Evaluating Data Representation and Chain-of-Thought Reasoning for Arithmetic Tasks in Character-Level Language Models

Pranav Azad

Abstract

This report investigates the impact of data representation and explicit reasoning pathways on the arithmetic capabilities of autoregressive language models. Using a heavily constrained, character-level transformer (nanoGPT, 0.79M parameters), performance was evaluated on 2-digit addition tasks across three data paradigms: a standard Baseline, a Reverse-Output format, and a Chain-of-Thought (CoT) format. Quantitative evaluation via an automated suite of 100 zero-shot prompts revealed a massive performance disparity. The Baseline and Reverse formats struggled to achieve absolute accuracy (1.0% and 7.0%, respectively), whereas the CoT model achieved 100.0% accuracy. Furthermore, error distribution analysis demonstrated that reversing the output sequence drastically reduced the Mean Absolute Error (MAE) from 24.24 to 5.60. These findings indicate that while manipulating output direction improves magnitude estimation, exposing explicit reasoning steps is required to overcome the latent computational bottleneck of exact arithmetic calculation in small-parameter models.

1. Introduction

Large Language Models (LLMs) often struggle with multi-digit arithmetic because autoregressive generation requires predicting the most significant digit first, before the less significant digits (and their potential "carries") have been calculated. This research explores whether teaching a model *how* to think (Chain-of-Thought) or changing the directional flow of the output data (Reverse Output) allows a small-scale model to achieve high arithmetic accuracy with limited computational resources.

2. Methodology

- **Model Architecture:** Experiments were conducted using a modified nanoGPT framework. To optimize for rapid experimentation and isolate the learning of arithmetic logic from complex language modeling, a highly constrained "Baby GPT" was deployed (4 layers, 4 attention heads, embedding dimension of 128, block size of 64). The vocabulary was restricted to 13 essential characters (0-9, +, =, \n, etc.).
- **Datasets:** A programmatic pipeline generated 20,000 unique 2-digit addition problems, formatted into three experimental datasets:
    1. *Baseline:* Standard formatting (15+19=34).
    2. *Chain of Thought (CoT):* Explicit step-by-step column addition exposing the carry operations (15+19=(5+9+0=14,1+1+1=3)34).
    3. *Reverse Output:* Answer digits reversed to align with right-to-left arithmetic logic (15+19=43).
- **Training & Evaluation:** Each model was trained independently for 1000 iterations. Models were evaluated on 100 dynamically generated, unseen mathematical prompts. Output was programmatically parsed, and incorrect responses were logged to calculate the Mean Absolute Error (MAE) and track specific carry-operation failures (errors of exactly ±10).

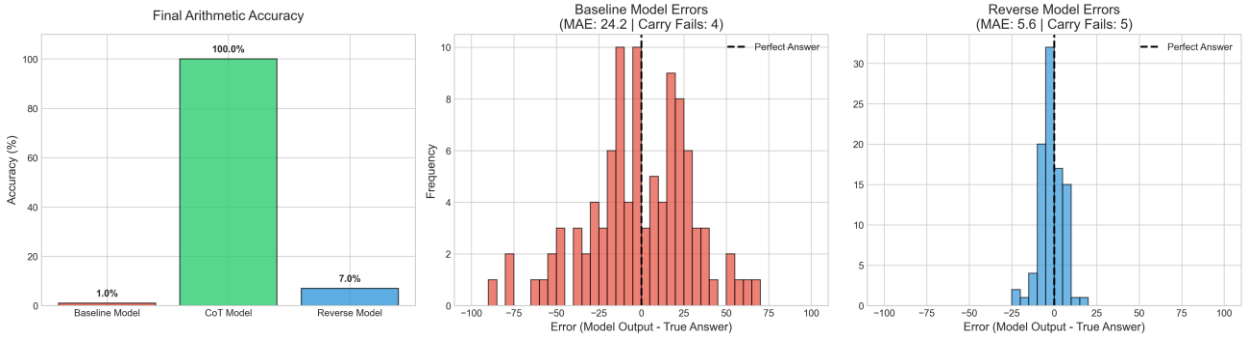
3. Results

The models exhibited extreme variance in both absolute accuracy and error magnitude:

- **Baseline Model:** 1.0% Accuracy | MAE: 24.24 | Exact Carry Fails: 4

- **Reverse-Output Model:** 7.0% Accuracy | MAE: 5.60 | Exact Carry Fails: 5
- **Chain-of-Thought Model:** 100.0% Accuracy | MAE: N/A (Perfect Accuracy)

4. Qualitative Error Analysis & Discussion



The quantitative data reveals critical insights into how small-parameter transformers manage latent representations.

- **The Latent State Bottleneck (Baseline):** The 1.0% accuracy and high MAE (24.24) of the Baseline model highlight a complete failure to internally route calculations. The broad spread of the error distribution indicates the model learned the superficial syntax of addition but struggled severely with magnitude and algorithmic logic.
- **Magnitude Correction via Reversal:** While the Reverse model only achieved 7.0% absolute accuracy, its MAE dropped drastically to 5.60. By predicting the least significant digit first, the model's generation order aligned with the mathematical algorithm. The tightened error distribution proves that this representation allowed the model to successfully estimate positional magnitude, even though it still lacked the depth to consistently execute exact carries (logging 5 specific carry-failures).

- **The Effectiveness of Explicit State Tracking (CoT):** The CoT model's 100.0% accuracy provides definitive proof that intermediate computation space is vastly more important than raw parameter count. By offloading the carry operations from the model's hidden layers into the visible token sequence, the complex algorithm of addition was reduced to simple, sequential pattern matching. The model did not need to "hold" a carry in its weights; it simply read the carry from its own generated context window.

5. Conclusion

This experiment demonstrates that the inability of small transformers to perform arithmetic is not a fundamental failure of the architecture, but rather a bottleneck of data representation. While aligning token generation with algorithmic order (Reverse) significantly reduces error magnitude, forcing a model to articulate its latent states via Chain-of-Thought prompting provides an immense boost to algorithmic execution, transforming a completely failing model into a perfect computational engine.