Back on Track: Characterizing Backtracking in Chain-of-Thought Reasoning

Siddharth Boppana [1] **Philip LaDuca** [1] **Nicholas Masi** [1] **Jack Merullo** [2]

Abstract

Training LLMs with reinforcement learning objectives has led to impressive capabilities in reasoning tasks, through mechanisms such as verification and backtracking. However, the chain of thought produced by these models is often unfaithful to their final answer. In this work, we explore the relationship between backtracking, faithfulness, and correctness in an answer by inserting incorrect intermediate results in a model's chain of thought. We show that models ignore these incorrect values for the majority of word problems, and either perform worse or act unfaithfully as a result. When models do backtrack, we find that their performance nearly recovers to the baseline. We also perform basic circuit analysis on a minimal example and show that certain answers are more likely to elicit backtracking based on their plausibility.

1. Introduction

Recent advancements in large language models (LLMs) have focused on scaling test time compute using chain of thought reasoning. Reasoning models like OpenAI's o1 and DeepSeekR1 have been trained using reinforcement learning fine-tuning to produce a chain of thought prior to the answer as a method of promoting "thinking". (OpenAI et al., 2024; DeepSeek-AI et al., 2025) These models exhibit strong performance on mathematical reasoning and coding benchmarks, having been trained on verifiable answers to these questions. One particular mechanism developed implicitly during reinforcement learning fine-tuning is backtracking, where a model decides to revisit a previous computation or go down a different reasoning path. Previous work has shown that increasing the length of response with backtracking tokens such as "Wait" leads to an improvement in accuracy on mathematics tasks. (Muennighoff et al., 2025) Reasoning models also have a measure of whether they are correct, which can be used to end a chain of thought early. (Zhang et al., 2025) Other work investigates whether models are faithful to their CoT, where the model's final answer differs from the answer it would have predicted by following its CoT. (Arcuschin et al., 2025; Turpin et al., 2023) Oftentimes, a model will produce a chain of thought containing hallucinated information to lead to a certain final answer as well. However, there has been little work understanding how backtracking affects faithfulness in a model's CoT.

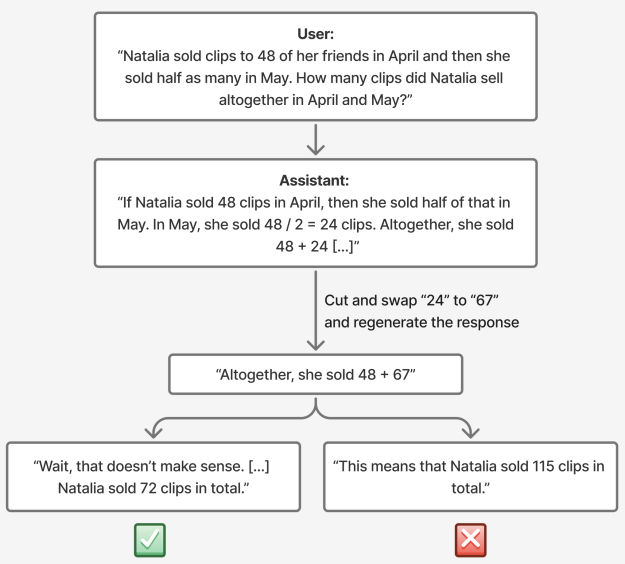

Figure 1. Our alteration process shown in an example prompt to the model. We swap numerical values such that the computation would result in an incorrect answer, and evaluate if the model explicitly backtracks after the swap. We then inspect whether the rest of the model's response is faithful based on the backtrack.

