Is Transformer a Stochastic Parrot? A Case Study in Simple Arithmetic Task

Peixu Wang [1] **Yu Chen** [2] **Ming Yu** [3]

Abstract

Large pretrained language models have demonstrated impressive capabilities, but there is still much to learn about how they operate mechanically. In this study, we conduct a multi-faceted investigation of the autoregressive transformer's ability to perform basic addition operations. Specifically, we use casual tracing to locate the information flow between attention and the fully-connected layer. For attention layers, we found that they exploit fixed patterns in the intermediate stage to perform the transfer of carry and numeric information. They project the input onto the distribution of a few neurons in later fully-connected layers, where the neurons activate the vocabulary distribution existing in the parameter space to implement the mapping relationship. In addition, our research can be further extended to the study of interpretability of general classification tasks like sentiment analysis. The findings suggest that, although the model appears to have learned some arithmetic rules, most of its reasoning still relies on statistical patterns.

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).

Current research has different views regarding the nature of the black box of the model. On the one hand, some studies believe that the model is just a model that relies on statistics (Bender & 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 & Pavlick, 2021).

Mathematics have solid rules and logic to follow compared with other linguistic tasks, however, the performance of the models in mathematics, especially arithmetic tasks, is often unsatisfactory (Brown et al., 2020). We believe that the reason behind this phenomenon could shed light on the nature of the model. To look into the mechanics of transformers, this paper studies how the current mainstream pre-trained language models complete simple addition tasks.

Recent interpretability studies of language models aim to identify circuits and components that are interpretable to humans(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 Multi-layer Perceptron (MLP) components, seek to locate the sub-graph responsible for the actual task computation within the entire computational graph. The goal of these methods is to identify the key components of the model that contribute to its performance on specific tasks.

In this paper, we focus on the current mainstream pre-training models, such as LLaMA (Touvron et al., 2023), Mistral (Jiang et al., 2023), and Qwen (Bai et al., 2023), and investigate their behavior on a simple addition task (e.g., 1482+2309=). Despite differences in architecture and training data, these models exhibit similar mechanisms. We employ causal tracing (Pearl, 2022; Meng et al., 2022; Vig et al., 2020) to locate the attention and MLP components critical to the model's reasoning process. By dividing the component influence into direct and indirect effects, we further narrow our research focus to the later MLP component.

To under the whole process of arithmetic, besides continuing to explore related model components, we also investigate on MLP neuron activation and parameters. By treating the MLP internal structure as Key-value pairs (Geva et al., 2020) and projecting the MLP parameters into the model's vocabulary space, we obtain human-understandable concepts related to the task.

We demonstrate that this framework can be extended to general classification tasks by verifying its applicability on a sentiment analysis task. Finally, we delve into the question of whether the model truly understands the mechanism of addition by examining its learning of the commutative law

[1]National University of Singapore [2]Independent Researcher [3]Power Automation. Correspondence to: Peixu Wang <e1143292@u.nus.edu>, Yu Chen <c_y@u.nus.edu>, Ming Yu <ALUM06YM0001@e.ntu.edu.sg>.

and its ability to perform carry calculations.

2. Causal tracing

2.1. Background

Consider two n-digit integers $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ and their addition result $\mathbf{z} = (z_1, z_2, \ldots, z_n)$. The numbers are tokenized into the sequence numbers of the vocabulary, mapping to the embedding d_i, and the hidden state of ith token in the first layer is $h_i^0 = d_i + pos(i)$, Recall that, in the autoregressive case, tokens only derive information from past (above) tokens:

$$h_i^{(l)} = h_i^{(l-1)} + a_i^{(l)} + m_i^{(l)} \quad (1)$$

$$a_i^{(l)} = \text{attn}^{(l)}\left(h_1^{(l-1)}, h_2^{(l-1)}, \ldots, h_i^{(l-1)}\right) \quad (2)$$

$$m_i^{(l)} = W_{down}^{(l)}\sigma\left(W_{up}^{(l)}\gamma\left(a_i^{(l)} + h_i^{(l-1)}\right)\right) \quad (3)$$

We adopt a clear and practical perspective on the hidden state h. The attention and MLP components of the model perform read and write operations: they read information from h, process it, and then update the residual stream (Elhage et al., 2021).

Our primary focus is on models that segment continuous numbers into individual digits and encode them ($x_i \to d_i$). Although our study is applicable to models with discontinuous encoding, such as LLaMA3 and GPT-2, these models introduce additional uncertainty to the research.

2.2. Activation intervention

Causal analysis (Pearl, 2022; Vig et al., 2020; Meng et al., 2022) is used to find the component for recovering output. Consider $\mathbf{x}+\mathbf{y}$, the autoregressive language model needs to output results in sequence, $z_1 \to z_2 \to \ldots$ We first consider the calculation without carry, that is, $x_i + y_i = z_i$. In this case, the model needs to master the simplest mapping ($x_i + y_i \to z_i$) without requiring other digital bit information. In causal analysis , we first create a set of two different inputs $(\mathbf{x} + \mathbf{y})$, $(\mathbf{x}' + \mathbf{y}')$, where $x_1 \neq x_1'$, $z_1 \neq z_1'$. We conduct three rounds of model inference.

- **In the first run:** $(\mathbf{x} + \mathbf{y})$ as the input to obtain the final probability output $p(z|x, y)$ and $p(z'|x, y)$ and collect all the activation $o \in \{m_1^{(1)}, \ldots, m_t^{(L)}, a_1^{(1)}, \ldots, a_t^{(L)}\}$.
- **In the second run:** $(\mathbf{x}' + \mathbf{y}')$ as the input and collect all the activation $o' \in \{m_1'^{(1)}, \ldots, m_t'^{(L)}, a_1'^{(1)}, \ldots, a_t'^{(L)}\}$.
- **In the third run:** $(\mathbf{x} + \mathbf{y})$ as the input and replace the activation o with o' to obtain the probability output $p^*(z'|x, y)$.

