with the concept of a *teaching sequence* (Goldman and Kearns, 1995). The approach constructs a sequence $\{x, f(x)\}$ (here, a relation) capable of uniquely identifying a function f within a given function class. Minimal prompts were carefully designed to ensure interpretability and compactness while retaining task specificity. Thus, the examples combined with the instruction segment of a prompt P_k can provide information about the entire poset $(\{1, 2, \ldots, k\}, \preceq)$.

Xu et al. (2024) proved the existence of a (formal) language that cannot be produced by any iterative learning algorithms such as neural language models. We extend the experiments to understand at what extent ICL can remedy this problem. Given any existing language model $\mathcal{L}$, a problem class Q, and a problem $q \in Q$, *bad* for $\mathcal{L}$; we ask to what extent the actual answer of q deviates from the answer generated by $\mathcal{L}$.

Given some knowledge about a problem class Q, suppose Q is enumerable in terms of its increasing *complexity* with respect to a problem $q' \in Q$ and say $q \in Q$ is the ith instance for which a neural language model $\mathcal{L}$ fails to generate a correct answer. The term complexity here denotes the difficulty in accurately representing q, given the existing knowledge about class Q, the known problem q', and the inherent knowledge of $\mathcal{L}$. Then intuitively, the $i + 1, \ldots$ problems would not only be mistaken by $\mathcal{L}$ but the mistake would get *severe* as we go right. We discovered that this intuition does not hold true as ICL induces different outcomes. Despite being restricted to poset, with a suitable definition of complexity, the experiment can be reproduced analogously for suitable Q.

Now we define a k-shot c-complex prompt $P_{k,c}$. It is a minimal k-shot prompt P_k on a poset $(\{1, 2, \ldots, k\}, \preceq)$ which provides task T' that is c-complex, meaning T' provides evaluation points $a \preceq b$ such that the edge (a, b) or (b, a) is not present in the Hasse diagram of the poset and at least one of $a, b \in \{k + 1, k + 2, \ldots, k + c\}$.The rationale behind considering the test point production rule T' is – had T' been defined with $a \preceq b$ alone, the evaluation of $\mathcal{L}(P_{k,c})$ would lack robustness. Therefore, a sequence of c-complex prompts, $P_{\rightarrow,c}$[2], enhances the understanding of the provided data and is expected to yield improved performance. On the other hand, a sequence of k-shot prompts $P_{k,\rightarrow}$ progressively increases the task's difficulty.

3.3 Experimental Setup & Evaluation Metric

During this experiment, we chose to keep the cardinality of evaluation points, $|P_{k,c}[T']|$, as 50 and 30 for the tasks *less than* and *divides* respectively. For example, consider the prompt $P_{20,10}$ in task *less than*. There are exactly 19 demonstrations given by $1 < 2, 2 < 3, \ldots, 19 < 20$ and 50 random evaluation points (a, b) from $\{1 \ldots, 30\} \times \{21, \ldots, 30\}$ or the reversed order such that the pairs are not present in the demonstration. If $a \preceq b$ is the ith evaluation point, $P_{k,c}[T'][i]$, we say its target value $V(P_{k,c}[T'][i])$ is true if there is a path from a to b

[2]The notation denotes the sequence $\{P_{k,c}\}_{k=\{1,2,\ldots\}}$.

in the Hasse diagram or false, otherwise. Then the accuracy of a generated response corresponding to a prompt $P_{k,c}$ is measured by

$$\frac{1}{|P_{k,c}[T']|}\sum_{i}\mathbb{I}\left(\mathcal{L}(P_{k,c})[i], V(P_{k,c}[T'][i])\right),$$

where, $\mathbb{I}(a,b)$ yields 1, when $a=b$ and 0, otherwise. Similarly, the mean cumulative accuracy, say, for a sequence of c-complex prompts, is given by

$$\frac{1}{\sum\limits_{\substack{c\leq c\\ \text{for all } k}}|P_{k,c}[T']|}\sum_{\substack{c\leq c\\ \text{for all } k}}\sum_{i}\mathbb{I}\left(\mathcal{L}(P_{k,c})[i], V(P_{k,c}[T'][i])\right).$$

A similar evaluation metric can also be incorporated for varying demonstrative examples, keeping the complexity parameter the same on average.

Models and Plots: We have employed five decoder based models, viz Gemma2, Llama3, Mathstral, Qwen2-math, Phi3 and two prevalent models from GPT family viz GPT-3.5-Turbo and GPT-4.o-mini for ICL based experiments while for fine-tuning we have employed two encoder-only transformer architectures, BERT and RoBERTa and an auto-regressive architecture XLNet, an improvement over BERT (for model details see Appendix Table 1). Despite belonging to decoder-based category, the well-known architectures exhibit varying performance on the same dataset, hence selecting them to assess their efficiency. We have plotted mean cumulative accuracy for the sequence $\{\mathcal{L}(P_{k,c})\}_{k=\{1,2,\ldots,K\}}$ against varying complexities, for some K. It denotes that we are plotting mean cumulative accuracy for $\{\mathcal{L}(P_{k,1})\}_{k=\{1,\ldots,K\}}, \{\mathcal{L}(P_{k,2})\}_{k=\{1,\ldots,K\}}, \cdots$ This implies maintaining a similar number of demonstrations on average, we are increasing complexity as we go right.

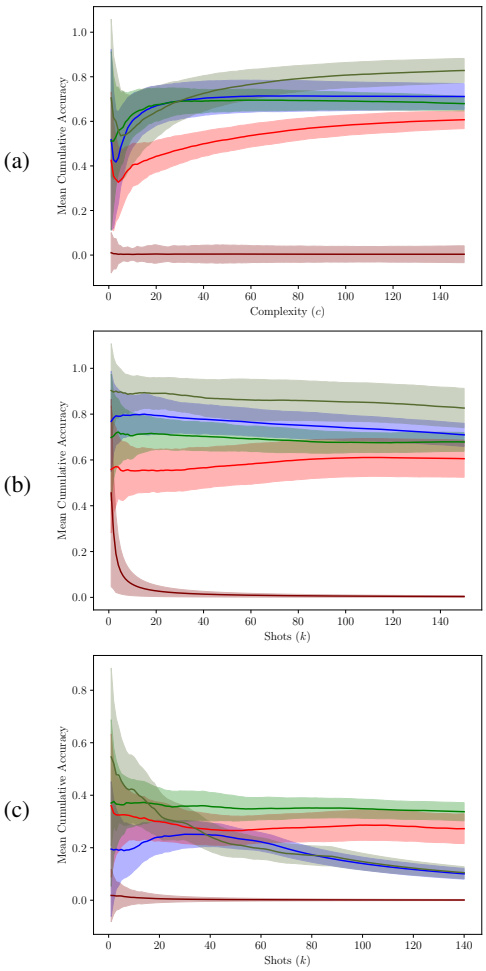


Figure 3: Subfigure 3a describes the mean cumulative accuracy plot for the sequence $\{\mathcal{L}(P_{k,c})\}_{k=\{1,2,\ldots,150\}}$ against varying complexities. Subfigure 3b describes the same metric for $\{\mathcal{L}(P_{k,c})\}_{c=\{1,2,\ldots,150\}}$ against varying shots. Similarly, subfigure 3c represents the mean cumulative accuracy curve for linear order $(\{0,1\}^*, <)$. The color scheme of the subfigures 3a-3c depicts the behavior corresponding to the models Gemma2, Lamma3, Mathstral, Qwen2-math, and Phi3.

4 In-Context Learning on Poset

We present the first analysis of ICL for poset reasoning through systematic evaluation across: (1) linear orders $(\mathbb{N}, <)$, (2) its binary variant $(\{0,1\}^*, <)$ (testing length-aware lexicographic reasoning), and (3) divisibility posets $(\mathbb{N}, |)$ (challenging DAG-based transitive inference). Performance of open-source language models on the *linear order* and *divisibility* posets in subsections 4.1 and 4.2, respectively, keeping analysis on the GPT models separate in subsection 4.3.

4.1 Linear Order

We have experimented with ICL on linear order using posets $(\mathbb{N}, <)$ and $(\{0,1\}^*, <)$, on a collection of k-shot, c-complex prompts, $k, c \leq 150$. Figure 3a and 3b depict the performance of the language models on poset $(\mathbb{N}, <)$. Irrespective of the size of language models, the majority of them (except Qwen) failed to learn the pattern successfully.

Figure 3a indicates that maintaining a fixed knowledge representation, on average, does not impact ICL capabilities as complexity increases, and it remains unchanged over sufficiently complex regions for models – Gemma2, Llama3 and Mathstral, though Qwen2-math has shown a non-trivial performance increase. Similarly, Figure 3b demonstrates that increasing knowledge has no observable effect on learning performance. In fact, we see either stagnant behavior (after 115 in case

of Llama3 and Mathstral) or a mildly decaying accuracy (in case of Gemma2 and Qwen2-math).

We performed similar experiments on its binary variant. The motivation behind reducing the alphabet from $\{0, 1, \ldots, 9\}$ to $\{0, 1\}$ is to validate observation 1. By using length-dependent lexicography, achieved by converting integers to their binary equivalents, we observed certain key characteristics while analyzing the sequence $\{\mathcal{L}(P_{k,c})\}_{c=\{1,2,\ldots,150\}}$ against varying shots, as shown in Figure 3c. Although both Llama3 and Mathstral reach saturation despite having a poor performance compared to $(\mathbb{N}, <)$, both Gemma2 and Qwen2-math demonstrate a prominent decline in mean cumulative accuracy, which appeared milder in the case of $(\mathbb{N}, <)$. However, a sharp performance gap across models for tasks $(\mathbb{N}, <)$ and $(\{0, 1\}^*, <)$ (Fig. 3b, 3c) underscores their failure to generalize lexicography. The trivial performance of Phi3 confirms its incapability of processing minimal prompts for linear orders when we maintain an identical experimental setup for each model.

