

Understanding Hidden Computations in Chain-of-Thought Reasoning

Aryasomayajula Ram Bharadwaj
Independent Researcher
`ram.bharadwaj.arya@gmail.com`

August 28, 2024

Abstract

Recent work has shown that transformer models can perform complex reasoning tasks using Chain-of-Thought (COT) prompting, even when the COT is replaced with filler characters. This paper attempts to investigate methods to decode these hidden/filler computations, focusing on the 3SUM task. We analyze a 34M parameter LLaMA model trained on filler COT sequences and propose a novel decoding method that successfully eliminates the filler tokens in the original COT without sacrificing the accuracy. Our findings provide insights into how transformers encode and process information in filler COT sequences, offering new perspectives on model interpretability and the nature of computation in language models.

1 Introduction

Chain-of-Thought (COT) prompting has emerged as a powerful technique for improving the performance of large language models on complex reasoning tasks [1]. However, recent work by [2] demonstrates that these improvements can be achieved even when the COT is replaced with hidden/filler characters (e.g., "..."), raising intriguing questions about the nature of computation being performed within these models.

This paper builds upon the findings of [2], focusing on the 3SUM task as a case study. We aim to decode the hidden computations embedded within the transformer architecture when trained on hidden COT sequences. Our work provides valuable insights into how these models encode and process information, potentially leading to improved model interpretability and more effective training strategies.

2 Background

2.1 The 3SUM Task

The 3SUM task involves finding three numbers in a given set that sum to zero. In this experiment it serves as a proxy for more complex reasoning tasks to study the computational capabilities of transformer models [1]. For this experiment

we have used the same data representation method and generating process or 3SUM sequences as in [1].

2.2 Chain-of-Thought using Filler Tokens

In this variant of Chain-of-Thought, intermediate reasoning steps are replaced with filler characters (e.g., "..."). It has been observed in [1] that the models trained on hidden/filler sequences still perform well, suggesting that meaningful computation occurs despite the lack of explicit reasoning steps.

3 Methodology

We used a 34M-parameter LLaMA model with 4 layers, 384 hidden dimension, and 6 attention heads [4], the size of the training and test datasets and all other hyperparameters are kept the same as in the "Let's think dot by dot" paper [2]. Our analysis focused on three main areas: Layer-wise Representation Analysis, Token Ranking, and Modified Greedy Decoding Algorithm.

4 Results and Discussion

4.1 Layer-wise Analysis

Our analysis revealed a gradual evolution of representations across the model's layers. The initial layers primarily contained raw numerical sequences associated with the 3SUM problem's chain of thought. However, starting from the third layer, we noticed the emergence of hidden tokens. As we progressed through subsequent layers, we observed a steady transition from purely numerical sequences to an increasing prevalence of filler characters. This pattern suggests that the model develops the ability to utilize filler tokens as proxies only in its deeper layers, which aligns with intuitive expectations of how neural networks process information. After conducting a comprehensive evaluation across numerous examples, we found that, on average, there is a marked increase in filler token usage immediately following the second layer. Furthermore, the final layers of the model demonstrate extensive reliance on these hidden tokens. For this analysis, we employed nostalgebraist's logit lens method [3].