When intervening in reasoning, we sequentially use z' to override the original activation z to change the model's probability output. Intuitively, this should lead to an increase in the model's output probability for z'. The total effect is defined as 4. We fixed the length of the numbers to the addition of 4-digit numbers, as the model exhibits higher accuracy at this length. Increasing the length leads to decreased accuracy, which could introduce unnecessary variability into the experiment. We used 100 sets of numbers as input for LLaMA2-7B and calculated their average total effect (TE).

$$\text{Total Effect (TE)} = p^*(z'|x, y) - p(z'|x, y) \quad (4)$$

We found that for the attention layers, the impact is primarily concentrated in the middle layers (14-17) and is focused on the last token position. In contrast, the MLP's impact is concentrated in the later layers and is also focused on the last token. These results are similar to those found by (Stolfo et al., 2023), who also studied arithmetic reasoning. However, our research aims to uncover the underlying mechanisms by which the model performs addition tasks, rather than merely examining the information flow.

2.3. Effect breakdown

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 Figure 2). To distinguish the degree of influence between the two, we set up an additional experimental process.

To calculate the DE, when we perform activation intervention on $m_1'^{l}$ we fix all the downstream components as their original activation $m_i^{l+1}, m_i^{l+2} \ldots, a_i^{l+1}, a_i^{l+2} \ldots$. To calculate IE, we first perform an activation intervention and collect the activation values of downstream components $m_i^{*l+1}, m_i^{*l+2} \ldots, a_i^{*l+1}, a_i^{*l+2} \ldots$. Then, we perform model inference and only cover the downstream components with the collected activation values.

As shown in Figure 3. We focus on the components with significant TE, which are the attention layers in the middle of the model and the MLP layers in the later stage, and calculate IE and DE. We found that intervention in the attention layer almost entirely depends on the impact on subsequent components to intervene in the model output, while for the MLP layer, the situation is the opposite. They directly in-

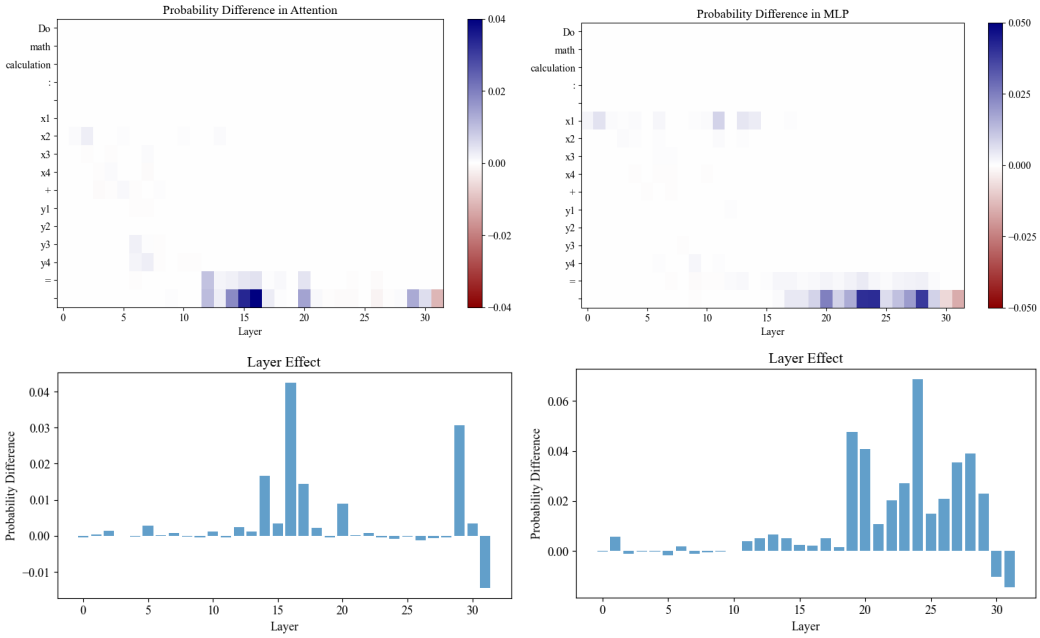


Figure 1: **Upper Left, Upper Right:** present the results of activation replacement by tokens in attention and MLP layers respectively. **Lower Left, Lower Right:** the outcomes of replacing all tokens in attention and MLP layers respectively, indicating the layer-level interventions.

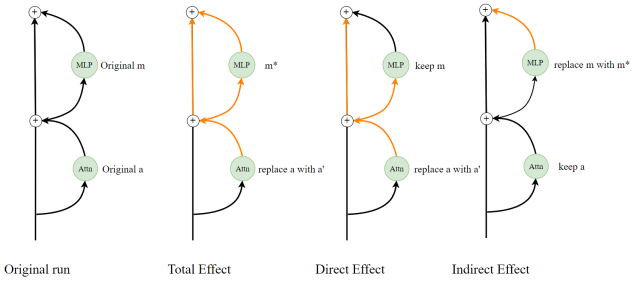


Figure 2: Total Effect, Direct Effect and Indirect Effect

tervene in the model output by updating the residual stream, and even compensate for negative IE values.

Since our goal is to identify the key components, we will continue tracing the indirect effect caused by the attention layer. This time, instead of analyzing different components in different layers, we adopt a more holistic approach. We classify the overall downstream components into two types: attention and MLP. To achieve this, we calculate the indirect effect by isolating the activation of each type of component $(m_i^{*l+1}, m_i^{*l+2} \ldots)$ or $(a_i^{*l+1}, a_i^{*l+2} \ldots)$, allowing us to determine the indirect effect specific to each type.

The results of the further breakdown of the indirect effect (IE) are shown in Figure 4. It is evident that the IE caused by the attention and MLP components in the middle layers is dominated by the MLP effect. This indicates that the components in the middle layers influence the model output primarily by indirectly affecting the downstream MLP components. These downstream MLP components, as we discovered, directly update the residual stream.

In conclusion, we found that the attention mechanisms primarily operate in the middle layers, while the MLPs exert their influence mainly in the later layers. The components in the middle layers, including both attention and MLP, significantly affect the downstream MLPs, causing an indirect effect. Conversely, the MLPs in the later layers directly modify the residual stream, impacting the final probability. Our research consistently highlights the crucial role of the later MLP layers.

3. MLP implements mapping relationship

Recent research regards MLPs as key-value memory structures (Geva et al., 2020; Meng et al., 2022), where the model stores knowledge in the corresponding relationships within the MLP. By adjusting the MLP parameters, we can modify the facts stored in the model. In this section, we examine how the model implements the mapping $x_i + y_i \rightarrow z_i$ through the MLP, focusing specifically on the MLPs in the later layers.

