How do autoregressive transformers solve full addition?

Peixu Wang[1], Yu Chen[1], Ming Yu[2], Cheng Xiang[1]
[1]College of Design and Engineering, National University of Singapore
[2]Power Automation, Singapore
{peixu.wang, c_y}@u.nus.edu

Abstract

Large pre-trained language models have demonstrated impressive capabilities, but there is still much to learn about how they operate. In this study, we conduct an investigation of the autoregressive transformer's ability to perform basic addition operations. Specifically, by using causal analysis we found that a few different attention heads in the middle layers control the addition carry, with each head processing carries of different lengths. Due to the lack of global focus on the sequence within these attention heads, the model struggles to handle long-sequence addition tasks. By performing inference intervention on mistral-7B, partial task performance can be restored, with the accuracy on 20-digit long-sequence additions from 2% to 38%. Through fine-tuning, a new mechanism branches out for handling complex cases, yet it still faces challenges with length generalization. Our research reveals how the models perform basic arithmetic task, and further provides insights into the debate on whether these models are merely statistical.

1 Introduction

As large pre-trained language models increase in scale, they demonstrate increasingly powerful performance on an increasing number of tasks (Brown et al., 2020). But their working principle is still a black box. As the application of large models expands, we have to start to care about safety and ethical issues (Weidinger et al., 2021). On the one hand, some studies believe that the model is just a model that relies on statistics (Bender and Koller, 2020; Merrill et al., 2021). On the other hand, some studies have found that the language model internally encodes other basic world concepts (Abdou et al., 2021; Patel and Pavlick, 2021).

Addition and subtraction, despite being the simplest arithmetic operations, are still challenging tasks for current large language models (Nogueira et al., 2021). Understanding how these models perform such operations internally is highly beneficial for improving their transparency and interpretability.

Figure 1: Two types of mistake commonly made by LLMs: the model only uses localized information for calculations, while the forward addition requires global information to handle carries. When the model cannot obtain information on whether to carry or not, it leads to incorrect outputs.

We focus on the mainstream pre-trained models and investigate their behavior on integer addition tasks, with a focus on challenging but critical cases. For example, when giving the following input to ChatGPT 4: "answer directly without programming: 633331+266667=", the model is highly likely to respond with 900,000 or another incorrect answer starting with 9 (the correct answer is 899,998). In this paper, we will investigate why such errors occur and implement mitigation measures.

We conducted experiments on pre-trained models, including Mistral-7B (Jiang et al., 2023) and LLaMA2-7B (Touvron et al., 2023). While these models demonstrate a baseline accuracy in performing integer addition, they are far from achieving precise results. Through causal analysis, we identified a subset of attention heads, primarily in the middle layers, that are responsible for encoding

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 12732–12756
November 4-9, 2025

digit information relevant to digit-wise addition. Visualizing the associated attention patterns showed a high degree of interpretability. Ablation studies further highlighted the critical role of these heads in determining the output, governing whether the model performs simple modular addition or full addition with carry. However, as the length of the carry chain increases, the information encoded in the attention head gradually loses its significance, accompanied by the rapid decline of interpretability of the attention pattern, resulting in a decrease in the accuracy of the model (See Fig 1).

Building on our discovery of the model's underlying mechanisms, we partially restored its performance through targeted inference intervention. Specifically, we manually adjusted the attention weights of selected attention heads, identified through causal analysis, by either reweighting or ablation during inference. This intervention led to a substantial improvement in accuracy, particularly on longer sequences.

Finally, we perform fine-tuning on Gemma2-2B on specialized addition tasks. The results reveal that while the model refines its original mechanism for simple cases, it implements a new and more reasonable method for summation judgment to handle more complex cases. However, it continues to face challenges with length generalization.

2 Related Work

Studying arithmetic on language models has become popular with the continuous improvement of model capabilities. Zhou et al. (2024) studies how the Transformer processes modular addition from the perspective of Fourier transforms, that the MLP layer mainly approximates the size of the results through low-frequency features, while the attention layer performs modular operations through high-frequency features. Quirke et al. (2023) shows how a one-layer model decomposes the task into parallel digit-specific computation streams and applies different algorithms to each digit by analyzing the training loss curve. Another similar work is Stolfo et al. (2023), which uses a causal mediation analysis framework to reveal the internal information flow in large language models during arithmetic reasoning tasks. Our research aims to go a step deeper and broader, providing a general framework to explain any pre-trained transformer-based autoregressive models' real internal operation process, the findings in our work could be work on a wide scale.

Lee et al. (2023); Zhou et al. (2022); Liu and Low (2023) focus on improving the performance of language models in arithmetic tasks. Their core idea is to extend the reasoning steps of the model, thereby reducing the complexity of serial calculations. This is done by employing training strategies like the "scratchpad" approach, where intermediate reasoning steps are made explicit, or by providing the model with hint prompts that guide it through mathematical expressions.

Broader interpretability research has recently focused on mechanistic interpretability (Geiger et al., 2021; Conmy et al., 2023; Wang et al., 2022). Mechanical interpretation methods, which treat the model as a computational graph composed of attention and MLP components, seek to locate the sub-graph responsible for the actual task computation within the entire computational graph. Hanna et al. (2024) explores the mechanistic interpretability of how GPT-2 performs mathematical comparison (greater-than) tasks through internal circuits without explicit training.

3 Method

3.1 Background

Consider two n-digit integers $X = (x_1, x_2, \ldots, x_n)$, $Y = (y_1, y_2, \ldots, y_n)$ and their addition result $Z = (z_1, z_2, \ldots, z_n)$. The numbers are tokenized digit by digit into the sequence numbers of the vocabulary, mapping to the hidden states h_i^0. In all our experiments, the models used the tokenizer that breaks numbers into individual digits.

$$\alpha_{ij}^{(h,l)} = \frac{e^{q_i^{(h,l)} \cdot k_j^{(h,l)} / \sqrt{d_k}}}{\sum_{j' \le i} e^{q_i^{(h,l)} \cdot k_{j'}^{(h,l)} / \sqrt{d_k}}} \quad (1)$$

$$\tilde{a}_i^{(h,l)} = \sum_{j \le i} \alpha_{ij}^{(h,l)} v_j^{(h,l)} \quad (2)$$

$$a_i^{(l)} = \text{Concat}\left(\tilde{a}_i^{(1,l)}, \ldots, \tilde{a}_i^{(H,l)}\right) W_O^{(l)} \quad (3)$$

$$m_i^{(l)} = W_{down}^{(l)} \sigma\left(W_{up}^{(l)} \gamma\left(a_i^{(l)} + h_i^{(l-1)}\right)\right) \quad (4)$$

Equations 1, 2, 3, and 4 outline how the transformer processes[1] internally. Specifically, the

[1]For simplicity, we omit details of rotary positional encoding in each layer, the implementation of the mixture of experts in Mistral, and the grouped query attention

query, key, and value vectors are defined as $q_i^{(h,l)} = W_Q^{(h,l)} h_i^{(l-1)}$, $k_i^{(h,l)} = W_K^{(h,l)} h_i^{(l-1)}$, $v_i^{(h,l)} = W_V^{(h,l)} h_i^{(l-1)}$. Here, W_Q, W_K, and W_V are learned linear transformations applied to the hidden states $h_i^{(l-1)}$. The matrix W_O acts as the output projection matrix, H denotes the number of attention heads, d_k represents the hidden dimensions, and l indicates the layer index, the hidden states are updated through the residual stream $h_i^{(l)} = h_i^{(l-1)} + a_i^{(l)} + m_i^{(l)}$.

The σ is a non-linear function. γ is a normalization function. W_{up} and W_{down} are learned weight matrices in the feed-forward network, where W_{up} expands the dimensionality of the input, and W_{down} reduces it back to the original dimension.

Different from the way humans calculate, the autoregressive model needs to output the answer from front to back, which has been found to be a more challenging task for the models (Lee et al., 2023; Zhang-Li et al., 2024). Consider the process needed for the model to output correct z_i, there are three cases. Case 1: When $x_{i+1} + y_{i+1} < 9$, $z_i = (x_i + y_i) \mod 10$. Case 2: When $x_{i+1} + y_{i+1} > 9$, then $z_i = (x_i + y_i) \mod 10 + 1$. Case 3: When $x_{i+1} + y_{i+1} = 9$, we need to check if there is a carry from subsequent digits. If $x_{i+2} + y_{i+2} > 9$, then $z_i = (x_i + y_i) \mod 10 + 1$; if $x_{i+2} + y_{i+2} < 9$, then $z_i = (x_i + y_i) \mod 10$; if $x_{i+2} + y_{i+2} = 9$, continue checking further. The value of z_i is not only decided by x_i and y_i, but also by a chain of conditions from subsequent digit pairs. In a worst case, it must evaluate digits all the way to the end of the sequence, making the task much more difficult due to the need for multiple conditional judgments.

While modular addition can be easily computed in parallel mathematically, full forward addition is fundamentally different. To better capture the nature of the task, we define two types of study objects. An equation with a carry chain of length d as *CCd* (Carry Chain). For example, 44 + 28 is *CC1*, and 35556 + 24447 is *CC4*. On the other hand, an equation that only has a chain format without an actual carry is called *OCd* (Only Chain). For example, 3445 + 2552 is *OC3*, and 35556 + 24442 is *OC4*, as their corresponding digit sums equal 9 repeatedly, creating a pattern allowing propagation without triggering actual carries. These two equations are similar in format but completely different in result.

We analyze the behavior of Mistral-7B on OCd and CCd tasks (Figure 2), where d ranges from 1 to 15. Each d corresponds to 200 samples (100 OC and 100 CC). For OC inputs $X + Y$ and CC inputs $X' + Y'$, we focus on the model's prediction of the first output digit z_1. Since z_1 depends on carry decisions propagated from all subsequent digit pairs, it presents the state for the entire problem. Once z_1 is predicted, the state is fixed and will directly influencing the remaining outputs. We compute the average correct probability distributions $p(z_1|X, Y)$, $p(z_1'|X', Y')$, and error probability $p(z_1' - 1|X', Y')$ (missed carry), $p(z_1 + 1|X, Y)$ (spurious carry). To simulate general scenarios, a randomly generated sequence of length d is appended to each number, resulting in a number length of $2d + 1$.