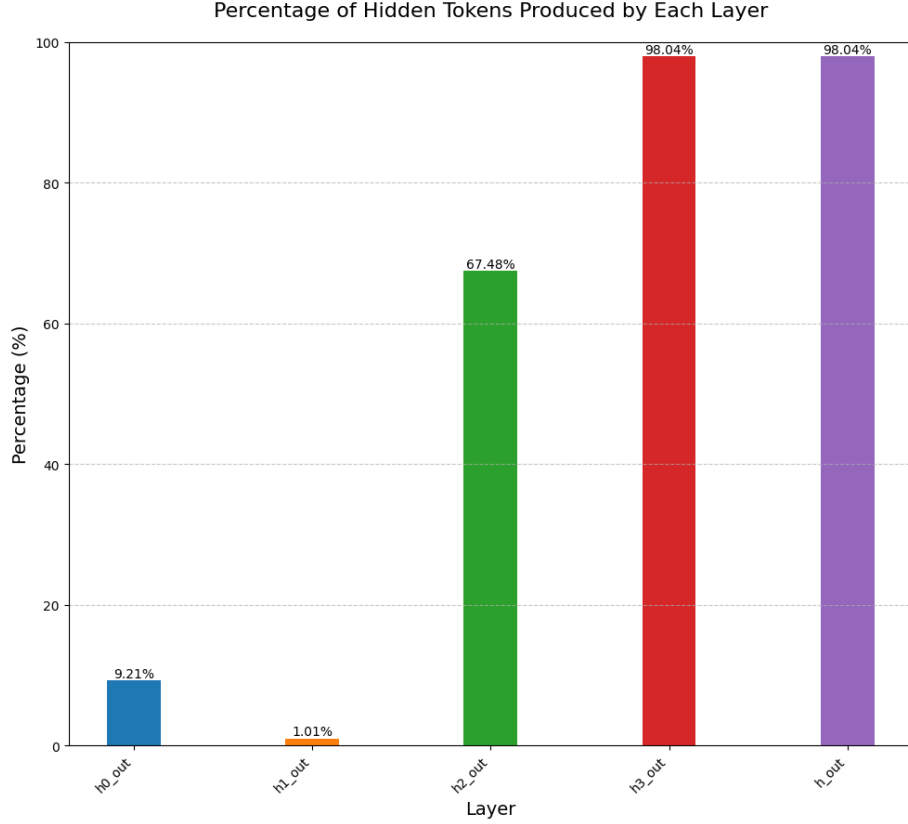


Figure 1: Filler token occurrence percentages in generated sequences across layers

4.2 Token Rank Analysis

The top-ranked token was consistently the filler character ("."), while lower-ranked tokens revealed the original, non-filler COT sequences. This supports the hypothesis that the model replaces all computation with filler tokens on top while keeping the original computation intact underneath. We have provided a sample snapshot of sequences decoded at each layer for both the top (rank-1) and rank-2 tokens in the appendix section.

4.3 Modified Greedy Decoding Algorithm

Based on the observations made in the token rank analysis, We implemented a modified greedy autoregressive decoding method. The steps include: performing standard greedy decoding, selecting the second-highest probability token when encountering a filler token, and continuing this process for the entire sequence. This resulted in a 100% match in 3SUM task results with and without filler tokens. To test the effectiveness of this method, we have also compared this with replacing the filler token with randomly sampled token instead of the next highest (rank-2) token. The percentages of the original and these modified decoding methods are visualized as plots below.

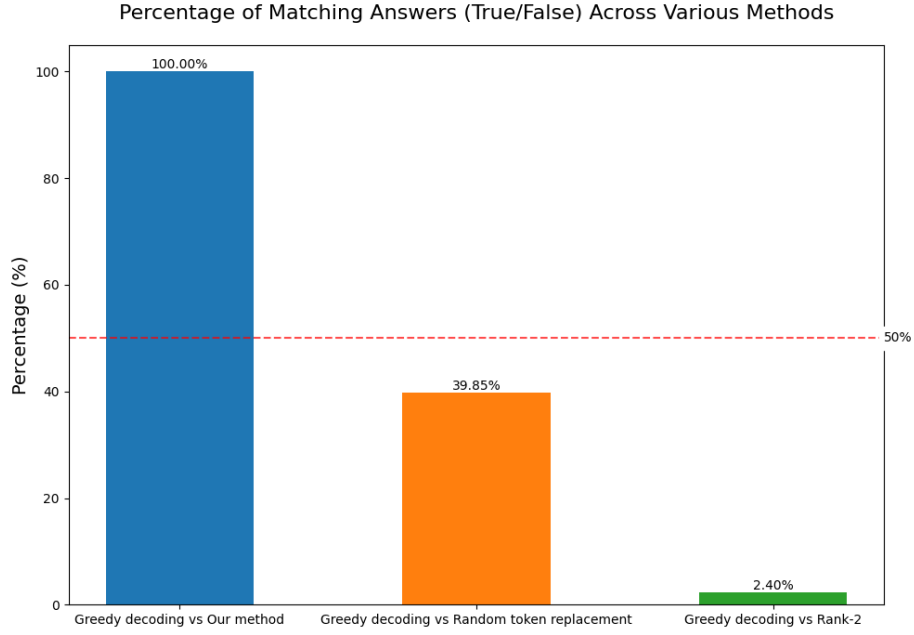


Figure 2: Comparison of decoding methods

5 Implications and Future Work

Our findings provide new tools for understanding internal reasoning processes and increase confidence in COT-based approaches for improving visibility. Future work should focus on developing better decoding methods or finding circuits that hide tokens, investigating generalizability to tasks beyond 3SUM (including natural language tasks), and improving token hiding methods (currently limited to one filler token which is simple to decode).

