Reasoning Without Generalization:

A Comparison of Direct, Chain-of-Thought, and Scratchpad Training Paradigms on Out-of-Distribution Arithmetic

Tayyaba Jadoon

Dickinson College

COMP 560: Independent Research, Spring 2026

Supervisor: Professor John MacCormick

Abstract

This paper investigates whether structured reasoning at training time improves out-of-distribution (OOD) generalization in small transformer language models. Using a 0.79M parameter character-level transformer trained on arithmetic addition tasks, we compare three training paradigms: Direct prediction, Chain-of-Thought (CoT), and Scratchpad reasoning. All three models achieve near-perfect accuracy on in-distribution 2-digit addition (99.2%–100.0%) following the adoption of a reversed-digit representation format. However, all three paradigms fail completely on 3-digit OOD addition, with 0.0% accuracy across the board. Error analysis reveals that the CoT and Scratchpad models generate structurally plausible reasoning traces but produce incorrect final answers, suggesting the models learn to imitate the format of reasoning rather than execute the underlying algorithm. Notably, the Scratchpad model spontaneously developed structured

digit-by-digit reasoning steps including carry notation, without being explicitly trained to do so, but consistently truncated its reasoning at 2 digit positions even on 3-digit inputs. These results contribute to growing evidence that small transformers may pattern-match to reasoning formats rather than acquire generalizable knowledge.

1. Introduction

A central question in the study of large language models is whether explicit reasoning processes, such as chain-of-thought prompting, improve a model's ability to generalize beyond its training distribution. While chain-of-thought and scratchpad methods have shown impressive results at scale (Wei et al., 2022; Nye et al., 2021), it remains unclear whether these benefits come from genuine algorithmic learning or from improved pattern matching within the training distribution.

This work examines this question in a controlled, small-scale setting. We train a character-level transformer (0.79M parameters, the nanoGPT architecture) on integer addition and evaluate whether three different ways: Direct prediction, Chain-of-Thought (CoT), and Scratchpad, differ in their ability to generalize from 2-digit to 3-digit addition. The addition task is well-suited to this investigation because its algorithmic structure (column-wise addition with carry propagation) is clearly defined, generalization can be executed clearly, and the computation can be done on normal hardware.

The central research question is: does training with intermediate reasoning steps, whether structured (CoT) or unstructured (Scratchpad), improve length generalization in small transformers? The findings suggest the answer is no, at least at this scale. All three models

achieve near-perfect in-distribution accuracy but fail completely on longer inputs, raising questions about what these models actually learn when trained with reasoning supervision.

2. Related Work

Chain-of-thought prompting (Wei et al., 2022) demonstrated that explicitly mentioning intermediate reasoning steps significantly improves performance on complex tasks in large language models. Concurrently, Nye et al. (2021) proposed the scratchpad approach, allowing models to write arbitrary intermediate tokens before producing a final answer, showing improvements in multi-step computation tasks. These works motivate a natural question: do the benefits of intermediate reasoning also exist at smaller model scales and under distributional shift?

Lee et al. (2023) directly examined whether small transformers can learn arithmetic generalization, finding systematic failure patterns when models are evaluated on longer sequences than seen during training. Their work on arithmetic in small transformers is particularly relevant to our experimental setup, as both investigate carry propagation failures as a mechanism for OOD breakdown. Similarly, Dziri et al. (2023) provide a broader theoretical argument in Faith and Fate: Limits of Transformers on Compositionality, arguing that transformers reduce reasoning tasks to pattern matching over training examples rather than acquiring genuine compositional rules. Their findings align closely with our observation that models generate structurally correct reasoning traces but fail on unseen problem lengths.

Prior work on representation choice (e.g., reversed digit ordering) has shown that input format significantly affects arithmetic learning in neural models. Our experiments confirm this: switching from standard to reversed digit representation transformed near-chance baseline performance (~1%) into near-perfect 2-digit accuracy (~99%), without any change to model architecture or training objective.