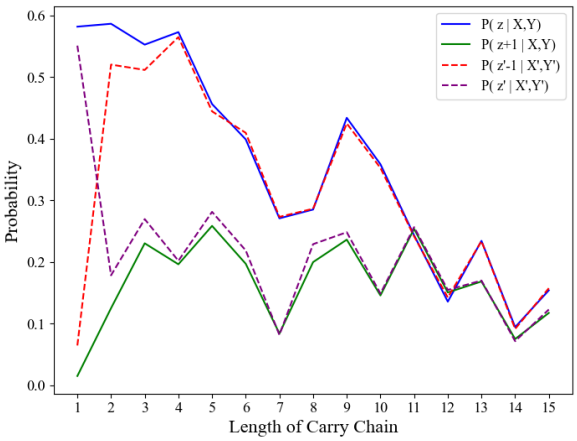


Figure 2: The average probability output of Mistral-7B on *OC* and *CC* tasks, with each value of length corresponding to 200 samples.

The result shows a reasonable overall decline in performance as the sequence length increases. However, beyond the simplest case $d = 1$, the model can hardly distinguish *CC* tasks from *OC* tasks. When $d > 3$, the lines nearly overlap, suggesting inputs like '13337+16663' may not be different from '13337+16662' for the model.

3.2 Causal Analysis

Causal analysis (Vig et al., 2020; Pearl, 2022; Meng et al., 2022) is a technique based on activation replacement, helping to reveal the causal roles of internal model components and understand how they contribute to outcomes. We first create a set of two similar but different inputs, *OCd* input $X + Y$ and *CCd* input $X' + Y'$. Naturally, $z_1 = x_1 + y_1$, $z_1' = x_1' + y_1' + 1$. And set restriction: $x_i = x_i', y_i = y_i' (i < d+1)$; $y_{d+1}' + x_{d+1} > 9$ or $x_{d+1}' + y_{d+1} > 9$ (randomly chosen). Figure

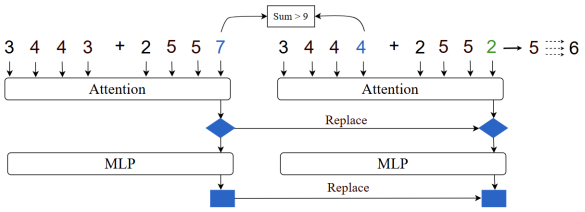


Figure 3: An interpretation of the causal analysis is presented in this example, where $y'_{d+1} + x_{d+1} > 9$, causing the output shift from 5 to 6.

3 explains how our method works. Two OC inputs can produce the same results as long as the constraints are maintained (see Appendix A).

We conduct three rounds of model inference. The symbols below refer to Equations 1,2, 3, 4.

- **In the first run:** $(X+Y)$ as the input to obtain the final probability output $p(z_1|X,Y)$ and $p(z'_1|X,Y)$, collect the activation o at y_{d+1} token position at l layer if $y_{d+1} + x'_{d+1} > 9$, or x_{d+1} token position if $x_{d+1} + y'_{d+1} > 9$, $o \in \{m^{(1)}_{d+1}, \ldots, m^{(L)}_{d+1}, v^{(1)}_{d+1}, \ldots, v^{(L)}_{d+1}\}$.
- **In the second run:** $(X' + Y')$ as the input and collect the activation o' at y'_{d+1} token position if $y'_{d+1} + x_{d+1} > 9$, or x'_{d+1} token position if $x'_{d+1} + y_{d+1} > 9$, $o' \in \{{m'_{d+1}}^{(1)}, \ldots, {m'_{d+1}}^{(L)}, {v'_{d+1}}^{(1)}, \ldots, {v'_{d+1}}^{(L)}\}$.
- **In the third run:** $(X + Y)$ as the input and replace the activation o with o' to obtain the probability output $p^*(z'_1|X,Y)$.

We sequentially use o' to override the original activation o to change the model's probability output during inferencing. Intuitively, this should lead to an increase in the model's output probability for z'_1. The total effect is defined as equation 5. We used 100 sets of number pairs as input for Mistral-7B and calculated their average TE. For activation replacement of the attention layer, v is chosen instead of $\tilde{a}$ or a because non-trivial result first and only occurs on v. Causal analysis of other components ($\tilde{a}$ and a) refer to Appendix A.

$$\text{Total Effect} = p^*(z'_1|X,Y) - p(z'_1|X,Y) \quad (5)$$

Through our experimental analysis, we have the following findings (See Fig 4): the most significant impact occurs in the middle to later layers, particularly between layers 15 and 20. For the simplest case $d = 1$, a few attention heads strongly influence the output, causing up to a 50% change in probability. However, as d increases, the maximum probability difference declines rapidly, with only 2% difference at $d = 2$. For MLP layers, their effect primarily arises from their indirect influence on the attention heads that are identified. Additionally, to have a general conclusion, we also conduct analysis with prefixes and suffixes padding on the input, and observe similar model behavior. (Causal analysis of MLP, other models, effect breakdown, analysis with padding, refer to Appendix A, a TSNE visualization in Appendix B).

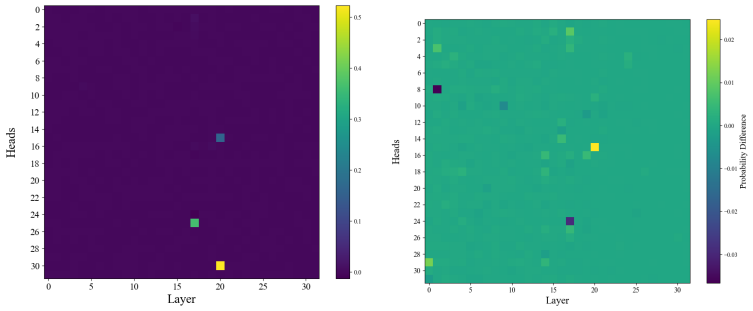
The equation 2 illustrates how the replaced v vector affects the output through the attention pattern. To recall that the v vector position we replaced in Section 3.2 is $pos(y'_{d+1})$ (or $pos(x'_{d+1})$ depending on the sample), so replaced v affects the output through the attention weight $\alpha_{n,pos(y'_{d+1})}$, where n the last sequence position.

Among the attention heads observed in the causal analysis, we visualize the top two attention heads that cause the largest TE for each value of d (See Fig 5).

It is observed that when "=" token appears, the model focuses on x_{i+1} and y_{i+1} in order to calculate z_i, just like a double pointer. As the length of the output sequence increases, it continues to move towards, forming a double staircase pattern. For more complex cases $d > 1$, the model ideally needs to attend to x_{i+d} and y_{i+d} to retrieve the correct carry information, in practice, the attention heads fail to maintain this staircase structure, and weights are unevenly distributed on each digit. This indicates that the model have far less control over more complex cases, which aligns with its declining accuracy (See table 1).

3.3 Attention implements incomplete carry

Our research focuses on two objectives: first, to assess the influence of the targeted attention heads on the corresponding CC task, the second is to investigate whether the model's significant performance decline on CC tasks with $d > 1$ is solely due to insufficient attention weights allocation. To address the first objective, we perform ablation on the top two attention heads for the corresponding CC task, by setting $\alpha_{n,pos(y'_{d+1})} = 0, \alpha_{n,pos(x'_{d+1})} = 0$ to eliminate the influence of the v vector. Additionally, we randomly select heads in the same layer and conduct a zero ablation on the same token position for comparison. For the second question, we



(a) d = 1 (b) d = 2

(c) d = 3 (d) d = 4

Figure 4: The attention heads located through causal analysis for mistral-7B under different values of d. Mistral-7B has 32 attention heads per layer, for a total of 32 layers.

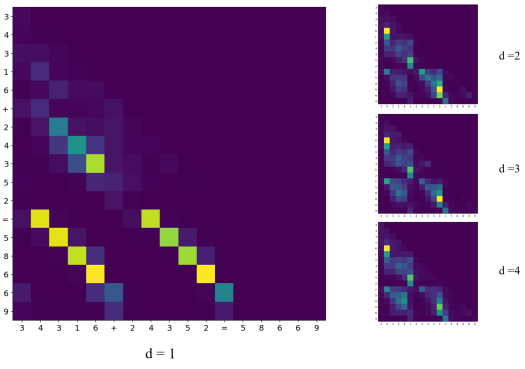


Figure 5: Superimposed top-2 attention heads on Mistral-7B, 34316+24352=58669 is used as a demonstration example. For simplicity, the pattern omits the prompt and only retains the question.

dynamically perform re-weighting (6) according to the input.

$$\alpha_{n,pos(x'_{d+1}/y'_{d+1})} = \alpha_{n,pos(x'_{d+1}/y'_{d+1})} + \lambda \quad (6)$$

The result shows that for the $CC1$ task with high accuracy, there is a strong correlation between attention and output (See Table 1). Ablating the top-2 heads has an immediate and dramatic effect. Notably, after ablating on Gemma, the accuracy of the $CC1$ task drops from 99.3% to just 1.47%. The ablation removes the carry operation and reverts the output to $z'_1 - 1$, while leaving the modular addition unaffected. (See Fig 6 and details of reverting in Appendix A). As d increases, the accuracy declines, and the impact of ablation diminishes accordingly.

When re-weighting weights to match $CC1$ level, we observed varying degrees of improvement across different CCd tasks. As d increases, the effect of re-weighting continues to diminish. This suggests that the model's difficulty in handling carries is not solely due to insufficient attention weight allocation, but rather a deeper issue in the model's mechanism for processing such operations. Additionally, the relatively low accuracy of Llama2 can be attributed to its inherently lower performance in modular addition (See Appendix C).

To summarize the experimental findings, pre-trained autoregressive models rely on the staircase

Table 1: Ablation study (with $\lambda = 0.6$) on Mistral-7B, Llama2-7B, and Gemma-7B models. Numbers in parentheses are the baseline for the OC task.

