Beyond In-Distribution Success: Scaling Curves of CoT Granularity for Language Model Generalization

Ru Wang[1], Wei Huang[2,3], Selena Song[1], Haoyu Zhang[1]
Qian Niu[1], Yusuke Iwasawa[1], Yutaka Matsuo[1], Jiaxian Guo[4]
[1]The University of Tokyo [2] RIKEN Center for Advanced Intelligence Project
[3] The Institute of Statistical Mathematics [4] Google Research Australia

Generalization to novel compound tasks under distribution shift is important for deploying transformer-based language models (LMs). This work investigates Chain-of-Thought (CoT) reasoning as a means to enhance OOD generalization. Through controlled experiments across several compound tasks, we reveal three key insights: (1) While QA-trained models achieve near-perfect in-distribution accuracy, their OOD performance degrades catastrophically, even with 10000k+ training examples; (2) the granularity of CoT data strongly correlates with generalization performance; finer-grained CoT data leads to better generalization; (3) CoT exhibits remarkable sample efficiency, matching QA performance with much less (even 80%) data. Theoretically, we demonstrate that CoT forces internalization of valid dependency structures, and thus can achieve better generalization. Further, we show that transformer positional embeddings can amplify generalization by emphasizing subtask condition recurrence in long CoT sequences. Our combined theoretical and empirical analysis provides compelling evidence for CoT reasoning as a crucial training paradigm for enabling LM generalization on multi-step reasoning tasks under structural distributional shifts..

1. Introduction

Transformer-based language models (LMs) [1–3] have demonstrated unprecedented capabilities in knowledge retrieval [4–6] and reasoning [7, 8], driven by large-scale pretraining on diverse text corpora [9–11]. While these models excel at tasks with clear input-output mappings, their ability to generalize to compound tasks—those requiring dynamic planning, multi-step reasoning, and adaptation to evolving contexts—remains a critical challenge. Real-world applications such as GUI automation [12–16], textual puzzle solving [17, 18], and strategic gameplay like poker [19–21] demand not only procedural knowledge but also the capacity to handle distribution shift between training and deployment environments.

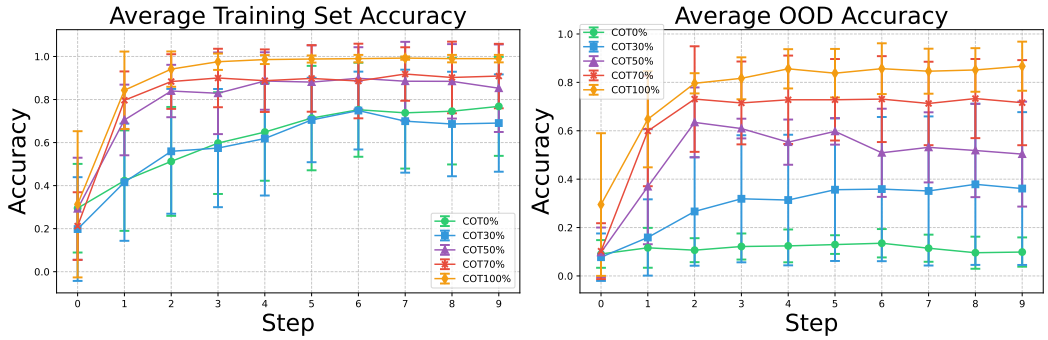


Figure 1: The illustration of the impact of the granularity of Chain-of-Thought on In-Distribution (IID) and Out-of-Distribution (OOD) performance. Left: IID performance. Right: OOD performance. Results are averaged over four compound tasks. While models trained without CoT achieve high IID accuracy (80%), they exhibit a substantially poor generalization performance (10%) on OOD data.

Third Conference on Parsimony and Learning (CPAL 2026).

A central challenge lies in the misalignment between training data and deployment environments. For instance, consider a medical diagnosis system trained on historical patient records: it may falter when encountering novel symptom combinations during a disease outbreak or misalign with updated clinical guidelines (e.g., revised thresholds for "high-risk" biomarkers [22]). Such distribution shifts can significantly degrade the performance of LMs trained on limited training datasets, underscoring the importance of generalization for reliable deployment.

In this paper, we investigate how data collection impacts the generalization of LMs in compound tasks. Traditional approaches that prioritize direct instruction-result pairs (analogous to question-answering frameworks) optimize for task-specific accuracy but inadequately prepare models for compositional reasoning in unseen contexts. Recent advances in Chain-of-Thought (CoT) prompting [5], which elicit intermediate reasoning steps, suggest that explicit process supervision can enhance generalization. However, acquiring high-quality CoT annotations at scale remains prohibitively expensive [23, 24], raising a pivotal question: **How do different data collection strategies, *e.g.*, result-oriented Q-A pairs and CoT—affect the generalization of LMs to novel compound tasks?**

To address this, we conduct controlled experiments using synthetic compound tasks that systematically introduce distribution shifts between training and evaluation environments. We evaluate LMs trained on two data paradigms: (1) result-oriented Q-A pairs and (2) CoT sequences of varying granularity. Our analysis reveals three critical insights(see Figure 1): (i) **Generalization Gap:** While both Q-A and CoT-trained LMs achieve near-perfect in-distribution accuracy, Q-A models exhibit severe performance degradation under distribution shifts—even with 10000k training examples. (ii) **Granularity-Generalization Tradeoff:** The granularity of CoT data strongly correlates with generalization performance; finer-grained CoT data leads to better generalization. (iii) **Sample Efficiency:** LMs trained with fine-grained CoT data demonstrate strong sample efficiency, achieving comparable generalization to Q-A pair training with substantially less data. These results suggest that even small amounts of high-quality CoT data can significantly enhance LM generalization, despite the challenges associated with its collection.

Inspired by the recent work [25], we further theoretically demonstrate that CoT training forces models to internalize valid reasoning paths. Leveraging transformer positional embeddings, we further show that explicitly emphasizing subtask conditions within CoT sequences further reduces shortcut reliance, especially in the long CoT setting.

In summary, our paper makes the following key contributions: (1) We establish a controlled experimental framework to quantify the impact of data collection strategies on LM generalization under distribution shifts. Based on it, we demonstrate that LMs trained directly on question-answer pairs can achieve high accuracy on in-distribution data for new tasks but exhibit poor generalization, even when trained on substantial datasets (e.g., 10000k examples). (2) Through systematic scaling experiments, we demonstrate that fine-grained CoT data enhances generalization and sample efficiency, even with limited training data. (3) We provide theoretical insights into how CoT reasoning helps capture the dynamic state transitions of compound tasks, thereby improving generalization. Furthermore, we demonstrate that longer CoT chains, by repeatedly conditioning on sub-tasks, can further enhance the generalization capabilities of language models. We believe these findings offer a reliable guide for data collection practices when leveraging LMs for novel tasks.

2. Related Work

Generalization in LLMs Transformer-based language models [26] demonstrate strong performance on in-distribution tasks [27–29] but often struggle with OOD tasks due to challenges such as distribution shifts [30, 31], shortcut learning [25, 32], and overfitting [33], where the correlations they exploit no longer hold [34, 35].

Various approaches have been proposed to mitigate shortcut learning, including data augmentation [36, 37] and adversarial training [38, 39], though many methods remain task-specific and lack generalization. Prior work [40] evaluates OOD performance by comparing models across datasets, but

without precise control over distribution shifts. Our work introduces a fully controlled experimental setup, explicitly defining and manipulating shifts to better analyze model adaptation.

Chain-of-Thought Reasoning CoT reasoning enables multi-step derivations, allowing models to articulate reasoning [4, 5, 41–43], though its mechanism remains unclear. Recent theoretical works apply circuit complexity theory to analyze CoT's capabilities [44–46], while other studies examine how CoT structures intermediate steps to decompose tasks [47–50].

CoT improves performance on tasks with shared reasoning patterns, even under distribution shifts [51, 52]. In zero-shot and few-shot settings, it leverages exemplars to apply pre-trained knowledge beyond training data [24]. Recent studies emphasize fine-grained CoT, where detailed intermediate steps enhance task learning and reduce shortcut reliance [53, 54].

Scaling Laws Scaling laws define the relationship between model size, data scale, and performance in LLMs [2, 55, 56]. However, scaling alone is insufficient under significant distribution shifts, as larger models may amplify shortcut learning or overfit to surface patterns [57]. Optimizing the ratio of CoT reasoning in training data and validating its real-world effectiveness remain open challenges. Structured intermediate representations further aid in overcoming global obstacles and improving generalization [45]. Our findings indicate that incorporating CoT reasoning into the training process enhances the robustness of LLMs when addressing distribution shifts, particularly in out-of-distribution tasks.

3. Preliminary

This paper investigates the generalization capabilities of transformer-based language models when applied to compound tasks characterized by dynamic dependencies among sub-tasks. Unlike standard sequential processing, compound tasks involve inter-related actions where the state of each action depends on a varying subset of previous actions. These dependencies are not static but evolve based on the task's progress, posing a unique challenge for models trained on sequential text. For example, consider preparing a complex meal with multiple dishes. The cooking state of pasta, for instance, depends on whether the sauce is freshly made or pre-made, demonstrating how dynamic dependencies can arise. This scenario reflects the intricate structure of many real-world tasks where a simple sequential reading fails to capture the underlying relationships between actions.

We formalize compound tasks through a state transition framework with dynamic dependencies, motivated by real-world scenarios like software development: subtask states (e.g., database setup, API integration) evolve through context-dependent interactions.

3.1. Compound Task Structure

A compound task organizes complex operations through a hierarchical tree of subtasks. The structure comprises atomic actions at leaf nodes and interconnected subtasks at higher levels, with dynamic dependencies governing their execution flow. As illustrated in Figure 2, the system processes these tasks sequentially, evaluating dependencies and aggregating results from completed subtasks to achieve the overall objective. This flexible architecture allows for modification and reorganization of subtasks to accommodate evolving requirements.

To accommodate transformer models' input format requirements, we define the compound task at the token level. Figure 6 in Appendix provides a step-by-step illustration demonstrating how this definition integrates into the task structure.

Definition 1. *Given the input sequence $S = (s_1, \ldots, s_n)$ and subtask state sequence $Q = (q_1, \ldots, q_n)$ where $Q^i = (q_1, \ldots, q_i)$ denotes the prefix subsequence contains the first i states of Q, $CP(n)$ is defined:*

Dynamic Graph Evolution: *At step $i+1$, the dependency graph updates with a function $B : S^{i+1} \times \mathbb{N} \to \mathcal{P}(\mathbb{N})$:*

$$\begin{aligned} G_{i+1} &= G(q_1, \ldots, q_i \mid s_1, \ldots, s_{i+1}) \\ &= \{q_k \mid k \in B(s_1, \ldots, s_{i+1}, i+1)\} \end{aligned} \tag{1}$$

where $B(s_1, \ldots, s_i, j)$ returns the set of indices that determine the dynamic dependencies based on the current

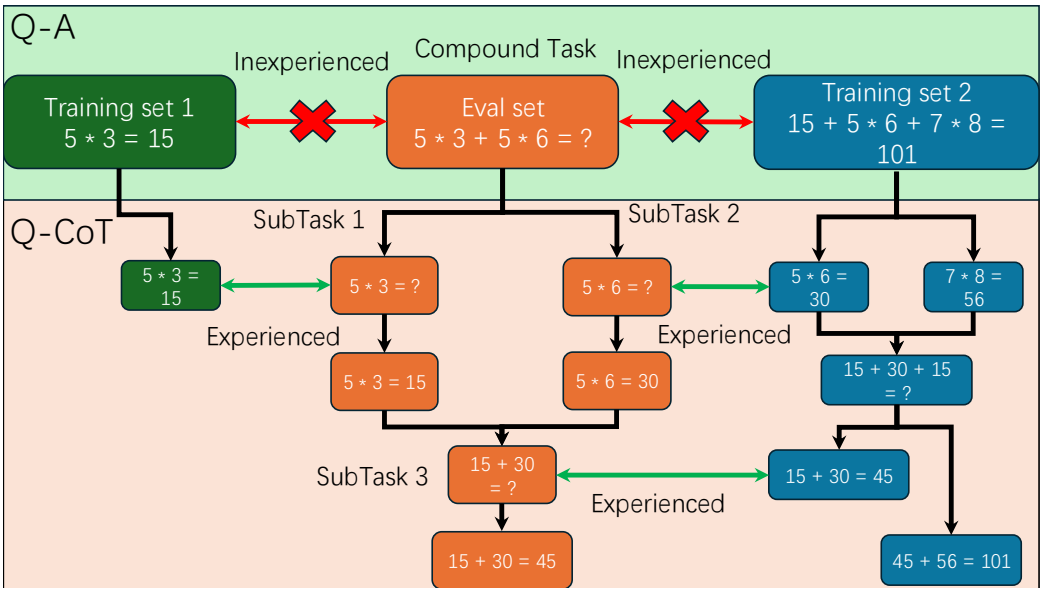


Figure 2: Chain-of-thought alleviates distribution shift by breaking down complex problems into simpler, familiar sub-problems.

state sequence and global semantic constraints.