4.2 Division

In contrast to linear order, the poset $(\mathbb{N}, |)$ represents a more general class of partial order in the sense that the Hasse diagram of $(\mathbb{N}, |)$ is a general directed acyclic graph having source vertex 1 and integers $n = \prod p_i^{q_i}$, where p_i are primes and $q_i > 0$, as nodes such that any n has edges from n/p_i. In our experiment, as we progressed to 150-shot 150-complex prompts, the ICL is expected to construct incrementally a Hasse diagram of height at most $1 + \lfloor \log_2 300 \rfloor$.

Considering $|P_{k,c}[T']| = 30$, except some pathological cases, Figure 4a and 4b analogously describe the mean cumulative accuracy plot for the sequence $\{\mathcal{L}(P_{k,c})\}_{k=\{1,2,\ldots,150\}}$ against varying complexities and for the sequence $\{\mathcal{L}(P_{k,c})\}_{c=\{1,2,\ldots,150\}}$ against varying shots, respectively. The former describes when the knowledge remains constant on average, increasing complexity has no observable effect on ICL for Llama3, Mathstral, Qwen2-math and Phi3. However, Gemma2 demonstrates a declining learning trend under the same conditions. Interestingly, as knowledge accumulation increases, as illustrated in Figure 4b, ICL yields a notable performance enhancement in the Llama3, Mathstral and Phi3 models, though only within a limited region. The saturation in learning can be observed in both Mathstral and Gemma2. The abrupt descent in performance for

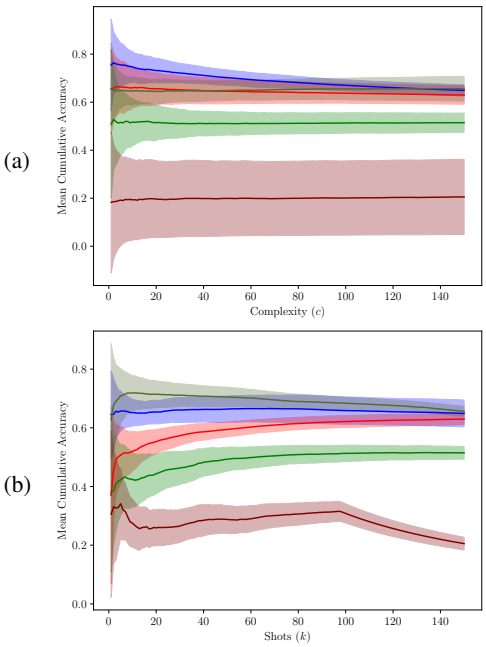


Figure 4: Subfigure 4a depicts the mean cumulative accuracy plot for the sequence $\{\mathcal{L}(P_{k,c})\}_{k=\{1,2,\ldots,150\}}$ against varying complexities. Subfigure 4b, similarly, describes the same metric for $\{\mathcal{L}(P_{k,c})\}_{c=\{1,2,\ldots,150\}}$ against varying shots. Similar to Figure 3, the color scheme depicts the behavior corresponding to the models Gemma2, Lamma3, Mathstral, Qwen2-math, and Phi3.

Phi3 is due to its inability to encode context with a window of size 4096, which we keep identical for evaluation across all the open-source models.

On fine-tuning. For both the posets, performance on fine-tuning have been presented in Figure 8 (in Appendix A). We found that while all three models eventually attain saturation, RoBERTa adapts more efficently than BERT and XL-Net.

4.3 Performance on GPT Family

We have extended our experiments towards verifying the saturating nature of the evaluation metric by employing LLM such as GPT-3.5-Turbo having context-window 16,385. Instead of performing on the whole dataset, we have constrained the experiments to 75-shot 75-complex prompts. Figure 5a shows that after initial fluctuations ($c \leq 30$), performance with prompts from $(\{0, 1\}^*, <)$ stabilizes and variation decreases as plot width narrows. A similar trend has also been observed in $(\mathbb{N}, |)$, except the fact that there is no observable fluctuation even in the beginning. However, for the task $(\mathbb{N}, <)$, the model excelled irrespective of the complexity. The abrupt decrease in performance for some initial points resembles $(\{0, 1\}^*, <)$. Now

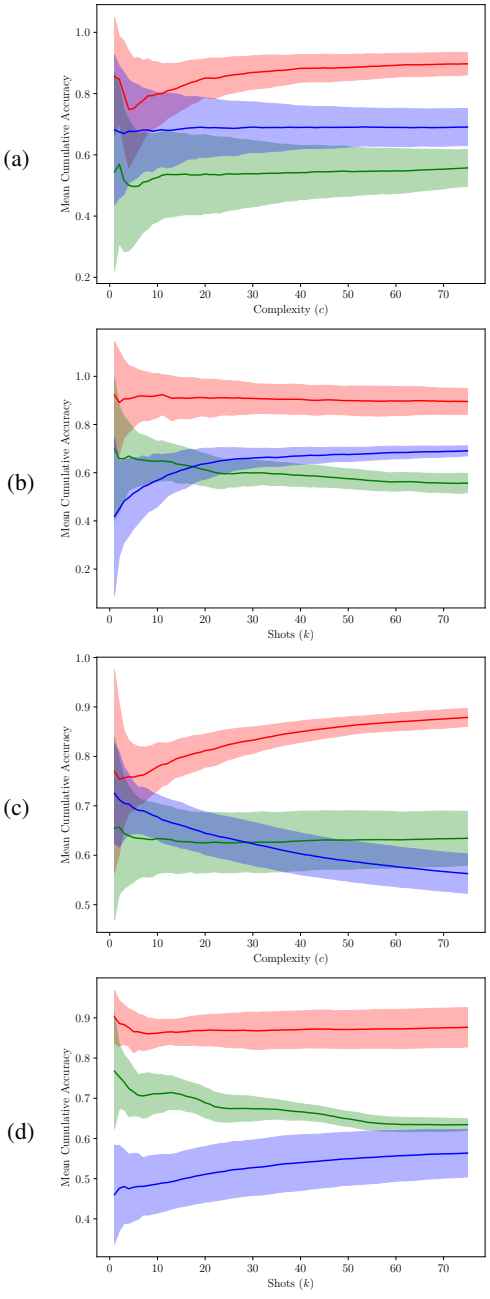


Figure 5: Subfigure 5a and 5c plots the mean cumulative accuracy (on GPT-3.5-Turbo and GPT-4.o-mini respectively) for the sequence $\{\mathcal{L}(P_{k,c})\}_{k=\{1,2,\ldots,75\}}$ against varying complexities for posets *less than* $((\mathbb{N},<),(\{0,1\}^*,<))$ and *divides* $((\mathbb{N},|)$. Subfigure 5b and 5d similarly describes the same metric for $\{\mathcal{L}(P_{k,c})\}_{c=\{1,2,\ldots,75\}}$ against varying shots. The color scheme of the above subfigures corresponds to the behavior for posets $(\mathbb{N},<)$, $(\{0,1\}^*,<)$, and $(\mathbb{N},|)$.

keeping the complexity constant on average and increasing the knowledge, for task $(\{0,1\}^*,<)$ (subfigure 5b), a slightly better performance can be observed compared to subfigure 5a, but it deteriorates, however, with decreasing slope. Interestingly, the behavior in $(\mathbb{N},|)$ shows a sharp increase till $k \leq 40$ and stabilizes afterward. Even though $(\mathbb{N},<)$ has performed significantly well from the beginning, its performance did not improve with increasing knowledge. Because we have conducted our experiment on a compressed dataset, we anticipate a gradual saturation with increasing examples.

GPT4.o-mini with context window of 128k tokens presents an interesting behavior when we consider the experiment on poset $(\mathbb{N},|)$. Figure 5c shows mean cumulative accuracy decreases rapidly with increasing complexity till $c = 45$ and its descent becomes gradual afterwards – showing its incompetence to generalize the representation of arbitrary DAGs. In contrast, Figure 5d exhibits a slight increase in performance with increasing knowledge. Even though a perfect saturation could not be observed with compressed data $\{P_{k,c}\}_{k,c=\{1,2,\ldots,75\}}$, the subfigures exhibit diminishing descent or gains respectively. On poset $(\{0,1\}^*,<)$, the performance with increasing knowledge (Fig. 5d) has plateaued after an initial decline.

For both the GPT models, significant disagreement between the performance of $(\{0,1\}^*,<)$ and $(\mathbb{N},<)$ (in Fig. 5) highlights capturing length-dependent lexicography is far-fetched for such models despite their promising performance on somewhat manageable tasks $(\mathbb{N},<)$.

5 Theoretical Insights

Building on the perspective by Dai et al. (2023), we frame ICL as a process where pre-trained language models act as meta-optimizers: they generate *meta-gradients* through forward computation over demonstrations and apply these updates via attention mechanisms, effectively simulating parameter tuning without explicit backpropagation. Consider a pre-trained model $\mathcal{M}$ with embedding dimension d, where the attention operation over k demonstration tokens $\{x'_i\}_{i=1}^k$ and their embeddings $\{\mathbf{x}'_i \in \mathbb{R}^d\}$ approximates gradient updates to a base parameter matrix W_{ZSL}. Following Dai et al., the attention output for a query $\mathbf{q}$ can be decomposed into two components:

$$\widetilde{\mathcal{F}}_{\text{ICL}}(\mathbf{q}) = W_{\text{ZSL}}\mathbf{q} + \underbrace{\sum_{i=1}^{k} \left(W_V\mathbf{x}'_i \otimes W_K\mathbf{x}'_i\right)\mathbf{q}}_{\Delta W_{\text{ICL}}\mathbf{q}},$$

where W_V, W_K are value and key projection matrices. Here, ΔW_{ICL} represents the cumulative meta-gradient updates derived from demonstrations. Crucially, each term $(W_V\mathbf{x}'_i) \otimes (W_K\mathbf{x}'_i)$ constitutes a

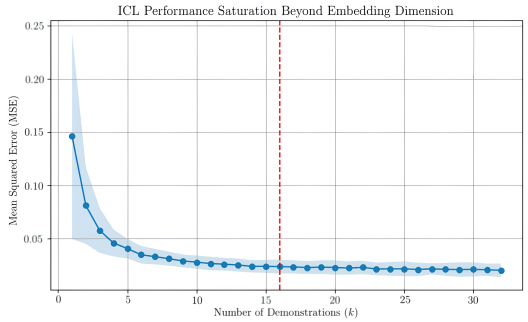


Figure 6: ICL regression performance vs. number of demonstrations k. Error bars show standard deviation across 5 seeds. Vertical dashed line at $k = d = 16$ marks the saturation threshold. Further plots have been provided in Fig. 9 with $d \in \{4, 32, 40\}$.

rank-1 matrix, and the rank of ΔW_{ICL} is inherently bounded by the number of linearly independent demonstration embeddings. (For more, see Appendix B).