The feed-forward network has the following form(bias ig-

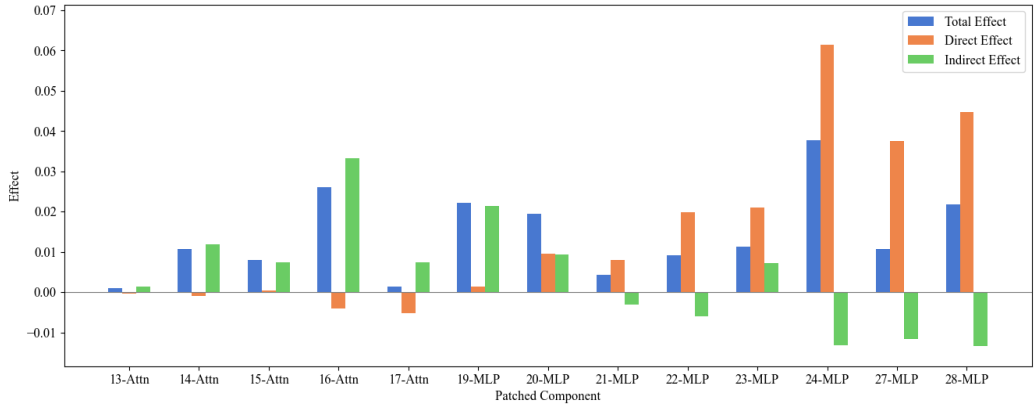


Figure 3: Total Effect breakdown, the effect caused by attention entirely depends on the indirect effect. As a comparison, the effect caused by MLP mostly relies on directly updating the residual stream

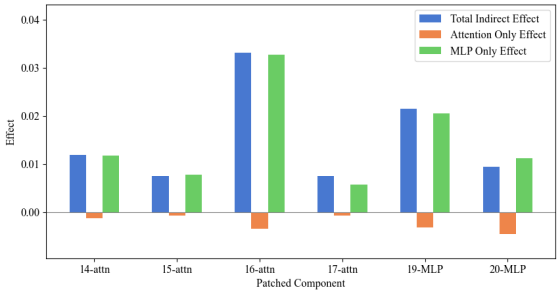


Figure 4: The breakdown of Indirect Effect

nored)

$$\text{FFN}(x) = f(xW_{up})W_{down} \tag{5}$$

where $x \in \mathbb{R}^{d_{\text{model}}}$ is a vector which has encoded the input text prefix, $W_{up} \in \mathbb{R}^{d_{\text{model}} \times d_{\text{mlp}}}$ and $W_{down} \in \mathbb{R}^{d_{\text{mlp}} \times d_{\text{model}}}$ are the parameter matrices, f is nonlinear activation function.

We take the Geva's (Geva et al., 2022) view of key-value in MLP, $f(xW_{up})$ represents the key k ($k \in \mathbb{R}^{d_{\text{mlp}}}$) which is a distribution captures the patterns of the input, while the value is a row vector v ($v \in \mathbb{R}^{d_{\text{model}}}$) of the W_{down} which represents the target distribution. By taking this view, the FFN network can be seen as projecting the hidden state h_i to k_i through the W_{up} matrix, and k is used as a weight to sum v in the W_{down} matrix and add the result to the residual stream. Since the results of the FFN are linearly added to the residual stream, we have the following expression

$$h_i^l = h_i^{l-1} + kW_{down} = h_i^{l-1} + \sum_{j=1}^{d_{mlp}} a_j v_j \tag{6}$$

$$p = softmax(h_i^{final} E_u) \tag{7}$$

where k is vector $(a_1, a_2 \dots a_{dmlp}) \in \mathbb{R}^{d_{\text{mlp}}}$. By observing 6, the increasing the value of a_j will cause the hidden state h_l to move closer to v_j. When a_j approaches infinity, $softmax(h_i^{final} E_u) = softmax(v_j E_u)$.

3.1. Finding value in vocabulary space

One of the benefits of viewing an MLP as a key-value pair is that the value can be analyzed separately. The value is explicitly stored within the model as internal knowledge. This knowledge is encoded into the model's parameters during training and is independent of the input.

In order to find the meaning of v, we uses the unembedding matrix $E_u \in \mathbb{R}^{d_{\text{model}} \times V}$ (as Geva did in (Geva et al., 2022)) to project v into the vocabulary space (8). Related research includes logits-lens (Nostalgebraist, 2020) and other improved versions (Belrose et al., 2023; Pal et al., 2023), which projects the hidden state h_i into the vocabulary space. This approach is highly effective for achieving interpretability.

$$w = softmax(vE_u) \tag{8}$$

where $w \in \mathbb{R}^V$ is a probability distribution of the vocabulary. We hypothesize that the model relies on the key-value pairs in MLP to implement the mapping relationship of $x_i + y_i \rightarrow z_i$. This is achieved by mapping the numbers x, y and operator $+$ to a specific key to activate the target value distribution. In other words, the key represents $x + y$, and the value represents z. v is a model parameter, which allows us to obtain the distribution represented by v before the

model receives input. Our goal is to identify the v values that encode single digits from 0 to 9.

Specifically, for the probability distribution w of the vocabulary space generated by each v, we truncate the top k words with the highest probability to represent the main meaning conveyed by v. For the top k words, if they contain more than r word that is related to number z, then we think that v has encoded the number z. The definition of related words can be found in Appendix C.

We explained total 32*11008 of v in LLaMA2-7b (with k=10, r=3). As shown in table 1, from digit 1 to 9, the quantity of v generally shows a decreasing trend (See Appendix C). Interestingly, the phenomenon follows the Benford's law. This may be because 1 and 2 account for the largest proportion of the numbers seen by the model during training. In addition, when applying TSNE (Van der Maaten & Hinton, 2008) visualization on the hidden state, the data that should output 1 or 2 will form clusters earlier than other data.

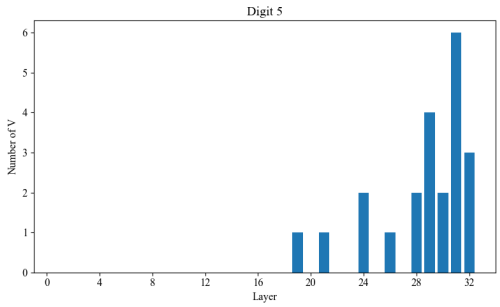


Figure 5: the number of v that encoded the concept of digit 5 in each layer. Most v appear in the later layers of the model, which is consistent with previous research (See Figure 4)).