State Transition Function $F : \mathcal{P}(Q^i) \times Q^i \to q$ *computes the next state using selected predecessors from the prefix subsequence* Q^i*:*

$$q_{i+1} = F(G(q_1, \ldots, q_i \mid s_1, \ldots, s_{i+1}), s_{i+1})$$
$$\textit{where} \quad F(\emptyset) = \textit{Constant}$$

Final Result Operation: *The system's final result is computed through a sequence of operations* L_i *that aggregate states progressively:* $L : q \times q \to R$

$$L_1 = H(\emptyset, q_1) = q_1$$
$$L_i = H(L_{i-1}, q_i) \quad \textit{for } i = 2, \ldots, N$$

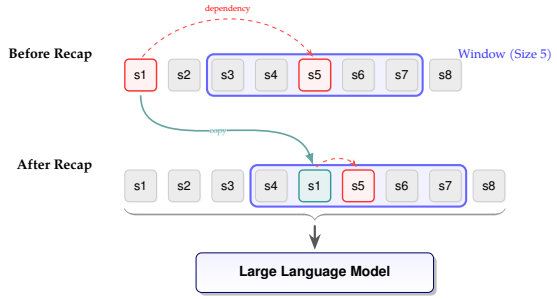
where H *is an aggregation function like maximum, minimum, or summation function and* $H(\emptyset, q) = q$ *that can be specialized as needed.*

3.2. Recap Conditions for Effective Chain of Thought

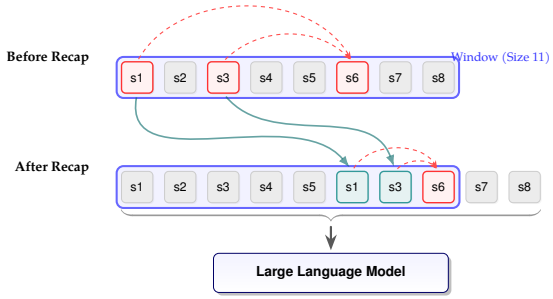
Before introducing the Chain of Thought [5] approach, we establish two fundamental conditions shown in Fig 3 that enable more effective reasoning chains. We emphasize that our analysis in this section rests on *finite floating-point precision constraints* of practical implementations rather than on idealized theoretical attention limits.

1. **Outside Window Recap Condition:** Due to finite precision constraints, only tokens with the top-k attention scores can be effectively recalled. When the distance between the target token position and current position exceeds a threshold, the attention scores become negligible within the given precision, preventing recall of tokens outside this window. This necessitates strategic recapping of tokens within the window to maintain information flow. See Appendix 1 for formal analysis.

2. **Inside Window Recap Condition:** For optimal model performance, tokens within the attention window should be organized following the ground truth causal order, avoiding irrelevant intermediary tokens that could impede convergence. As proven in Appendix 2, including irrelevant tokens in the sequence slows down model convergence. Therefore, we must carefully structure the token sequence within the window (e.g., through techniques like reverse ordering or selective token repetition) to align with the underlying causal structure. Combined with these two recap conditions, for each inference step in a compound task, we must ensure all dependent tokens are both compactly positioned and arranged in their causal order. This strategy enables effective information flow while maintaining computational efficiency. In implementation, since tokens may have multiple dependencies, it is preferable to copy tokens rather than move them.



(a) **Outside Window Recap:** s_1 is copied into the window.



(b) **Inside Window Recap:** s_1 and s_3 are copied closer to s_6.

Figure 3: **Recap mechanism visualization.** Selected tokens are copied to restructure the input sequence so that relevant dependencies fall within the effective attention window before being processed by the LLM.

3.3. CoT Captures the Dynamic State Transitions

Inspired by recent work [44, 46] and building upon our recap conditions, we now introduce a CoT formalization specifically designed to capture the dynamic state transitions in compound tasks. Figure 2 conceptually illustrates how CoT helps to generalize. Rather than allowing the model to learn superficial shortcuts, our CoT format explicitly tracks the evolving dependencies and state transitions that characterize the ground truth recurrent solution.

Definition 2 (Chain of Thought)**.**

$$\begin{aligned} \textit{Input: } & s_1 \mid \cdots \mid s_N \\ \textit{CoT Steps: } & \langle sep \rangle\ s_2 \mid G_2 \mid q_2 \mid L_1 \mid L_2 \\ & \langle sep \rangle\ \cdots \\ & \langle sep \rangle\ s_N \mid G_N \mid q_N \mid L_{N-1} \mid L_N \end{aligned}$$

This format structures each step to track dependencies (G_i), states (q_i), and intermediate results (L_i), guiding the model toward learning true solution dynamics. Remarkably, this principled approach can be implemented with minimal architectural requirements:

Proposition 1. *For any compound problem satisfying Definition 1, and for any input length bound $n \in \mathbb{N}$, there exists an autoregressive Transformer with: 1.Constant depth L, 2.Constant hidden dimension d, 3.Constant number of attention heads H where L, d, and H are independent of n, such that the Transformer correctly generates the Chain-of-Thought solution defined in Definition 2 for all input sequences of length at most n. Furthermore, all parameter values in the Transformer are bounded by $O(poly(n))$.*

4. Experiment

In order to investigate how data collection methods impact LMs' generalization ability, we conducted controlled studies on three compound reasoning tasks with systematic distribution shifts. Our experiments address three key questions: (1) How does direct QA training compare to CoT in OOD generalization? (2) Can increasing data help direct QA training to generalize to OOD data? (3) Can repeat the conditions for each subtask of one compound task help generalization?

4.1. Setting

4.1.1. Task and Experiment setting

We evaluate the generalization ability of LMs through three compound reasoning tasks: (1) Longest Increasing Subsequence (LIS), (2) Multi-Step Path Counting (MPC), and (3) Equation Restoration with Variable Computation (ERVC). Representative examples for all tasks are provided in Appendix F.1.1, F.1.2, and F.1.3.

Longest Increasing Subsequence (LIS): Given an input sequence X^n of integers, compute the length of the longest increasing subsequence. The model is trained on sequences of complexity levels $n_1 = 4$ and $n_2 = 16$, and evaluated on $n_3 = 10$, testing generalization across sequence lengths and positional dependencies.
Multi-Step Path Counting (MPC): Given an input sequence X^n specifying available steps (0 for forbidden, 1 for available), compute the number of unique paths to reach position n using steps of size $\{1, 2, 3\}$. Training uses $n_1 = 20$ and $n_2 = 40$, and evaluation uses $n_3 = 30$, testing generalization under varying combinatorial constraints.
Equation Restoration and Variable Computation (ERVC): Given observations of n variables and their values, recover m underlying linear equations, then compute target variables under new assignments. Training and testing involve distribution shifts in both n and m.

4.1.2. Training Dataset Preparation

For each task, we created datasets in two formats: **1. Question-Answer (Q-A):** Questions paired directly with final answers, representing supervision without reasoning steps (CoT-0%). **2. Question-Chain-of-Thought (Q-CoT):** Questions paired with step-by-step CoT explanations leading to answers, providing process supervision (CoT-100%). To investigate the impact of partial CoT supervision, we also conducted ablation studies using probabilistic CoT dropout. In these settings, each step within a complete CoT demonstration has a probability of being retained in the training data. We explored dropout rates of 30%, 50%, and 70% (CoT-30%, CoT-50%, CoT-70%). We focus on probabilistic dropout as preliminary experiments with fixed-portion CoT dropout showed significant OOD performance degradation.

For scaling experiments, we varied the dataset size from 1k to 30k samples for both Q-A and Q-CoT regimes. Models were trained on the ID data splits and evaluated on both ID and OOD splits to measure generalization gaps.

4.1.3. Model Training and Evaluation

For the Longest Increasing Subsequence and Multi-Step Path Counting tasks, we implemented a 6-layer transformer architecture trained from scratch, featuring an embedding size of 256/512 and 16 attention heads. The Equation Restoration tasks were approached differently, utilizing the Phi-3.5-mini-instruct model as the foundation for task-specific fine-tuning. All model variants incorporate Rotary Position Embedding. In evaluating model performance, we employed different strategies for Q-A and Q-CoT data. For Q-A data in both the Longest Increasing Subsequence and Multi-Step Path Counting tasks, correctness is determined by comparing the token immediately preceding the `<eos>` token. For Q-CoT data, our evaluation extends beyond the final answer, incorporating intermediate computational tokens (such as Q and L) that contribute to the solution. This comprehensive evaluation approach helps mitigate cases where incorrect reasoning processes might accidentally yield correct final answers through guessing.

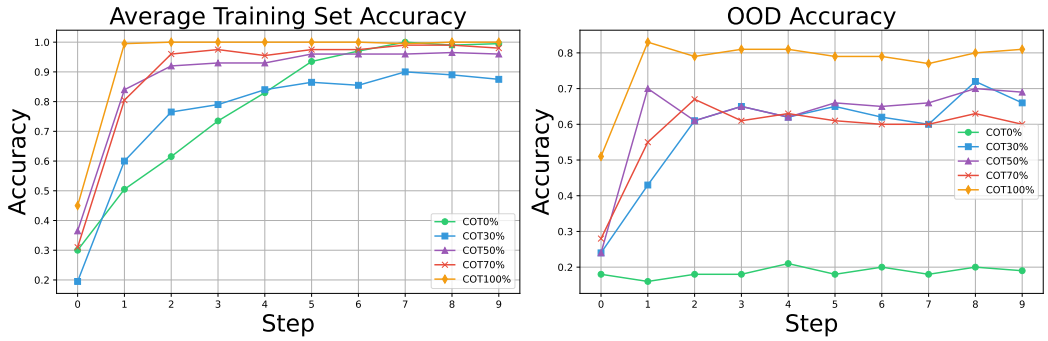
4.2. Results Analysis

Table 1: Performance comparison across MPC, LIS, ERVC tasks, where ID denotes the in-distribution task, OOD denotes out-of-distribution task, ID-OOD denotes the different between ID and OOD performance.

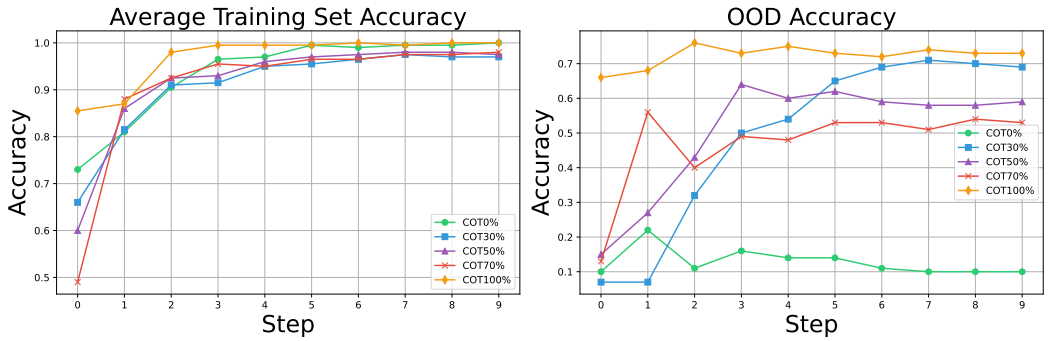
Task	MPC		LIS		ERVC21-43		ERVC21-41	
	Q-A	Q-CoT	Q-A	Q-CoT	Q-A	Q-CoT	Q-A	Q-CoT
ID ↑	0.83	1.0	0.97	0.995	0.39	1.0	0.49	0.95
OOD ↑	0.21	0.81	0.14	0.75	0.02	0.959	0.126	0.905
ID-OOD ↓	0.62	0.19	0.83	0.245	0.37	0.041	0.364	0.045

4.2.1. Generalization Gap and Scaling Curves

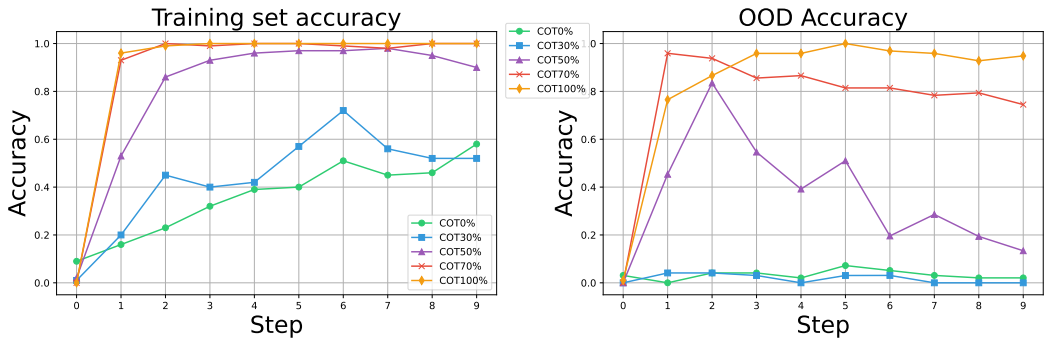
Figure 5 and Table 1 show accuracy scaling curves for MPC and LIS as dataset size increases, comparing CoT ratios (0%, 30%, 50%, 70%, 100%). Consistent with Theorem 3 in Appendix E, models trained with higher CoT (70–100%) achieve better OOD generalization than direct QA (0%), with faster gains and higher sustained accuracy.