3. Experimental Setup

3.1 Model Architecture

We use nanoGPT, a minimal GPT-2-style character-level transformer with 0.79M parameters. All experiments use the same architecture; only the training data and supervision format differ across conditions. Training was conducted on Apple M-series hardware (MPS backend) for approximately 8,000 iterations per model, with a batch size of 128 and a block size sufficient to accommodate the longest training sequence format. Final training losses converged to approximately 0.19 (train) and 0.22 (validation) for the scratchpad model.

3.2 Dataset and Representation

We generated synthetic datasets of integer addition problems. A key design decision, informed by preliminary experiments, was to represent numbers in reversed-digit format: writing digits from least significant to most significant (e.g., 47 becomes "74", 58 becomes "85"). This format aligns the addition algorithm with left-to-right token generation and

dramatically improved in-distribution accuracy. Without reversal, the Direct baseline achieved only ~1% accuracy on 2-digit addition even after 5,000 training iterations. With reversal, it reached 99.2%.

Training data consists of 2-digit addition problems. Test sets include 500 in-distribution 2-digit problems and 500 out-of-distribution 3-digit problems, stratified by number of carry operations (0, 1, 2, and for 3-digit: 3 carries).

3.3 Training Paradigms

We compare three training conditions:

- Direct: The model is trained to output the sum directly after the input expression (e.g., "74+85=" → "361").
- Chain-of-Thought (CoT): The training target includes explicit digit-by-digit addition steps with carry values before the final answer, following the format established by Wei et al. (2022) and Nye et al. (2021).
- Scratchpad: The model is given a designated scratch region (delimited by <scratch> tokens) in which it may write any intermediate tokens before producing the final answer. No specific format is prescribed for the scratch region.

3.4 Evaluation

We evaluate using exact-match accuracy: a prediction is correct only if it exactly matches the gold answer. Results are reported both overall and stratified by number of carry

operations, since carry propagation is the key algorithmic challenge that increases in complexity with digit length.

4. Results

4.1 Overall Accuracy

Table 1 summarizes accuracy across all three paradigms on in-distribution (2-digit) and out-of-distribution (3-digit) test sets.

Table 1: Accuracy by paradigm, digit length, and carry count (n=500 per condition)

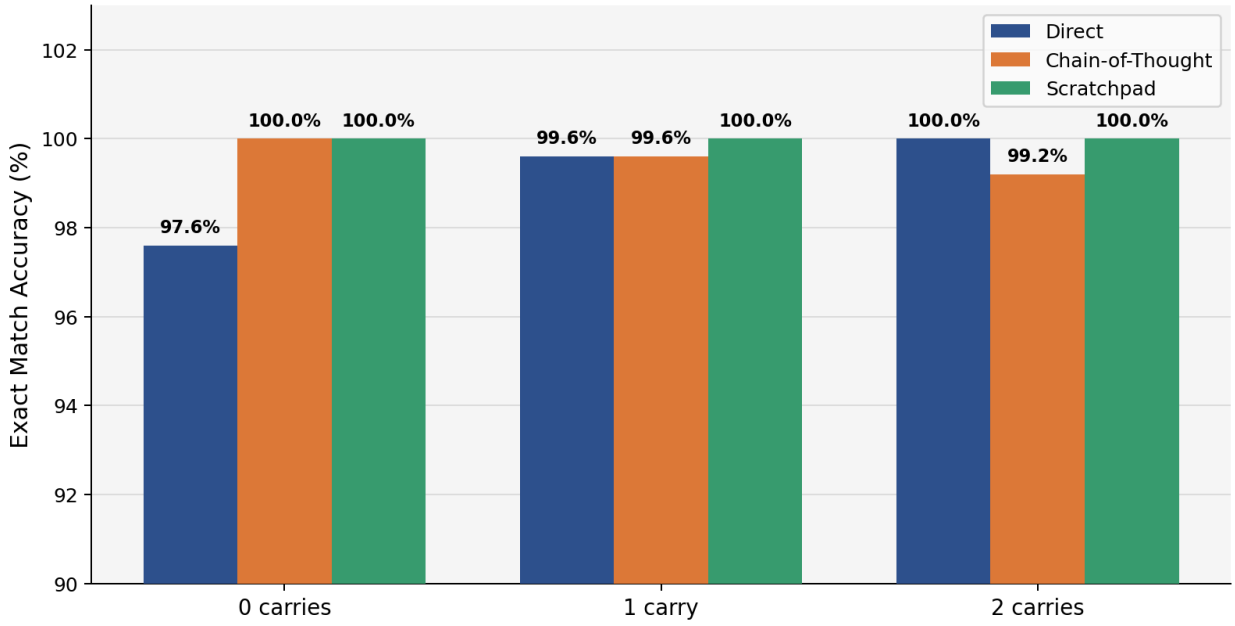
Paradigm	2-Digit Acc.	2-Digit (c=0)	2-Digit (c=1)	2-Digit (c=2)	3-Digit Acc.
Direct	99.2%	97.6%	99.6%	100.0%	0.0%
Chain-of-Thought	99.6%	100.0%	99.6%	99.2%	0.0%
Scratchpad	100.0%	100.0%	100.0%	100.0%	0.0%