Model	Method	$CC1$	$CC4$	$CC6$	$CC10$
Mistral-7B	Baseline	99.21(98.32)	29.99(80.24)	20.31(67.18)	17.93(23.26)
	Zero ablation	**34.80**	24.51	19.21	17.09
	Random ablation	96.99	29.46	21.32	17.45
	Re-weighting	98.12	**41.83**	**23.87**	**19.20**
Llama2-7B	Baseline	91.17(98.09)	48.21(44.23)	17.33(12.41)	0.67(0.21)
	Zero ablation	**17.51**	42.14	17.23	0.67
	Random ablation	88.92	49.21	17.30	0.68
	Re-weighting	89.32	**58.43**	**22.78**	**4.63**
Gemma-7B	Baseline	99.33(99.21)	80.60(37.26)	49.98(31.65)	25.17(26.88)
	Zero ablation	**1.47**	49.72	28.69	20.64
	Random ablation	96.84	78.53	48.57	25.07
	Re-weighting	95.10	**92.73**	**58.48**	**31.46**

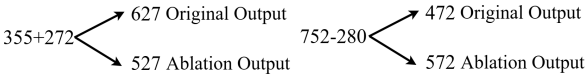


Figure 6: Zero ablation causes the output of the model to revert back to modular operation.

attention patterns to transmit carry-encoded value tokens to the final token for prediction. When this information transmission is disrupted, the model defaults to modular addition. As the carry length increases, the model loses its ability to transmit this information, leading to disordered attention patterns and a loss of carry-related information in the value vectors.

4 Inference intervention

Based on the findings in Section 3, we performed a full intervention enhancement experiment on the model's addition task. Many of the model's errors stemmed from incorrect handling of carries— missing or introducing an unnecessary one. In contrast, errors related to modular addition were relatively minor (See Appendix B). The primary goal of this experiment is to explore the potential for improving the model's performance by attempting to restore its internal mechanisms without training, rather than transforming the model into an accurate calculator.

4.1 Experiments

Our experimental procedure is summarized in Algorithm 1. When the model processes $X + Y$, it dynamically intervenes in the internal activations. During the generation of z_i, we determine whether a carry occurs from the subsequent sequence. If no carry is detected, we perform zero ablations on the attention weights for x_{i+1} and y_{i+1}, allowing the model to carry out modular addition. If a carry arises from a chain of length m, the corresponding attention weight $a_{n,i+m}$ is re-weighted. The attention heads are chosen from the top two that produce the largest TE when $d = m$. For further details, see Appendix C.

In the experiment, number pairs of each length n were randomly sampled from the interval $(10^{n-1}, 10^n)$. For every length, we constructed a dataset of 9,000 questions using greedy sampling to ensure adequate coverage across the input space. Since evaluating numbers of length n requires n sequential interferences, the computational cost grows rapidly with sequence length. As a result, we limited the experiment to sequences of length at most 20, which already represent a demanding computational setting.

Our methodology is related to prior work on activation intervention, a technique that manipulates the activations of specific components in the reasoning process to adjust or steer the behavior of the model. This line of work originates from interpretability studies of deep neural networks (Adi et al., 2016; Finlayson et al., 2021; Vig et al., 2020), where interventions are typically applied to gain insights into internal mechanisms. In contrast, our experiments involve only lightweight modifications, often adjusting no more than a few dozen scalar values, yet these small-scale interventions are suffi-

cient to produce measurable changes in the model's behavior.

Algorithm 1 Inference intervention

Input: $X + Y$, L: The last sequence position, m: Length of the carry chain, A_m: Top 2 attention heads located in causal analysis when $d = m$.
Output: Z
for $i \leftarrow 0$ **to** $n - 1$ **do**
  if $x_{i+1} + y_{i+1} < 9$ **then**
    $z_i \leftarrow$ Inference with ablation $(a_{L,pos(x_{i+1})}, a_{L,pos(y_{i+1})})$ on Head A_1
  else
    if Carry exist: **then**
      $z_i \leftarrow$ Inference with re-weighting $(a_{L,pos(x_{i+m})}, a_{L,pos(y_{i+m})})$ on Head A_m
    else
      $z_i \leftarrow$ Inference with ablation $(a_{L,pos(x_{i+1})}, a_{L,pos(y_{i+1})})$ on Head A_1
    end if
  end if
end for

4.2 Results

The results (Fig. 7) indicate consistent improvement across most cases, with longer sequences showing more substantial gains (see additional models in Appendix C). For short sequences, however, the improvement is negligible. This is largely because the proportion of randomly sampled short sequences that include $CC2$ or higher is relatively small. For example, among sequences of length 3, only about 17% contain a CC_d instance where $d \neq 1$. This proportion rises to 56% at length 10 and reaches 86% by length 20. As shown in Table 1, the benefits of re-weighting are concentrated in more challenging cases, while performance slightly deteriorates for $CC1$, where the baseline model already performs well without intervention.

Despite these improvements, the effect has a clear upper bound. Accuracy remains close to zero for sequences of length 60 (Appendix C), suggesting that re-weighting alone cannot overcome fundamental limitations. One reason, as discussed in Section 3, is that adjusting weights only partially mitigates the degradation caused by information loss and cannot fully recover long-range dependencies. Another limiting factor is that while the intervention enhances the model's ability to manage carry operations, it does not improve its core ability to perform modular addition. Furthermore, a large fraction of errors stem from misalignment issues, where the model incorrectly adds x_i with y_j for $i \neq j$ (Appendix B). Prior studies (McLeish et al., 2024; Shen et al., 2023) suggest that positional encodings are a major contributing factor to such alignment errors, further constraining the effectiveness of re-weighting.

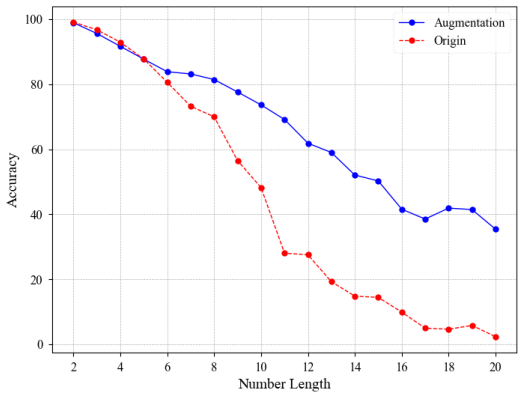
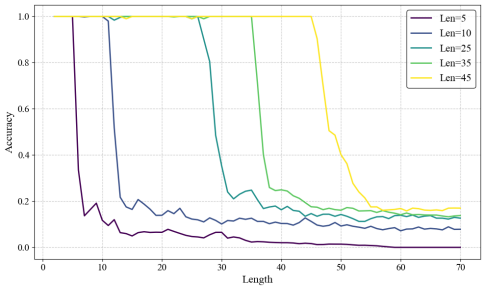
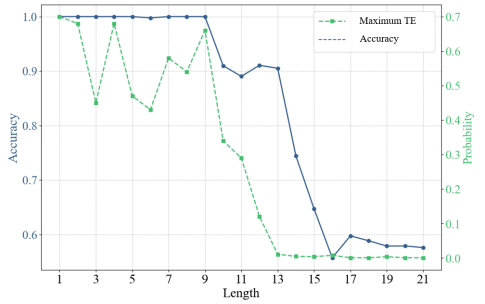


Figure 7: Comparison of accuracy between the baseline and inference intervention augmentation on Mistral-7B.



(a) Performance on CC tasks



(b) Accuracy on CC tasks and maximum TE trained on a maximum length of 10

Figure 8: (a): Performance of Gemma2-2B on CCd under different max training length d. (b): The accuracy of CCd task on training length $d = 10$, along with the corresponding and maximum TE tracked throughout.

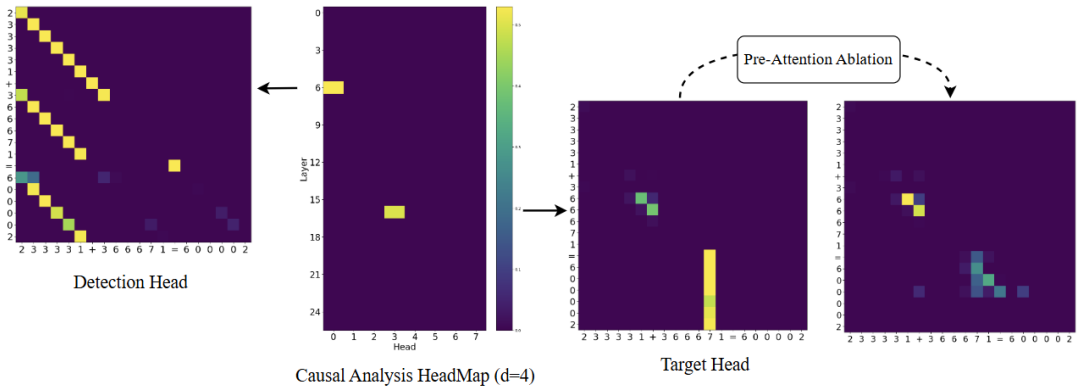


Figure 9: Causal Analysis on fine-tuned Gemma2-2B. "233331+366671=600002" is used as the demonstration example. Left: The Detection Head Pattern. Middle Left: Causal analysis on all layers and heads. Middle Right: Target head patterns. Right: Target head patterns with detection head melted.

5 Fine-tuning

Furthermore, we extend our investigation to the model's internal processing after fine-tuning on more complex tasks, conducting full-parameter fine-tuning experiments on the Gemma2-2B. The dataset consists of 30% CCd, 30% OCd equations, and 40% randomly generated number pairs with a length upper limit of 80 for ensuing basic modular addition accuracy. The fine-tuned model results are shown in Figure 8. The detailed training parameters and dataset are provided in Appendix E.

5.1 Generalization

In circuit complexity theory, addition and subtraction belong to AC^0, while studies have found that transformers have a best upper bound of TC^0 (Merrill and Sabharwal, 2023). There are also studies showing the difficulties transformers face in generalizing addition and subtraction. Zhou et al. (2023) show that, due to the existence of complex carry, addition cannot be expressed by RASP-L languages and therefore cannot be generalized. Likewise, Huang et al. (2024) demonstrates that addition cannot be represented by limit transformers, as it violates the PERIODIC and LOCAL conditions. Even simpler tasks, such as counting (Chang and Bisk, 2024) and PARITY (Hahn and Rofin, 2024), are empirically difficult for transformers to generalize.