6 Conclusion

We have presented a novel approach to understanding hidden computations in transformer models through the analysis of token rankings and layer-wise representations, and the development of a modified decoding algorithm. Our insights into how models encode and process information in filler COT sequences open new avenues for improving interpretability, efficiency, and safety in language models. This work strengthens the belief in COT visibility research for interpretability.

The code used for the experiments and analysis in this paper is available on GitHub at https://github.com/rokosbasilisk/filler_tokens/tree/v2.

7 Appendix: Layerwise view of sequences generated via various decoding methods

[illegible]

Figure 3: Greedy Decoding

h0_out: 0-0 2 [EOS] 0-2 A 5 A 0-8 A 3 [EOS] 0-9 0-2 4 4 5 0-6 5 0-2 5 3 2 0-8 2 A 9 1 9 0-5 5 0-8 6 0-7 3 0-9 0-8
 0-2 0-6 0-6 [EOS] 0-9 0-2 0-5 0 [EOS] [EOS] [EOS] 2 0-6 2 0-7 5 0-6 0 A 5 [EOS] 1 0-6 2 0-4 5 0-7 . [EOS] 5 A 6
 A 7 0-6 0-6 0-6 0-6 5 0-6 2 0-6 5 0-6 5 0-7 1 [EOS] 4 0-6 0-2 0-6
 h1_out: 0-0 0-9 0-2 9 0-2 1 0-2 1 0-2 6 0-2 1 0-2 5 0-2 8 0-2 7 0-2 4 9 0-2 6 0-2 9 0-2 4 0-6 8 0-2 2 0-9 6 0-9 0-9
 4 0-7 6 0-7 2 0-6 6 0-2 1 0-6 7 0-7 0-8 0-7 8 0-4 7 0-6 3 0-9 3 0-7 0-9 0-7 0-7 0-8 0-6 0 0-6 8 0-3 0-7 8 0-9 5 0-7 0 0-9
 2 0-6 4 0-7 6 0-7 3 0-6 1 0-7 9 0-7 9 0-6 8 0-9 7 0-7 7 0-7
 h2_out: 0-1 9 0-2 9 0-3 1 0-4 2 0-5 6 0-6 0 0-7 9 0-8 0-6 7 0-2 7 0-3 6 0-4 2 0-5 4 0-1 8 0-7 2 0-8 5 0-9 1 0-3
 6 0-2 6 0-5 1 [EOS] 5 0-7 1 0-2 7 0-2 8 A 8 0-5 2 0-3 7 0-7 3 0-3 9 0-3 3 0-4 0 0-4 2 0-7 3 [EOS] 6 0-4 0 0-5 2 0-7
 7 2 0-8 5 0-5 5 [EOS] 3 0-8 1 0-6 6 [EOS] 7 0-7 5 0-8 7 0-3 4 0-4
 h3_out: 0-0 0-9 0-0 0-0 1 0-4 2 0-6 0 1 0-0 1 0-0 5 0-8 8 0-0 7 0-1 0 0-1 0 6 0-1 0-5 0-5 4 0-1 2 0-1 2 0-1 3 0-9 1 0-2
 2 0-6 4 0-6 0-2 2 0-2 1 0-2 1 0-2 7 0-2 8 0-3 8 0-5 2 0-6 8 0-7 3 0-3 9 0-9 0-0 0-4 8 0-6 0 0-7 8 0-8 7 0-9 5 0-5 0-0
 7 2 0-8 4 0-9 0-0 0-6 3 0-8 6 0-9 1 0-8 0-9 0-4 0-9 7 3 0-7 [EOS]
 h_out: 0-0 9 0-0 0-0 0-0 1 0-4 2 0-0 1 0-0 1 0-0 5 0-8 8 0-0 7 0-1 0 0-1 0 6 0-1 0-5 0-5 4 0-1 2 0-1 2 0-1 3 0-9 1 0-2
 6 0-4 0-6 0-2 2 0-2 1 0-2 1 0-2 7 0-2 8 0-3 8 0-5 2 0-6 8 0-7 3 0-3 9 0-9 0-0 0-4 8 0-6 0 0-7 8 0-8 7 0-9 5 0-5 0 0-7
 2 0-8 4 0-9 0-0 0-6 3 0-8 6 0-9 1 0-8 0-9 0-4 0-9 7 0-3 7 [EOS]