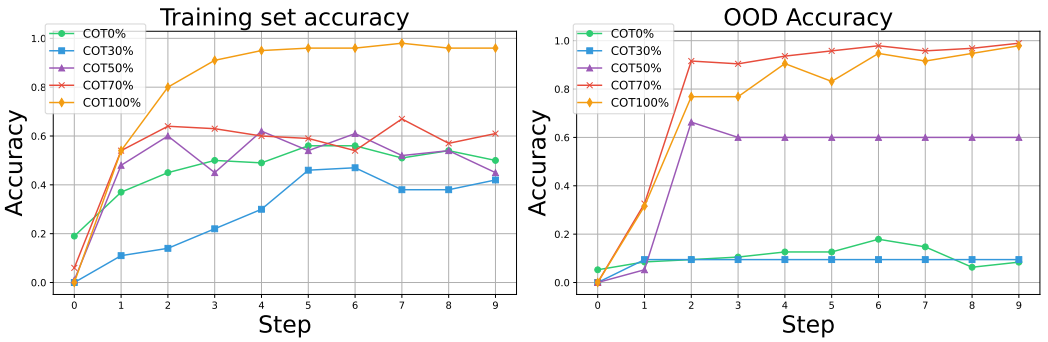
(a) Multi-Step Path Count (MPC)



(b) Longest Increasing Sequence (LIS)

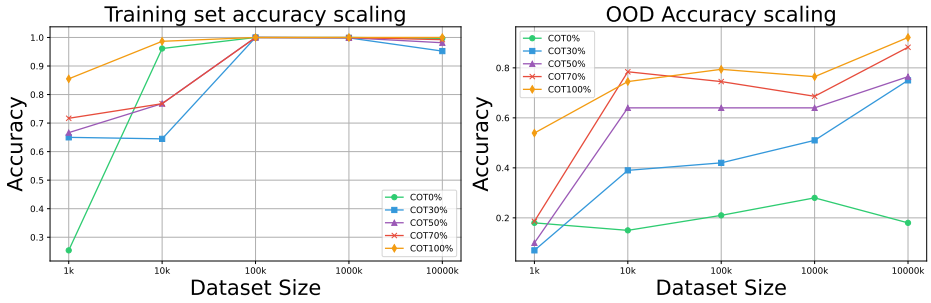


(c) Equation Restoration and Variable Computation (ERVC21-43)

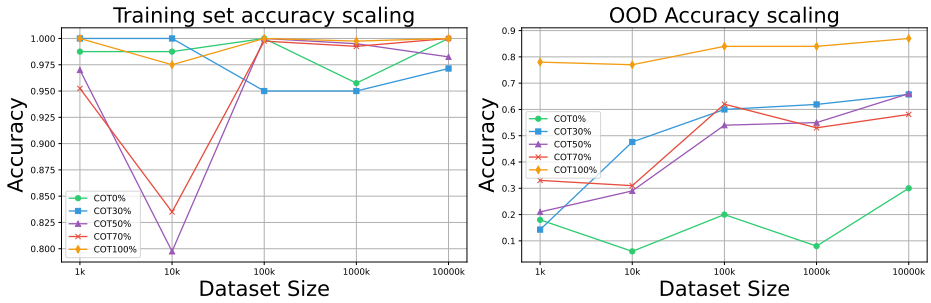


(d) Equation Restoration and Variable Computation (ERVC21-41)

Figure 4: Impact of CoT granularity on In-Distribution and Out-of-Distribution performance across four tasks. Left panels show ID accuracy, right panels show OOD accuracy. While non-CoT models achieve high ID accuracy, they demonstrate significantly lower OOD generalization performance.



(a) Data Scaling Curves in MPC task



(b) Data Scaling Curves in LIS task

Figure 5: The illustration of the impact of data scale on ID and OOD performance in the MPC and LIS tasks. Left: ID performance. Right: OOD performance. We can clearly see that even when the training data increases to 10M, LMs trained without CoT can exhibit near-perfect performance on training tasks, yet they still achieve only about 30% accuracy on OOD data.

For Q-A (0%), OOD performance plateaus quickly, revealing poor generalization. In contrast, Q-CoT models continue improving with scale, reducing the gap. Partial CoT (30–50%) degrades OOD performance, supporting the claim that incomplete reasoning chains weaken prefix coverage, increase KL divergence, and harm generalization.

Overall, the results validate the theory: CoT supervision improves OOD generalization by encoding reasoning steps, effectively transforming OOD problems into in-distribution-like ones.

4.2.2. Granularity-Generalization Tradeoff

Figure 4 demonstrates a clear relationship between CoT step coverage and OOD generalization performance across tasks. Specifically, models trained with more complete CoT steps exhibit superior generalization capabilities on OOD samples. To formally characterize this empirical observation, we develop a theoretical framework in with a simple linear model that quantitatively explains how the omission of CoT steps during training systematically degrades model performance. Our analysis reveals that the degradation follows an exponential decay pattern with respect to the number of missing steps, highlighting the critical importance of maintaining granular reasoning steps for robust generalization.

4.2.3. Data Scaling Experiments

To investigate the effect of training data scaling on generalization, we examined the performance trends as dataset size increased. As depicted in Figure 5, a key observation emerges: models trained without CoT supervision exhibit limited improvement in OOD generalization despite substantial increases in training data size (up to 10,000k samples). While ID accuracy improves with larger datasets for these models, their OOD performance plateaus, indicating that even the increased data cannot help generalization to novel compound tasks with direct Q-A training. In stark contrast, models trained with CoT supervision demonstrate a distinct capacity to translate larger datasets into improved OOD generalization.

4.2.4. Sample Efficiency

As shown in Figure 5, models trained with a higher percentage of CoT examples demonstrate superior sample efficiency in both MPC and LIS scaling experiments. Specifically, models with 70% and 100% CoT (CoT-70% and CoT-100%) achieve a higher OOD accuracy with significantly smaller dataset sizes compared to models trained with lower CoT percentages. This effect is particularly pronounced in the small data regime (1k-10k samples), where CoT-100% maintains a relatively robust OOD performance (0.55-0.75 accuracy) while models with less CoT supervision struggle (below 0.4 accuracy). This suggests that incorporating more reasoning steps through CoT not only improves overall performance but also enables more efficient learning from limited training data.

4.2.5. Recap Condition Ablation

Table 2: Model performance comparisons on the with and without recap condition on the Equation Restoration Task

Task	ERVC21-43		ERVC21-41	
Dataset	Training	OOD	Training	OOD
w/ recap condition	0.974	0.947	0.997	0.952
w/o recap condition	0.436	0.126	0.638	0.558

We investigate the impact of the recap condition on model performance in Equation Restoration tasks in Table 2. For ERVC21-43, models trained with the recap condition achieve a high training accuracy of 0.974 and maintain this level of performance on the OOD set (0.974). In contrast, models trained without the recap condition exhibit significantly lower accuracy on both the training set (0.436) and the OOD set (0.126). A similar trend is observed for ERVC21-41. The model with the recap condition achieves near-perfect training accuracy (0.997) and strong OOD performance (0.952). Conversely, the model without the recap condition shows substantially reduced accuracy in both training (0.638) and OOD settings (0.558). These results strongly suggest that the recap condition of each subtask plays a critical role in the generalization of LMs and and validate our theoretical propositions outlined in Section 3.2

5. Limitation

Limitations We acknowledge three limitations in our study. First, to rigorously isolate the impact of reasoning density, we rely on synthetic tasks; quantifying "granularity" in unstructured natural language remains an open challenge. Second, there is a distinct trade-off in efficiency: while finer-grained CoT achieves sample efficiency (training parsimony), it inevitably increases inference costs (compute parsimony) due to longer sequence generation. Third, we focus on fundamental Transformer dynamics; while our theoretical analysis suggests our findings are structurally invariant, we have not empirically verified these specific scaling laws on massive 100B+ parameter foundation models.

6. Conclusion

In this paper, through controllable systematic experimentation and theoretical analysis in compound tasks, we demonstrate that (1) Direct QA-trained models exhibit dramatic performance degradation under shifts despite high in-distribution accuracy with even 100k data in the single task, (2) CoT granularity directly governs generalization strength, with fine-grained reasoning steps enabling higher OOD accuracy, and (3) CoT training achieves comparable performance to QA paradigms with much less data, underscoring its sample efficiency. Theoretically, we prove that CoT enforces structured reasoning paths, a process amplified by transformer positional embeddings through subtask condition recurrence, and these trends are visually corroborated in Appendix Figure 7, where models trained with 100% CoT maintain high accuracy across varying sequence lengths while direct QA models collapse under OOD conditions despite strong in-domain performance. These findings provide a reliable guide for data collection practices when leveraging LMs for novel tasks. We will explore automated CoT generation and dynamic granularity adaptation to further reduce annotation costs in the future work.

References

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

[2] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

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

[4] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213, 2022.

[5] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[6] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[7] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[8] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[9] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744, 2022.

[10] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[11] Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. The flan collection: Designing data and methods for effective instruction tuning. *arXiv preprint arXiv:2301.13688*, 2023.

[12] Shuai Wang, Weiwen Liu, Jingxuan Chen, Weinan Gan, Xingshan Zeng, Shuai Yu, Xinlong Hao, Kun Shao, Yasheng Wang, and Ruiming Tang. Gui agents with foundation models: A comprehensive survey. *arXiv preprint arXiv:2411.04890*, 2024.

[13] Yuhao Yang, Yue Wang, Dongxu Li, Ziyang Luo, Bei Chen, Chao Huang, and Junnan Li. Aria-ui: Visual grounding for gui instructions. *arXiv preprint arXiv:2412.16256*, 2024.

[14] Huawen Shen, Chang Liu, Gengluo Li, Xinlong Wang, Yu Zhou, Can Ma, and Xiangyang Ji. Falcon-ui: Understanding gui before following user instructions. *arXiv preprint arXiv:2412.09362*, 2024.

[15] Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Weixian Lei, Lijuan Wang, and Mike Zheng Shou. Showui: One vision-language-action model for gui visual agent. *arXiv e-prints*, pages arXiv–2411, 2024.

[16] Gaurav Verma, Rachneet Kaur, Nishan Srishankar, Zhen Zeng, Tucker Balch, and Manuela Veloso. Adaptagent: Adapting multimodal web agents with few-shot learning from human demonstrations. *arXiv preprint arXiv:2411.13451*, 2024.

[17] Panagiotis Giadikiaroglou, Maria Lymperaiou, Giorgos Filandrianos, and Giorgos Stamou. Puzzle solving using reasoning of large language models: A survey. *arXiv preprint arXiv:2402.11291*, 2024.

[18] Soumadeep Saha, Sutanoya Chakraborty, Saptarshi Saha, and Utpal Garain. Language models are crossword solvers. *arXiv preprint arXiv:2406.09043*, 2024.

[19] Jiaxian Guo, Bo Yang, Paul Yoo, Bill Yuchen Lin, Yusuke Iwasawa, and Yutaka Matsuo. Suspicion-agent: Playing imperfect information games with theory of mind aware gpt-4. *arXiv preprint arXiv:2309.17277*, 2023.

[20] Wenqi Zhang, Ke Tang, Hai Wu, Mengna Wang, Yongliang Shen, Guiyang Hou, Zeqi Tan, Peng Li, Yueting Zhuang, and Weiming Lu. Agent-pro: Learning to evolve via policy-level reflection and optimization. *arXiv preprint arXiv:2402.17574*, 2024.

[21] Chenghao Huang, Yanbo Cao, Yinlong Wen, Tao Zhou, and Yanru Zhang. Pokergpt: An end-to-end lightweight solver for multi-player texas hold'em via large language model. *arXiv preprint arXiv:2401.06781*, 2024.

[22] Paul Welsh, Carole Hart, Olia Papacosta, David Preiss, Alex McConnachie, Heather Murray, Sheena Ramsay, Mark Upton, Graham Watt, Peter Whincup, et al. Prediction of cardiovascular disease risk by cardiac biomarkers in 2 united kingdom cohort studies: does utility depend on risk thresholds for treatment? *Hypertension*, 67(2):309–315, 2016.