3.2. Neurons activation

The advantage of locating these v is that we can, in turn, focus on the specific neuron activation (key) corresponding to these v. In this paper, we leverage the one-to-one correspondence between activation k and v to quickly locate specific neurons in a particular layer. Recent research on neurons is mainly troubled by the large number of neurons and their ambiguity (Olah et al., 2020; Arora et al., 2018). Neurons are activated simultaneously in different contexts, which makes it difficult to focus on specific locations. Related research uses dictionary learning (Yun et al., 2021; Zhang et al., 2019) or sparse models (Cunningham et al., 2023) to solve the problem of ambiguity and automatically interpret neurons with a much more powerful model (Bills et al.).

If the model relies on the key-value mechanism of MLP to achieve a mapping relationship of $x_i + y_i \rightarrow z_i$, then the neurons a corresponding to the v vector encoding the number z should be activated.

We use n-digit plus n-digit inputs to observe the activation of neurons a_i^l. Specifically, when the autoregressive model is required to output the digit z as the first digit,, then we collect all the activations a_i^j corresponding to z. We use the sum of activation values and average them on a dataset of 200 data points. Finally, we conduct experiments on different digit lengths from 2 to 25 to observe the universality of the conclusion. On one hand, we record the activation values of neurons $\sum_{j=1}^{total} abs(a_j)$, and on the other hand, we record the output probability $p(z)$ of the model for the answer. As a comparison, we selected random neurons as the baseline and neurons a_i^l related to v_i^l which also has encoded other digit z', $(z' \neq z)$.

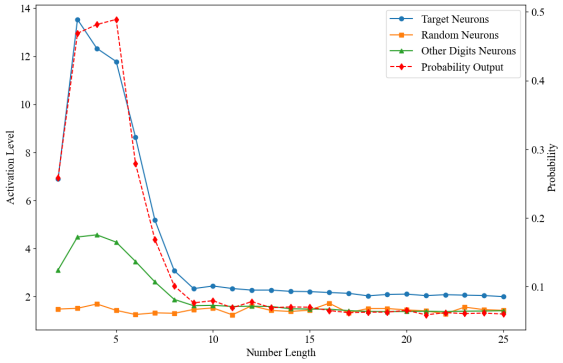


Figure 6: The correlation between neurons activation and probability output $p(z)$ on LLaMA2-7B, results of more models refer to Appendix D.

The activation and probability output $p(z)$ of the target neuron activation showed the same trend of change (See Figure 6). This indicates that the neuron activation corresponding to result z does indeed reflect the output probability of the final output. It should be noted that the proportion of neurons we studied in the entire model is very small, usually less than 50, but they still demonstrate a strong correlation with the results. As a comparison, the activation of random neurons showed lower levels of activation. For neurons corresponding to the v encoding other numbers z', $(z' \neq z)$, they had activation values higher than random but lower than the target neurons.

3.3. Task expansion

The arithmetic task can be seen as a continuous ten-class classification problem. To expand the research scope, consider a sentiment binary classification task, where the model will receive a segment of text to determine whether it is positive or negative. As before, we first searched for v encoding positive and negative semantics, as we did not need

Table 1: Some representative v that encode digit in LLaMA2-7B.

Position of v	Encoded digit	Tokens with top probability
$\mathbf{v}_{5643}^{31}$	9	nine, 9, nine (In Chinese), September, IX, Sep, Sept, september, nin, III
$\mathbf{v}_{5643}^{29}$	8	8, eight, eight(In Chinese), VIII, _acht, huit, eig, otto
$\mathbf{v}_{4958}^{24}$	3	third, Third, III, troisième, III, 3, three, _thirty
$\mathbf{v}_{10820}^{26}$	2	2, two, twenty, ², two(In Chinese), II, ²), II

to consider the issue of discontinuous encoding of numbers. Therefore, our experimental subjects also included LLaMA3,

Similarly, for the samples with label y, $y \in \{\text{positive}, \text{negative}\}$, we take the a_y as the target neurons which we intervene on, $a_{y'}$ as the opposite label neurons. As discussed earlier, the model can increase the activation a_y to assign larger weights to v_y, or it can decrease the activation $a_{y'}$ to assign negative weights to $v_{y'}$ to indirectly increase the probability $p(y)$.

We conduct our experiment on the sst-2 dataset. We define four types of neuron activations, target positive neurons activation: $\sum_{j=1}^{total} a_j^y (a_j^y > 0)$, target negative neurons activation: $\sum_{j=1}^{total} a_j^y (a_j^y < 0)$, opposite positive neurons activation: $\sum_{j=1}^{total} a_j^{y'} (a_j^{y'} > 0)$, opposite negative neurons activation: $\sum_{j=1}^{total} a_j^{y'} (a_j^{y'} < 0)$. As shown in Figure 7, surprisingly, except for Mistral, the rest of the models indirectly increase the probability $p(y)$ by assigning negative weight to the wrong $v_{y'}$. Correspondingly, a^y has consistently maintained a relatively low positive activation value which is counterintuitive.

4. Stochastic parrot

We have studied how the model achieves mapping from $x + y$ to z. However, implementing this mapping relationship is an easy task for today's models. For the number n, the model only needs to remember all n+1 combinations from (0+n) to (n+0) to achieve an operation (without carry and borrow). The key to the problem lies in whether the model learns knowledge beyond simple mappings, such as commutative laws, carry, and number alignment operations. In this section, the discussion will focus on whether the model has learned commutative laws and carry mechanisms, which are higher than general mapping relationships

4.1. Commutative law

The model does not need to know $a+b = b+a$. It only needs to remember two different mapping rules $a+b = z, b+a = z$. However, we still investigate from the perspective of neurons. $a+b$ and $b+a$ are essentially the same in arithmetic, but they are in different states when they enter the model through the embedding layer because of positional encoding. We believe that if the model ultimately maps these two different inputs to the same neuron activation distribution, it can be said that it has learned the commutative law. While it's not practical to directly obtain the activation distribution of all neurons, the model will capture unrelated features of other concepts, resulting in inconsistent final distributions.

We also used the neurons located in section 3 for analysis, which are strongly correlated with the task and are capable of filtering out irrelevant activation caused by other input features. Specifically, considering a target output number $\mathbf{z} = (z_1, z_2, \ldots, z_n)$, we created a dataset containing $\mathbf{x} + \mathbf{y}$ (without carry), where number starts form $(x_1 = 1, y_1 = z_1 - 1)$ to $(x_1 = z_1 - 1, y_1 = 1)$. For each number pair, we collected their neuron activations, and each pair consisted of twenty sets for averaging. We calculated the cosine similarity matrix $N \in \mathbb{R}^{z_1-1 \times z_1-1}$.