The results reveal an interesting pattern: all three paradigms achieve near-perfect accuracy on in-distribution 2-digit addition (99.2%–100.0%) but fail completely on 3-digit OOD problems (0.0% across all carry levels). This complete collapse is uniform; there is no gradient of degradation, no paradigm that performs better than another at 3 digits, and no carry category that proves easier or harder. The failure is absolute and consistent.

4.2 Carry-Stratified Analysis

Within the 2-digit condition, performance is very stable across carry counts for all three paradigms. The Direct model shows slight variation (97.6% at c=0, 100.0% at c=2), while CoT and Scratchpad are near-ceiling across all carry levels. This suggests that all three models successfully internalize the carry operation for the training distribution.

Figure 2: 2-Digit Accuracy by Number of Carries



For 3-digit problems, all carry categories (c=0, c=1, c=2, c=3) achieve exactly 0.0% accuracy for every paradigm. The failure is not concentrated in high-carry problems as might be expected if carry propagation was the problem.

4.3 Qualitative Error Analysis

Qualitative inspection of 3-digit errors reveals important differences across paradigms despite equivalent accuracy. Direct model errors typically produce short outputs that

match the structure of 2-digit answers; the model appears to ignore or truncate the third digit of the input, generating answers in the range of 2-3 digits rather than the expected 3-4 digit sums.

CoT model errors are more revealing. The model generates full digit-by-digit reasoning traces that are structurally correct; it performs digit additions and carry tracking at each position; but the final assembled answer is wrong. This suggests the model has learned to produce reasoning-like output without performing the actual computation.

The Scratchpad model produces the most interesting qualitative behavior. Despite receiving no explicit supervision about the format of intermediate reasoning, it spontaneously developed structured digit-by-digit steps within the scratch region, including the use of 'c' notation for carry values, the same notation used by the CoT model. However, like the CoT model, it produces plausible-looking reasoning steps that lead to incorrect final answers on 3-digit inputs. Closer inspection of the 3-digit outputs reveals a more specific failure mode: the model consistently writes exactly 2 digit steps inside the scratch region and then stops, never attempting the third digit position. For example, given 542+775=(gold=822), the model generates *<scratch> d0 : 5 + 7 + 0 = 12 -> 2, c1 ; d1: 4 + 7 + 1 = 12 -> 2, c1 </scratch>*, and outputs 221 — correctly computing the first two digit positions but failing to extend the procedure to the third. This suggests the model learned the reasoning procedure specifically for 2-digit problems, not a generalizable algorithm that could be extended to longer inputs.