Figure 4: Greedy Decoding with Rank-2 Tokens

```

h0_out: 0-0 5 [EOS] [EOS] [EOS] [EOS] [EOS] [EOS] 6 A [EOS] [EOS] 0 A [EOS] 0-9 5 [EOS] [EOS] A 3 0-7 3 0-8 2 [EOS] 0-8
A 7 True True 0-6 [EOS] [EOS] 0-6 0-8 0-6 2 [EOS] [EOS] [EOS] 2 0-6 9 0-6 0-8 0-6 2 A [EOS] [EOS] 2 0-6 6 A 6 [E
S] [EOS] [EOS] [EOS] [EOS] 5 [EOS] 0 [EOS] 2 [EOS] True [EOS] A 0-8 A 5 [EOS] [EOS] [EOS] [EOS] A 5 A 1 A [E
OS] A 6 0-7 [EOS] [EOS] [EOS] [EOS] [EOS] [EOS] [EOS] [EOS]
h1_out: 0-0 2-0 0-3 [EOS] 5 0-0 2-0 0-7 0-0 0-0 9 0-0 5 0-0 3 0-0 0-0 0-7 2 0-7 4 [EOS] 8 0-2 8 0-2 0-0 0-5 0-8 8
0-2 0-2 1 0-2 0-0 1-6 0-7 4 0-6 0-0 0-7 2-0 6 3 0-6 2 0-7 8 0-7 6 0-6 2 0-6 4 0-6 9 0-7 2 0-7 2 0-6 0-7 0-0 7-9
0-7 8 0-6 5 0-9 5-6 2 0-6 0-8 0-6 0-0 6-6 6-7 5-7 0-7 4 0-9 4 0-6 False
h2_out: 0-0 1-1 0-3 0-5 0-0 1-0 1-0 1-0 1-0 0-5 0-0 5 0-9 3 0-1 0-1 0-1 2-0 0-1 0-9 0-1 8 0-6 9 0-1 0 0-1 3 0-9 8 0-
2 0-0 4 1-0 2 1-0 6 1-0 2 4 0-8 0-9 2-0 4 3 0-3 3 0-6 8 0-3 6 0-8 2 0-9 0 A 8 0-6 0-0 0-4 2 0-4 8 0-9 1 0-6 0 0-5
8 0-5 4 0-9 0-0 6-2 0-6 8 0-9 1 A 6 0-8 8 0-9 4 0-9 4 A
h3_out: 0-0 1-0 1-2 0-3 5 0-6 0-6 5-6 0-6 6-2 0-7 9 0-0 5 0-9 3 0-2 7 0-1 9 0-4 0-9 0-1 8 0-6 8 0-7 4 0-8 6 0-1 8 0-
3 0-2 1-0 5 1-0 6 5-7 4 0-8 0-9 2-0 4 3 0-3 3 0-3 7 0-3 6 0-8 2 0-9 4 0-5 0-0 0-6 2 0-4 2 0-4 7 0-4 0 0-5 9 0-
5 8 0-5 0-5 5-7 2 0-6 6 0-6 9 0-7 6 0-7 8 0-8 4 0-9 4 A True
h4_out: 0-0 1 0-2 3 0-3 5 0-6 0-6 0-5 6 0-6 2 0-7 9 0-0 5 0-9 3 0-2 7 0-1 9 0-4 0-9 0-1 8 0-6 8 0-7 4 0-8 6 0-1 8 0-3
0 0-2 1 0-5 1-0 6 5-7 4 0-8 0-9 2-0 4 3 0-3 3 0-3 7 0-3 6 0-8 2 0-9 4 0-5 0-0 0-6 2 0-4 2 0-4 7 0-4 0 0-5 9 0-5
8 0-5 5 0-5 5-7 2 0-6 6 0-6 9 0-7 6 0-7 8 0-8 4 0-9 4 A True