In this work, we systematically introduce incorrect reasoning steps in a CoT to evaluate whether the model will (1) backtrack and (2) remain faithful to its incorrect step when producing a final answer. By directly modifying a model's existing CoT with incorrect numerical values, we provide opportunities to backtrack. We find that reasoning models do not consistently backtrack after these values, as models

*Equal contribution [1]Department of Computer Science, Brown University, Providence, RI, USA [2]GoodFire AI, San Francisco, CA, USA. Correspondence to: Siddharth Boppana <siddharth_boppana@brown.edu>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

either ignore the value unfaithfully or it affects the final computation with an incorrect answer. We swap with different types of incorrect values and altering where in the CoT the swap occurs. Finally, we conduct basic circuit analysis to identify the backtracking mechanism and show that the contribution for the backtracking token can be traced to the last MLP.

2. Methods

2.1. Swapping Tokens to Elicit Backtracking

We use the DeepSeekR1-Distill-Qwen-1.5B model for all of our experiments. (DeepSeek-AI et al., 2025) We begin by collecting the full response of the model to each of the 7.5k problems in the GSM-8K dataset's training split, which consists of grade school mathematical word problems. (Cobbe et al., 2021) We query the model using the prompt template provided in the DeepSeekR1 technical report, which instructs the model to think first, then answer to the user. A max token limit of 1k is set, and responses which exceed 1k are included for recording backtracking but are generally marked as incorrect.

To create different situations to backtrack for the model, we swap numerical values at various positions in the original response, and generate the new response beginning from the swapped value. We identify any numerical values (any substring containing digits 0-9, but not written out numbers such as one, two, etc.), and swap them one-third of the time. Then, a swap is performed by sampling one of the potential replacement types outlined in Table 1. The new responses are collected from each prompt.

Swap Type	Example
Positive Integer [1, 100]	$2 \rightarrow 57$
Negative Integer [-100, -1]	$2 \rightarrow -3$
Positive Decimal [1, 100]	$2 \rightarrow 6.7$
Additive Inverse	$2 \rightarrow -2$

Table 1. Types of swaps conducted in our experiment. Each swap ensures that the computation produces a different value which would require backtracking

We determined whether a model has backtracked by searching for key terms in the model's response. The terms "wait", "double-check", "alternatively", "make sure", "another way", "verify", and "to confirm" are used to identity backtracking, in line with the collection process from (Zhang et al., 2025). Correctness is determined by comparing the final numerical output of the model with the correct answer provided in the GSM-8K dataset.

2.2. Circuit Analysis

One of the issues with studying reasoning models is their long chain of thought, which makes it difficult to use methods that investigate hidden states of the transformer. To circumvent this, for the following experiments the model was given a question and was provided a concise mock CoT (which skipped straight to the answer), as part of the input. This prompt setup has the model operate as if it has already found a potential answer, without having to generate the full chain of thought which otherwise would be hundreds of tokens long. With this setup, we can now modify the model's internal chain of thought such that it arrives at different solutions, and observe whether the model decides to backtrack, assume the answer is correct, or restart the chain of thought completely. We swap the answer token in the shortened CoT to a variety of swaps, similarly to the previous setup.

First, the logit difference between the token "<" (representing the start of the `<answer>` tag, indicating a final answer) and "Wait" was computed to see how likely the model was to backtrack. By varying the CoT answer with different swap types, we observe whether certain implausible answers are more likely to trigger a backtrack than others. We use two approaches to mechanistically analyze the underlying circuit when "Wait" is predicted after the concise mock CoT. A clean (with correct answer) and corrupted prompt (with a swapped implausible answer) are used to get activations at different layers. Activation patching, a method of replacing activations at individual components from another run, is used to evaluate which components of the model are necessary for a specific mechanism. (Meng et al., 2023; Wang et al., 2022) In this case, we patch in activations from the swapped prompt into the correct answer prompt. Logit lens is an approach where the residual stream between different layers is passed into the unembedding matrix W_U to get predicted logits. (nostalgebraist, 2020)This was also applied to both original and swapped prompts, to observe at which point the residual stream represents either the backtracking or final answer token. Further analysis uses the logit lens approach in the final layer of the model to further narrow down where this contribution emerges. When interpreting the multi-layer perceptrons (MLPs) within a layer of the model, we use singular value decomposition (SVD) to decompose the down-projection matrix of the MLP into a low-rank approximation where each singular vector represents a contribution to the residual stream. (Stolfo et al., 2024)