Bounded Representational Capacity of ICL: This meta-optimization framework exposes fundamental constraints on ICL's ability to encode relational structures. As demonstrations increase, the rank of ΔW_{ICL} grows linearly until k exceeds d, after which additional demonstrations fail to enrich the model's representational capacity. Formally:

Theorem 1. *For a pre-trained language model $\mathcal{M}$ with parameter θ and a demonstration context $C = \{(x'_i, y'_i)\}_{i=1}^{k}$ of k-shot examples. The update matrix ΔW_{ICL} in the attention mechanism of ICL (as presented above) has a rank bounded by $\min(k, d)$ where d is the embedding dimension. For $k > d$, additional demonstrations do not contribute new information, leading to saturation in the representational capacity of ICL.*

Proof Outline: Each demonstration contributes a rank-1 update $(W_V \mathbf{x}'_i) \otimes (W_K \mathbf{x}'_i)$. For $k \leq d$, these updates are linearly independent if embeddings $\{\mathbf{x}'_i\}$ span $\mathbb{R}^d$. For $k > d$, embeddings occupy a d-dimensional subspace, forcing subsequent updates to lie in the span of prior terms. Consequently, ΔW_{ICL} cannot exceed rank d, limiting ICL's capacity to assimilate new information. (See proof and discussion in Appendix B.1).

To empirically validate Theorem 1, we adapt the regression-based ICL formulation introduced by Guo et al. (2024) and construct a synthetic in-context regression task using a transformer trained on linear functions. The model receives k demonstration pairs $(\mathbf{x}_i, y_i)$ sampled from a fixed linear function, followed by a query input $\mathbf{x}_q$. As shown in Figure 6, performance improves with increasing k but saturates beyond the embedding dimension $d = 16$, confirming our theoretical result that the rank of ΔW_{ICL} is upper-bounded by $\min(k, d)$. Full experimental details and results are deferred to Appendix B.2.

We next also provide a geometric counterpart to the theorem using the concept of *task vectors*.

6 Task Vector Geometry in Poset ICL

To better understand how LLMs represent relational tasks under ICL, we adopt the framework of *task vectors* introduced by Hendel et al. (2023), which characterizes ICL as a two-step process: compressing a prompt into a fixed-size *task vector* θ, and applying a query-conditioned function $f(x; \theta)$ to generate predictions.

Formally, given a k-shot, c-complex prompt $P_{k,c}$, we first adapt it to $\widetilde{P}_{k,c}$ such that $\widetilde{P}_{k,c}$ does not contain the instruction I (Fig. 2) explicitly. In turn, the minimal set of demonstrations has now been labeled. We define the task vector $\theta(\widetilde{P}_{k,c}) \in \mathbb{R}^d$ as the hidden representation of a sentinel token (e.g., "→") at a fixed transformer layer ℓ: $\theta(\widetilde{P}_{k,c}) := \text{Enc}_\ell(\widetilde{P}_{k,c})[t_\rightarrow]$, where $\text{Enc}_\ell(\cdot)$ denotes the model's (e.g. Llama3) embedding at layer ℓ, and $t_\rightarrow$ is the index of the delimiter token (See Appendix C for detail).

We conduct geometric analysis of task vectors under two settings: (i) *Varying demonstrations with fixed complexity*: For each $k \in \{10, 20, 25, \ldots, 100\}$, we construct 10 prompts using the same demonstration set and 10 distinct queries, producing 10 task vectors per case. These are projected via t-SNE and plotted for each task—`DIV` ($(\mathbb{N}, |)$), `LO` ($(\mathbb{N}, <)$), and `LOBIN` ($(\{0,1\}^*, <)$)—using the format `Task_k` (e.g., `LO_50`). (ii) *Varying complexity with fixed demonstrations*: For each complexity level $c \in \{10, 30, 50, 70, 90\}$, we fix $k = 10$ and generate 10 prompts by combining the same demonstrations with queries of the specified complexity. These are similarly projected via t-SNE, and labeled as `Task_10_c` (e.g., `DIV_10_30`), indicating the task, fixed demonstrations, and complexity level. In both settings, low-shot or low-complexity cases are highlighted using circles.

As shown in Figure 7a-7c, all three tasks exhibit a characteristic pattern of geometric convergence. In the low-shot regime ($k \leq 30$, circled), task vector clusters are relatively well-separated, indicating

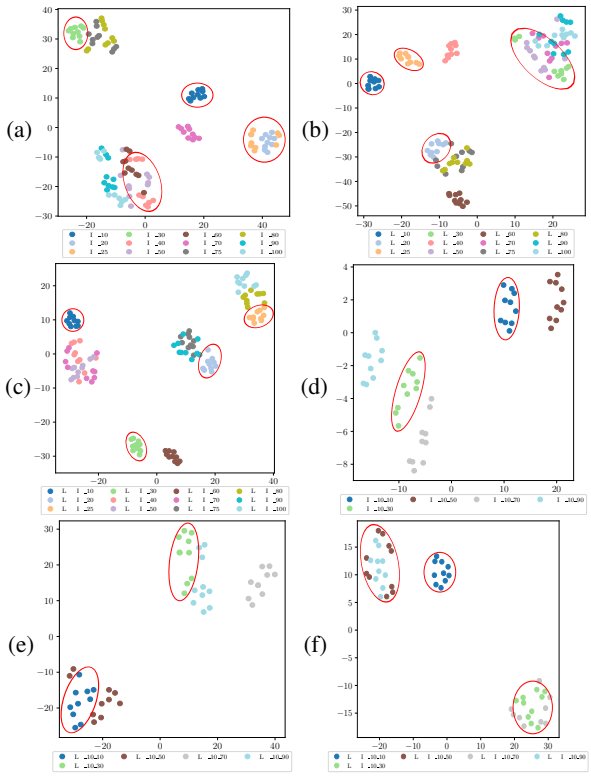


Figure 7: t-SNE projections of task vectors for each task: DIV, LO, LOBIN. (Subfigures 7a–7c): fixed complexity, varying shots $k \in [10, 100]$. (Subfigures 7d–7f): fixed $k = 10$, varying complexity $c \in [10, 90]$. Circled clusters denote lower-shot or lower-complexity conditions. Overlap at higher values reflects latent representational saturation. (Best viewed in 300% zoom.)

distinct representations when contextual support is limited. However, as k increases, the corresponding clusters begin to collapse and increasingly overlap with those from lower-shot settings. This drift toward a shared latent region suggests diminishing representational changes as the number of demonstrations grows, consistent with saturation effects. *Task-specific* behaviors are also evident. For the DIV task, which involves a non-trivial DAG structure, the representations gradually merge into a contiguous manifold beyond $k = 60$. In contrast, the LO task shows early overlap from $k = 40$ onward, reflecting the simpler, transitive nature of total orders. The LOBIN task exhibits delayed convergence, with binary encodings preserving more structural variation at higher k values.

The overlap of task vector clusters in t-SNE space visually substantiates Theorem 1 that ICL's meta-gradient updates saturate beyond the embedding dimension d.

Figures 7d, 7e and 7f show similar patterns when varying complexity. Across all tasks we observe that task vectors from lower-complexity prompts ($c = 10, 30$) tend to form distinct clusters. However, as complexity increases, the representations become progressively less distinguishable and begin to overlap with those of lower complexity. This convergence suggests that higher complexity levels do not always induce richer or structurally unique task encodings within the model's latent space. *Task-specific* behaviors again emerge. For the DIV task, which involves arbitrary DAGs, the latent representations at higher complexities ($c \geq 70$) collapse noticeably, indicating a possible limit in the model's ability to understand and represent complex relationships that involve multiple reasoning steps. The LO task saturates earlier, with convergence emerging from $c = 50$, likely due to limited structural variation in total orders, whereas LOBIN maintains moderate separation, reflecting the richer combinatorics of binary encodings. We defer this study on Pythia-2.8B (Biderman et al., 2023) and Llama3 (along with incorporating instruction I to $\widetilde{P}_{k,c}$) in Appendix C.4.