```

Figure 5: Our Method: Greedy Decoding with Filler Tokens Replaced by Rank-2 Tokens

```

n0_out: 0-0 5 [EOS] [EOS] A [EOS] 2 A [EOS] A 3 A 6 0-9 4 4 7 A 5 0-2 2 A 5 True 2 A 5 1 5 0-5 [EOS] 0-8 6 0-6 0
-5 0-9 0 8 [EOS] 2 0-6 5 0-6 2 0-6 0-8 0-2 [EOS] [EOS] A [EOS] 2 A 2 A [EOS] 5 A [EOS] [EOS] 5 [EOS] 0 [EOS] 2
[EOS] 0-8 0-7 5 0-6 0-8 0-2 5 [EOS] 6 0 [EOS] A 5 A 5 A 5 0-7 1 [EOS] True [EOS] 4 0-6
h1_out: 0-0 2 2-2 0-9 0-0 1 0-0 2 0-2 7 0-2 0-0 0-5 0-5 0-0 3 0-2 0-2 0-7 6 0-2 9 0-2 4 0-2 9 0-2 0-2 6 0-9 0-0 0-2
0-0 2-2 1 0-2 0-6 1 0-2 1 0-0 0-6 0-6 2 0-6 3 0-6 4 0-6 3 0-7 3 0-6 2 0-6 4 0-7 9 0-7 0 0-9 0-7 6 0-9 5 0-7 9 0-9
2 0-6 5 0-9 6 0-6 2 0-6 1 0-7 0 0-7 6 0-9 8 0-6 7 0-7 0-6
h2_out: 0-1 1 0-2 0-9 0-3 1 0-0 2 0-0 1 0-0 1 0-8 5 0-8 8 0-0 3 0-2 0-2 0-3 0-1 0-9 0-5 0-8 0-6 9 0-1 0-0 1-5 0-1 8 0-3
3 6 0-4 6 0-2 1 [EOS] 5 0-7 1 0-8 7 0-9 8 A 3 0-3 2 0-6 8 0-3 3 0-8 2 0-3 0-0 4-8 0-6 0-4 3 A 7 0-9 0-5 5 2 0-7
2 0-5 4 0-5 6 0-6 3 0-6 1 0-6 1 A 7 0-7 5 0-8 4 0-9 4 0-4
h3_out: 0-0 9 0-9 0-9 0-5 0-2 4 0-0 7 0-6 0-2 0-7 5 0-8 8 0-9 7 3 0-2 7 0-3 9 0-1 0-9 0-5 4 0-1 8 0-1 4 0-1 6 0-1 1 0-2
0-2 0-2 1 0-5 0-0 6 0-5 0-2 4 0-8 0-9 9 0-8 0-3 8 0-3 2 0-8 0-7 3 0-8 9 0-3 4 0-5 8 0-4 0-4 0-8 8 0-7 0-4 0-6 9 0-9
7 0-8 0-4 0-5 0-7 3 0-6 6 0-9 0-8 6 0-7 8 0-8 7 0-7 A
h_out: 0-0 1 0-2 9 0-9 0-1 0-4 2 0-5 7 0-0 1 0-7 5 0-8 5 0-9 7 0-2 0 0-3 9 0-1 0-9 0-1 4 0-1 8 0-1 4 0-8 3 0-9 8 0-3
6 0-2 6 0-5 1 0-2 1 0-7 1 0-2 7 0-9 8 0-8 0-8 0-5 2 0-6 8 0-7 6 0-8 2 0-0 0-5 0-6 0-2 0-7 0-4 7 0-9 0-0 0-5 2 0-5
8 0-8 4 0-5 0-0 0-7 3 0-8 1 0-6 0-0 8 0-9 0-8 0-9 7 0-9 7 [EOS]

```

Figure 6: Greedy Decoding with Filler Tokens Replaced by Randomly Selected Tokens

References

- [1] Pfau, J., Merrill, W., & Bowman, S. R. (2023). Let’s Think Dot by Dot: Hidden Computation in Transformer Language Models. arXiv:2404.15758.
- [2] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., ... & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. arXiv:2201.11903.
- [3] nostalgebraist (2020). interpreting GPT: the logit lens interpreting-gpt-the-logit-lens
- [4] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., ... & Lample, G. (2023). LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971.