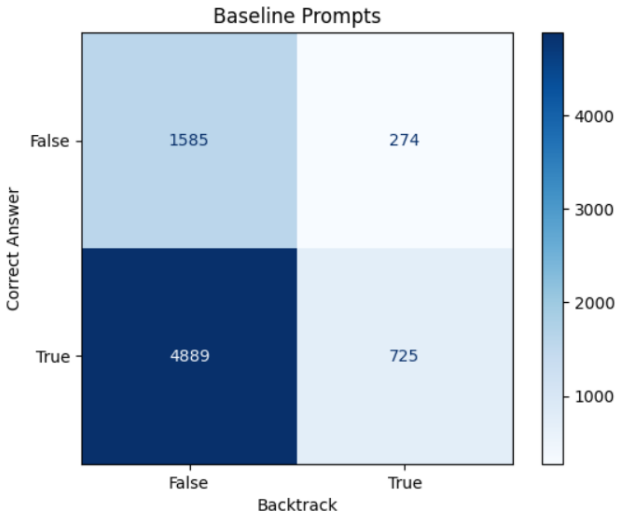

Figure 2. A matrix showing the number of baseline prompts in which the model achieve either correct or incorrect answers as well as whether the model backtracked during the CoT.

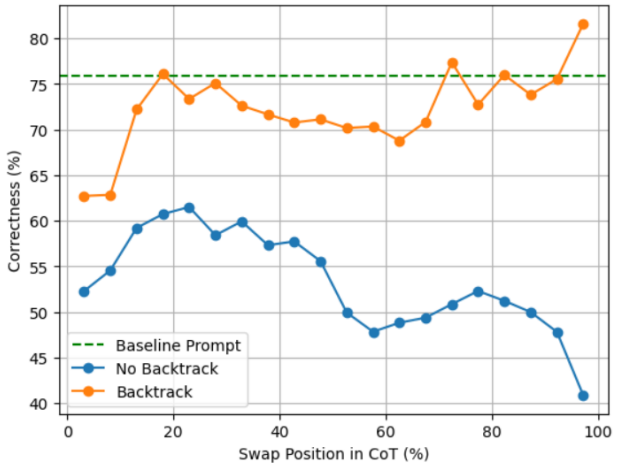

Figure 3. The accuracy of the model when swapping tokens throughout the CoT. The orange indicates results from swapping when a backtrack was triggered while the blue indicates results from swapping when no backtrack was triggered. The green indicates the overall accuracy from the baseline model for prompts which do not trigger a backtrack.

3. Results

3.1. Swapping Analysis

To obtain a baseline for the model's performance on the GSM-8K dataset, we queried the model with each prompt, recording when the model obtained a correct answer and when the model backtracked. In Figure 2 we show the resulting counts with the model backtracking in 15% of its response while archiving a 76% accuracy in responses with no backtracking and a 72% accuracy in responses with backtracking.

In Figure 3 we show the accuracy for both responses with backtracking and responses without backtracking. The responses where the model did backtrack have a much higher accuracy, outperforming the responses which did backtrack at all swap positions. Interestingly, the model's performance when backtracking is similar to the model's baseline performance despite having incorrect values injected into the CoT, even beating the baseline performance given very late swap positions. The overall accuracy for CoTs with backtracking in the swapping procedure was 73%, only 3% less than the accuracy of the baseline model, and the overall accuracy for CoTs without backtracking was 53% with exact values given in Figure 4. When looking specifically at the prompts where the baseline model got an incorrect answer, we find that the swapping procedure results in an accuracy of 10% where CoTs without backtracking result in an accuracy of 8% while CoTs with backtracking result in an accuracy of 17%, further showing the benefits of backtracking on model accuracy. The responses with backtracking show a clear positive correlation with position in the CoT while the responses without backtracking show a clear negative correlation. This means that as a swap is done further into the chain of thought the model becomes more likely to produce an incorrect answer, by staying faithful to the swapped value, or if the model backtracks it becomes more likely to get a correct answer, implying that having the model backtrack and evaluate an important mistake can help improve its overall performance.