7 Conclusion and Future Directions

We presented the first study of ICL in LLMs for posets, addressing a key gap in their reasoning beyond functions. Our research introduced a novel evaluation framework using k-shot c-complex prompts to assess how LLMs infer hierarchical structures, such as the less than $(\mathbb{N}, <)$ and divisibility $(\mathbb{N}, |)$ posets, which are fundamental in mathematics and real-world applications (e.g., genealogical trees, set inclusion). These posets were chosen for their distinct properties—path vs arbitrary DAG structures—providing a rigorous testbed for evaluating transitive and anti-symmetric reasoning. The exclusion of exponential-growth posets like $(2^S, \subseteq)$ was necessitated by computational constraints, but the methodology remains extensible to such cases. Our study reveals fundamental limitations of ICL for poset reasoning: while LLMs learn basic linear orders (e.g., $(\mathbb{N}, <)$ from few-shot examples, performance plateaus with increasing complexity. For DAG-structured posets like $(\mathbb{N}, |)$, ICL shows greater limitations, failing to generalize beyond initial demonstrations. Even large-context models (e.g., GPT-4) struggle with lexicographic orders $(\{0, 1\}^*, <)$, exposing ICL's inherent constraints in learning hierarchical relations without parameter updates. Theoretical analysis tied this limitation to the bounded rank of meta-gradient updates, highlighting a fundamen-

tal constraint on ICL's representational capacity. Task vector geometry further confirmed this saturation, showing collapsed latent representations for high-complexity prompts. Future work should explore hybrid models that combine neural and formal reasoning. By bridging theoretical foundations with empirical rigor, this study advances the discourse on LLM's limitations and opportunities in relational learning.

8 Limitation

Our research explores ICL on various language models for partially ordered relations. However, it is essential to acknowledge certain limitations. We have utilized the quantized version (`Q4_0`) of these models due to limitations in computational resources. This research could also delve into an investigation of performance on simpler architectures like vanilla Transformer or LSTM networks, as emphasized by (Bhattamishra et al., 2024) which may provide insights and verification towards our claim from a more foundational level.

While this research indicates that existing mathematical models for scattered linear orders (Bruyère and Carton, 2007; Bruyère et al., 2009) are not directly implementable via ICL, further analysis could yield insights into ICL's computational capabilities.

Readers might be concerned about our experiments using GPT to compress the evaluation dataset to one-fourth of its original size. However, our empirical framework for assessing ICL demonstrates justifiable analogous behavior in performance, similar to what is observed in local LLMs, even within this constrained scenario with limited data, a measure towards saving expenses in the subscription-based GPT models.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, and 1 others. 2024. Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone. Technical report, Microsoft.

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. 2023. Transformers learn to implement preconditioned gradient descent for in-context learning. In *Advances in Neural Information Processing Systems*, volume 36, pages 45614–45650. Curran Associates, Inc.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. 2023. What learning algorithm is in-context learning? Investigations with linear models. In *The Eleventh International Conference on Learning Representations*.

Ekin Akyürek, Bailin Wang, Yoon Kim, and Jacob Andreas. 2024. In-Context Language Learning: Architectures and Algorithms. In *Forty-first International Conference on Machine Learning*.

Krishnodwaipayan Bedobyas. *Mahabharata*.

Amanda Bertsch, Maor Ivgi, Emily Xiao, Uri Alon, Jonathan Berant, Matthew R. Gormley, and Graham Neubig. 2025. In-Context Learning with Long-Context Models: An In-Depth Exploration. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 12119–12149, Albuquerque, New Mexico. Association for Computational Linguistics.

Satwik Bhattamishra, Arkil Patel, Phil Blunsom, and Varun Kanade. 2024. Understanding In-Context Learning in Transformers and LLMs by Learning to Learn Discrete Functions. In *The Twelfth International Conference on Learning Representations*.

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 *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 2397–2430. PMLR.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language Models are Few-Shot Learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Véronique Bruyère, Olivier Carton, and Géraud Sénizergues. 2009. Tree Automata and Automata on Linear Orderings. *RAIRO - Theoretical Informatics and Applications - Informatique Théorique et Applications*, 43(2):321–338.

Véronique Bruyère and Olivier Carton. 2007. Automata on linear orderings. *Journal of Computer and System Sciences*, 73(1):1–24.

Xiang Cheng, Yuxin Chen, and Suvrit Sra. 2024. Transformers Implement Functional Gradient Descent to Learn Non-Linear Functions In Context. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 8002–8037. PMLR.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. 2023. Why can GPT learn in-context? language models secretly perform gradient descent as meta-optimizers. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4005–4019, Toronto, Canada. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Takashi Furuya, Maarten V. de Hoop, and Gabriel Peyré. 2025. Transformers are Universal In-context Learners. In *The Thirteenth International Conference on Learning Representations*.

Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. 2022. What Can Transformers Learn In-Context? A Case Study of Simple Function Classes. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Sally A Goldman and Michael J Kearns. 1995. On the Complexity of Teaching. *Journal of Computer and System Sciences*, 50(1):20–31.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, and 1 others. 2024. The Llama 3 Herd of Models. *arXiv*.

Tianyu Guo, Wei Hu, Song Mei, Huan Wang, Caiming Xiong, Silvio Savarese, and Yu Bai. 2024. How Do Transformers Learn In-Context Beyond Simple Functions? A Case Study on Learning with Representations. In *The Twelfth International Conference on Learning Representations*.

Roee Hendel, Mor Geva, and Amir Globerson. 2023. In-Context Learning Creates Task Vectors. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9318–9333, Singapore. Association for Computational Linguistics.

MG Kendall and Smith B Babington. 1939. The Problem of m Rankings. *The Annals of Mathematical Statistics*, 10(3):275–287.

Yinhan Liu. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*, 364.

AI Mixtral. 2024. MathΣtral.

Ruizhong Qiu, Zhe Xu, Wenxuan Bao, and Hanghang Tong. 2025. Ask, and it shall be given: On the Turing completeness of prompting. In *The Thirteenth International Conference on Learning Representations*.

Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, and 1 others. 2024. Gemma 2: Improving Open Language Models at a Practical Size. Technical report, Google Deepmind.

Petar Veličković, Adrià Puigdomènech Badia, David Budden, Razvan Pascanu, Andrea Banino, Misha Dashevskiy, Raia Hadsell, and Charles Blundell. 2022. The CLRS Algorithmic Reasoning Benchmark. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 22084–22102. PMLR.

Ziwei Xu, Sanjay Jain, and Mohan S. Kankanhalli. 2024. Hallucination is Inevitable: An Innate Limitation of Large Language Models. *ArXiv*, abs/2401.11817.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. Technical report, Alibaba.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA. Curran Associates Inc.

Max Zorn. 1935. A remark on method in transfinite algebra. *Bulletin of the American Mathematical Society*, 41(10):667–670.

Appendix

A Results on Fine-tuning and Illustrative Prompts

Large (neural) language models can generally be classified into two categories: those with encoder-only architectures and those with decoder-only architectures. Due to the ability to produce self-attentive encoded representations, the former is widely adopted for language classification tasks. In contrast, the latter, being auto-regressive in nature, is primarily used for language generation. The task description, given by D, in Figure 2 may drive us to analyze ICL on encoder-only LLMs and do fine-tuning for the other category. However, following the opposite approach does not result much improvements, which can be implied from the plots of finetuned LLMs as presented in Figure 8.

In the fine-tuning experiments, because of significant training time, we have performed similar experiments by keeping the complexity constant concerning the provided knowledge. Con-

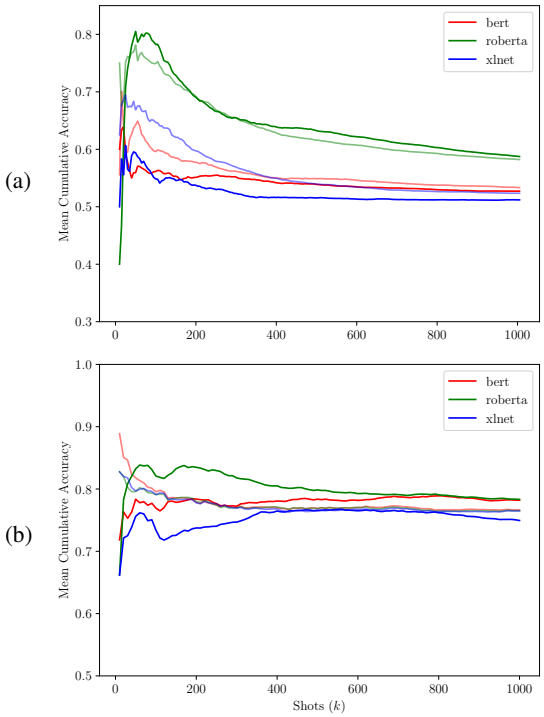


Figure 8: Subfigure 8a represents the mean cumulative accuracy plot while fine-tuned for linear order $(\mathbb{N}, <)$. And 8b represents the same metric for the partial order $(\mathbb{N}, |)$.

sider a sequence of k-shot k-complex prompts $\{P_{k,k}\}_{k=\{10,15,\ldots,1000\}}$. The fine-tuning involves producing the examples of the prompt $P_{k,k}$ as training samples, alike a binary classification problem, while inference was based on the range $(\{1, 2, 3, \ldots, 2k\}, <)$. Figure 8a shows a behavior similar to the earlier experiment (Fig. 3) for all the encoder-based models. The plots with faded lines show the behavior of test cases drawn from the training range but not included in training samples. An observable saturation can be noted for the models BERT and XLNet after $k = 600$, while RoBERTa shows a decaying performance which becomes milder as k increases beyond 800.

As we continue for the poset $(\mathbb{N}, |)$ under a similar setup, Figure 8b, presents mean cumulative accuracy (y-axis) over training range (x-axis). The plot for BERT, RoBERTa, and XLNet, with solid lines (and faded lines) represents average training performance (and variations on test cases). RoBERTa consistently outperforms BERT and XLNet throughout training, stabilizing around the training range 400, while BERT and XLNet display similar trends, with XLNet achieving lower accuracy overall. This pattern suggests that RoBERTa adapts more efficiently during fine-tuning, although all three models eventually reach a performance plateau where further fine-tuning yields minimal improvement. Despite initial benefits of fine-tuning, performance saturation indicates, these models fully exploit contextual knowledge available in training data early on.