It can be observed that when $z_1 = 9$, the similarity matrix presents an X-shaped pattern(See Figure 9), which means the model combines $a + b$ and = $b + a$ well. However, when $z_1 = 8$, their similarity is not so obvious, some less relevant inputs (5-3 and 1-7) have a rather higher similarity.

Overall, the model seems to have learned some knowledge about commutative laws, but it is not obvious in some cases and further confirmation is needed in the future.

4.2. Carry propagation

Looking back at the causal analysis experiment (See section 2), we found that the attention layer played a dominant role in the mid-term. Attention naturally has interpretability, we visualize attention heads(See Appendix E) with distinct features and notice that attention presents attention to computational digit x_1, y_1, and x_2, y_2. This makes sense, models need to calculate the first digit, so x_1 and y_1 are the calculation digits that should be considered, while x_2 and y_2 are the basis for judging whether carry will occur. When we melt the attention head's focus on x_2 and y_2 with zero ablation, we find that the model cancels carry or borrow in the calculation equations that should have a carry or

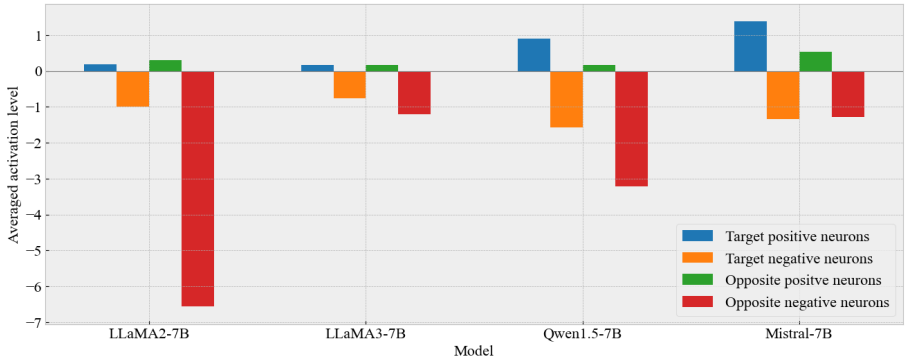
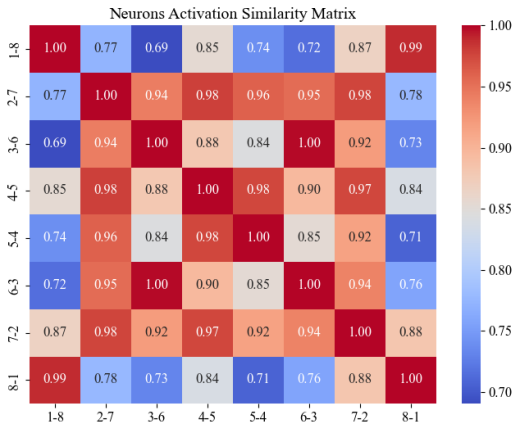
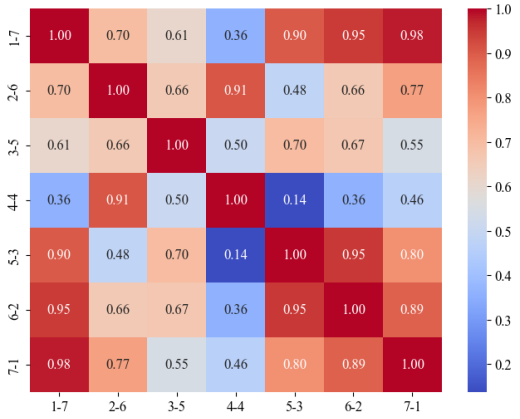


Figure 7: Four types of averaged activation representing how the model chooses the way to output the probability,



A case of $z_1 = 9$



A case of $z_1 = 8$

Figure 8: The cosine similarity of neuronal activation, where each pair of numbers $a + b$ represents the first digit of two numbers, is averaged over 100 sets of data for each activation. This figure shows cases for digits 9 and 8. More result see Appendix E.

borrow(See Appendix E).

Attention naturally has interpretability, and we visualize attention heads with distinct features, noticing that attention presents an effect on the computational digit x_1, y_1, and x_2, y_2's attention. This makes sense. Unlike humans, the model needs to first calculate the result, so x_1, y_1 is the computational bit that should be considered, while x_2 and y_2 is used as a basis to determine whether carry will occur. When we perform a zero value ablation of the attention head's focus on x_2 and y_2, we find that the model cancels carry in or borrow from the original formula(See Figure), such as 140+180 $\rightarrow$ 220. This phenomenon not only exists in explicit addition formulas but also in arithmetic application problems. If the carry occurs at a later position $x_{carry}, y_{carry}, carry > 2$, the zero ablation will still stop the carry.

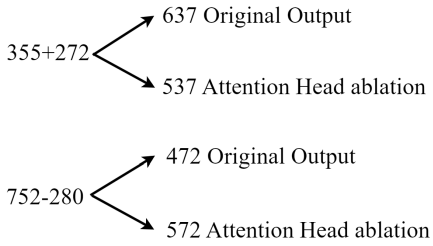


Figure 9: Ablation on the position of x_{carry}. For the case in the figure, $carry = 2$.