The rate of backtracking does not increase with position in CoT, where backtracking seems to be have more impact. In Figure 5 we show the rate of backtracking with respect to the position of a swap in the CoT which has a clear dependence on the thinking and answer phase denoted in the prompt. Prior to the thinking phase, the model is much more likely to backtrack, reaching a peak of around 25% before sharply falling after the thinking phase ends. The average thinking token appears around 50% of the way through each CoT with the majority occurring between 40% and 60% as seen by the region of steep decline in the backtracking rate. After the thinking phase the model is much less likely to backtrack only reaching a rate of about 17%. The overall backtracking rate in the swapping procedure was 22%.

We also determine the number of times the model backtracks by counting the occurrences of the backtracking key words. In Figure 6 we show the baseline model's accuracy as well as the swapping procedure accuracy with respect to the number of backtracks. There is a consistent negative trend which both models share; however, the swapping model

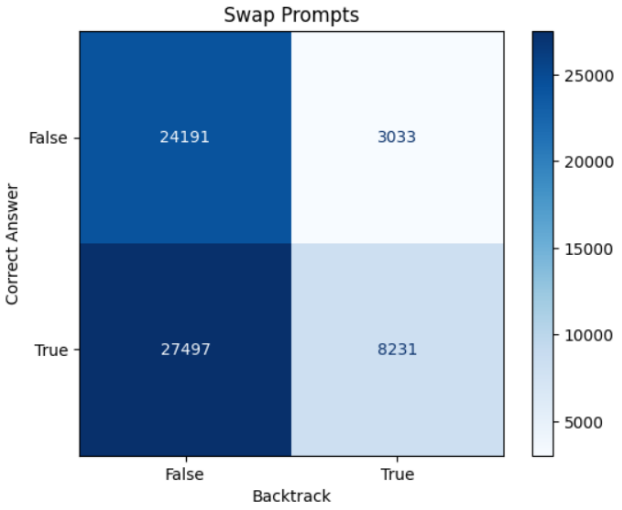

Figure 4. A matrix showing the number of swapped prompts in which the model achieve either correct or incorrect answers as well as whether the model backtracked during the CoT.

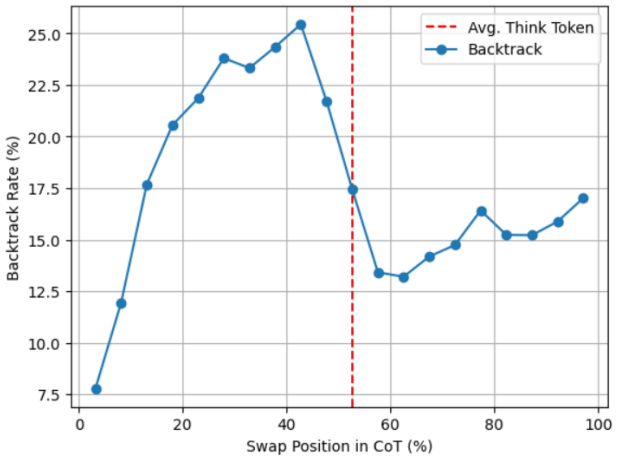

Figure 5. The rate at which backtracking was triggered in the swapped prompts with respect to the position of the swap. The red indicates the average position of the think token.

has the highest accuracy when there is only one backtrack while the baseline model has it's peak accuracy when there is three backtracks.

3.2. Circuit Analysis

In Figure 7, we show six different answer schemes where the swapped value is varied in magnitude for each scheme. In the leftmost graph of the figure, we see that as swapped value increases in magnitude, the model becomes increasingly more confident in its prediction for the final answer token. Certain answer schemes appear to be more implausible to the model, such as the negative and decimal swaps which predict the wait token with higher confidence than the answer token consistently regardless of magnitude. This suggests that the model has an internal state of plausibility for this question, where the correct answer is a positive number. We generally see large spikes in logits at the value 48, which is provided in the problem description.