[23] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

[24] Seungone Kim, Se June Joo, Doyoung Kim, Joel Jang, Seonghyeon Ye, Jamin Shin, and Minjoon Seo. The cot collection: Improving zero-shot and few-shot learning of language models via chain-of-thought fine-tuning. *arXiv preprint arXiv:2305.14045*, 2023.

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

[26] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[27] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey, 2024. URL `https://arxiv.org/abs/2402.06196`.

[28] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. A comprehensive overview of large language models, 2024. URL `https://arxiv.org/abs/2307.06435`.

[29] Xingcheng Xu, Zihao Pan, Haipeng Zhang, and Yanqing Yang. It ain't that bad: Understanding the mysterious performance drop in ood generalization for generative transformer models. In *Proceedings of the Thirty-ThirdInternational Joint Conference on Artificial Intelligence*, IJCAI-2024, page 6578–6586. International Joint Conferences on Artificial Intelligence Organization, August 2024. doi: 10.24963/ijcai.2024/727. URL `http://dx.doi.org/10.24963/ijcai.2024/727`.

[30] Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. Exploring length generalization in large language models, 2022. URL https://arxiv.org/abs/2207.04901.

[31] Chongzhi Zhang, Mingyuan Zhang, Shanghang Zhang, Daisheng Jin, Qiang Zhou, Zhongang Cai, Haiyu Zhao, Xianglong Liu, and Ziwei Liu. Delving deep into the generalization of vision transformers under distribution shifts. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 7277–7286, 2022.

[32] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[33] Yingcong Li, Muhammed Emrullah Ildiz, Dimitris Papailiopoulos, and Samet Oymak. Transformers as algorithms: Generalization and stability in in-context learning. In *International Conference on Machine Learning*, pages 19565–19594. PMLR, 2023.

[34] Jing Qian, Hong Wang, Zekun Li, Shiyang Li, and Xifeng Yan. Faith and fate: Limits of transformers on compositionality, 2022. URL https://arxiv.org/abs/2208.05051.

[35] Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. Investigating the limitations of transformers with simple arithmetic tasks, 2021. URL https://arxiv.org/abs/2102.13019.

[36] Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. Pretrained transformers improve out-of-distribution robustness, 2020. URL https://arxiv.org/abs/2004.06100.

[37] Yi Zhang, Arturs Backurs, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, and Tal Wagner. Unveiling transformers with lego: a synthetic reasoning task, 2023. URL https://arxiv.org/abs/2206.04301.

[38] Yichen Jiang and Mohit Bansal. Avoiding reasoning shortcuts: Adversarial evaluation, training, and model development for multi-hop qa. *arXiv preprint arXiv:1906.07132*, 2019.

[39] Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. When robustness doesn't promote robustness: Synthetic vs. natural distribution shifts on imagenet, 2020. URL https://openreview.net/forum?id=HyxPIyrFvH.

[40] Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. Alphamath almost zero: Process supervision without process, 2024. URL https://arxiv.org/abs/2405.03553.

[41] Tinghui Zhu, Kai Zhang, Jian Xie, and Yu Su. Deductive beam search: Decoding deducible rationale for chain-of-thought reasoning. *COLM*, 2024.

[42] Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. Complexity-based prompting for multi-step reasoning. In *The Eleventh International Conference on Learning Representations*, 2022.

[43] Juno Kim and Taiji Suzuki. Transformers provably solve parity efficiently with chain of thought. *arXiv preprint arXiv:2410.08633*, 2024.

[44] Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. Chain of thought empowers transformers to solve inherently serial problems, 2024. URL https://arxiv.org/abs/2402.12875.

[45] Emmanuel Abbe, Samy Bengio, Aryo Lotfi, Colin Sandon, and Omid Saremi. How far can transformers reason? the globality barrier and inductive scratchpad, 2024. URL https://arxiv.org/abs/2406.06467.

[46] Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing the mystery behind chain of thought: A theoretical perspective, 2023. URL https://arxiv.org/abs/2305.15408.

[47] Jean-Francois Ton, Muhammad Faaiz Taufiq, and Yang Liu. Understanding chain-of-thought in llms through information theory, 2024. URL https://arxiv.org/abs/2411.11984.

[48] Keito Kudo, Yoichi Aoki, Tatsuki Kuribayashi, Shusaku Sone, Masaya Taniguchi, Ana Brassard, Keisuke Sakaguchi, and Kentaro Inui. Think-to-talk or talk-to-think? when llms come up with an answer in multi-step reasoning, 2024. URL https://arxiv.org/abs/2412.01113.

[49] Yijiong Yu. Do llms really think step-by-step in implicit reasoning?, 2025. URL https://arxiv.org/abs/2411.15862.

[50] Chenxiao Yang, Zhiyuan Li, and David Wipf. Chain-of-thought provably enables learning the (otherwise) unlearnable. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=N6pbLYLeej.

[51] Hongkang Li, Meng Wang, Songtao Lu, Xiaodong Cui, and Pin-Yu Chen. How do nonlinear transformers acquire generalization-guaranteed cot ability? In *High-dimensional Learning Dynamics 2024: The Emergence of Structure and Reasoning*, 2024. URL https://openreview.net/forum?id=8pM8IrT6Xo.

[52] Xinyang Hu, Fengzhuo Zhang, Siyu Chen, and Zhuoran Yang. Unveiling the statistical foundations of chain-of-thought prompting methods, 2024. URL https://arxiv.org/abs/2408.14511.

[53] Hoang H Nguyen, Ye Liu, Chenwei Zhang, Tao Zhang, and Philip S Yu. Cof-cot: Enhancing large language models with coarse-to-fine chain-of-thought prompting for multi-domain nlu tasks. *arXiv preprint arXiv:2310.14623*, 2023.

[54] Zheng Chu, Jingchang Chen, Zhongjie Wang, Guo Tang, Qianglong Chen, Ming Liu, and Bing Qin. Towards faithful multi-step reasoning through fine-grained causal-aware attribution reasoning distillation. In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert, editors, *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2291–2315, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics. URL https://aclanthology.org/2025.coling-main.157/.

[55] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

[56] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022. URL https://arxiv.org/abs/2203.15556.

[57] Antonio Valerio Miceli-Barone, Alexandra Birch, and Rico Sennrich. Distributionally robust recurrent decoders with random network distillation, 2022. URL https://arxiv.org/abs/2110.13229.

[58] Xin Men, Mingyu Xu, Bingning Wang, Qingyu Zhang, Hongyu Lin, Xianpei Han, and Weipeng Chen. Base of rope bounds context length, 2024. URL https://arxiv.org/abs/2405.14591.

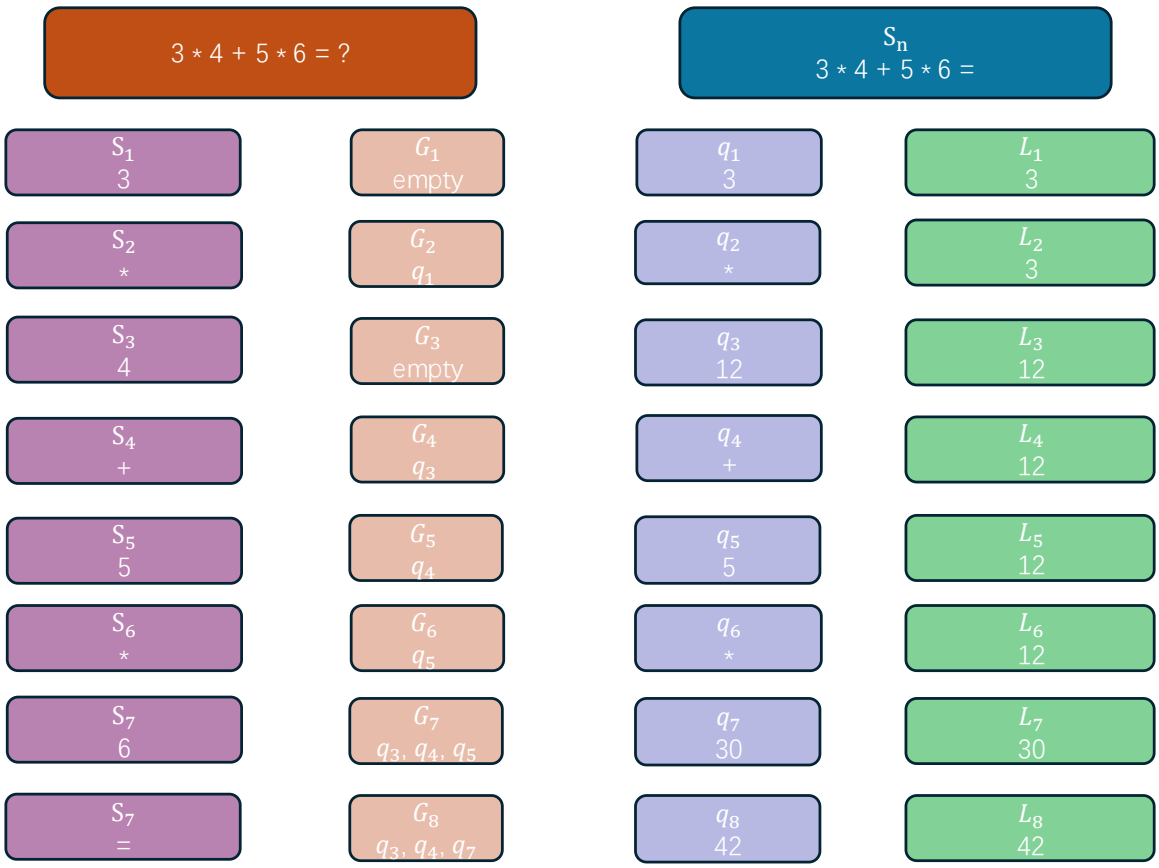


Figure 6: Step by Step explanation on definition 1

A. Explain Compound task in formal definition

As shown in the figure, original tree like data structure can be converted to an array which is easily feeded into language model.

B. Code and Data

We provide an anonymous page, you can following the instruction to generate data, training model and evaluation. `https://github.com/physicsru/Scaling-Curves-of-CoT-Granularity-for-Language-Model-Generalization`

C. Recap Condition Analysis

Theorem 1 (Outside Token Recap Condition under RoPE)**.** *Consider a transformer with Rotary Positional Embedding (RoPE) using angles $\theta_j = 10000^{-2j/d_{model}}$. Given finite computational precision s and minimum resolvable attention score ϵ, there exists a threshold distance $\tau > 0$ such that for all positional distances $d > \tau$:*

$$|A(d)| < \epsilon,$$

where $A(d)$ is the attention score between tokens at distance d. Consequently, tokens beyond $[i_{current} - \tau, i_{current} + \tau]$ cannot be recalled.

Proof. Step 1: Attention Score Formulation The RoPE attention score between positions m and n (distance $d = |m - n|$) is:

$$A(d) = \text{Re}\left[\sum_{j=0}^{d_{\text{model}}/2-1} h_j e^{\mathbf{i}d\theta_j}\right], \quad h_j := q_{[2j:2j+1]}\mathbf{k}^*_{[2j:2j+1]}$$

where h_j encodes query-key interactions for the j-th dimension pair.

Step 2: Abel Transformation[58] Let $S_{j+1} = \sum_{k=0}^{j} e^{\mathbf{i}d\theta_k}$ with $S_0 = 0$. Using summation by parts:

$$\sum_{j=0}^{d_{\text{model}}/2-1} h_j e^{\mathbf{i}d\theta_j} = \sum_{j=0}^{d_{\text{model}}/2-1} (h_j - h_{j+1})S_{j+1}$$

Taking absolute values:

$$|A(d)| \leq \sum_{j=0}^{d_{\text{model}}/2-1} |h_{j+1} - h_j| \cdot |S_{j+1}|$$

Step 3: Bounding Query-Key Differences. Assume that the query and key representations are uniformly bounded so that $\|q\|, \|k\| \leq C$. In particular, since each h_j results from a dot product between sub-vectors from q and k, we have $|h_j| \leq C^2$. Moreover, if we assume that the embeddings vary smoothly with the index j (as expected from the continuity of underlying network nonlinearities and weight matrices), then the mean value theorem implies that the difference

$$|h_{j+1} - h_j|$$

is bounded by a Lipschitz constant. That is, there exists a constant $M = \mathcal{O}(C^2)$ such that for every j

$$|h_{j+1} - h_j| \leq M.$$

A more refined analysis might track this difference in terms of the network's smoothness, but the key point is that each difference is uniformly bounded by a constant depending on C.