Studies (Sabbaghi et al., 2024; McLeish et al., 2024) emphasize the importance of positional encoding in enabling length generalization for arithmetic tasks. Effective positional encoding enhances generalization by capturing the structural alignment (Sabbaghi et al., 2024; McLeish et al., 2024). However, long carry chains remain challenging (Sabbaghi et al., 2024), as they require precise digit alignment and conditional reasoning based on digit sums. While Gemma2-2B utilizes RoPE, Kazemnejad et al. (2024) highlights its limitations in achieving length generalization.

Figure 8 shows the fine-tuning performance of Gemma2-2B, where accuracy drops sharply beyond the maximum training length, indicating poor generalization. We attribute this to attention head formation, as results in Figure 8b reveal that maximum TE declines beyond the training length, suggesting insufficient attention heads to transfer carry-digit information effectively. Detailed causal analysis heatmaps and OC task performance refer to Appendix D.

5.2 Emergence of new strategy

Pre-trained models address cases such as $OC1$ and $CC1$ by statically focusing attention on the digit right after the current calculation digit. After fine-tuning, the model not only retains and refines this original mechanism (see Appendix D) but also develops a complementary and dynamical strategy (See Fig 9) for handling more complex cases ($d > 2$). The original mechanism remains active for simpler cases ($d \leq 2$) and operates in parallel with the new strategy.

Notably, fine-tuning introduces a specialized attention head, termed the **Target Head**, which directly focuses on the actual carry digit. For example, in Figure 9, the attention weight is centralized on the digit "7". Additionally, a second attention head, referred to as the **Detection Head**, emerges. This head becomes active prior to the appearance of the "=" token, enabling it to transmit the rele-

Table 2: Ablation study on the fine-tuned Gemma2-2B model (trained on a maximum length of 10).

Method/Task	$CC1$	$CC2$	$CC4$	$CC10$
Baseline	100	100	99.67	91.72
Detection Head Ablation	100	100	**55.42**	**54.32**
Target Head Ablation	100	100	**51.58**	**54.21**
Combined Ablation	100	99.47	44.11	48.12
Random Ablation	99.68	99.02	99.12	90.47

vant x_i information to the current y_i token when it appears (See result of other model in Appendix D).

A guess is that the detection head obtains the information of x_i and y_i to determine whether $x_i + y_i$ is greater than 9. The carry chain length-related information is then passed to the Target Head, enabling it to focus on the actual carry token and execute the carry operation. To validate this, we performed zero ablation on the Detection Head and observed a disruption in the Target Head's attention patterns, resulting in a significant accuracy decrease for complex cases, while simpler cases remained unaffected (see ablation studies in table 2). Intuitively, the differentiation of attention functions is more reasonable, since it involves the concept of sum judgment. From the perspective of mechanism formation, it is recommended to explicitly include the infrequent arithmetic cases in the training dataset rather than relying on randomly sampled numbers. But the underlying reason for the phase change remains unclear and addressing this question will require further studies.

6 Conclusion

In this study, we investigated how pre-trained autoregressive language models perform addition operations.

We found the model relies on localized attention distribution for handling carry operations, which makes it challenging to process inputs with long sequences. Some task performance can be recovered by intervening in attention during inference without training, but the inherent limitations remain. Finally, fine-tuning on specialized addition tasks led the model to develop a more efficient mechanism for complex cases while retaining the original strategy for simpler ones, yet still challenging to achieve generalization, likely due to its inability to form functional attention heads.

Our findings offer valuable insights into how language models process arithmetic tasks and help assess whether they rely on statistical patterns or deeper reasoning.

Limitations

While our study focuses on addition tasks, this represents only a small subset of arithmetic. Multiplication and division are much more complex operations that require further investigation. Although we propose a general framework for analyzing autoregressive models, one limitation is the tokenization, as some models like GPT 3.5/4 don't apply a 'single-digit' tokenization approach, and our method needs to be adjusted. Our conclusions are also constrained by the architecture of the specific pre-trained models studied. Models with larger scales or different components (particularly positional encoding) may exhibit different behaviors. Training different models from scratch with customized architectures can get more generalizable insights.

References

Mostafa Abdou, Artur Kulmizev, Daniel Hershcovich, Stella Frank, Ellie Pavlick, and Anders Søgaard. 2021. Can language models encode perceptual structure without grounding? a case study in color. *arXiv preprint arXiv:2109.06129*.

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv preprint arXiv:1608.04207*.

Emily M Bender and Alexander Koller. 2020. Climbing towards nlu: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 5185–5198.

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

Yingshan Chang and Yonatan Bisk. 2024. Language models need inductive biases to count inductively. *arXiv preprint arXiv:2405.20131*.

Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. 2023. Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, 36:16318–16352.

Matthew Finlayson, Aaron Mueller, Sebastian Gehrmann, Stuart Shieber, Tal Linzen, and Yonatan Belinkov. 2021. Causal analysis of syntactic

agreement mechanisms in neural language models. *arXiv preprint arXiv:2106.06087*.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. 2021. Causal abstractions of neural networks. *Advances in Neural Information Processing Systems*, 34:9574–9586.

Michael Hahn and Mark Rofin. 2024. Why are sensitive functions hard for transformers? *arXiv preprint arXiv:2402.09963*.

Michael Hanna, Ollie Liu, and Alexandre Variengien. 2024. How does gpt-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. *Advances in Neural Information Processing Systems*, 36.

Xinting Huang, Andy Yang, Satwik Bhattamishra, Yash Sarrof, Andreas Krebs, Hattie Zhou, Preetum Nakkiran, and Michael Hahn. 2024. A formal framework for understanding length generalization in transformers. *arXiv preprint arXiv:2410.02140*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, and 1 others. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. 2024. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*, 36.

Nayoung Lee, Kartik Sreenivasan, Jason D Lee, Kangwook Lee, and Dimitris Papailiopoulos. 2023. Teaching arithmetic to small transformers. *arXiv preprint arXiv:2307.03381*.

Tiedong Liu and Bryan Kian Hsiang Low. 2023. Goat: Fine-tuned llama outperforms gpt-4 on arithmetic tasks. *arXiv preprint arXiv:2305.14201*.

Sean McLeish, Arpit Bansal, Alex Stein, Neel Jain, John Kirchenbauer, Brian R Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, Jonas Geiping, Avi Schwarzschild, and 1 others. 2024. Transformers can do arithmetic with the right embeddings. *arXiv preprint arXiv:2405.17399*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372.

William Merrill, Yoav Goldberg, Roy Schwartz, and Noah A Smith. 2021. Provable limitations of acquiring meaning from ungrounded form: What will future language models understand? *Transactions of the Association for Computational Linguistics*, 9:1047–1060.

William Merrill and Ashish Sabharwal. 2023. A logic for expressing log-precision transformers. *Advances in neural information processing systems*, 36:52453–52463.

Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. 2021. Investigating the limitations of transformers with simple arithmetic tasks. *arXiv preprint arXiv:2102.13019*.

Roma Patel and Ellie Pavlick. 2021. Mapping language models to grounded conceptual spaces. In *International conference on learning representations*.

Judea Pearl. 2022. Direct and indirect effects. In *Probabilistic and causal inference: the works of Judea Pearl*, pages 373–392.

Philip Quirke and 1 others. 2023. Understanding addition in transformers. *arXiv preprint arXiv:2310.13121*.

Mahdi Sabbaghi, George Pappas, Hamed Hassani, and Surbhi Goel. 2024. Explicitly encoding structural symmetry is key to length generalization in arithmetic tasks. *arXiv preprint arXiv:2406.01895*.

Ruoqi Shen, Sébastien Bubeck, Ronen Eldan, Yin Tat Lee, Yuanzhi Li, and Yi Zhang. 2023. Positional description matters for transformers arithmetic. *arXiv preprint arXiv:2311.14737*.

Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. 2023. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. *arXiv preprint arXiv:2305.15054*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33:12388–12401.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, and 1 others. 2021. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*.

Daniel Zhang-Li, Nianyi Lin, Jifan Yu, Zheyuan Zhang, Zijun Yao, Xiaokang Zhang, Lei Hou, Jing Zhang, and Juanzi Li. 2024. Reverse that number! decoding order matters in arithmetic learning. *arXiv preprint arXiv:2403.05845*.

Hattie Zhou, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Josh Susskind, Samy Bengio, and Preetum Nakkiran. 2023. What algorithms can transformers learn? a study in length generalization. *arXiv preprint arXiv:2310.16028*.

Hattie Zhou, Azade Nova, Hugo Larochelle, Aaron Courville, Behnam Neyshabur, and Hanie Sedghi. 2022. Teaching algorithmic reasoning via in-context learning. *arXiv preprint arXiv:2211.09066*.

Tianyi Zhou, Deqing Fu, Vatsal Sharan, and Robin Jia. 2024. Pre-trained large language models use fourier features to compute addition. *arXiv preprint arXiv:2406.03445*.

A Appendix - Additional information for casual analysis

This section discusses details for other models, general addition task analysis, output reverting, and some interactions we observed between attention and MLP.

The causal analyses of Llama2-7B and Gemma-7B are shown in Figure 10 and Figure 12, respectively. In Llama2-7B, the influential attention heads are located earlier than those in Mistral, with significant TE emerging as early as layer 13. As d increases, the maximum TE decreases rapidly. Some heads, such as (4,14) and (11,17), show a degree of robustness across different values of *d*. In contrast, the key attention heads in Gemma-7B are found much later, between layers 20 and 23.