In Figure A.2, activation patching reveals a clear mechanism for backtracking, where information moves from the swapped token (the minus sign) to the newline token. Then, another information transfer seems to be occurring from the newline token to the final token, with an increase in logit different at the final layer. This indicates that there is a significant contribution to the residual stream in the last layer to promote the answer token vs. wait token.

To further investigate this, we apply logit lens to the final layer specifically (Figure A.2, after each component of the transformer block. We find that the final MLP's contribution leads to a significant increase in the logit for "Wait" without patching on the negative number swap. Further investigation into the individual neurons of the MLP indicates that there is not a single neuron leading to this contribution. Applying SVD, seen in A.2 shows that there appear to be several singular vectors contributing, without a clear sparse set which contribute.

4. Conclusion

Our work presents both an empirical and mechanistic evaluation of whether reasoning LLMs backtrack when presented with incorrect values in their chain of thought. We show that models which catch these mistakes with a backtrack nearly recover their original accuracy on mathematical word problems. However, models largely do not backtrack when presented with incorrect values, showing that they either fail to get the correct answer or are unfaithful to their chain of thought. Our circuit analysis shows that there is a mechanism for backtracking which ends with a combination of neurons in the final MLP promoting the backtracking token, and that specific tokens are more likely to lead to a backtrack based on their plausibility. It is not yet clear how models ignore incorrect values in their chain of thought to recover the original correct answer, and whether steering or ablation of certain components can improve both performance and faithfulness.

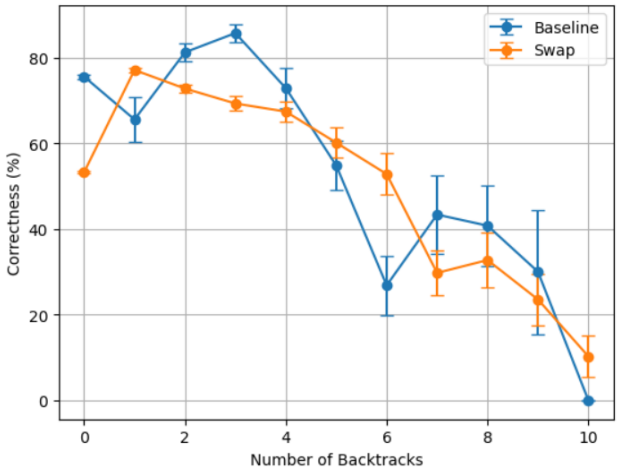

Figure 6. The accuracy of the model outputs given the number of times a backtrack keyword was used. The blue indicates results for the baseline model while orange indicates results for the swapping procedure.

References

Arcuschin, I., Janiak, J., Krzyzanowski, R., Rajamanoharan, S., Nanda, N., and Conmy, A. Chain-of-thought reasoning in the wild is not always faithful, 2025. URL https://arxiv.org/abs/2503.08679.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

DeepSeek-AI, Guo, D., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and editing factual associations in gpt, 2023. URL https://arxiv.org/abs/2202.05262.

Muennighoff, N., Yang, Z., Shi, W., Li, X. L., Fei-Fei, L., Hajishirzi, H., Zettlemoyer, L., Liang, P., Candès, E., and Hashimoto, T. s1: Simple test-time scaling, 2025. URL https://arxiv.org/abs/2501.19393.

nostalgebraist. interpreting gpt: the logit lens, 2020. URL https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens.

OpenAI et al. Openai o1 system card, 2024. URL https://arxiv.org/abs/2412.16720.

Stolfo, A., Wu, B., Gurnee, W., Belinkov, Y., Song, X., Sachan, M., and Nanda, N. Confidence regulation neurons in language models, 2024. URL https://arxiv.org/abs/2406.16254.