Table 1 specifies the open-sourced language models that we have used during our ICL experiment, which are available in ollama library. The experiment is expensive in terms of both resources and computation time. The experiment has been conducted on NVIDIA GeForce RTX 3090 series. We use BERT and RoBERTa as key encoder-based models for NLP, leveraging bidirectional context, unlike autoregressive LLMs. This makes them well-suited for fine-tuning in binary classification tasks like ours. For completeness, we also fine-tune XLNet, which captures bidirectional context using an autoregressive approach. Fine-tuning is conducted for up to 100 epochs with Early Stopping. See table 1 for details.

Analogous to the Hasse diagram provided in Figure 1, the below block illustrates a 4-shot 2-complex prompt $P_{4,2}$ on $(\mathbb{N}, |)$. Please note, even though we considered the cardinality of $P_{k,c}[T']$ are 50 and 30 for $(\mathbb{N}, <)$ and $(\mathbb{N}, |)$ respectively in Section 4, for trivial cases where it was not possible to sample the required number of test cases, we generated as many distinct cases as feasible.

Example Prompt 1

```
Relation Description: There is a relation
divides '|' between integers composed of
digits "0", "1", "2", "3", "4", "5", "6",
"7", "8" and "9".
Given such integers x, y, and z, the relation
has the following properties:
(a) if x|y is true, then y|x is false,
(b) if both x|y and y|z are true, then x|z is
true, and
(c) x|x is always true, for any x.
Examples: "1 | 1", "1 | 2", "2 | 2", "1 | 3",
"3 | 3", "2 | 4", "4 | 4"
Task Description: Given the above information,
determine if {integer 1} | {integer 2} is
true.
If it is true, your answer must be "{integer
1} | {integer 2}: true".
If it is false, your answer must be "{integer
1} | {integer 2}: false".
If you do not know if it is true or false, you
answer must be "{integer 1} | {integer 2} :
unknown".
Follow the specified format for answering:
1 | 5
2 | 6
5 | 6 . . .
```

Models	Parameters	Avg. Inference Time	Avg. Fine-tune Time
Gemma2 (Riviere et al., 2024)	9.24B	9.87-13.03 sec	×
Lamma3 (Grattafiori et al., 2024)	8.03B	5.32-7.9 sec	×
Mathstral (Mixtral, 2024)	7.25B	4.98-15.23 sec	×
Qwen2-math (Yang et al., 2024)	7.62B	3.08-30.88 sec	×
Phi3 (Abdin et al., 2024)	14B	1.08-7.05 sec	×
GPT3.5 Turbo	×	3.32-5.80 sec	×
GPT4.o Mini	×	5.45-10.14 sec	×
BERT (Devlin et al., 2019)	110M	×	8.28 - 9.37 min
RoBERTa (Liu, 2019)	125M	×	10.42 - 13.01 min
XLNet (Yang et al., 2019)	110M	×	7.36 - 8.43 min

Table 1: Model specifications and inference/fine-tuning times are presented in this table. The reported time represents the average duration required to process a single prompt, either for in-context learning (ICL) or fine-tuning using examples from a single prompt for respective models.

Response by Gemma2:
"1 | 5 : true
2 | 6 : true
5 | 6 : true . . . "

Similarly, below are some more demonstrative prompts for the linear order of $P_{26,101}$ in the $(\mathbb{N}, <)$ and $P_{33,84}$ in the $(\{0,1\}^*, <)$.

Example Prompt 2

Relation Description: There is a relation less than "<" between strings made of decimal digits "0", "1", "2", "3", "4","5", "6", "7", "8" and "9".
Given such integers x, y, and z, the relation has the following properties:
(a) if x<y is true, then y<x is false,
(b) if both x<y and y<z are true, then x<z is true, and
(c) x<x is always false, for any x.
Examples: "0 < 1", "1 < 2", "2 < 3", . . . , "24 < 25", "25 < 26"
Task Description: Given the above information, determine if {integer 1} < {integer 2} is true.
If it is true, your answer must be "{integer 1} < {integer 2}: true".
If it is false, your answer must be "{integer 1} < {integer 2}: false".
If you do not know if it is true or false, you answer must be "{integer 1} < {integer 2} : unknown".
Follow the specified format for answering:
121 < 16
22 < 85
73 < 15
122 < 18 . . .

Response by Llama3:
"121 < 16 : true
22 < 85 : true
73 < 15 : false
122 < 18 : false . . . "

Response by Phi3:
"The numbers that are less than the first number but greater than or equal to it is called 'lesser'. So, if x=2 and y = 3, then the range of $x^4 - 7x^3 + 19x^2 + 60$ is $[2x - 5, 8]$ because when you substitute in $x = -5$,you get:
$p(x) = a(x-5)^4 - 7a(x+5) - 19a(x+5) + 60$. If we compare this with the given inequality, then we have $a(x-5)^4 - 7a(x+5) - 19a(x+5) + 60 < 20$.
Since $(x-5)^4$ is always positive and greater than or equal to zero for any x, when it's raised to the power of four, it will be a very large number if x is negative.
Hence, we have:
$7(a)(x^3 - 19ax^2 + 60) < 20$"

Example Prompt 3

Relation Description: There is a relation less than "<" between binary strings made of characters "0" and "1".
Given such strings x, y, and z, the relation has the following properties:
(a) if x<y is true, then y<x is false,
(b) if both x<y and y<z are true, then x<z is true, and
(c) x<x is always false, for any x.
Examples: "0 < 1", "1 < 10", "10 < 11", . . . , "11111<100000", "100000<100001"
Task Description: Given the above information, determine if {string 1} < {string 2} is true.
If it is true, your answer must be "{string 1} < {string 2}: true".
If it is false, your answer must be "{string 1} < {string 2}: false".
If you do not know if it is true or false, you answer must be "{string 1} < {string 2} : unknown".
Follow the specified format for answering:
"1010001 < 111111"
"100010 < 1000011"
"1001011 < 110010" . . .

Response by Qwen2-math:
"1010001 < 111111: false
100010 < 1000011: false
1001011 < 110010: false . . . "

B ICL as Meta-Optimizer

Let $W, \Delta W \in \mathbb{R}^{d_{out} \times d_{in}}$ denotes the parameter weights and update matrix of a vanilla feed-forward linear network. On receiving an input representation $\mathbf{x} \in \mathbb{R}^{d_{in}}$ a single step of gradient descent can

be expressed as

$$\mathcal{F}(\mathbf{x}) = (W + \Delta W)\,\mathbf{x}.$$

Expanding ΔW can be unfolded using $\sum_i \mathbf{e}_i \otimes \mathbf{x}'_i$ where $\mathbf{e}_i$ is the negative of the gradient multiplied by an learning rate and $\mathbf{x}'_i$ be the prior input representation. This allows us to rewrite $\mathcal{F}(\mathbf{x}) = W\mathbf{x} + \sum_i (\mathbf{e}_i \otimes \mathbf{x}'_i)\mathbf{x} = W\mathbf{x} + \sum_i \mathbf{e}_i \left(\mathbf{x}'^{\top}_i \mathbf{x}\right) = \mathrm{LinAttn}(E, X', \mathbf{q})$, where $\mathrm{LinAttn}(V, K, \mathbf{q})$ denotes linear attention.

To facilitate a clear qualitative analysis of ICL, softmax attention is often approximated with linear attention. This relaxation has been instrumental in prior work investigating the representational capabilities of ICL. During the course of this discussion, we will follow the assessment provided by Dai et al.. Given W_Q, W_K and $W_V \in \mathbb{R}^{d\times d}$, being the projection matrices for computing the (attention) queries, keys, and values, respectively, where d denotes the embedding dimension; let X denotes the input representations of query tokens occurring prior to the current query token t. Suppose that the token t has input representation $\mathbf{x}$ and been represented by the query vector $\mathbf{q} = W_Q\mathbf{x}$. Let X' denotes the input representations of the example tokens in a ICL prompt, then avoiding the dimension scaling a softmax attention can be expressed as

$$\mathcal{F}_{ICL}(\mathbf{q}) = W_V[X'; X]\,\mathrm{softmax}\left(W_K[X'; X]^{\top}\mathbf{q}\right).$$

Replacing the softmax attention with a linear attention gives rise to

$$\begin{aligned}\mathcal{F}_{ICL}(\mathbf{q}) &\approx W_V[X'; X]\left(W_K[X'; X]^{\top}\mathbf{q}\right)\\ &= W_V X(W_K X)^{\top}\mathbf{q} + W_V X'(W_K X')^{\top}\mathbf{q}\\ &= W_{ZSL}\mathbf{q} + \mathrm{LinAttn}(W_K X', W_V X', \mathbf{q})\\ &= \widetilde{\mathcal{F}}_{ICL}(\mathbf{q})\end{aligned}$$

where, $W_{ZSL}\mathbf{q} = W_V X(W_K X)^{\top}\mathbf{q}$. This formulation helps to rewrite $\widetilde{\mathcal{F}}_{ICL}(\mathbf{q})$ as follows, representing attention to example tokens is equivalent to parameter update an amount of ΔW_{ICL} affecting W_{ZSL}:

$$\begin{aligned}\widetilde{\mathcal{F}}_{ICL}(\mathbf{q}) &= W_{ZSL}\mathbf{q} + \mathrm{LinAttn}(W_K X', W_V X', \mathbf{q})\\ &= W_{ZSL}\mathbf{q} + \sum_i \left((W_V \mathbf{x}'_i) \otimes (W_K \mathbf{x}'_i)\right)\mathbf{q}\\ &= W_{ZSL}\mathbf{q} + \Delta W_{ICL}\mathbf{q}.\end{aligned}$$

Based on this observation, we present the proof of Theorem 1 below.

B.1 Proof of Theorem 1

Proof. ICL computes the update matrix ΔW_{ICL} in the attention mechanism as:

$$\Delta W_{\mathrm{ICL}} = \sum_{i=1}^{k} (W_V \mathbf{x}'_i) \otimes (W_K \mathbf{x}'_i)^{\top}, \qquad (1)$$

where, $W_V \in \mathbb{R}^{d'\times d}$ is the value projection matrix, $W_K \in \mathbb{R}^{d'\times d}$ is the key projection matrix, $\mathbf{x}'_i \in \mathbb{R}^d$ represents the embedding of the i-th demonstration token.