We examine whether the model has learned the carry mechanism from the perspective of carry propagation. Consider an addition equation $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ that generates carry when $x_2 + y_2 > 9$, and does not generate carry when $x_2 + y_2 < 9$. When $x_2 + y_2 = 9$, the model needs to focus on x_3 and y_3 to continue determining whether $x_3 + y_3$ will generate carry,

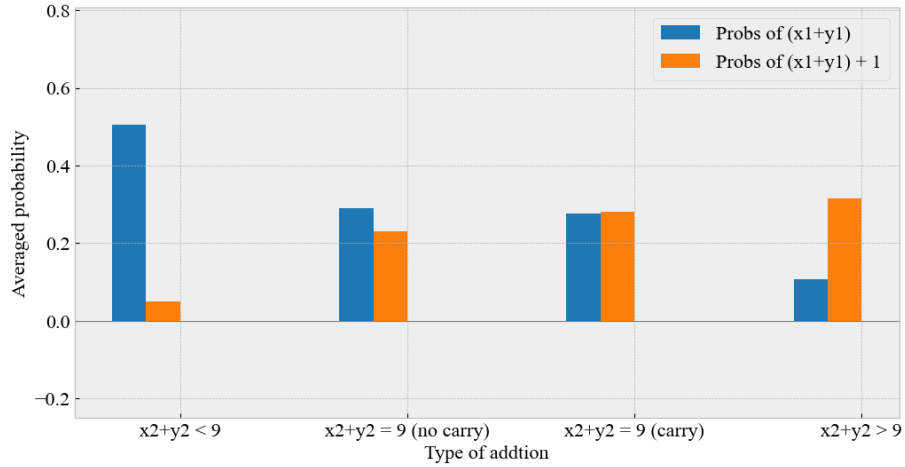


Figure 10: Averaged probability representing four types of addition. When $x_2 + y_2 = 9$, it is difficult for the model to determine whether there is a carry, and the output probabilities of both results are at a high level.

the carry propagation detection stops until it is determined whether carry exists.

We found the model only uses a fixed attention weight in the attention head to weigh the computational digit and the digit after it. This is a clear statistical pattern, as detection of later digits is only required when $x_i + y_i = 9$, which is relatively rare in the training data.

When $x_i + y_i = 9$, the model simply just increases the probability of carrying (See Figure 10), rather than truly propagating the carry mechanism. This is also based on statistical patterns: "When $x_i + y_i = 9$, carry is likely to occur.". Until now, the ChatGPT4 has not been able to escape this phenomenon. Given the input 'answer directly: 652734181+247265817=', the output is 900000000 (answer: 899999998), the first digit of the model output is 9, which causes errors to accumulate and makes subsequent output become 0.

5. Conclusion

We investigated how the model performs addition tasks. By using causal tracing, we investigated the information flow path in the model and located it on the attention as the carry operator and the MLP layer as the execution mapping relationship. Our research found that the model achieves mapping relationships to implement addition by activating the parameter and prompting a certain vocabulary distribution in later MLP, verifying the hypothesis that MLP is a memory network. The framework can be extended to general mapping relationships, especially for classification tasks. We also explain from the perspectives of carry and commutation laws whether the model truly learns addition rules, providing insights for the debate on the nature of the model.

Ethics Statement

Our research on the model has enhanced its interpretability and transparency. We can judge the decision results of the model from within, which is of great help in managing increasingly powerful models in the future.

The main concern for the article comes from its description of mapping relationships. Like other studies on memory networks, the mapping relationships of the model's memory can be modified, leading to the model being manipulated to output harmful content. Therefore, strict control is required for the modification of memory networks, and content should not be redirected to other risky content at any time.

References

Abdou, M., Kulmizev, A., Hershcovich, D., Frank, S., Pavlick, E., and Søgaard, A. Can language models encode perceptual structure without grounding? a case study in color. *arXiv preprint arXiv:2109.06129*, 2021.

Arora, S., Li, Y., Liang, Y., Ma, T., and Risteski, A. Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association for Computational Linguistics*, 6:483–495, 2018.

Bai, J., Bai, S., Chu, Y., Cui, Z., Dang, K., Deng, X., Fan, Y., Ge, W., Han, Y., Huang, F., et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Belrose, N., Furman, Z., Smith, L., Halawi, D., Ostrovsky, I., McKinney, L., Biderman, S., and Steinhardt, J. Eliciting

latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023.

Bender, E. M. and Koller, A. 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*, pp. 5185–5198, 2020.

Bills, S., Cammarata, N., Mossing, D., Tillman, H., Gao, L., Goh, G., Sutskever, I., Leike, J., Wu, J., and Saunders, W. Language models can explain neurons in language models. URL https://openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

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

Cunningham, H., Ewart, A., Riggs, L., Huben, R., and Sharkey, L. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*, 2023.

Dauphin, Y. N., Fan, A., Auli, M., and Grangier, D. Language modeling with gated convolutional networks. In *International conference on machine learning*, pp. 933–941. PMLR, 2017.

Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1:1, 2021.

Geiger, A., Lu, H., Icard, T., and Potts, C. Causal abstractions of neural networks. *Advances in Neural Information Processing Systems*, 34:9574–9586, 2021.

Geva, M., Schuster, R., Berant, J., and Levy, O. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*, 2020.

Geva, M., Caciularu, A., Wang, K. R., and Goldberg, Y. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. *arXiv preprint arXiv:2203.14680*, 2022.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.

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

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

Olah, C., Cammarata, N., Schubert, L., Goh, G., Petrov, M., and Carter, S. Zoom in: An introduction to circuits. *Distill*, 5(3):e00024–001, 2020.

Pal, K., Sun, J., Yuan, A., Wallace, B. C., and Bau, D. Future lens: Anticipating subsequent tokens from a single hidden state. *arXiv preprint arXiv:2311.04897*, 2023.

Patel, R. and Pavlick, E. Mapping language models to grounded conceptual spaces. In *International conference on learning representations*, 2021.

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

Stolfo, A., Belinkov, Y., and Sachan, M. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7035–7052, 2023.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Vig, J., Gehrmann, S., Belinkov, Y., Qian, S., Nevo, D., Singer, Y., and Shieber, S. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33: 12388–12401, 2020.

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. *arXiv preprint arXiv:2211.00593*, 2022.

Weidinger, L., Mellor, J., Rauh, M., Griffin, C., Uesato, J., Huang, P.-S., Cheng, M., Glaese, M., Balle, B., Kasirzadeh, A., et al. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*, 2021.

Yun, Z., Chen, Y., Olshausen, B. A., and LeCun, Y. Transformer visualization via dictionary learning: contextualized embedding as a linear superposition of transformer factors. *arXiv preprint arXiv:2103.15949*, 2021.

Zhang, J., Chen, Y., Cheung, B., and Olshausen, B. A. Word embedding visualization via dictionary learning. *arXiv preprint arXiv:1910.03833*, 2019.

A. Limitations

We have conducted research on how pre-trained language models complete addition tasks, and our research has profoundly revealed the basic principles of using attention heads to transmit numerical and carry information, implementing basic mapping relationships using MLP, and discovering the possibility of the model learning addition rules. We believe that this study takes a firm step toward interpretability research.