Turpin, M., Michael, J., Perez, E., and Bowman, S. R. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting, 2023. URL https://arxiv.org/abs/2305.04388.

Wang, K., Variengien, A., Conmy, A., Shlegeris, B., and Steinhardt, J. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small, 2022. URL https://arxiv.org/abs/2211.00593.

Zhang, A., Chen, Y., Pan, J., Zhao, C., Panda, A., Li, J., and He, H. Reasoning models know when they're right: Probing hidden states for self-verification, 2025. URL https://arxiv.org/abs/2504.05419.

A. Circuit Analysis Additional Information.

A.1. Prompt Templates

We use the following prompt for CoT generation. The generated CoT is then used in the swapping procedure.

```
A conversation between User and Assistant. The user asks a question, and the Assistant
solves it. The assistant first thinks about the reasoning process in the mind and then
provides the user with the answer. The reasoning process and answer are enclosed within
<think> </think> and <answer> </answer> tags, respectively, i.e., <think> reasoning process
here </think> <answer> answer here </answer>. User: Natalia sold clips to 48 of her friends
in April, and then she sold half as many clips in May. How many clips did Natalia
sell altogether in April and May? Assistant:
```

We use the following prompt for our circuit analysis. Note that the value after $c =$ is swapped based on our answer schemes.

```
A conversation between User and Assistant. The user asks a question, and the Assistant
solves it. The assistant first thinks about the reasoning process in the mind and then
provides the user with the answer. The reasoning process and answer are enclosed within
<think> </think> and <answer> </answer> tags, respectively, i.e., <think> reasoning process
here </think> <answer> answer here </answer>. User: Natalia sold clips to 48 of her friends
in April, and then she sold half as many clips in May. How many clips did Natalia
sell altogether in April and May? Assistant:Let c be the number of clips Natalia sold
altogether in April and May.
\[
c = -20
\]
```

A.2. Plots

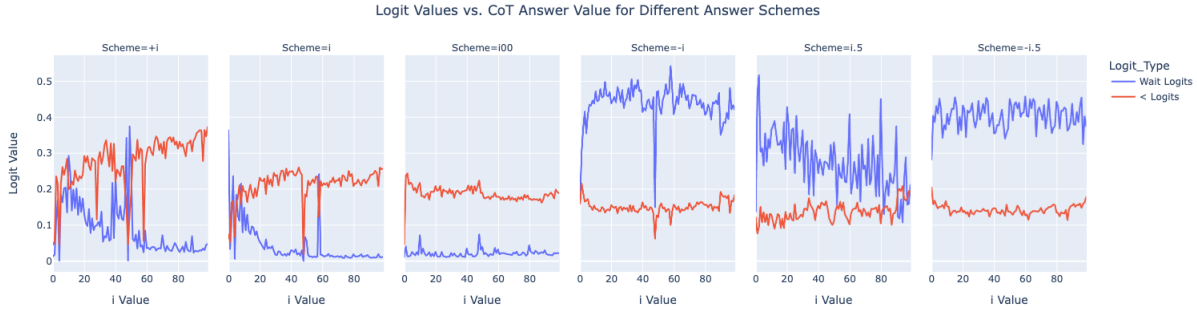

Figure 7. We vary the magnitude of the swapped value within each scheme, and observe the logits of the backtracking vs. final answer token. Answer schemes from left to right: positive integer, positive integer without sign, positive integer multiplied by 100, negative integer, positive decimal value (always 0.5), negative decimal integer (always -0.5).