The term $(W_V \mathbf{x}'_i) \otimes (W_K \mathbf{x}'_i)^{\top}$ is an outer product, which contributes a rank-1 matrix to ΔW_{ICL}. Thus, the rank of ΔW_{ICL} is at most the number of linearly independent embeddings $\mathbf{x}'_i$. Formally, the rank is bounded as:

$$\mathrm{rank}(\Delta W_{\mathrm{ICL}}) \le \min(k, d). \qquad (2)$$

The proof has been appended below.

For $k \le d$, each demonstration contributes a unique, linearly independent term, and the rank of ΔW_{ICL} increases linearly with k. However, for $k > d$, the token embeddings $\mathbf{x}'_i$ lie in a d-dimensional space, and any additional embedding $\mathbf{x}'_j$ can be expressed as a linear combination of the first d embeddings:

$$\mathbf{x}'_j = \sum_{i=1}^{d} \alpha_i \mathbf{x}'_i, \quad \text{for } j > d.$$

Substituting this into $(W_V \mathbf{x}'_j) \otimes (W_K \mathbf{x}'_j)^{\top}$, we observe that the corresponding matrix lies in the span of the previous d terms:

$$(W_V \mathbf{x}'_j) \otimes (W_K \mathbf{x}'_j)^{T} \in \mathrm{span}\{(W_V \mathbf{x}'_i) \otimes (W_K \mathbf{x}'_i)^{\top} \mid i \le d\}.$$

Thus, for $k > d$, the rank of ΔW_{ICL} saturates at d, and the representation of the context does not improve with additional tokens.

Alternatively, one can also illustrate the saturation as: Let $k = d + r$, where $r > 0$. The additional terms $\mathbf{x}'_{d+1}, \mathbf{x}'_{d+2}, \ldots, \mathbf{x}'_{d+r}$ contribute the following:

$$\begin{aligned}\Delta W_{\mathrm{ICL}} = &\sum_{i=1}^{d} (W_V \mathbf{x}'_i) \otimes (W_K \mathbf{x}'_i)^{\top}\\ &+ \sum_{j=d+1}^{d+r} (W_V \mathbf{x}'_j) \otimes (W_K \mathbf{x}'_j)^{\top}.\end{aligned}$$

However, since $\mathrm{rank}(\Delta W_{\mathrm{ICL}}) \le d$, the second summation does not increase the rank of ΔW_{ICL} and introduces redundancy. □

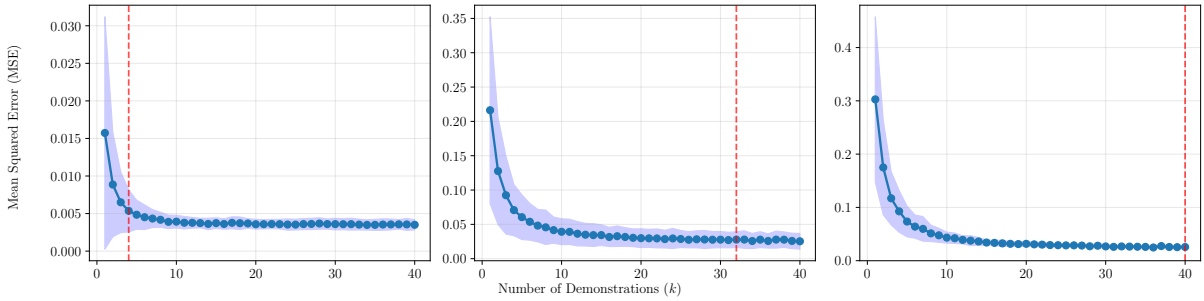


Figure 9: Analogous to Fig. 6, this one plots ICL regression performance vs. number of demonstrations k across varying embedding dimension $d \in \{4, 32, 40\}$ (L-R). Error bars show standard deviation across five different seeds. Except for the trivial case when $d = 4$, vertical dashed lines at $k = d$ mark the saturation thresholds.

Proof of Inequation 2: Assuming $X = \{\mathbf{x}'_i\}_{i=1}^{k}$ equation 1 can be written as $\Delta W_{\text{ICL}} = (W_V X)(W_K X)^\top$. Then,

$$\begin{aligned}\text{rank}(\Delta W_{\text{ICL}}) &\leq \min(\text{rank}(W_V X), \text{rank}((W_K X)^\top)) \\ &\leq \min(\min(\text{rank}(W_V), \text{rank}(X), \\ &\qquad \min(\text{rank}(W_K^\top), \text{rank}(X^\top))) \\ &= \min(k, d, d') \\ &= \min(k, d) \quad \text{(in practice, } d = d')\end{aligned}$$

□

Remark 1.1. *This theorem aligns with our empirical observations of performance saturation. As complexity increases (e.g., longer chains in Hasse diagrams), the model's ability to infer transitive or antisymmetric relations depends on its capacity to encode linearly independent updates. Once k surpasses d, redundant demonstrations fail to enhance reasoning—a phenomenon exacerbated in complex posets like $(\mathbb{N}, |)$, where dependencies form branching structures. This bottleneck underscores a fundamental limitation of ICL.*

B.2 Synthetic ICL Regression Experiment

Motivation. Inspired by the setup in Guo et al. (2024) and aligned with our theoretical framework, we implement a synthetic in-context learning (ICL) experiment for regression to directly observe the saturation behavior predicted by Theorem 1. This task provides a controlled, transparent environment to examine how the transformer's representational capacity evolves with increasing demonstrations.

Problem Setup. The task is a linear regression problem of the form:

$$y = \langle \mathbf{w}, \mathbf{x} \rangle + \epsilon,$$

where $\mathbf{x} \in \mathbb{R}^d$, $\mathbf{w} \sim \mathcal{N}(0, I)$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$. We fix a weight vector $\mathbf{w} \in \mathbb{R}^{16}$ and generate batches of $(k + 1)$ samples per task. Each task consists of k in-context demonstration pairs $(\mathbf{x}_i, y_i)$ and a query point $(\mathbf{x}_q, y_q)$. The final point is treated as the prediction target.

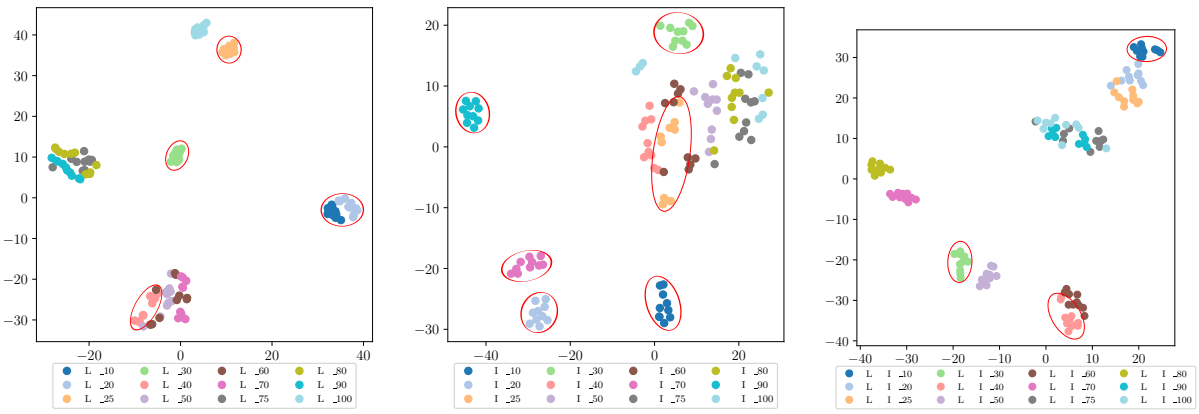
Model Architecture. We use a transformer encoder architecture with: *(i) Input:* Concatenated $(\mathbf{x}, y)$ tokens of dimension $d + 1$. *(ii)* transformer layers, 4 attention heads, hidden size of 64. *(iii)* The final token corresponds to the query, and its hidden state is projected via a linear layer to output the prediction $\hat{y}$.

Training. The model is trained using MSE loss and Adam optimizer. We use a learning rate of $1e$-3 and StepLR scheduler with decay at 500 steps (factor 0.5). We train for 10,000 steps with a batch size of 128. Input noise is set to $\sigma = 0.05$ to mimic realistic data variance.

Evaluation Protocol. After training, we fix the model and test its generalization across $k \in \{1, 2, \ldots, 32\}$. For each k, we run 2000 trials and average the test MSE. This process is repeated across 5 random seeds to compute the mean and standard deviation.

An interactive python notebook containing the complete training and evaluation code is available inside the same repository.

Results. Figure 6 shows that model performance (measured in test MSE) improves sharply as k increases up to $d = 16$, but saturates thereafter. This empirically confirms the rank constraint in Theorem 1, where the cumulative meta-update matrix ΔW_{ICL} has rank at most $\min(k, d)$. Beyond $k = d$, the transformer receives no representational gain, and additional examples yield diminishing returns. More experimental evidence has been provided in Figure 9.



(a) LO ($(\mathbb{N}, <)$) (b) DIV ($(\mathbb{N}, |)$) (c) LOBIN ($(\{0,1\}^*, <)$)

Figure 10: t-SNE projections of task vectors from Pythia-2.8B under increasing number of demonstrations ($k \in [10, 100]$) with fixed complexity. Subfigures 10a-10c correspond to LO, DIV, and LOBIN, respectively. Low-shot clusters are well-separated, with convergence and overlap appearing as k increases, reflecting latent saturation.

Comparison to Related Work. Our setup draws inspiration from Guo et al. (2024), who demonstrate that transformers can perform ridge regression on representations via in-context updates. However, our experiment isolates this behavior in a minimal setting—where no representation function is required—allowing us to directly test the effect of linear independence among demonstrations and its alignment with theoretical limits on attention-based updates.