Step 4: Analyzing Oscillatory Sums. We now study the partial sum

$$S_{j+1} = \sum_{k=0}^{j} e^{\mathbf{i}d\theta_k}.$$

Two regimes are considered:

(*i*) *Low-frequency regime* ($k \leq j_0$)*:* For sufficiently small indices k, we have θ_k being relatively large so that

$$d\theta_k \gg 1.$$

In this regime the phases $e^{\mathbf{i}d\theta_k}$ change rapidly with k, leading to cancellations among the terms. A standard bound for such oscillatory sums yields

$$\Big|\sum_{k=0}^{j_0} e^{\mathbf{i}d\theta_k}\Big| \leq \frac{2}{|1 - e^{\mathbf{i}d\theta_0}|} = \mathcal{O}(1),$$

since the denominator remains bounded away from zero when $d\theta_0$ is large.

(*ii*) *High-frequency regime* ($k > j_0$)*:* For larger k, the angle θ_k becomes very small (because θ_k decays exponentially with k), so that $d\theta_k \ll 1$. In this case we can use the first-order Taylor expansion:

$$e^{\mathbf{i}d\theta_k} \approx 1 + \mathbf{i}d\theta_k.$$

Thus, for indices $j > j_0$, the sum becomes approximately

$$\sum_{k=j_0+1}^{j} e^{\mathbf{i}d\theta_k} \approx (j - j_0) + \mathbf{i}d \sum_{k=j_0+1}^{j} \theta_k.$$

While the real part grows (almost) linearly in the number of terms, the alternating phases and the small magnitude of $d\theta_k$ imply that additional cancellation occurs when the two regimes are combined. In the worst case one may bound

$$|S_{j+1}| \leq \mathcal{O}(\log d)$$

by appealing to harmonic series type estimates; this bound is loose but captures the fact that the cancellation improves with larger d.

Step 5: Combining the Results. Returning to the bound from Step 2, we have

$$|A(d)| \leq \sum_{j=0}^{d_{\text{model}}/2-1} |h_{j+1} - h_j|\,|S_{j+1}|$$

and using the bounds from Steps 3 and 4,

$$|A(d)| \leq M \sum_{j=0}^{d_{\text{model}}/2-1} \mathcal{O}(\log d) = \mathcal{O}(M\, d_{\text{model}} \log d).$$

In practice, the alternating signs in the summands produce even stronger decay in d, and empirical observations suggest a polynomial decay of the form

$$|A(d)| \leq \mathcal{O}\Big(\frac{M}{\sqrt{d}}\Big).$$

Step 6: Precision Threshold. For a given minimum resolvable attention score ϵ, we require

$$\frac{M}{\sqrt{d}} < \epsilon \quad \Longrightarrow \quad d > \left(\frac{M}{\epsilon}\right)^2.$$

Defining

$$\tau = \left(\frac{M}{\epsilon}\right)^2,$$

we conclude that for any $d > \tau$, it holds that $|A(d)| < \epsilon$. That is, tokens beyond the window indexed by $[i_{\text{current}} - \tau,\, i_{\text{current}} + \tau]$ effectively contribute an attention score below the resolvable threshold.

□

Interpretation:

- *The θ_j schedule creates frequency-dependent decay: high frequencies (small j) attenuate rapidly*
- *Window size $\tau \propto (M/\epsilon)^2$ explains memory limitations in long contexts*
- *Practical implementations must balance d_{model} and precision s for optimal τ*

Theorem 2 (Inside window recap condition under Rope)**.** *Assume: The true function is $s_2 = {w^\circ}^\top e(s_1) + \epsilon$, with $s_2 \perp s_x$ in expectation. Consider two linear models:*

- M_s: $y_s = {w_s}^\top e(s_1)$
- M_l: $y_l = {w_1}^\top e(s_1) + {w_2}^\top e(s_x)$

Under mean-squared-error training:

- $g_s = \frac{\partial L_s}{\partial w_s}$
- $(g_1, g_2) = \left(\frac{\partial L_l}{\partial w_1}, \frac{\partial L_l}{\partial w_2}\right)$

Then in finite-data regimes or with random noise, the gradient component of g_1 along $(w^\circ - w_1)$ is typically smaller than the corresponding component of g_s along $(w^\circ - w_s)$ Formally,

$$\mathbb{E}\left[\langle \mathbf{g}_1, \mathbf{w}^\circ - \mathbf{w}_1 \rangle\right] < \mathbb{E}\left[\langle \mathbf{g}_s, \mathbf{w}^\circ - \mathbf{w}_s \rangle\right],$$

leading to slower convergence for $\mathcal{M}_l$ when s_x is irrelevant.

Proof. We analyze the gradients of both models and demonstrate how irrelevant features in $\mathcal{M}_l$ reduce the effective gradient signal.

Step 1: Express Gradients for Both Models

For model $\mathcal{M}_s$, the loss is:

$$L_s = \mathbb{E}\left[(s_2 - \mathbf{w}_s^\top \mathbf{e}(s_1))^2\right]$$

The gradient becomes:

$$\mathbf{g}_s = -2\mathbb{E}\left[\mathbf{e}(s_1)\mathbf{e}(s_1)^\top\right](\mathbf{w}^\circ - \mathbf{w}_s) \tag{2}$$

For model $\mathcal{M}_l$, the gradient for $\mathbf{w}_1$ is:

$$\mathbf{g}_1 = -2\mathbb{E}\left[\mathbf{e}(s_1)\mathbf{e}(s_1)^\top\right](\mathbf{w}^\circ - \mathbf{w}_1) + 2\mathbb{E}\left[\mathbf{w}_2^\top \mathbf{e}(s_x)\mathbf{e}(s_1)\right] \tag{3}$$

Step 2: Compare Gradient Components

The inner product for $\mathcal{M}_l$ contains two terms:

$$\begin{aligned}\mathbb{E}\left[\langle \mathbf{g}_1, \mathbf{w}^\circ - \mathbf{w}_1 \rangle\right] = &\underbrace{-2(\mathbf{w}^\circ - \mathbf{w}_1)^\top \mathbb{E}\left[\mathbf{e}(s_1)\mathbf{e}(s_1)^\top\right](\mathbf{w}^\circ - \mathbf{w}_1)}_{\text{Matches } \mathcal{M}_s} \\ &+ \underbrace{2\mathbb{E}\left[\mathbf{w}_2^\top \mathbf{e}(s_x)\mathbf{e}(s_1)^\top (\mathbf{w}^\circ - \mathbf{w}_1)\right]}_{\text{Additional term}}\end{aligned} \tag{4}$$

Step 3: Effect of Irrelevant Features

Since s_x is irrelevant ($s_2 \perp s_x$):

- Population truth: $\mathbf{w}_2 = \mathbf{0}$
- Finite data allows $\mathbf{w}_2^\epsilon$ to fit noise ϵ
- Induces spurious correlation: $\mathbb{E}\left[\mathbf{e}(s_x)\mathbf{e}(s_1)^\top\right] \neq \mathbf{0}$

This makes the additional term:

$$2\mathbb{E}\left[\mathbf{w}_2^\top \mathbf{e}(s_x)\mathbf{e}(s_1)^\top (\mathbf{w}^\circ - \mathbf{w}_1)\right] \neq 0$$

which *reduces* the magnitude of the gradient component.

Step 4: Convergence Comparison

For $\mathcal{M}_s$:

$$\mathbb{E}\left[\langle \mathbf{g}_s, \mathbf{w}^\circ - \mathbf{w}_s \rangle\right] = -2(\mathbf{w}^\circ - \mathbf{w}_s)^\top \mathbb{E}\left[\mathbf{e}(s_1)\mathbf{e}(s_1)^\top\right](\mathbf{w}^\circ - \mathbf{w}_s)$$

For $\mathcal{M}_l$, the additional term in Equation 3 creates:

$$\mathbb{E}\left[\langle \mathbf{g}_1, \mathbf{w}^\circ - \mathbf{w}_1 \rangle\right] < \mathbb{E}\left[\langle \mathbf{g}_s, \mathbf{w}^\circ - \mathbf{w}_s \rangle\right]$$

Conclusion
The irrelevant features in $\mathcal{M}_l$ reduce the gradient component in the direction of $\mathbf{w}^\circ$, leading to slower convergence compared to $\mathcal{M}_s$. □

D. CoT simulates the target solution

Definition 3 (Chain of Thought)**.**

$$\begin{aligned} &\textit{Input: } s_1 \mid \cdots \mid s_N \\ &\textit{CoT Steps: } \langle \textit{sep} \rangle\ s_2 \mid G_2 \mid q_2 \mid L_1 \mid L_2 \\ &\qquad\qquad \langle \textit{sep} \rangle\ \cdots \\ &\qquad\qquad \langle \textit{sep} \rangle\ s_N \mid G_N \mid q_N \mid L_{N-1} \mid L_N \end{aligned}$$

Proposition 2. *For any compound problem satisfying Definition 1, and for any input length bound $n \in \mathbb{N}$, there exists an autoregressive Transformer with:*

- *Constant depth L*
- *Constant hidden dimension d*
- *Constant number of attention heads H*

where L, d, and H are independent of n, such that the Transformer correctly generates the Chain-of-Thought solution defined in Definition 2 for all input sequences of length at most n. Furthermore, all parameter values in the Transformer are bounded by $O(poly(n))$.

D.1. Constructive Proof

We prove this theorem by constructing a Transformer architecture with 4 blocks, where each block contains multiple attention heads and feed-forward networks (FFNs). The key insight is that we can simulate each step of the Chain-of-Thought solution using a fixed number of attention heads and a fixed embedding dimension. The attention mechanism is primarily used to select and retrieve relevant elements from the input and previous computations, while the FFNs approximate the required functions G, B, etc. By maintaining constant depth, width, and number of heads per layer, we ensure the Transformer's architecture remains independent of the input length, while still being able to generate arbitrarily long Chain-of-Thought solutions. The parameter complexity of $O(\text{poly}(n))$ arises from the need to handle inputs and intermediate computations of length n, but importantly, this only affects the parameter values and not the model architecture itself.

D.2. Embedding Structure

For position k, define the input embedding:

$$x_k^{(0)} = (e_k^{\text{isInput}}, e_k^{\text{isState}}, e^{\text{isDependence}}, e^{\text{isL}}, e_k^{\text{q}}, e_k^{\text{d}}, e_k^{\text{L}}, e_k^{\text{sep}}, e_k^{\text{step}}, k, 1)$$

where:

- $e_k^{\text{isInput}} \in \{0, 1\}$: Input token indicator
- $e_k^{\text{isState}} \in \{0, 1\}$: State position indicator
- $e^{\text{isDependence}} \in \{0, 1\}$: Dependency marker
- $e^{\text{isL}} \in \{0, 1\}$: Aggregation result indicator
- $e_k^{\text{q}} \in \mathbb{R}^{d_q}$: State value embedding
- $e_k^{\text{d}} \in \mathbb{R}^{d_d}$: Dependency graph embedding
- $e_k^{\text{L}} \in \mathbb{R}^{d_L}$: Aggregation value embedding
- $e_k^{\text{sep}} \in \{0, 1\}$: Step separator indicator
- $e_k^{\text{step}} \in \mathbb{N}$: Current step index
- $k \in \mathbb{N}$: Position encoding
- 1: Bias term

D.3. Block Constructions

Block 1: Input Processing and State Identification Define attention heads $A_1^{(1)}, A_2^{(1)}, A_3^{(1)}$ with parameters:

$$\begin{aligned} Q_1^{(1)} &= W_1^q[e_k^{\text{isInput}}] \\ K_1^{(1)} &= W_1^k[e_j^{\text{isInput}}]_{j<k} \\ V_1^{(1)} &= W_1^v[j]_{j<k} \end{aligned}$$

The second head tracks state positions:

$$\begin{aligned} Q_2^{(1)} &= W_2^q[e_k^{\text{isState}}] \\ K_2^{(1)} &= W_2^k[e_j^{\text{isState}}]_{j<k} \\ V_2^{(1)} &= W_2^v[j]_{j<k} \end{aligned}$$

The third head tracks step indices through separators:

$$\begin{aligned} Q_3^{(1)} &= W_3^q[e_k^{\text{sep}}] \\ K_3^{(1)} &= W_3^k[e_j^{\text{sep}}]_{j<k} \\ V_3^{(1)} &= W_3^v[\text{count}(e_j^{\text{sep}})]_{j<k} \end{aligned}$$

Lemma 1. *The first block correctly identifies positions through attention scoring:*

1. *For input positions, $A_1^{(1)}$ scoring gives:*

$$\textit{score}_1(q_k, k_j) = \begin{cases} 1 & \textit{if } e_j^{\textit{isInput}} = 1 \\ 0 & \textit{otherwise} \end{cases}$$

   Thus $V_1^{(1)}$ returns positions of input tokens

2. *For state positions, $A_2^{(1)}$ scoring gives:*

$$\textit{score}_2(q_k, k_j) = \begin{cases} 1 & \textit{if } e_j^{\textit{isState}} = 1 \\ 0 & \textit{otherwise} \end{cases}$$