5. Discussion

5.1 Imitation of Reasoning vs. Algorithmic Execution

The central finding of this work is that structured reasoning supervision does not improve OOD generalization in small transformers, at least for the arithmetic task studied here. All three paradigms fail equally at 3-digit addition, despite substantial differences in the richness of their training signal.

The qualitative error patterns suggest a specific failure mode: the models learn to imitate the format of correct reasoning rather than to execute the underlying algorithm. CoT and Scratchpad models produce reasoning traces that look like they should lead to correct answers; they perform the right steps in the right structure; but the final outputs are wrong. This is consistent with the hypothesis advanced by Dziri et al. (2023) that transformers pattern-match to training examples rather than acquiring compositional rules.

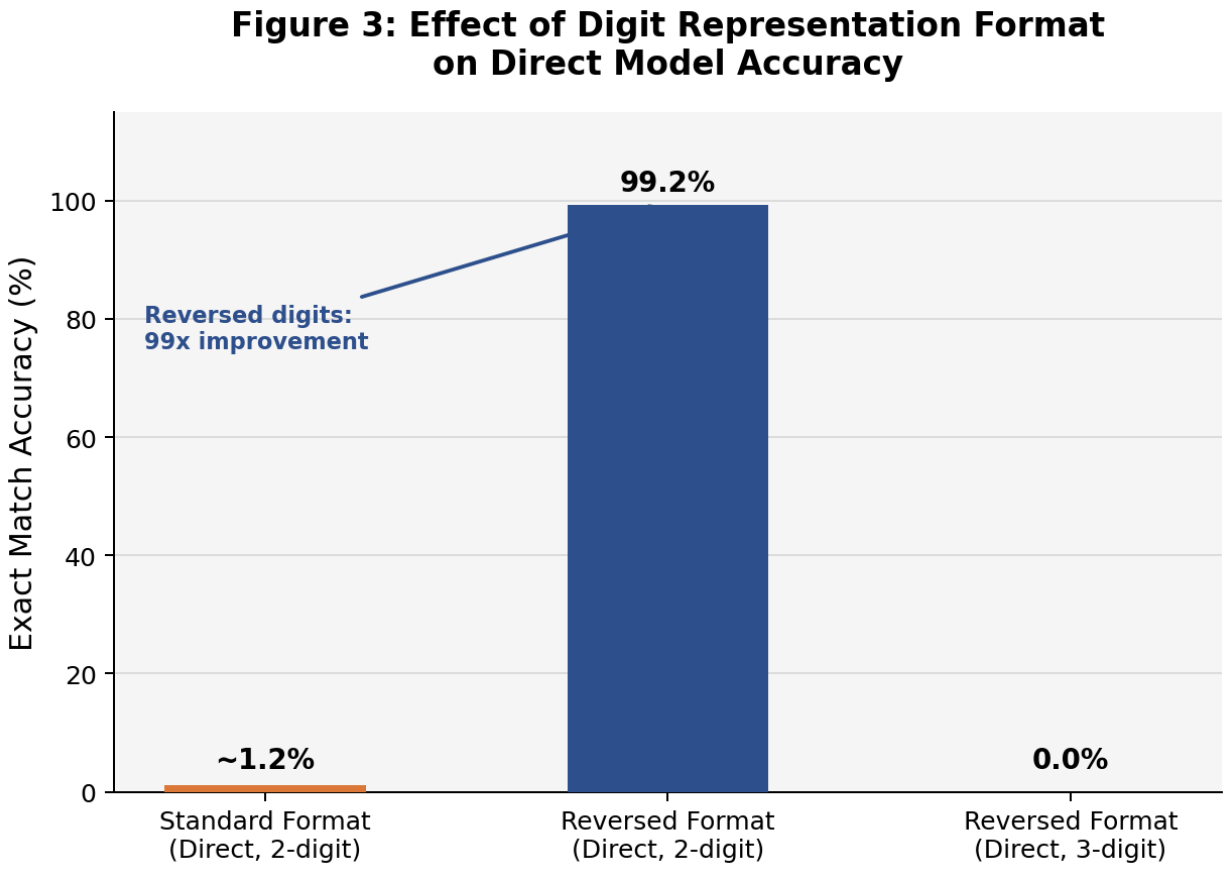
The spontaneous development of structured carry notation in the Scratchpad model is particularly interesting. The model learned that reasoning-like intermediate tokens are associated with correct outputs, and reproduced the reasoning format — but this did not translate into correct computation on unseen problem lengths.