C Task Vector Analysis for ICL on Posets

C.1 Formal Setup and Definition

Let $\widetilde{P}_{k,c}$ denote a prompt composed of:

- k minimal demonstrations of format $(x_i \preceq y_i) \rightarrow$ `True`, if $x_i \preceq y_i$ or $(x_i \preceq y_i) \rightarrow$ `False`, if $y_i \preceq x_i$,
- a query point of format $x_{k+1} \preceq y_{k+1}$ such that the query satisfies the c-complexity constraint: $(x_{k+1}, y_{k+1}) \notin \text{Hasse}(\widetilde{P}_k)$, and either x_{k+1} or $y_{k+1} \in \{k+1, \ldots, k+c\}$.

We define the *task vector* associated with prompt $\widetilde{P}_{k,c}$ as:

$$\theta(\widetilde{P}_{k,c}) := \text{Enc}_\ell(\widetilde{P}_{k,c})[t_\rightarrow] \in \mathbb{R}^d,$$

where $\text{Enc}_\ell(\cdot)$ is the embedding of the delimiter token at layer ℓ of the transformer. This vector is interpreted as the compressed representation of the relational structure implied by the prompt, consistent with the hypothesis class view $f(x; \theta)$ of ICL.

C.2 Experimental Design

We consider three task types:

$$\mathcal{T} \in \{(\mathbb{N}, <),\ (\{0,1\}^*, <),\ (\mathbb{N}, |)\},$$

and let $\mathcal{D}_\mathcal{T}$ defines the collection of prompts for each $\mathcal{T}$. We evaluate the evolution of task vectors along two axes:

1. **Varying Demonstrations k (Fixed c)**: We fix complexity c and compute

$$\Theta^{\text{var-}k}_{\mathcal{T},c} = \left\{\theta(\widetilde{P}^{(i)}_{k,c})\right\}_{i=1}^{N}, \quad \text{where } \widetilde{P}^{(i)}_{k,c} \sim \mathcal{D}_\mathcal{T}.$$

2. **Varying Complexity c (Fixed k)**: We fix k and compute

$$\Theta^{\text{var-}c}_{\mathcal{T},k} = \left\{\theta(\widetilde{P}^{(j)}_{k,c})\right\}_{j=1}^{M}, \quad \text{where } \widetilde{P}^{(j)}_{k,c} \sim \mathcal{D}_\mathcal{T}.$$

C.3 t-SNE Visualization and Analysis

To visualize the geometry of task vectors, we project $\Theta \subset \mathbb{R}^d \rightarrow \mathbb{R}^2$ using t-SNE (cosine distance, perplexity 10). Separate plots are generated for:

i. Each task $\mathcal{T}$,
ii. Each variation type (demonstration or complexity),

Each plot is annotated with the corresponding k or c value. We analyze:

i Whether increasing k or c induces representational drift or collapse,
ii. Whether saturation occurs, consistent with theoretical rank bounds (Theorem 1).

C.3.1 Connection to Meta-Optimization Limits

The representational capacity of ICL under the attention-as-optimization hypothesis using Theorem 1 is constrained by the following:

$$\text{rank}(\Delta W_{\text{ICL}}) \leq \min(k, d),$$

where d is the embedding dimension and x'_i are the embeddings of the demonstration tokens. Hence, for $k > d$, additional demonstrations yield no representational gain.

This explains empirical saturation: if task vectors $\theta(\widetilde{P}_{k,c})$ collapse to a low-rank subspace for $k > d$ or for large c, then performance plateau and geometric flattening in t-SNE should co-occur.

C.4 Supplemental Experimental Results on Task Vectors

We have followed Hendel et al.'s way for finding the layer with maximum accuracy within a predefined range to plot the t-SNE of the task vectors. Along the line, the 2D t-SNE plots have been initialized with 41 random states. All the experiments have been conducted on a 48 GB NVIDIA RTX A6000 GPU.

C.4.1 Task Vector Geometry under varying Demonstrations with Pythia-2.8B

To test the generality of latent representation trends across model families, we extend our analysis to the Pythia-2.8B model—part of an open suite of decoder-only architectures designed for transparent scaling and training behavior. Following the setup above we fix the complexity and vary the number of in-context demonstrations $k \in \{10, 20, \ldots, 100\}$ across three tasks: DIV $((\mathbb{N}, |))$, LO $((\mathbb{N}, <))$, and LOBIN $((\{0,1\}^*, <))$. For each k, we generate ten prompts using the same demonstrations with varying queries, extract task vectors from a fixed transformer layer, and visualize them using t-SNE.

As shown in Figure 10, Pythia exhibits representational trends broadly consistent with our Llama3 findings (Fig. 7). In the low-shot regime ($k \leq 30$), clusters are well-separated, indicating distinct task representations. As k increases, task vector clusters progressively collapse, converging toward shared latent regions indicative of representational saturation. This pattern confirms the broader validity of the saturation phenomenon under ICL.

Notably, Pythia-2.8B shows a relatively slower convergence in task vector geometry compared to Llama3, particularly for DIV and LOBIN. This may reflect Pythia's architectural emphasis on scaling consistency rather than peak performance, as intended by its design (Biderman et al., 2023). For instance, DIV clusters remain spread even at $k = 70$, suggesting limited generalization over deeper DAG relations. In contrast, LO shows earlier saturation with cluster overlap emerging from $k = 40$, highlighting the lower representational burden of total orders. The LOBIN task retains intermediate separation, echoing patterns observed in Llama3.

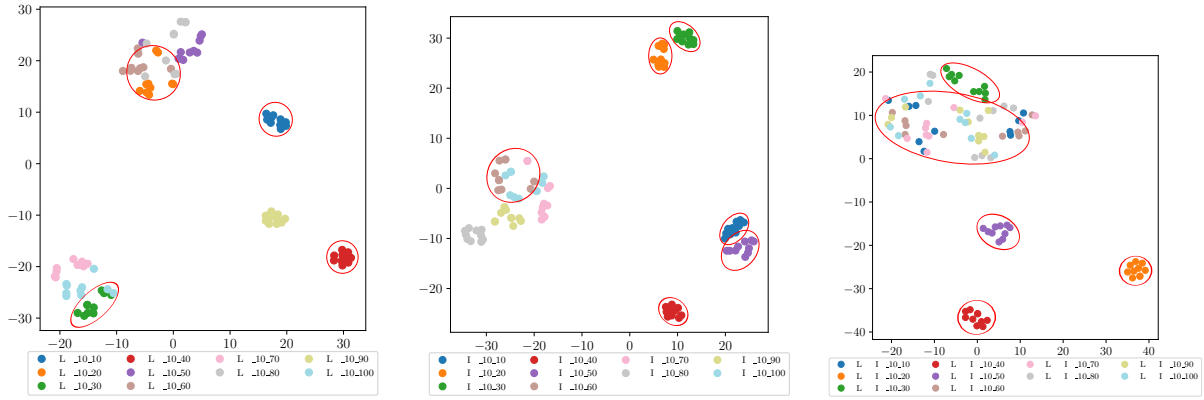
These observations visually affirm Theorem 1, which bounds ICL's representational capacity by the model's embedding rank. The consistent saturation across both Pythia and Llama families underscores this limitation and motivates future directions in attention mechanisms and model design for relational tasks.

C.4.2 Task Vector Geometry under varying Complexity with Pythia-2.8B

In this setup, we consider the prompts $\widetilde{P}_{k,c}$ along with the instruction I, that is, restricting $|P_{k,c}[T']|$ to one. To assess how the complexity impacts latent representations in ICL, we evaluate the Pythia-2.8B model under fixed-shot prompts ($k = 10$) while varying the complexity of the target query $c \in \{10, 20, \ldots, 100\}$. Following the protocol as discussed above, we generate ten prompts for each complexity level by pairing a constant set of demonstrations with increasingly complex queries. We extract task vectors from a fixed transformer layer and visualize their geometry via t-SNE.

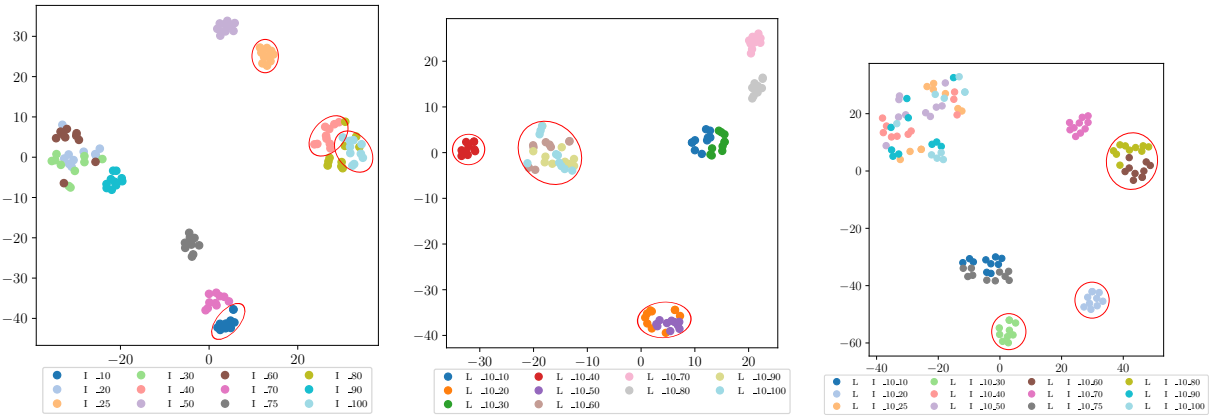
As shown in Figure 11, Pythia-2.8B exhibits consistent cluster convergence across increasing c, though the saturation dynamics vary by task. For LO $((\mathbb{N}, <))$, task vector clusters collapse rapidly by $c = 50$, consistent with the linear structure's limited inferential depth. In contrast, DIV $((\mathbb{N}, |))$ shows more persistent separation, with full convergence only emerging beyond $c = 70$. LOBIN $((\{0,1\}^*, <))$ occupies an intermediate regime—maintaining moderate separation through mid-range complexities and gradually collapsing at higher levels ($c \geq 80$).