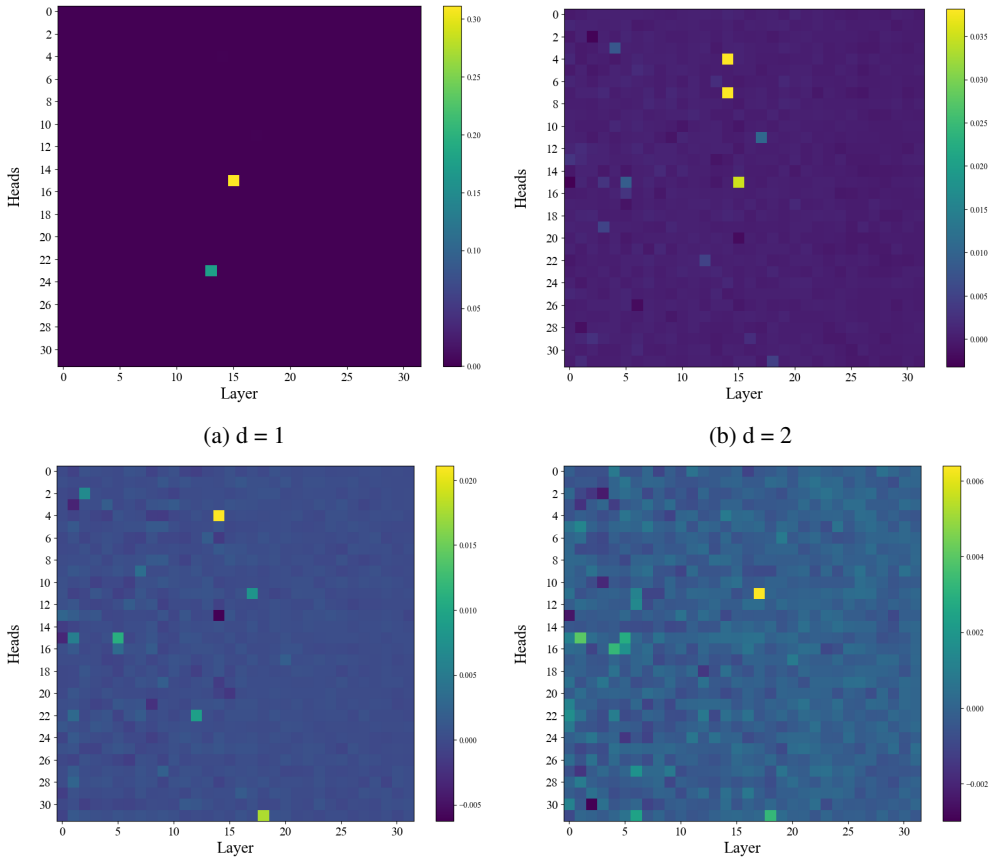
When applying causal analysis, using two *OC* inputs does not produce significant differences in results compared to one *CC* and one *OC* input, figure 11 shows the causal analysis results based on two OCd inputs. The targeted heads are similar to the heads in shown in figure 4.

When performing activation intervention on a certain component, its total effect can be divided into two parts (Vig et al., 2020): one is that the component directly affects the output probability by writing the residual stream value to cause direct effects (*DE*), and the other is that the residual stream passes the influence to downstream components to cause indirect effects (*IE*) (See Fig 13).

We found that the TE caused by MLP mainly comes from the indirect effect on downstream attention components, especially on the Top 2 attention heads. We set up an additional experimental process to distinguish the degree of influence between the two. Specifically, when we perform activation intervention on $m_1'^{l}$ we fix the top 2 attention heads components as their original activation a_i^{Top2} .

The results (See Fig 14) show that the impact of MLP on the results was concentrated in the early stage of the model, and most of it was earlier than the influence range of the attention heads. After restoring the top 2 attention heads, the impact of MLP decreased to an insignificant level. This may represent that the role of MLP in the early stage is to provide the basic work of information processing for attention layers.

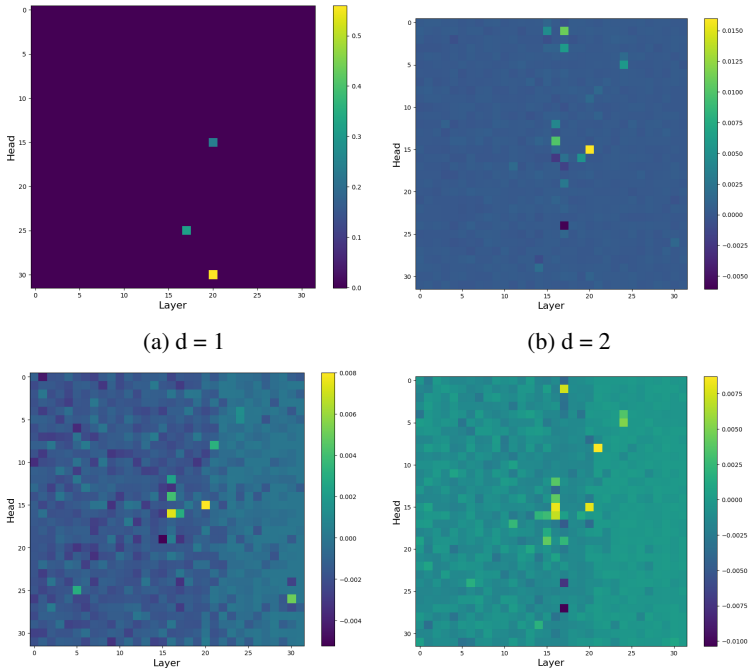
In addition to the activation replacement of v, we also conducted experiments on $\tilde{a}$ and a. (there is no concept of head index in a.) The results (See Fig 15b, 15a, 16b, 16a) show that these compo-



(a) d = 1 (b) d = 2

(c) d = 3 (d) d = 4

Figure 10: The attention heads located through causal analysis show that for Llama2-7B under different values of d. Llama2-7B has 32 attention heads per layer, for a total of 32 layers.



(a) d = 1 (b) d = 2

(c) d = 3 (d) d = 4

Figure 11: The attention heads located through causal analysis based on two OC inputs in Mistral-7B

nents are difficult to significantly impact the output probability (even if $d = 1$).

To generalize our findings, we conduct an additional analysis by applying prefix and suffix padding to the input numbers (See table 3,4). These paddings do not form carry chains and therefore do not affect the chain length d, but they allow us to extend the study from a specialized setting to a more general addition scenario. For example, a $CC2$ case like `445 + 357` can be transformed into `21`**`445`**`52 + 32`**`357`**`15` with 2-digit prefix and suffix padding. To avoid interference with modular addition accuracy, the padding length is set as $2d$. The results show similar model behavior: the locations of the Top-2 attention heads remain unchanged, and the maximum TE caused by these heads does not significantly differ from the analysis without padding.

Zero ablation on Top-2 attention heads can greatly influence the model output, and most of the changes of the output are reverting to modular addition (See Table 5), each task corresponds to 500 samples. This confirms that ablating the Top-2 attention heads primarily disrupts the carry mechanism while having minimal impact on modular addition.

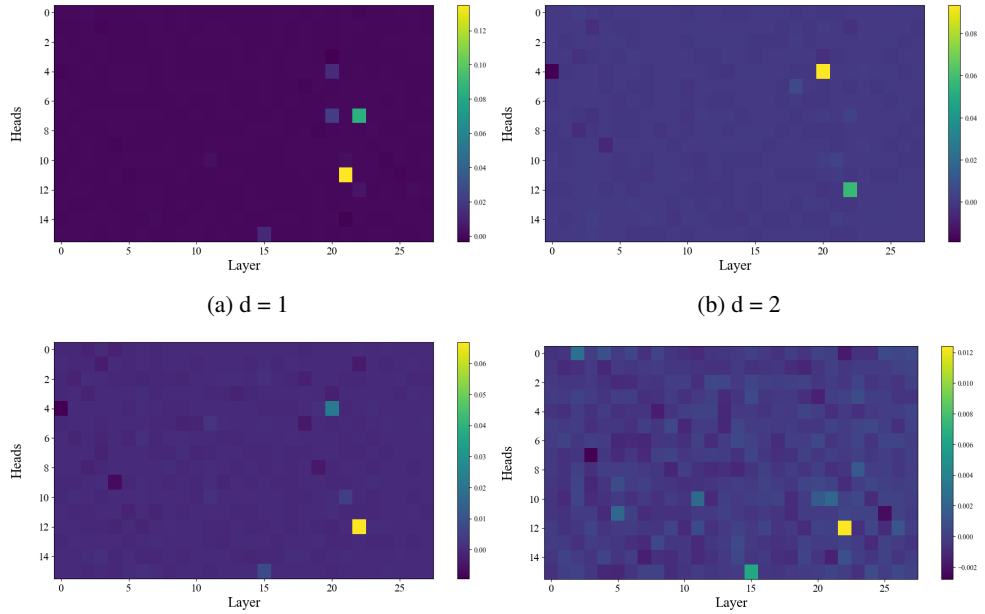
B Appendix - Visualization

We provide some visualizations in this section, mainly including the process of handling modular additions and additions with carry inside the model.

In Figure 17, layer 12 shows that data points with similar labels (indicated by similar colors) are intermixed. Each color represents an addition question categorized by its first-digit answer. For example, 34+12 and 25+17 fall into the same category since both yield a first-digit answer of 4. The activations are collected at the "=" token position. At layer 13, the clustering pattern immediately undergoes a sudden change, and the model distinguishes between carry and non-carry equations, which are much closer to the output of the final layer. Layer 13 is also the earliest layer to be located through causal analysis (See Fig 10a).

Similarly, a similar process also occurs in the mistral-7B model (See Fig 18). After passing through the 17th layers, the model quickly distinguishes between CC and OC tasks, the causal analysis location of the 17th layer refers to 4a.

For modular addition, the visualization results show that the model is implemented in a progressive manner, rather than dealing with abrupt changes like handling carry.



(a) d = 1 (b) d = 2

(c) d = 3 (d) d = 4

Figure 12: The attention heads located through causal analysis show that for Gemma-7B under different values of d. Gemma-7B has 16 attention heads per layer, for a total of 26 layers.

Table 3: Causal analysis on Mistral-7B. The maximum TE and the top-2 attention heads under different input formatting conditions.