However, the characteristic of language models lies in their comprehensiveness. The mechanism by which the model specifically locates each digit that needs to participate in the operation is not yet understood. How changes in attention affect the distribution of MLP activation in the later stages has not been studied. Why does the model assign negative activation to the answer, which may decrease the probability.

Another crucial point is that our research builds on the findings of previous researchers on memory networks, which still have many unexplained aspects. For example, a significant portion of model parameters cannot be accounted for using vocabulary projection, highlighting the need for further investigation into these unexplained parameters.

B. Additional information for casual tracing

In this section, we discuss casual tracing for other models and some interactions we observed between attention and MLP.

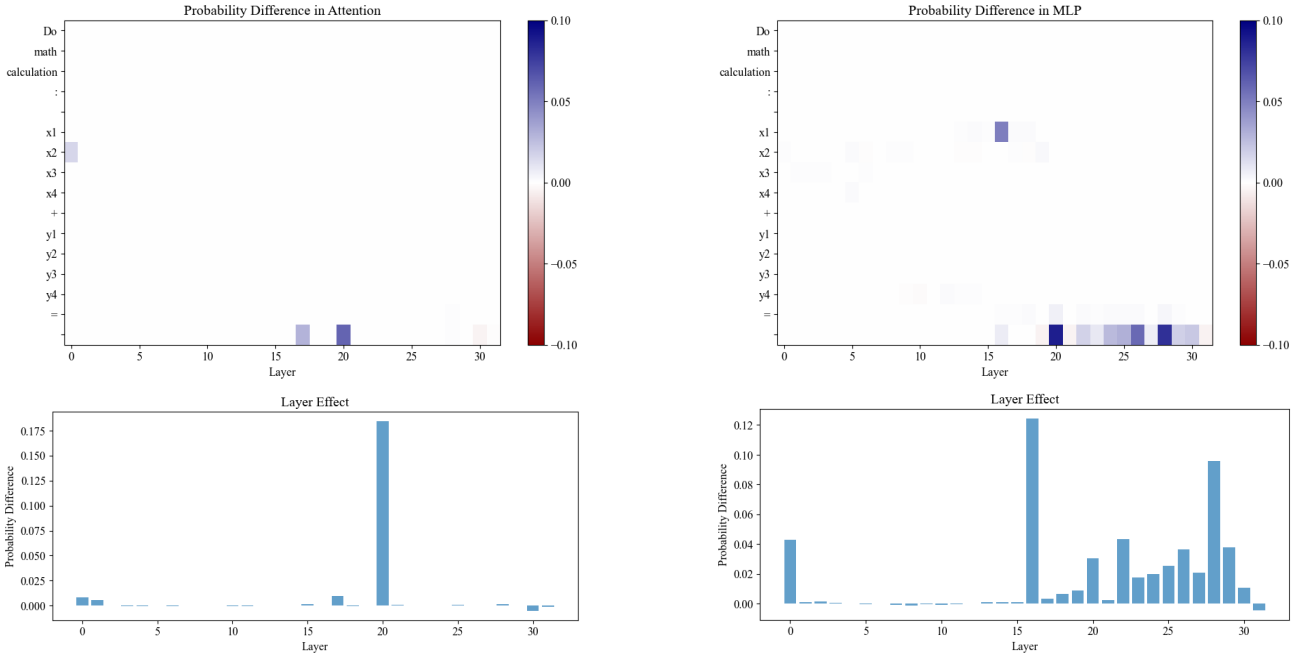


Figure 11: Total effect in Mistral-7B. Figures demonstrate the impact of activation replacement by tokens and interventions at the layer level, highlighting significant effects primarily concentrated on the last token and specific layers. For the attention layer, impacts are mainly on layer 20, while for MLP, impacts are notably on layer 20 and beyond.

Mistral has a special pattern, as it concentrates almost all attention effects on the 20th layer. By visualizing the attention heads of the 20th layer, we found that this layer contains attention to both the computational digit and the last digit, while the attention heads of other models separate these two functions. MLP suddenly experienced a peak in the 16th layer, which was caused by the first number of the first digit. This may be due to the model transmitting information between tokens.

Gemma also has a similar situation as it concentrates almost all attention effects on the 21st layer. The difference is that Mistral has more information on the first number of the first digit.

The influence of the attention layers on output probability may occur in the mid to late stage, but it will be earlier than the MLP layer, which is intuitively reasonable. On the one hand, attention transmits information between tokens, while MLP processes information. In addition, most of the v distribution related to answers only exists in the MLP layer in the later