Closer inspection of the 3-digit scratchpad outputs reveals a more specific failure mode: the model consistently writes only 2 digit steps inside the scratch region before stopping, never attempting the third digit position at all. This is consistent with what Lee et al. (2023) find about small transformers failing to extend learned procedures to longer sequences, the model learned the reasoning procedure for 2-digit problems specifically, not a generalizable algorithm.

One possible explanation for how the scratchpad developed its carry notation is that in a character-level model, the scratch region gets filled with whatever intermediate token sequence most reliably predicts the correct output during training. The simplest symbolic way to represent the carry state using single characters: c0 and c1, converged to match the same structure as the explicitly supervised CoT format. Nye et al. (2021) discuss how models can develop useful intermediate representations in the scratch region without explicit supervision, which is consistent with this interpretation. It should be noted that this remains speculative, as understanding the exact mechanism would require further investigation.

5.2 The Role of Representation

A secondary finding is the dramatic effect of input representation on in-distribution learning. The shift from standard to reversed digit format increased 2-digit accuracy from roughly 1% to over 99% for the Direct model, without any change to architecture or training objective. This aligns with prior work on how positional alignment between the generation order and the addition algorithm affects learnability (Lee et al., 2023). The representation result confirms that format choices can be decisive at small scales, independent of the reasoning paradigm.



5.3 Limitations

This study has several important limitations. The model scale (0.79M parameters) is far below the regime where chain-of-thought methods have shown the most dramatic benefits. It is possible that reasoning supervision improves OOD generalization at larger scales, or that the benefits of CoT and Scratchpad are primarily emergent phenomena in large models. Additionally, all experiments were conducted on a single arithmetic task (addition); generalization to other compositional tasks is not established. Training was conducted on consumer hardware with no GPU access, which imposed practical limits on the number of training iterations and model sizes explored.

6. Conclusion

We trained three small character-level transformers on arithmetic addition under different supervision paradigms: Direct, Chain-of-Thought, and Scratchpad, and evaluated their generalization from 2-digit to 3-digit addition. All three models achieve near-perfect in-distribution accuracy but fail completely on out-of-distribution inputs. Qualitative analysis suggests the models learn to reproduce the format of reasoning rather than acquire the underlying algorithm.

The Scratchpad model's behavior is particularly revealing. It spontaneously developed structured carry notation identical in form to the explicitly supervised CoT format, suggesting gradient descent converged on the same compact symbolic representation independently. However, inspection of 3-digit outputs shows the model truncated its reasoning at exactly 2 digit steps in every case, never attempting the third digit: evidence that it learned a 2-digit procedure, not a generalizable algorithm. These findings together suggest that at small model scales, structured reasoning supervision may improve output plausibility without improving generalization.

Future work should examine whether this failure exists across model scales, whether longer training or larger datasets change the picture, and whether alternative supervision strategies such as symbolic scaffolding or progressive digit-length training can induce genuine algorithmic generalization in small transformers.

References

1.Dziri, N., Lu, X., Sclar, M., Li, X., Jiang, L., Lin, B. Y., ... & Hajishirzi, H. (2023). Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems*, 36.

2. Lee, N., Sreenivasan, K., Lee, J. D., Lee, K., & Papailiopoulos, D. (2023). Teaching arithmetic to small transformers. *arXiv preprint arXiv*:2307.03381.

3. Nye, M., Andreassen, A. J., Gur-Ari, G., Michalewski, H., Austin, J., Bieber, D., ... & Odena, A. (2021). Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv*:2112.00114.

4. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., ... & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35, 24824–24837.