Metric/Task	**OC1 and CC1**	**2-digit Prefix**	**2-digit Suffix**	**2-digit Prefix + Suffix**
Maximum TE	0.521	0.506	0.542	0.536
Top-2 Attention Heads (Layer, Head)	(20, 30), (17, 25)	(20, 30), (17, 25)	(20, 30), (17, 25)	(20, 30), (17, 25)
Metric/Task	**OC2 and CC2**	**4-digit Prefix**	**4-digit Suffix**	**4-digit Prefix + Suffix**
Maximum TE	0.023	0.021	0.023	0.023
Top-2 Attention Heads (Layer, Head)	(15, 20), (3, 17)	(15, 20), (3, 17)	(15, 20), (3, 17)	(15, 20), (3, 17)

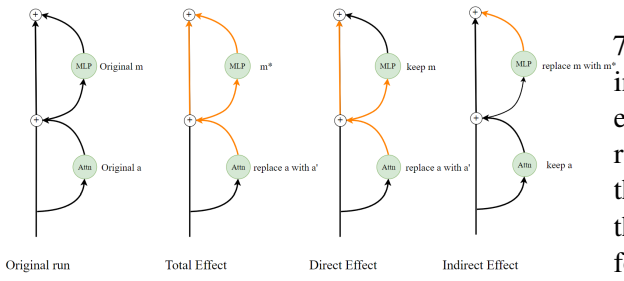


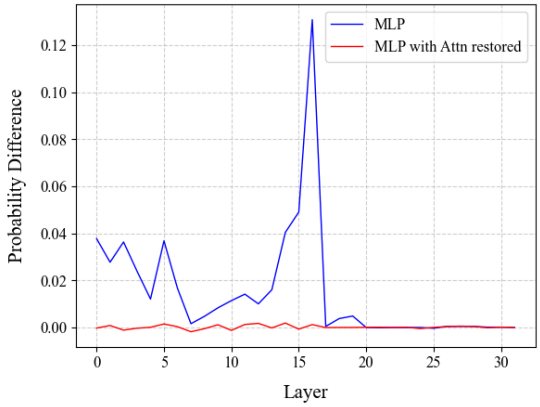
Figure 13: Total Effect, Direct Effect and Indirect Effect

C Appendix - Details in Inference Intervention

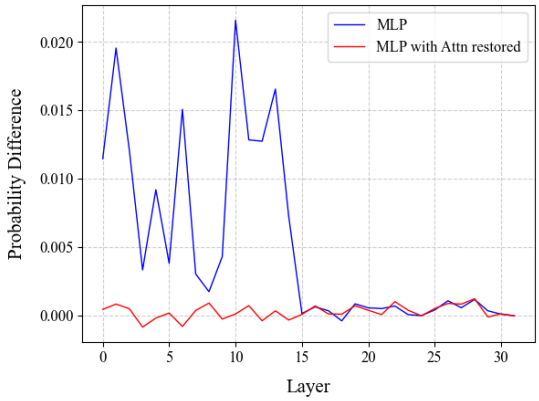
In this section, we give more details about the inference intervention.

The inference intervention results of LLaMA2-7B are shown in Table 6, the length stops at 11 instead of 20 because the inference intervention experiment has to be conducted under a basic accuracy since inference intervention doesn't improve the modular addition performance. Additionally, the input format could massively influence the performance. Specifically speaking, 'X+Y=' and 'X + Y =' are different, while Llama and Mistral work better on the former, Gemma only works well on the later.

We first classify the incorrect answer obtained by the model into four situations. When the model treats the CC question as the OC question, it is called a missing carry; When treating the OC ques-



(a) Mistral-7B



(b) Llama2-7B

Figure 14: The *TE* caused by MLP activation replacement and the effect caused by restoring the top 2 attention heads, the results are averaged among 100 samples.

Table 4: Causal analysis on LLaMA2-7B. The maximum TE and the top-2 attention heads under different input formatting conditions.

Metric/Task	**OC1 and CC1**	**2-digit Prefix**	**2-digit Suffix**	**2-digit Prefix + Suffix**
Maximum TE	0.315	0.313	0.309	0.321
Top-2 Attention Heads (Layer, Head)	(15, 15), (13, 23)	(15, 15), (13, 23)	(15, 15), (13, 23)	(15, 15), (13, 23)
Metric/Task	**OC2 and CC2**	**4-digit Prefix**	**4-digit Suffix**	**4-digit Prefix + Suffix**
Maximum TE	0.036	0.032	0.036	0.034
Top-2 Attention Heads (Layer, Head)	(14, 4), (14, 7)	(14, 4), (14, 7)	(14, 4), (14, 7)	(14, 4), (14, 7)

tion as a CC question, it is called extra carry; alignment errors caused by incorrect numerical alignment; basic modular addition errors.

The model runs on a dataset of randomly generated numbers of a specified length. By classifying each error, it can be found that the most common error made by the model in low sequence lengths is missing carry (See Fig 19). As the length increases, errors caused by alignment account for the vast majority, rather than basic modular additions. Our inference intervention is only optimized for missing carry and extra carry situations.

We give a more detailed explanation of the algorithm for inference intervention (See Algorithm2). The algorithm explains more concretely about the carry detection progress.

The improved accuracy data beyond length of 20 is listed in table 7.

D Appendix - Details in Fine-tuning

We first perform a causal analysis on the fine-tuned Gemma-2B (trained on a maximum length of 10) under different d values (See Fig 21). In the case of length=12, which is beyond the maximum training length, the attention heads did not exhibit significant *TE*, and the attention heads heatmap pattern is different from others in the case of d=1 case, representing a functional differentiation of these attention heads. The training details are listed in Appendix E.

Figure 20 shows the *OC* tasks performance after fine-tuning. Similar to the result of *CC* task (Figure 8a), the accuracy drops sharply after surpassing the maximum training length.

In Section 5.2, we discussed the emergence of a new pattern, the original mechanism remains and is refined. Figure 24 shows the top-2 attention heads overlay pattern located in casual analysis. 23333+36667=60000 is used as a demonstration example, figure 24a shows how the fine-tuned model tries to output the first-digit 6 by gathering attention weights on the following digit(2 and 3), figure 24b shows the model tries to extend one more step to focus on the second digit.

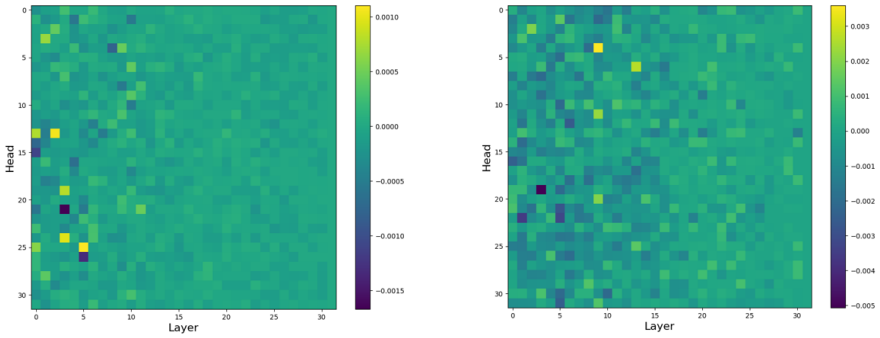
The new strategy is also observed in the fine-tuned Llama2-7B. The detection and target heads are located in layers 7 and 24, respectively (see Figure 22). These heads exhibit similar attention patterns as well (see Figure 23). And the corresponding ablation study results are show in Table

Table 5: Output distribution under zero ablation on tasks CC1, CC4, and CC10

Model / Output Case	Revert to Modular Addition (%)	Error on Modular Addition (%)	Fail to Output Number (%)
Task CC1			
LLaMA2-7B	98.3	1.3	0.4
Mistral-7B	97.8	2.2	0.0
Gemma-7B	99.4	0.6	0.0
Task CC4			
LLaMA2-7B	98.0	0.9	1.1
Mistral-7B	97.3	2.7	0.0
Gemma-7B	99.4	0.6	0.0
Task CC10			
LLaMA2-7B	97.8	1.3	0.9
Mistral-7B	97.7	2.3	0.0
Gemma-7B	99.6	0.4	0.0

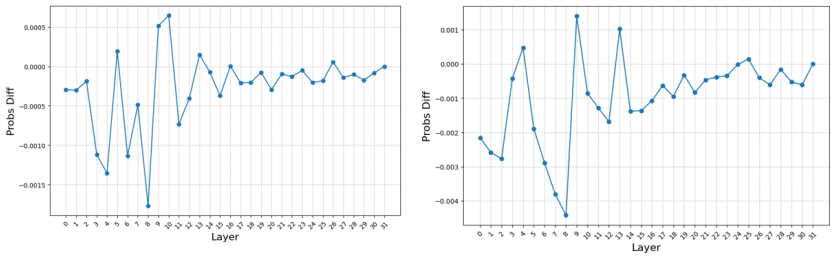
Algorithm 2 Model Inference with Ablation and Reweighting

Input: $X + Y$. L: The last position in the sequence. m: Length of the carry chain. A_m: Top 2 attention heads when $d = m$ in causal analysis.
Output: Z
for $i \leftarrow 0$ **to** $n - 1$ **do**
  Initialize: $m \leftarrow i$
  if $x_{i+1} + y_{i+1} < 9$ **then**
    Set ablation: $a_{L,x_{i+1}}, a_{L,y_{i+1}} \leftarrow 0$
    Perform model inference: $z_i \leftarrow \text{ModelInference}(X, Y, \text{ablation}(a_{L,x_{i+1}}, a_{L,y_{i+1}}))$
    Output: z_i
  else if $x_{i+1} + y_{i+1} > 9$ **then**
    Re-weighting: $a_{L,x_{i+1}}, a_{L,y_{i+1}} \leftarrow A_m$
    Perform model inference: $z_i \leftarrow \text{ModelInference}(X, Y, \text{reweighting}(a_{L,x_{i+1}}, a_{L,y_{i+1}}))$
    Output: z_i
  else
    while $x_{i+1} + y_{i+1} == 9$ **do**
      $m \leftarrow m + 1$
      if $x_m + y_m < 9$ **then**
        Set ablation: $a_{L,x_m}, a_{L,y_m} \leftarrow 0$
        Perform model inference: $z_i \leftarrow \text{ModelInference}(X, Y, \text{ablation}(a_{L,x_m}, a_{L,y_m}))$
        Output: z_i
      else if $x_m + y_m > 9$ **then**
        Re-weighting: $a_{L,x_m}, a_{L,y_m} \leftarrow A_m$
        Perform model inference: $z_i \leftarrow \text{ModelInference}(X, Y, \text{reweighting}(a_{L,x_m}, a_{L,y_m}))$
        Output: z_i
      end if
    end while
    Set ablation: $a_{L,x_{i+1}}, a_{L,y_{i+1}} \leftarrow 0$
    Perform model inference: $z_i \leftarrow \text{ModelInference}(X, Y, \text{ablation}(a_{L,x_m}, a_{L,y_m}))$
    Output: z_i
  end if
  $X + Y \leftarrow X + Y + z_i$
end for



d=1 d=2

(a)



d=1 d=2

(b)

Figure 15: Casual analysis of other components in Llama2-7B. (a) Replacing $\tilde{a}$. (b) Replacing a.