   Thus $V_2^{(1)}$ returns positions of states

3. *For step indices, $A_3^{(1)}$ counts separators up to position k:*

$$\textit{count}(e_j^{\textit{sep}}) = \sum_{l \le j} e_l^{\textit{sep}}$$

   Thus $V_3^{(1)}$ returns the current step index

Block 2: Dependency Graph Construction Define three attention heads $A_1^{(2)}, A_2^{(2)}, A_3^{(2)}$ implementing dependency selection:

$$\begin{aligned}
A_1^{(2)} : Q_1^{(2)} &= W_2^q[e_k^{\text{step}}] \\
K_1^{(2)} &= W_2^k[e_j^{\text{input}}]_{j<k} \\
V_1^{(2)} &= W_2^v[j]_{j<k} \\
A_2^{(2)} : Q_2^{(2)} &= W_3^q[e_k^{\text{step}}] \\
K_2^{(2)} &= W_3^k[e_j^{\text{step}}]_{j<k} \\
V_2^{(2)} &= W_3^v[B(s_1, \ldots, s_{i+1}, i+1)]_{j<k} \\
A_3^{(2)} : Q_3^{(2)} &= W_4^q[e_k^{\text{step}}] \\
K_3^{(2)} &= W_4^k[j]_{j<k} \\
V_3^{(2)} &= W_4^v[e_j^{\text{q}}]_{j<k}
\end{aligned}$$

Lemma 2. *Block 2 correctly implements $G_{i+1} = \{q_k | k \in B(s_1, \ldots, s_{i+1}, i+1)\}$ through:*

1. First attention head $A_1^{(2)}$ gathers input sequence up to current step i+1:

$$z_1^{(2)} = \{s_j | j \leq i+1\}$$

2. Second attention head $A_2^{(2)}$ computes indices from B using gathered inputs:

$$z_2^{(2)} = B(z_1^{(2)}, i+1)$$

3. Third attention head $A_3^{(2)}$ selects states using computed indices:

$$z_3^{(2)} = \{e_j^q | j \in z_2^{(2)}\}$$

Therefore, the composition $z_3^{(2)}(z_2^{(2)}(z_1^{(2)}))$ correctly implements G_{i+1} by:

1. *Gathering relevant input sequence*
2. *Computing dependency indices using B*
3. *Selecting corresponding states*

The correctness follows from attention scoring:

$$\text{score}_1(q_k, k_j) = \begin{cases} 1 & \text{if } j \leq i+1 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{score}_2(q_k, k_j) = \begin{cases} 1 & \text{if } j \in B(s_1, \ldots, s_{i+1}, i+1) \\ 0 & \text{otherwise} \end{cases}$$

$$\text{score}_3(q_k, k_j) = \begin{cases} 1 & \text{if } j \in z_2^{(2)} \\ 0 & \text{otherwise} \end{cases}$$

Block 3: State Transition Define attention mechanism implementing F:

$$\begin{aligned}
A_1^{(3)} : Q_1^{(3)} &= W_3^q[e_k^{\text{isState}}] \\
K_1^{(3)} &= W_3^k[e_j^{\text{isDependence}}]_{j<k} \\
V_1^{(3)} &= W_3^v[e_j^{\text{q}}]_{j<k} \\
A_2^{(3)} : Q_2^{(3)} &= W_4^q[e_k^{\text{isState}}] \\
K_2^{(3)} &= W_4^k[e_j^{\text{isInput}}]_{j<k} \\
V_2^{(3)} &= W_4^v[e_j^{\text{input}}]_{j<k}
\end{aligned}$$

Lemma 3. *The state transition function F is correctly computed through:*

$$q_{i+1} = F(G_{i+1}, s_{i+1}) = FFN(z_1^{(3)}, z_2^{(3)})$$

where $z_1^{(3)} = A_1^{(3)}(e_j^q \mid j \in B(s_1, \ldots, s_{i+1}, i+1))$ represents the states selected by G_{i+1} from Block 2, and $z_2^{(3)} = A_2^{(3)}(s_{i+1})$ represents the current input token.

Block 4: Result Aggregation Define two attention heads $A_1^{(4)}, A_2^{(4)}$ for implementing H:

$$\begin{aligned} A_1^{(4)} : Q_1^{(4)} &= W_4^q[e_k^{\text{isL}}] \\ K_1^{(4)} &= W_4^k[e_j^{\text{isL}}]_{j<k} \\ V_1^{(4)} &= W_4^v[e_j^{\text{L}}]_{j<k} \\ A_2^{(4)} : Q_2^{(4)} &= W_5^q[e_k^{\text{isL}}] \\ K_2^{(4)} &= W_5^k[e_j^{\text{isState}}]_{j<k} \\ V_2^{(4)} &= W_5^v[e_j^{\text{q}}]_{j<k} \end{aligned}$$

Lemma 4. *Block 4 correctly implements the aggregation function H through:*

1. For $i = 1$ (base case):

$$score_1(q_k, k_j) = 0, \quad score_2(q_k, k_j) = \begin{cases} 1 & \text{if } e_j^{isState} = 1 \\ 0 & \text{otherwise} \end{cases}$$

Therefore $L_1 = H(\emptyset, q_1) = q_1$ since only $A_2^{(4)}$ activates to select q_1

2. For $i > 1$:

$$score_1(q_k, k_j) = \begin{cases} 1 & \text{if } e_j^{isL} = 1 \text{ and j is the latest L position} \\ 0 & \text{otherwise} \end{cases}$$

$$score_2(q_k, k_j) = \begin{cases} 1 & \text{if } e_j^{isState} = 1 \text{ and j corresponds to } q_i \\ 0 & \text{otherwise} \end{cases}$$

Therefore:

$$\begin{aligned} z_1^{(4)} &= A_1^{(4)}(e_k^L) = L_{i-1} \text{ (previous aggregation result)} \\ z_2^{(4)} &= A_2^{(4)}(e_k^q) = q_i \text{ (current state)} \\ L_i &= FFN(z_1^{(4)}, z_2^{(4)}) = H(L_{i-1}, q_i) \end{aligned}$$

The FFN is constructed to implement the specific aggregation operation of H (e.g., max, min, or sum).

Proposition 3 (Block Transitions)**.** *The blocks connect sequentially where:*

1. *Block 1 output provides input positions, state positions and step indices*
2. *Block 2 implements dependency function G to gather required states*
3. *Block 3 uses gathered dependencies and current input to compute new states via F*
4. *Block 4 implements H to aggregate states into final result*

Each transition preserves information through residual connections.

E. Distribution Shift Analysis for Q-A vs. Q-CoT

Understanding the distribution shift between traditional Q-A and CoT approaches is crucial for analyzing OOD generalization. This section examines how CoT's intermediate reasoning steps

influence the alignment between training and evaluation distributions. We analyze a dynamic state-transition system where problems of different lengths are processed through either direct Q-A or through CoT reasoning 2. Specifically, we compare: 1. **Q-A training/evaluation** on problem lengths $\{n_1, n_2\}$ (train) and n_3 (eval). 2. **Q-CoT training/evaluation** on problem lengths $\{n_1, n_2\}$ (train) and n_3 (eval). We assume $n_1 < n_3 < n_2$, and let

$$\mathcal{D}_{\text{train}}^{\text{Q-A}} = \Big\{(X^{n_1}, Y^{n_1}),\, (X^{n_2}, Y^{n_2})\Big\},$$

$$\mathcal{D}_{\text{eval}}^{\text{Q-A}} = \Big\{(X^{n_3}, Y^{n_3})\Big\},$$

be the respective datasets in the Q-A setup. Analogously, in the Q-CoT setup, each problem instance is expanded into subproblem–subsolution pairs (the chain-of-thought states). Denote:

$$\mathcal{D}_{\text{train}}^{\text{Q-CoT}} = \Big\{\big(X^{n_1}, \{q_i^{(n_1)}\}, Y^{n_1}\big),\, \big(X^{n_2}, \{q_i^{(n_2)}\}, Y^{n_2}\big)\Big\},$$

$$\mathcal{D}_{\text{eval}}^{\text{Q-CoT}} = \Big\{\big(X^{n_3}, \{q_i^{(n_3)}\}, Y^{n_3}\big)\Big\}.$$

Here $\{q_i^{(n)}\}$ denotes the chain-of-thought states or subsolutions for a length-n problem.

We let $P_{\text{train}}^{\text{Q-A}}$, $P_{\text{eval}}^{\text{Q-A}}$, $P_{\text{train}}^{\text{Q-CoT}}$, and $P_{\text{eval}}^{\text{Q-CoT}}$ be the corresponding data or model-induced distributions over Q-A setting or Q-CoT setting.

Our analysis begins with a fundamental result about the structure of CoT sequences in our dynamic system.

E.1. Prefix-Substructure Theorem in a Dynamic State-Transition System

A key property of chain-of-thought sequences is that prefix relationships between input sequences carry over to their states, enabling shorter problems $\mathrm{CP}(n_3)$ to embed within longer ones $\mathrm{CP}(n_2)$.

Lemma 5 (Prefix Substructure)**.** *Let $S^{short} = (s_1, s_2, \ldots, s_{n_2})$ and let $S^{long} = (s_1, s_2, \ldots, s_{n_3})$ be its prefix, with $n_3 < n_2$. Suppose their chain-of-thought sequences under the same (G, F) are*

$$Q^{short} = \big(q_1^{short}, q_2^{short}, \ldots, q_{n_2}^{short}\big) \quad \textit{and}$$

$$Q^{long} = \big(q_1^{long}, q_2^{long}, \ldots, q_{n_3}^{long}\big).$$

Then for all $1 \le i \le n_3$, we have

$$q_i^{long} = q_i^{short}.$$

Hence $\big(q_1^{long}, \ldots, q_{n_3}^{long}\big)$ is exactly the prefix of $\big(q_1^{short}, \ldots, q_{n_2}^{short}\big)$.

This property suggests CoT's intermediate steps provide natural bridges between problems of different lengths, potentially easing distribution shift concerns.

E.2. Training Size Effects on Distribution Shift Through Prefix Coverage

Building upon the prefix-substructure property, we now quantify the distribution shift between training and evaluation sets for both Q-A and Q-CoT approaches. This analysis reveals how CoT's intermediate reasoning steps can potentially mitigate distribution shift effects.

Theorem 3 (KL Divergence Reduction via Chain-of-Thought Coverage)**.** *Let m_1 and m_2 denote the number of training sequences of length n_1 and n_2, respectively, with total training size $M = m_1 + m_2$. Suppose the evaluation set consists of m_3 sequences of length n_3, with $n_1 < n_3 < n_2$. Let k be the input vocabulary size.*

(*Coverage Probability Definition*)

Let the coverage probability $P_{cover}(M)$ *be defined as the probability that an intermediate state $q_{1:n_3}^{(\text{eval})}$, corresponding to a prefix of length n_3 from an evaluation sequence, is contained within the support of the training*

distribution; that is, $q_{1:n_3}^{(\text{eval})}$ exists as the prefix of some chain-of-thought sequence $q_{1:n_2}^{(\text{train})}$ in the training data. Formally,

$$P_{cover}(M) = \mathbb{P}\left(q_{1:n_3}^{(\text{eval})} \in \left\{q_{1:n_3}^{(\text{train})} \,:\, q_{1:n_3}^{(\text{train})} \text{ is prefix of length } n_3 \text{ from some training } q_{1:n_2}^{(\text{train})}\right\}\right).$$

Then, the following holds:

1. ***Prefix Coverage:*** *The coverage probability for evaluation prefixes of length n_3 is*

$$P_{cover}(M) = \frac{m_2}{m_3 k^{n_3}},$$

where m_3 is the size of the evaluation set and $m_2 \leq m_3 k^{n_3}$.

2. ***KL Divergence Reduction:*** *The Kullback-Leibler (KL) divergence between evaluation and training distributions over intermediate reasoning steps (Q-CoT supervision) satisfies*

$$D_{\text{KL}}\Big(P_{\text{eval}}^{\text{Q-CoT}} \,\|\, P_{\text{train}}^{\text{Q-CoT}}\Big) \leq (1 - P_{cover}(M))\, D_{\text{KL}}\Big(P_{\text{eval}}^{\text{Q-A}} \,\|\, P_{\text{train}}^{\text{Q-A}}\Big).$$

Thus, supervision on intermediate steps—i.e., Chain-of-Thought (CoT)—dramatically reduces the distributional gap between training and evaluation compared to direct QA supervision.