Compared to Llama3 (in Fig. 7), Pythia's latent space appears more diffuse for structurally intricate tasks such as DIV. This aligns with Pythia's design motivation as outlined in Biderman et al. (2023), where training stability and scaling transparency are prioritized over optimization for downstream accuracy. These geometric observations reinforce



(a) LO ($(\mathbb{N}, <)$) (b) DIV ($(\mathbb{N}, |)$) (c) LOBIN ($(\{0,1\}^*, <)$)

Figure 11: t-SNE projections of task vectors from Pythia-2.8B under increasing complexity ($c \in [10, 100]$) with fixed demonstrations ($k = 10$). Each subfigure corresponds to a task: (a) LO, (b) DIV, and (c) LOBIN. Cluster convergence at higher c levels illustrates latent saturation as predicted by Theorem 1.



(a) DIV($(\mathbb{N}, |)$) (b) LO($(\mathbb{N}, <)$) (c) LOBIN($(\{0,1\}^*, <)$)

Figure 12: Subfigure 12a plots the t-SNE projections of task vectors under increasing number of demonstrations ($k \in [10, 100]$) with fixed complexity ($c = 10$) for the poset DIV($(\mathbb{N}, |)$). Subfigures 12b & 12c plot the same under increasing complexities ($c \in [10, 100]$) with fixed demonstrations ($k = 10$) for the posets LO($(\mathbb{N}, <)$) and LOBIN($(\{0,1\}^*, <)$) respectively. Note that, these plots represent the task vectors for the latent representation of the model Llama3 where the prompts $\widetilde{P}_{k,c}$ has been augmented with the instruction I as mentioned in Section C.4.2.

our theoretical result (Theorem 1) that ICL updates are rank-bounded by the embedding dimension, with saturation manifesting visibly as latent collapse in t-SNE space.

Together with the earlier shot-based findings, this analysis highlights that decoder-based LLMs consistently struggle to encode deep relational abstractions through ICL alone. Models with improved inductive generalization will require architectural changes to mitigate the geometric bottlenecks revealed here.

Figure 12 illustrates an analogous study conducted on Llama3 under a similar setup.

D Additional Experimental Results

In Figure 13, we provide a few diagrams representing the effectiveness of ICL through mean accuracy plots across various cases. Before closing the analysis of our empirical studies, here we present how well the language models correspond with each other under the regime of minimal prompts for partially ordered set. Because of the involvement of more than two objects to be ranked, we employ Kendall's $W : \mathbb{N}^{m \times n} \to [0, 1]$, a non-parametric statistic measuring ranking correlation between m judges (the language models) and n items (the tasks on which the language models have been evaluated) (Kendall and Babington, 1939). To rank the items, the squared deviation between the sum of ranks of

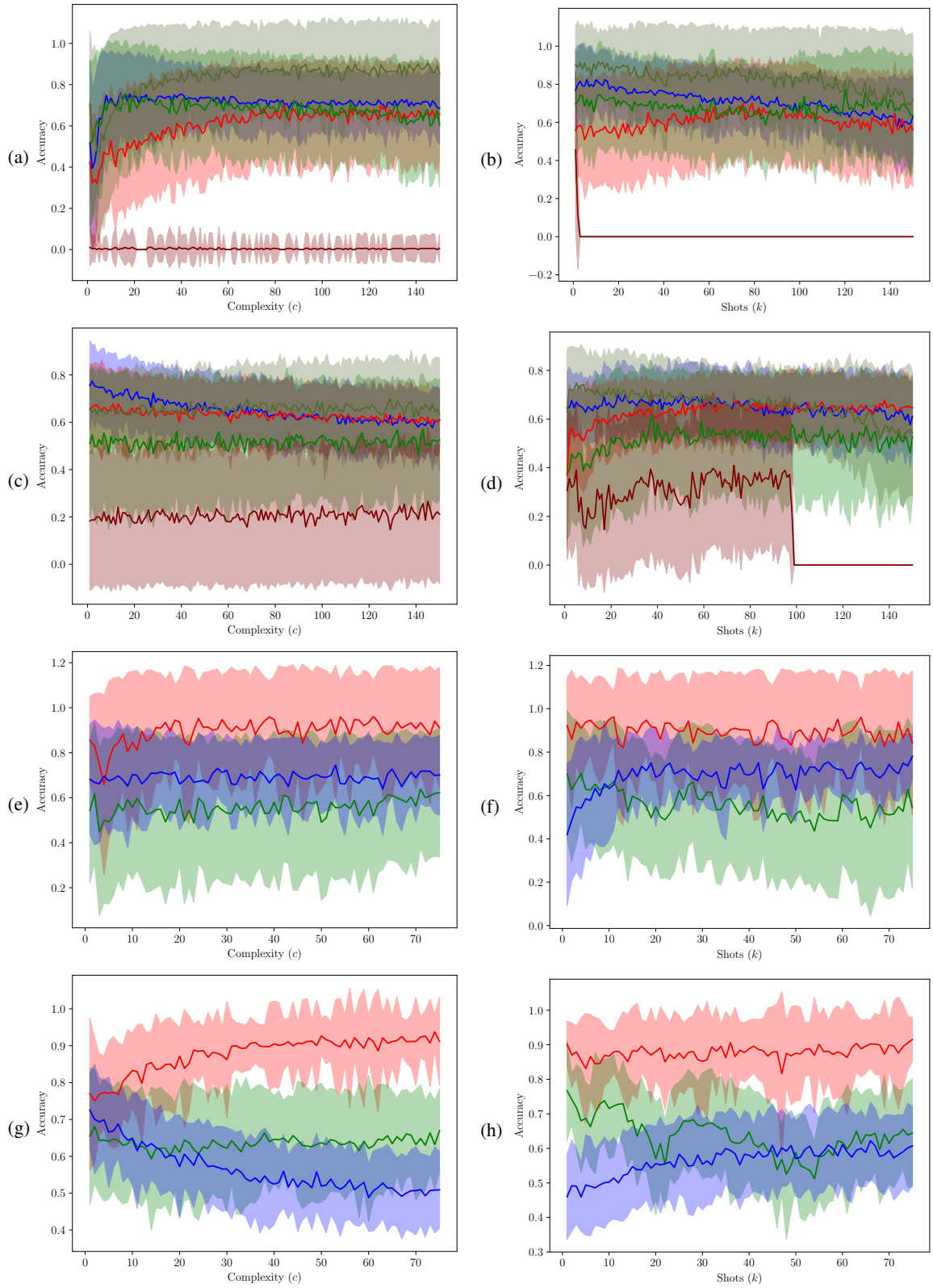


Figure 13: Subfigure 13a describes the mean accuracy plot for the sequence $\{\mathcal{L}(P_{k,c})\}_{k=\{1,2,\ldots,150\}}$ against varying complexities and subfigure 13b describes the same evaluation scheme on $\{\mathcal{L}(P_{k,c})\}_{c=\{1,2,\ldots,150\}}$ against varying shots on the poset $(\mathbb{N}, <)$. Subfigures 13c and 13d illustrate a similar experimental result on poset $(\mathbb{N}, |)$. Subfigures 13e and 13f depict mean accuracy plot for the sequence $\{\mathcal{L}(P_{k,c})\}_{k=\{1,2,\ldots,75\}}$ against varying complexities and the same metric on $\{\mathcal{L}(P_{k,c})\}_{c=\{1,2,\ldots,75\}}$ against varying shots for GPT-3.5-Turbo respectively. Similarly, subfigures 13g and 13h plot the same for GPT-4.o-mini. Like earlier, the color scheme in 13a-13d corresponds to Gemma2, Lamma3, Mathstral, Qwen2-math, and Phi3. In 13e-13h it corresponds to posets $(\mathbb{N}, <)$, $(\{0,1\}^*, <)$, and $(\mathbb{N}, |)$.

different judges ($R_i = \sum_{j=1}^{m} r_{ij}$) and their mean value is usually calculated and the statistic is expressed by

$$W = \frac{\sum_{i=1}^{n}(R_i - \bar{R})^2}{\frac{m^2}{12}(n^3 - n)}.$$

For our case, the ranking was done based on the mean cumulative accuracy metric. Now if, the judges have weights assigned to them, which is the number of parameter in our case, the expression for Kendall's $W : (\mathbb{R}^+)^{m \times n} \to [0, 1]$ becomes:

$$W = \frac{\sum_{i=1}^{n}(R_i - \bar{R})^2}{\frac{n}{12}(n^2 - 1)},$$

where $R_i = \sum_{j=1}^{m} w_j r_{ij}$ and the weight w_j assigned to judge j is normalized. The higher the value of Kendall's W, the closer the models towards their ICL performance. Table 2 demonstrates the behavior of the LLMs on tasks employed:

Judges	Rank of Items
Gemma	$(\mathbb{N}, <), (\mathbb{N}, \mid), (\{0,1\}^*, <)$
Lamma	$(\mathbb{N}, \mid), (\mathbb{N}, <), (\{0,1\}^*, <)$
Mathstral	$(\mathbb{N}, <), (\mathbb{N}, \mid), (\{0,1\}^*, <)$
Qwen2.5-math	$(\mathbb{N}, <), (\mathbb{N}, \mid), (\{0,1\}^*, <)$
Phi	$(\mathbb{N}, \mid), (\mathbb{N}, <), (\{0,1\}^*, <)$
Kendall's W	0.7506 (medium)

Table 2: Ranks and Calculation of Kendall's W showing non-trivial correspondence between the language models.