Table 6: Comparison of accuracy between the baseline and inference-time intervention augmentation on LLaMA2-7B. Performance is measured on numbers of varying length (in digits).

Number Length	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Baseline (LLaMA2-7B)	0.955	0.933	0.853	0.693	0.363	0.132	0.094	0.061	0.041	0.008
Augmentation	0.942	0.923	0.883	0.793	0.663	0.432	0.264	0.200	0.155	0.131

8.

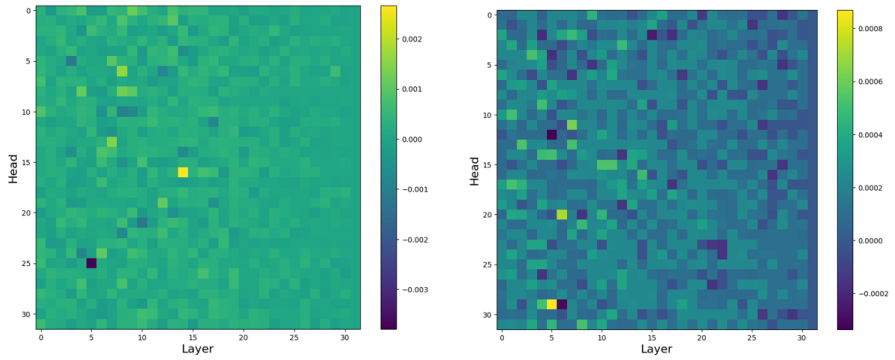
E Appendix - Implementation details

First, we provide a consolidated summary of the models used throughout our experiments, as shown in Table 9.

The prompts in Table 10 are applied in the fine-tuning experiment (randomly sampled), in the experiments related to model inference (Section 3, Section 4), the prompt is fixed to the first prompt shown in the table. The complete format is a prompt plus the question '$X + Y =$' format as input. The temperature is set as 1 and use greedy sampling. The reason for choosing accuracy instead of digit match as the evaluation metric is due to the accumulation of errors during model inference.

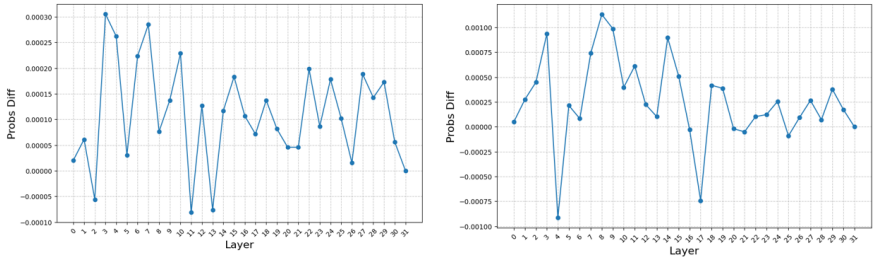
The detailed fine-tuning parameters are listed in table 11. The batch size (varies from 4 to 64) and taken epoch (usually 6-9) varies depending on the specific CCd and OCd tasks. The entire training process is done with one Nvidia A800 GPU, all experiments in the paper could be done within 15 hours.

The dataset includes questions of CCd, OCd, and randomly generated number pairs. To ensure that modular addition does not affect the results, 40% of the dataset consists of randomly sampled numbers with a length upper limit of 80 (sampled between 10 and 10^{79}), 30% consists of *CCd* tasks, and 30% consists of *OCd* tasks. The dataset includes 10^6 samples for each length d, which means $4 * 10^5$ randomly sampled numbers pairs, $3 * 10^5$ CCd samples, and $3 * 10^5$ OCd samples. For the CCd and OCd tasks, we add a number padding of length d in the front and behind. For



d=1 d=2

(a)

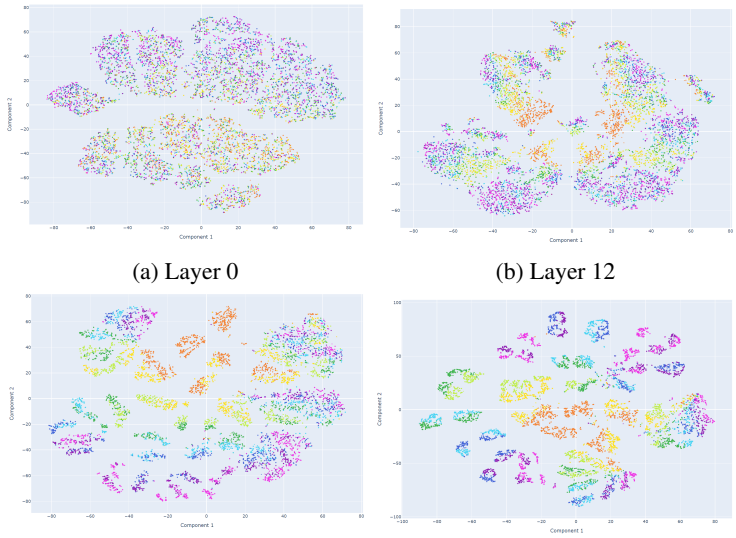


d=1 d=2

(b)

Figure 16: Casual analysis of other components in Mistral-7B. (a) Replacing $\tilde{a}$. (b) Replacing a.

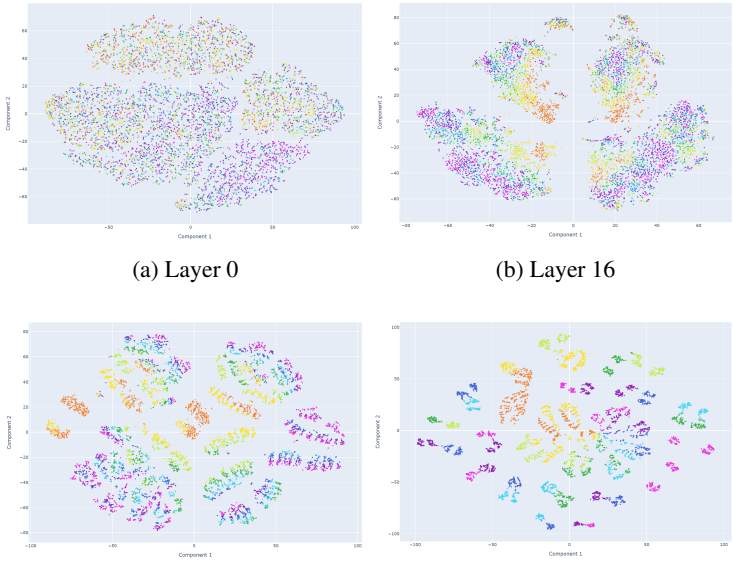
example, a $CC2$ input "234+468=702" could be "22344+34682=57026" in the dataset. The fine-tuning training process has musked the prompt, '$X + Y =$" sequence, and only predicts for Z.



(a) Layer 0 (b) Layer 12

(c) Layer 13 (d) Layer 31

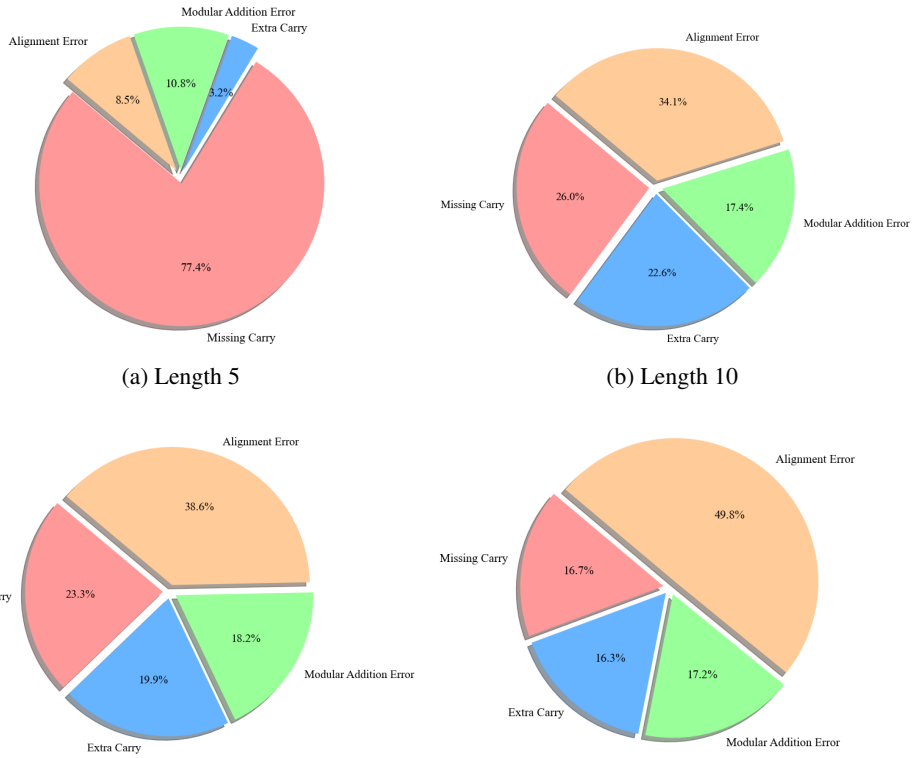
Figure 17: TSNE visualization of the last token hidden state on Llama2-7B, each color represents the label z_1.