3. ***Complete Coverage Yields No Shift:*** *In the limit where $m_2 = m_3 k^{n_3}$ (all evaluation prefixes are seen in training),*

$$D_{\text{KL}}\Big(P_{\text{eval}}^{\text{Q-CoT}} \,\|\, P_{\text{train}}^{\text{Q-CoT}}\Big) = 0.$$

Remark (Interpretation and Practical Role of CoT). This result explicitly quantifies how CoT mitigates the KL divergence (distribution shift) between training and evaluation. The coverage probability $P_{\text{cover}}(M)$ measures how likely it is for a generated intermediate (reasoning-step) prefix in evaluation to appear in the CoT-augmented training data. As $P_{\text{cover}}(M)$ increases with more or better-constructed CoT training data, the effective distribution shift shrinks, ensuring robust generalization—even where direct QA fails. In practice, incomplete or lower-granularity CoT chains reduce $P_{\text{cover}}(M)$, revealing a concrete trade-off between annotation effort and OOD generalization.

E.3. Proof for Theorem 3

Proof. We prove the KL divergence bound by decomposing the distributions over covered and uncovered prefixes. Let $P_{\text{train}}^{\text{Q-CoT}}$ and $P_{\text{eval}}^{\text{Q-CoT}}$ denote the training and evaluation distributions under Q-CoT, respectively.

Step 1: Event Space Partitioning

Define two disjoint events for any evaluation sample $x = (X^{n_3}, \{q_i^{(n_3)}\}, Y^{n_3})$:

- $\mathcal{E}_{\text{cover}}$: The prefix $\{q_i^{(n_3)}\}_{i=1}^{n_3}$ exists in some length-n_2 training sample.
- $\mathcal{E}_{\text{uncover}}$: The prefix $\{q_i^{(n_3)}\}_{i=1}^{n_3}$ is absent from all training samples.

By Lemma 5, $\mathcal{E}_{\text{cover}}$ occurs when the evaluation prefix matches at least one length-n_2 training sequence's prefix. The probabilities satisfy:

$$P_{\text{cover}} = \mathbb{P}(\mathcal{E}_{\text{cover}}), \quad 1 - P_{\text{cover}} = \mathbb{P}(\mathcal{E}_{\text{uncover}}).$$

where P_{cover} is calculated as:

$$P_{\text{cover}} = \frac{m_2}{m_3 k^{n_3}}$$

Derivation: Each length-n_2 training sample contains a unique prefix of length n_3 (Lemma 5). With m_2 samples, we can cover m_2 distinct prefixes. The total number of possible prefixes is $m_3 k^{n_3}$ (m_3

evaluation problems, each with k^{n_3} possible prefixes). Thus, the coverage probability follows the ratio.

Step 2: Distributional Decomposition

Using the law of total probability, we express:

$$P_{\text{eval}}^{\text{Q-CoT}} = P_{\text{cover}} \cdot P_{\text{eval}|\mathcal{E}_{\text{cover}}} + (1 - P_{\text{cover}}) \cdot P_{\text{eval}|\mathcal{E}_{\text{uncover}}}$$
$$P_{\text{train}}^{\text{Q-CoT}} = P_{\text{cover}} \cdot P_{\text{train}|\mathcal{E}_{\text{cover}}} + (1 - P_{\text{cover}}) \cdot P_{\text{train}|\mathcal{E}_{\text{uncover}}}$$

where:

- $P_{\text{eval}|\mathcal{E}_{\text{cover}}}$: Evaluation distribution restricted to covered prefixes
- $P_{\text{train}|\mathcal{E}_{\text{cover}}}$: Training distribution restricted to covered prefixes
- $P_{\text{eval}|\mathcal{E}_{\text{uncover}}}$: Evaluation distribution for uncovered prefixes
- $P_{\text{train}|\mathcal{E}_{\text{uncover}}}$: Training distribution for uncovered prefixes

Step 3: KL Divergence Expansion with Total Expectation

From the KL divergence definition:

$$D_{\text{KL}}\left(P_{\text{eval}}^{\text{Q-CoT}} \,\|\, P_{\text{train}}^{\text{Q-CoT}}\right) = \mathbb{E}_{x \sim P_{\text{eval}}^{\text{Q-CoT}}}\left[\log \frac{P_{\text{eval}}^{\text{Q-CoT}}(x)}{P_{\text{train}}^{\text{Q-CoT}}(x)}\right]$$

Apply the law of total expectation by conditioning on $\mathcal{E}_{\text{cover}}$ and $\mathcal{E}_{\text{uncover}}$:

$$= \mathbb{P}(\mathcal{E}_{\text{cover}}) \cdot \mathbb{E}_{x|\mathcal{E}_{\text{cover}}}\left[\log \frac{P_{\text{eval}}^{\text{Q-CoT}}(x|\mathcal{E}_{\text{cover}})}{P_{\text{train}}^{\text{Q-CoT}}(x|\mathcal{E}_{\text{cover}})}\right] + \mathbb{P}(\mathcal{E}_{\text{uncover}}) \cdot \mathbb{E}_{x|\mathcal{E}_{\text{uncover}}}\left[\log \frac{P_{\text{eval}}^{\text{Q-CoT}}(x|\mathcal{E}_{\text{uncover}})}{P_{\text{train}}^{\text{Q-CoT}}(x|\mathcal{E}_{\text{uncover}})}\right]$$

Step 4: Handling Covered Cases

Under $\mathcal{E}_{\text{cover}}$, Lemma 5 guarantees that the CoT states $\{q_i^{(n_3)}\}$ in evaluation samples exactly match those in training samples. This implies:

$$P_{\text{eval}|\mathcal{E}_{\text{cover}}}(x) = P_{\text{train}|\mathcal{E}_{\text{cover}}}(x), \quad \forall x \in \mathcal{E}_{\text{cover}}$$

Therefore:

$$\mathbb{E}_{x|\mathcal{E}_{\text{cover}}}\left[\log \frac{P_{\text{eval}|\mathcal{E}_{\text{cover}}}}{P_{\text{train}|\mathcal{E}_{\text{cover}}}}\right] = \mathbb{E}_{x|\mathcal{E}_{\text{cover}}}[\log 1] = 0$$

Step 5: Uncovered Cases Reduce to Q-A

For $x = (X^{n_3}, \{q_i^{(n_3)}\}, Y^{n_3}) \in \mathcal{E}_{\text{uncover}}$, the absence of matching prefixes in training data implies the model cannot leverage CoT states $\{q_i^{(n_3)}\}$ during inference. We formally analyze this degradation:

Under Q-CoT, the generation process factors as:

$$P^{\text{Q-CoT}}(Y|X) = \sum_{\{q_i\}} P(Y|X, \{q_i\}) P(\{q_i\}|X)$$

where:

- $P(\{q_i\}|X)$: Probability of generating CoT states $\{q_i\}$ given input X
- $P(Y|X, \{q_i\})$: Probability of answer Y given X and CoT states

When $\{q_i^{(n_3)}\}$ is uncovered ($\mathcal{E}_{\text{uncover}}$), the model lacks training data to estimate either:

- The CoT state distribution $P(\{q_i\}|X)$
- The answer likelihood $P(Y|X, \{q_i\})$

Thus, the model *cannot* utilize the CoT decomposition and must marginalize over all possible $\{q_i\}$:

$$P^{\text{Q-CoT}}(Y|X) = \mathbb{E}_{\{q_i\}\sim P(\{q_i\}|X)}\left[P(Y|X, \{q_i\})\right]$$

Without CoT supervision on $\{q_i^{(n_3)}\}$, two condition assumes:

1. **Untrained CoT States**: If $\{q_i^{(n_3)}\}$ never appears in training, $P(\{q_i\}|X)$ becomes a *uniform prior* over possible CoT sequences (by maximum entropy principle).
2. **Uninformative Likelihood**: The answer likelihood $P(Y|X, \{q_i\})$ reduces to $P^{\text{Q-A}}(Y|X)$ because the model cannot associate $\{q_i\}$ with Y without training signals.

Thus:

$$P^{\text{Q-CoT}}(Y|X) = \sum_{\{q_i\}} \underbrace{P^{\text{Q-A}}(Y|X)}_{\text{Uninformative}} \cdot \underbrace{\frac{1}{k^{n_3}}}_{\text{Uniform } P(\{q_i\}|X)} = P^{\text{Q-A}}(Y|X)$$

with expansion of KL divergence of Q-A

$$D_{\text{KL}}\left(P^{\text{Q-A}}_{\text{eval}} \,\|\, P^{\text{Q-A}}_{\text{train}}\right) = \mathbb{E}_{x\sim P^{\text{Q-A}}_{\text{eval}}}\left[\log \frac{P^{\text{Q-A}}_{\text{eval}}(x)}{P^{\text{Q-A}}_{\text{train}}(x)}\right]$$

$$= \mathbb{P}(\mathcal{E}_{\text{cover}}) \cdot \mathbb{E}_{x|\mathcal{E}_{\text{cover}}}\left[\log \frac{P^{\text{Q-A}}_{\text{eval}}(x|\mathcal{E}_{\text{cover}})}{P^{\text{Q-A}}_{\text{train}}(x|\mathcal{E}_{\text{cover}})}\right] + \mathbb{P}(\mathcal{E}_{\text{uncover}}) \cdot \mathbb{E}_{x|\mathcal{E}_{\text{uncover}}}\left[\log \frac{P^{\text{Q-A}}_{\text{eval}}(x|\mathcal{E}_{\text{uncover}})}{P^{\text{Q-A}}_{\text{train}}(x|\mathcal{E}_{\text{uncover}})}\right]$$

Notice that

$$\mathbb{E}_{x|\mathcal{E}_{\text{cover}}}\left[\log \frac{P^{\text{Q-A}}_{\text{eval}}(x|\mathcal{E}_{\text{cover}})}{P^{\text{Q-A}}_{\text{train}}(x|\mathcal{E}_{\text{cover}})}\right] \le \mathbb{E}_{x|\mathcal{E}_{\text{uncover}}}\left[\log \frac{P^{\text{Q-A}}_{\text{eval}}(x|\mathcal{E}_{\text{uncover}})}{P^{\text{Q-A}}_{\text{train}}(x|\mathcal{E}_{\text{uncover}})}\right]$$

since covered prefix will decrease the KL divergence via probability decomposition

$$D_{\text{KL}}\left(P^{\text{Q-A}}_{\text{eval}} \,\|\, P^{\text{Q-A}}_{\text{train}}\right) \ge \mathbb{E}_{x|\mathcal{E}_{\text{uncover}}}\left[\log \frac{P^{\text{Q-A}}_{\text{eval}}(x|\mathcal{E}_{\text{uncover}})}{P^{\text{Q-A}}_{\text{train}}(x|\mathcal{E}_{\text{uncover}})}\right] = D_{\text{KL}}\left(P^{\text{Q-A}}_{\text{eval}|\mathcal{E}_{\text{uncover}}} \,\|\, P^{\text{Q-A}}_{\text{train}|\mathcal{E}_{\text{uncover}}}\right)$$

Therefore, for $x \in \mathcal{E}_{\text{uncover}}$:

$$D_{\text{KL}}\left(P_{\text{eval}|\mathcal{E}_{\text{uncover}}} \,\|\, P_{\text{train}|\mathcal{E}_{\text{uncover}}}\right) = D_{\text{KL}}\left(P^{\text{Q-A}}_{\text{eval}|\mathcal{E}_{\text{uncover}}} \,\|\, P^{\text{Q-A}}_{\text{train}|\mathcal{E}_{\text{uncover}}}\right) \le D_{\text{KL}}\left(P^{\text{Q-A}}_{\text{eval}} \,\|\, P^{\text{Q-A}}_{\text{train}}\right) = \text{KL}_{\text{base}}$$

Step 6: Final Inequality

Combining all terms:

$$D_{\text{KL}}\left(P^{\text{Q-CoT}}_{\text{eval}} \,\|\, P^{\text{Q-CoT}}_{\text{train}}\right) = \underbrace{P_{\text{cover}} \cdot 0}_{\text{Covered term}} + \underbrace{(1 - P_{\text{cover}}) \cdot \text{KL}_{\text{base}}}_{\text{Uncovered term}}$$

Hence:

$$D_{\text{KL}}\left(P^{\text{Q-CoT}}_{\text{eval}} \,\|\, P^{\text{Q-CoT}}_{\text{train}}\right) \le (1 - P_{\text{cover}}) \cdot \text{KL}_{\text{base}}$$

The equality holds when $P_{\text{cover}} \in [0, 1]$. When $m_2 = m_3 k^{n_3}$, we have $P_{\text{cover}} = 1$, making the KL divergence zero. □

F. Quantitation Analysis Drop of CoT

Theorem 4 (CoT Accuracy Degradation)**.** *Let s_{input} be the input text, s_{ans} be the unique correct answer, and $s_1, \ldots, s_k$ be the exact required sequence of perfect Chain-of-Thought (CoT) tokens where:*