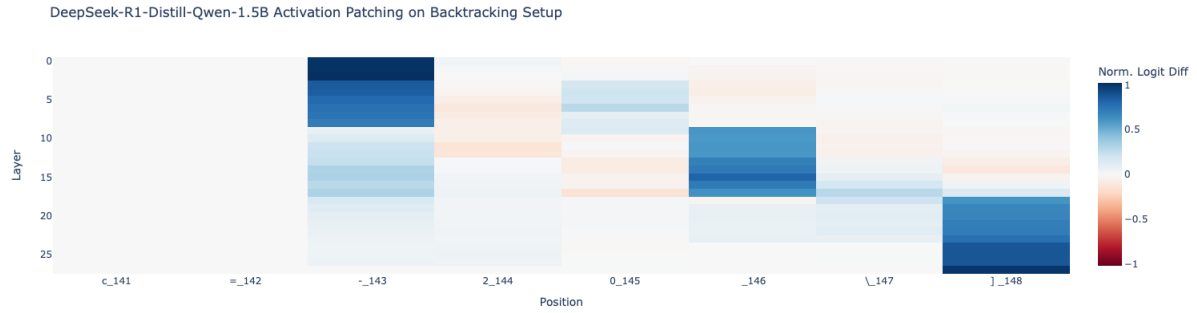

Figure 8. Results from activation patching with a clean prompt (containing the swapped minus token) and a patched in corrupted prompt (containing a plus token). The logit difference changes begin with the swapped token, and show two clear layers at which there is a transfer of information between tokens.

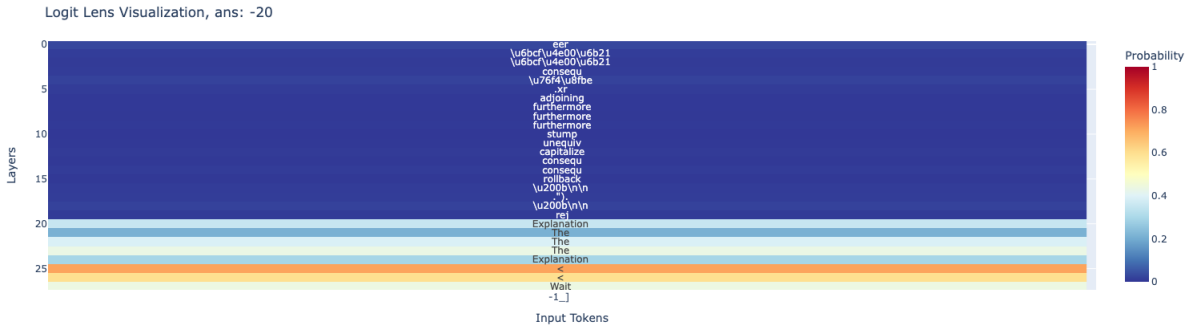

Figure 9. Logit lens, specifically on activations of the last token in a prompt with an artificial answer of -20 inserted. Note that the very last layer leads to a shift from the answer token to the wait token.

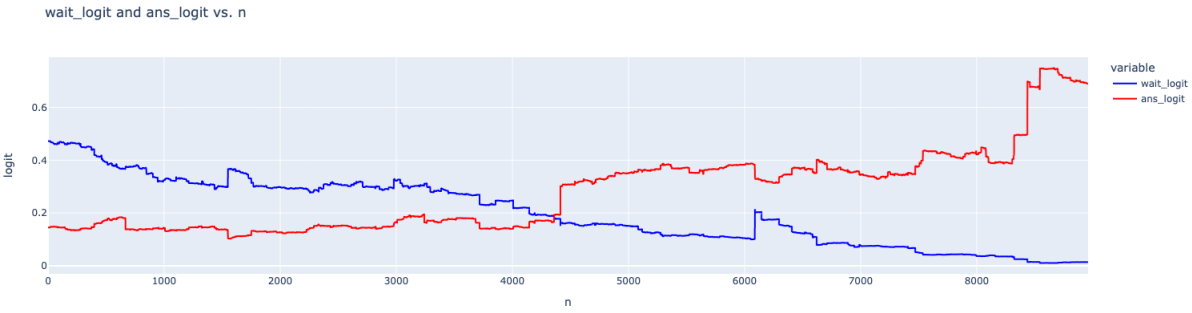

Figure 10. The effect of adding the first n singular vectors to the residual stream at the final layer of the final token in our minus token prompt. The singular vectors are sorted in order by largest singular value.