(a) Layer 0 (b) Layer 16

(c) Layer 17 (d) Layer 31

Figure 18: TSNE visualization of the last token hidden state on Mistral-7B, each color represents the label z_1.



(a) Length 5 (b) Length 10

(c) Length 15 (d) Length 20

Figure 19: Four types of error that Mistral-7B makes on different lengths.

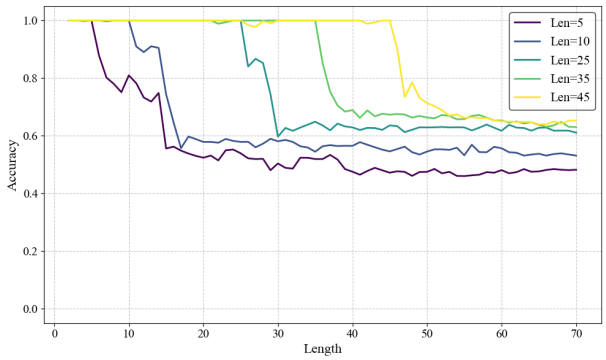


Figure 20: *OC* tasks performance after fine-tuning

Table 7: Further accuracy information about inference intervention on Mistral-7B, Llama-7B, and Gemma-7B.

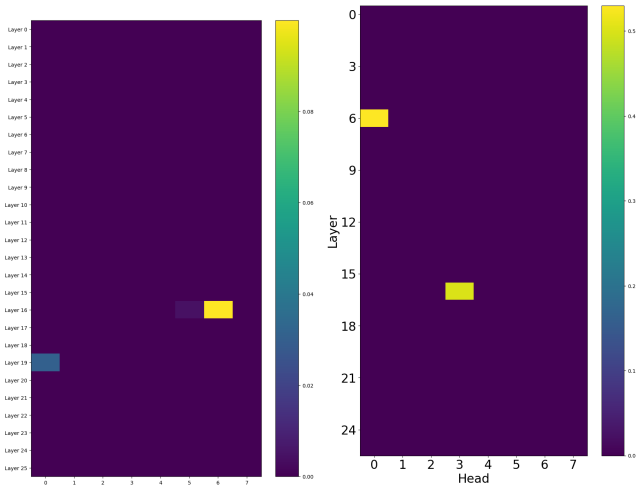
Model	**Method**	Length					
		10	20	30	40	50	60
Mistral-7B	Baseline (%)	48.21	2.98	0.05	0.00	0.00	0.00
	Inference Intervention (%)	72.45	38.21	14.12	4.63	1.12	0.22
Llama-7B	Baseline (%)	0.04	0.00	0.00	0.00	0.00	0.00
	Inference Intervention (%)	3.73	3.51	0.06	0.00	0.00	0.00
Gemma-7B	Baseline (%)	7.73	0.08	0.01	0.00	0.00	0.00
	Inference Intervention (%)	15.45	4.32	1.14	0.42	0.11	0.00

Table 8: Fine-tuned Llama2-7B ablation study (%)

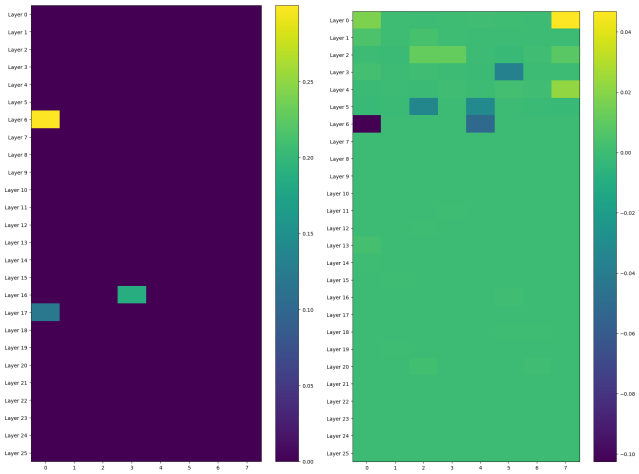
Method / Task	*CC1*	*CC2*	*CC4*	*CC10*
Baseline	100.00	100.00	100.00	97.27
Detection Head Ablation	100.00	100.00	**45.22**	**42.22**
Target Head Ablation	100.00	100.00	**42.58**	**43.27**
Combined Ablation	100.00	99.15	41.51	40.12
Random Ablation	99.91	98.02	99.12	95.20

Section	**Models Used**
3.1	Mistral-7B
3.2	Mistral-7B (main text); LLaMA2-7B, Gemma-7B (Appendix A)
3.3	Mistral-7B, LLaMA2-7B, Gemma-7B (main text)
4	Mistral-7B (main text); LLaMA2-7B, Gemma-7B (Appendix C)
5.1	Gemma2-2B (main text)
5.2	Gemma2-2B (main text); LLaMA2-7B (Appendix D)

Table 9: Summary of model usage across experiments.



(a) d = 1 (b) d = 3



(c) d = 5 (d) d = 12

Figure 21: The attention heads located through causal analysis show that for fine-tuned Gemma2-2B under different values of d, $d = 12$ analysis is based on OOD input.

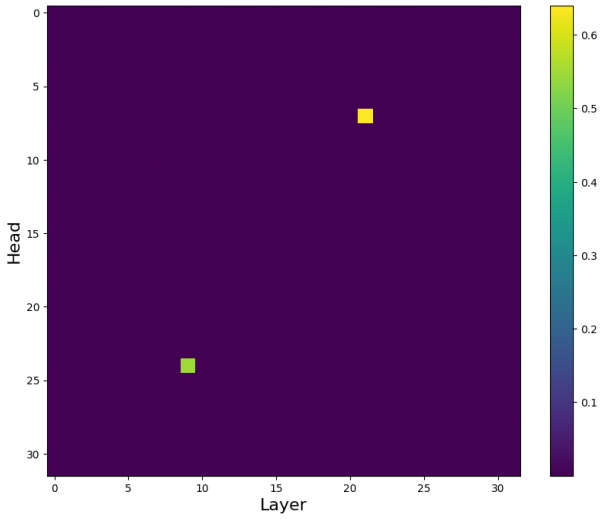
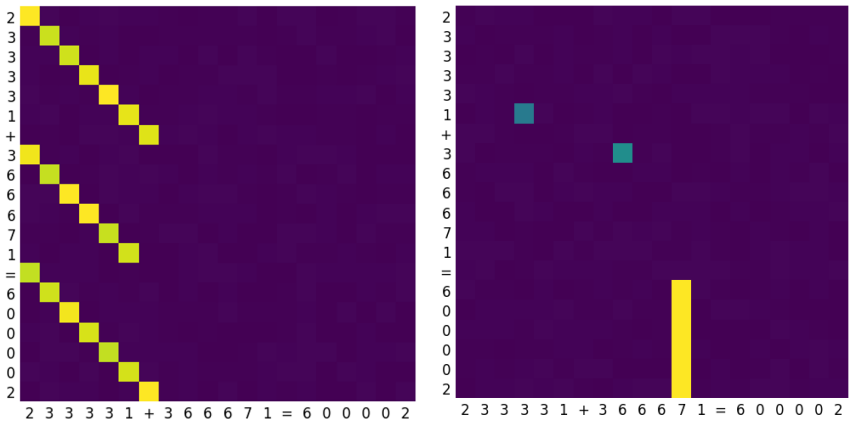
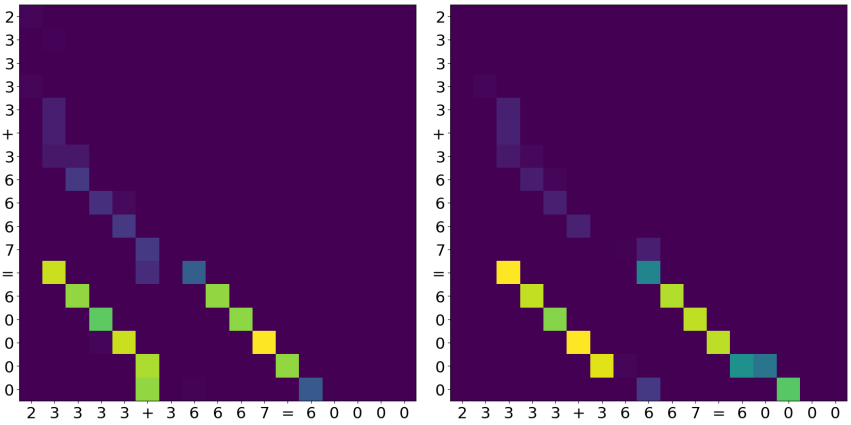


Figure 22: Causal Analysis on fine-tuned Llama2-7B.



(a) Detection head (b) Target head

Figure 23: The result of superimposing the attention heads on fine-tuned LLama2-2B, 233331+366671=600002 is used as a demonstration example. For simplicity, the attention pattern displayed omits the prompt and only retains the question.



(a) d=1 (b) d=2

Figure 24: The result of superimposing the attention heads on fine-tuned Gemma2-2B, 23333+36667=60000 is used as a demonstration example. For simplicity, the attention pattern displayed omits the prompt and only retains the question. For d from 1 to 2, the selected top 2 attention heads (Head, Layer) are (16, 6) and (19, 0); (16, 6) and (19, 9).

Table 10: Examples of Prompts

Prompt Examples
Do math calculations:
Calculate:
Compute the following sum:
Solve the addition:
Calculate the result of:
Solve the following problem:
Perform the calculation:
Determine the result of:
Find the value of:
Complete the calculation:
What is the solution to:
Solve this equation:
Compute the answer for:
What is the sum of:
Figure out the result of:
Determine the answer to:
Find the solution to:
Perform the operation:

Table 11: Key Training Arguments Configuration, d represents the CCd and OCd task.

Parameter	**Value**
num_train_epochs	15
number of training tokens	about $(6d+1) * 10^6$
learning_rate	5e-5
bf16	True
weight_decay	0.0
adam_beta1	0.9
adam_beta2	0.999
adam_epsilon	1e-08
gradient_accumulation_steps	1
seed	42
lr_scheduler_type	linear
optim	adamw_torch