1. ***Completeness**: $P(s_{ans} \mid s_1, \ldots, s_k, s_{input}) = 1$*
2. ***Uniqueness**: No other token sequence produces s_{ans}*
3. ***Conditional Independence**: $P(s_1, \ldots, s_k \mid s_{input}) = \prod_{i=1}^{k} P(s_i \mid s_{input})$*
4. ***Training Deficiency**: For any CoT token s_j excluded during training, $P(s_j \mid s_{input})$ drops from 1 to $1 - \epsilon$*

When $l < k$ CoT tokens are lost/mishandled during inference, the final answer accuracy satisfies:

$$P(s_{ans} \mid s_{input}) = (1 - \epsilon)^l$$

Proof. By the uniqueness condition, only the full sequence $s_1, \ldots, s_k$ guarantees s_{ans}. Let $\mathcal{L}$ be the set of l compromised tokens. The probability of maintaining correctness is:

$$P(s_{\text{ans}} \mid s_{\text{input}}) = \underbrace{\prod_{j \in \mathcal{L}} P(s_j \mid s_{\text{input}})}_{\text{Lost tokens}} \cdot \underbrace{\prod_{i \notin \mathcal{L}} P(s_i \mid s_{\text{input}})}_{\text{Preserved tokens}}$$

For preserved tokens ($i \notin \mathcal{L}$), full training ensures $P(s_i \mid s_{\text{input}}) = 1$. For lost tokens ($j \in \mathcal{L}$), training deficiency gives $P(s_j \mid s_{\text{input}}) = 1 - \epsilon$. Thus:

$$P(s_{\text{ans}} \mid s_{\text{input}}) = (1 - \epsilon)^l \cdot 1^{k-l} = (1 - \epsilon)^l$$

This equality holds because any deviation from the exact CoT sequence (due to lost tokens) eliminates the chance of correctness by the uniqueness condition. □

F.1. Experiments

For the Longest Increasing Subsequence and Multi-Step Path Counting tasks, we implemented a 6-layer transformer architecture trained from scratch, featuring an embedding size of 256/512 and 16 attention heads. The model training utilized the following hyperparameters: maximum sequence length (`-maxlen`) 524, maximum data samples (`-maxdata`) 524, vocabulary size (`-vocab`) 59, number range (`-num_range`) 50, weight decay 0.05, learning rate 1×10^{-3}, dropout 0.1, batch size 256, 1 training epoch, warmup ratio 0.1, model dimension (`-dmodel`) 256, number of heads 16, number of layers 6, with chain-of-thought supervision (`-chain`), rotary position embedding (`-rpe`), and supervised fine-tuning (`-sft`). Model training was distributed on 4 GPUs using `torchrun -nproc_per_node=4`.

For the Equation Restoration tasks, we adopted a different approach using the Phi-3.5-mini-instruct model as the backbone for task-specific fine-tuning. The restoring equation experiments used the following additional hyperparameters: maximum equation length (`max_len`) 300, maximum normalization factor (`max_norm`) 1, maximum training samples (`max_samples`) 200,000, and micro train batch size (`micro_train_batch_size`) 32.

F.1.1. LIS

Chain of thought is like the following:

$$
\begin{aligned}
&48 \quad 49 \quad 26 \quad 47 < sep > \\
&48| < empty >= 48 \quad 1:1 \to 1 < sep > \\
&49|48 \quad 1 = 49 \quad 2:1 \to 2 < sep > \\
&26| < empty >= 26 \quad 1:2 \to 2 < sep > \\
&47|26 \quad 1 = 47 \quad 2:2 \to 2
\end{aligned}
$$

F.1.2. MPC

Chain of thought is like following:

$$
\begin{aligned}
&0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 10,8 < sep > \\
&1,0,1 \to 0 < sep > \\
&2,1,1 \quad 0 \to 1 < sep > \\
&3,1,1 \quad 0 \quad 1 \to 2 < sep > \\
&4,0,0 \quad 1 \quad 2 \to 0 < sep > \\
&5,0,1 \quad 2 \quad 0 \to 0 < sep > \\
&6,1,2 \quad 0 \quad 0 \to 2 < sep > \\
&7,1,0 \quad 0 \quad 2 \to 2 < sep > \\
&8,0,0 \quad 2 \quad 2 \to 0
\end{aligned}
$$

F.1.3. Equation Restoration and Variable Computation

Input is: Data: $data_1 : Condor = 6, Cheetah = 1$.
$data_2 : Condor = 12, Cheetah = 3$.
Question: Assume all relations between variables are linear combinations. If the number of Cheetah equals 5, then what is the number of Condor?

Question: Assume all relations between variables are linear combinations. If the number of Leopard equals 5, the number of Rhino equals 3, the number of Koala equals 6, then what is the number of Black_Bear?

Solution

Defining Variables
Known Variables:
Cheetah as $c_1 = 5$

Unknown Variables:
Target Variable: Condor as c_2

Restoring Relations
List all variable names in each data point: $[c_2, c_1], [c_2, c_1]$
Deduplicate them: $[c_2, c_1]$
There is 1 distinct group, implying 1 distinct linear relationship to be determined.
Examining each relationship:

Relation 1:
Exploring relation for c_2:
There are 2 variables in the data beginning with c_2: Hence, 2 coefficients are required, and at least 2

data points are needed.

Let the coefficients on the right side of the equation be K_1 and K_2.
Recap variables: $['c_2', 'c_1']$
Define the equation of relation 1:
$c_2 = K_1 \cdot c_1 + K_2$

Using data points data_1 and data_2:
$\text{data}_1 : c_2 = 6, c_1 = 1$
Equation 1: $6 = K_1 \cdot 1 + K_2$
$\text{data}_2 : c_2 = 12, c_1 = 3$
Equation 2: $12 = K_1 \cdot 3 + K_2$

Solve the system of equations using Gaussian Elimination:
Initialize:
Equation 1: $1 \cdot K_1 + 1 \cdot K_2 = 6$
Equation 2: $3 \cdot K_1 + 1 \cdot K_2 = 12$

Swap Equation 1 with Equation 2:
Equation 1: $3 \cdot K_1 + 1 \cdot K_2 = 12$
Equation 2: $1 \cdot K_1 + 1 \cdot K_2 = 6$

Multiply Equation 1 by 1 and subtract 3 times Equation 2:
$(\text{Equation 1}) \cdot 1 : 3 \cdot K_1 + 1 \cdot K_2 = 12$
$(\text{Equation 2}) \cdot 3 : 3 \cdot K_1 + 3 \cdot K_2 = 18$
New Equation 2: $-2 \cdot K_2 = -6$

Recap updated equations:
Equation 1: $3 \cdot K_1 + 1 \cdot K_2 = 12$
Equation 2: $-2 \cdot K_2 = -6$

Solve for K_2:
$-2 \cdot K_2 = -6$
$K_2 = \frac{-6}{-2} = 3$

Solve for K_1:
$3 \cdot K_1 = 12 - 1 \cdot K_2$
$3 \cdot K_1 = 12 - 3 = 9$
$K_1 = \frac{9}{3} = 3$

Recap the equation:
$c_2 = K_1 \cdot c_1 + K_2$
Estimated coefficients: $K_1 = 3, K_2 = 3$
Final equation: $c_2 = 3 \cdot c_1 + 3$

Calculation with Restored Relations:
Using the equation $c_2 = 3 \cdot c_1 + 3$:
Known variables: $c_1 = 5$

$c_2 = 3 \cdot 5 + 3 = 15 + 3 = 18$

Recap Target Variable:
Condor (c_2) = 18

Conclusion: The number of Condor equals 18.

F.2. Out-of-distribution Comparison Across Input length

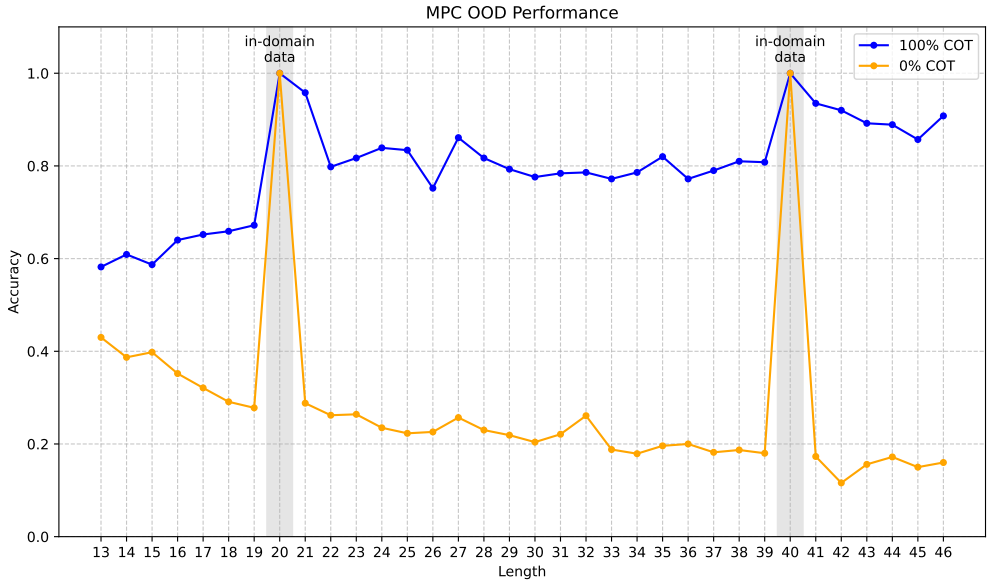
The comparison 7 reveals the critical role of Chain-of-Thought prompting in improving models' OOD generalization. Both MPC (a) and LIS (b) demonstrate substantially higher accuracy when equipped with 100% COT (blue lines) compared to without COT. This performance gap is particularly pronounced in out-of-domain regions, where models without COT show severe degradation (dropping below 0.2 accuracy). The consistent superior performance of COT-enabled models, especially in maintaining accuracy above 0.8 across different sequence lengths, underscores how COT prompting serves as a crucial mechanism for enhancing models' ability to generalize beyond their training distribution.

Impact Statement

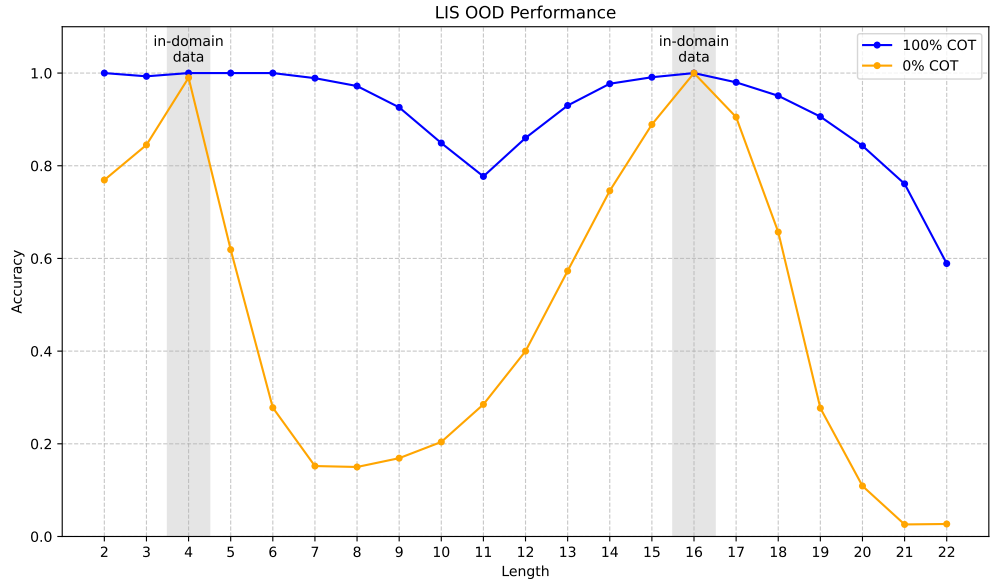
This paper offers a novel perspective, demonstrating the indispensable role of CoT in enhancing the generalization capabilities of LMs. Through theoretical analysis and comprehensive empirical experimentation, we establish CoT as a critical enabler of robust out-of-distribution generalization. Crucially, this work provides valuable guidance for the development of effective data curation strategies, specifically for collecting data that maximizes the benefits of CoT training. This guidance is directly applicable to the industrial deployment of LMs and the fine-tuning of large models for novel tasks, offering a pathway to improve the generalization and real-world utility of these models through informed data acquisition methodologies.

Use of Large Language Models

Large Language Models (LLMs) were used exclusively for language polishing, including grammar refinement and improving readability. All research ideas, methodological contributions, experimental design, analysis, and writing of technical content were conceived and carried out solely by the authors. The LLMs did not generate or influence any scientific claims, results, or interpretations. The authors take full responsibility for the content of this paper.



(a)



(b)

Figure 7: Comparison of Out-Of-Distribution (OOD) performance between MPC and LIS models under different Chain-of-Thought (COT) conditions across varying sequence lengths.