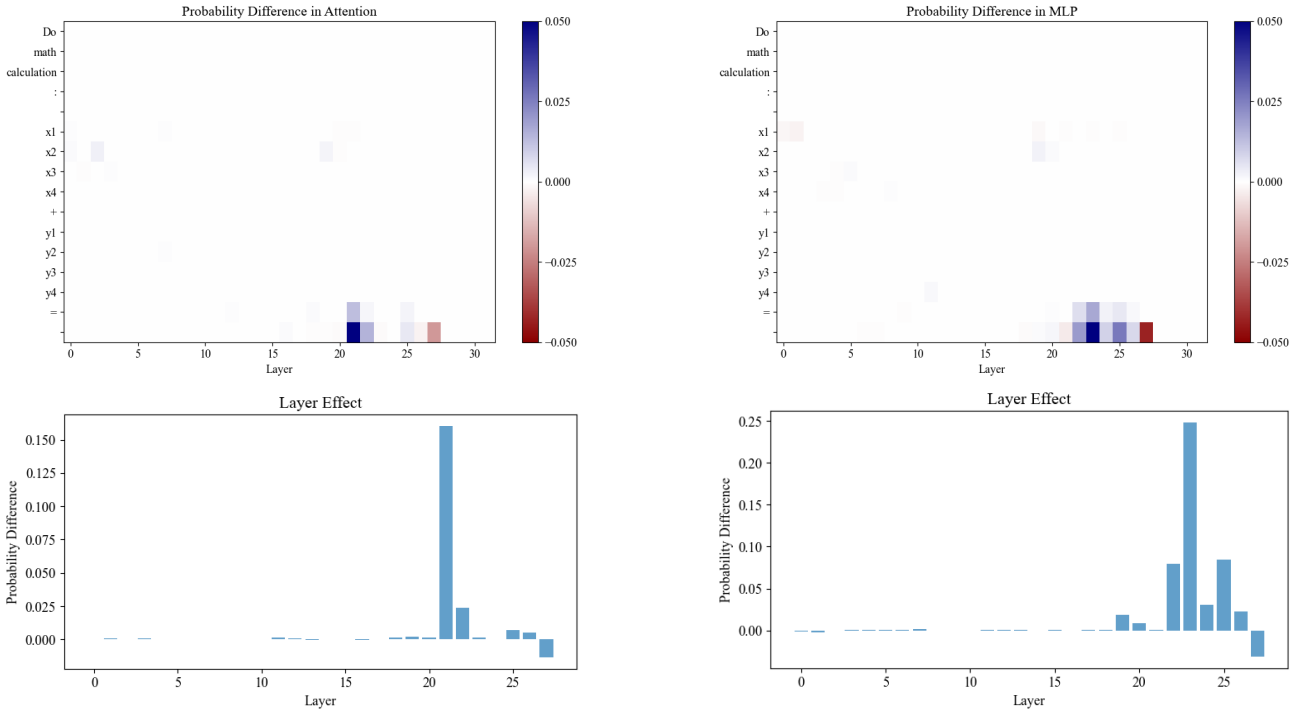


Figure 12: Total effect in Gemma-7B. For the attention layer, its impact is mainly on layer 21, while for MLP, its impact is mainly on layers 22 and later.

stage.

C. Finding v in parameter space

In this section, we give details of how we find the v.

Table 2 shows the definition of related words. For each v , we take the top k token with the highest probability of the top k and intersect it with the digit in the table. If the number of intersections is greater than r , v is classified as v_i which has encoded i.

From 1 to 9, the quantity generally shows a decreasing trend (See Figure 13). According to Benford's law, the probability of a number with 1 as the first digit appearing in a pile of real-life data is about 30% of the total. In terms of promotion, the larger the number, the lower the probability of its first few digits appearing.

We hypothesize that this is due to the presence of more 1 and 2 in the training data, resulting in a higher degree of training by the model. Furthermore, we use TSNE (Van der Maaten & Hinton, 2008) visualization to observe the internal clustering situation of the model. Specifically, we tested on a dataset with 9000 data points, where the first digit of the calculation result represents its label, with 1000 samples in each group of labels from 1 to 9.

As for the sentiment classification task, we use a similar approach to find the v, the related words are listed in table 3. The distribution of v that has encoded the sentiment concepts is shown in Figure 15. We found that the positive and negative tokens are usually mixed in one v, so we increase r from 3 to 5 to get more distinct v.

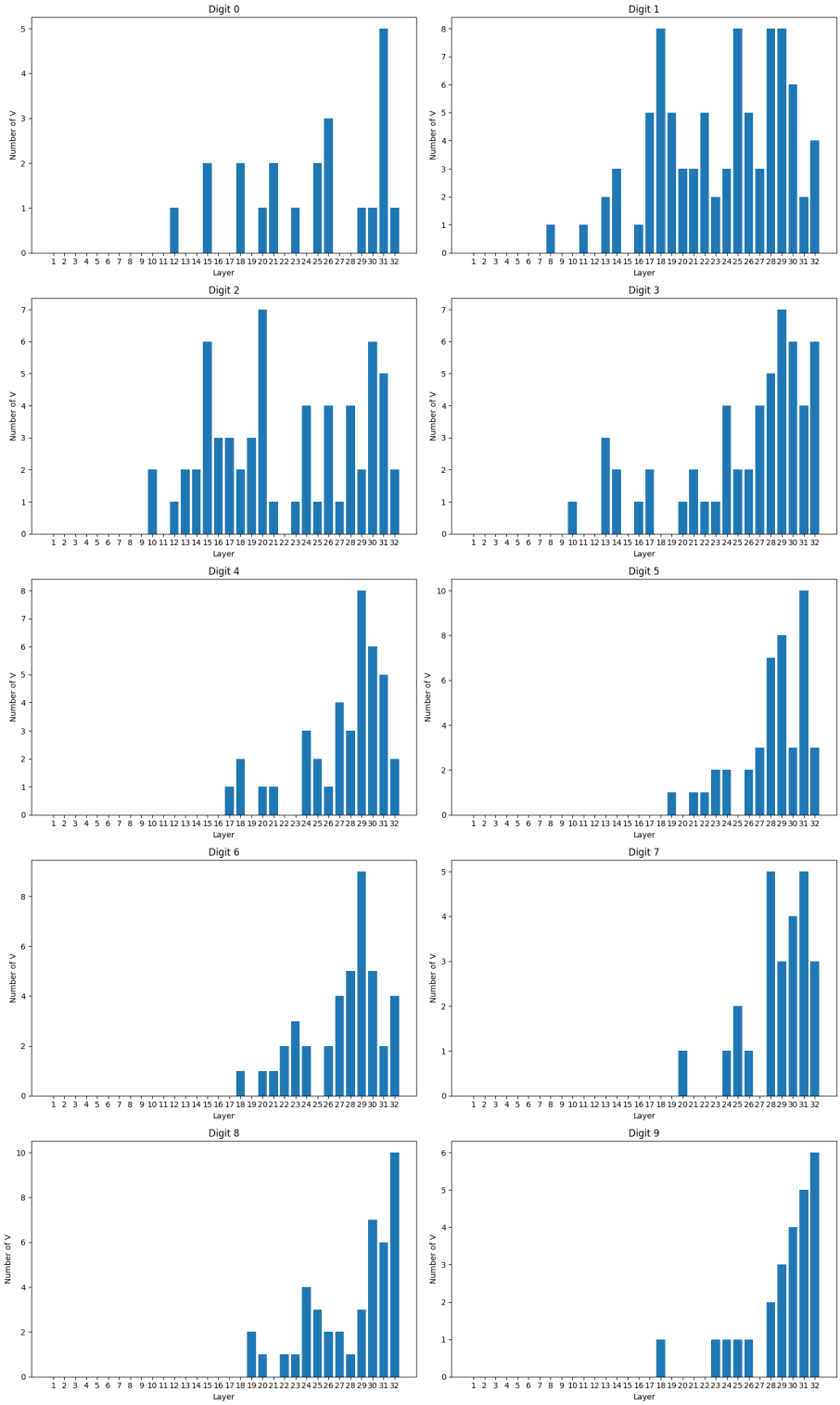


Figure 13: Total v_z that have encoded ten digits in LLaMA2-7B (k=10, r=3)

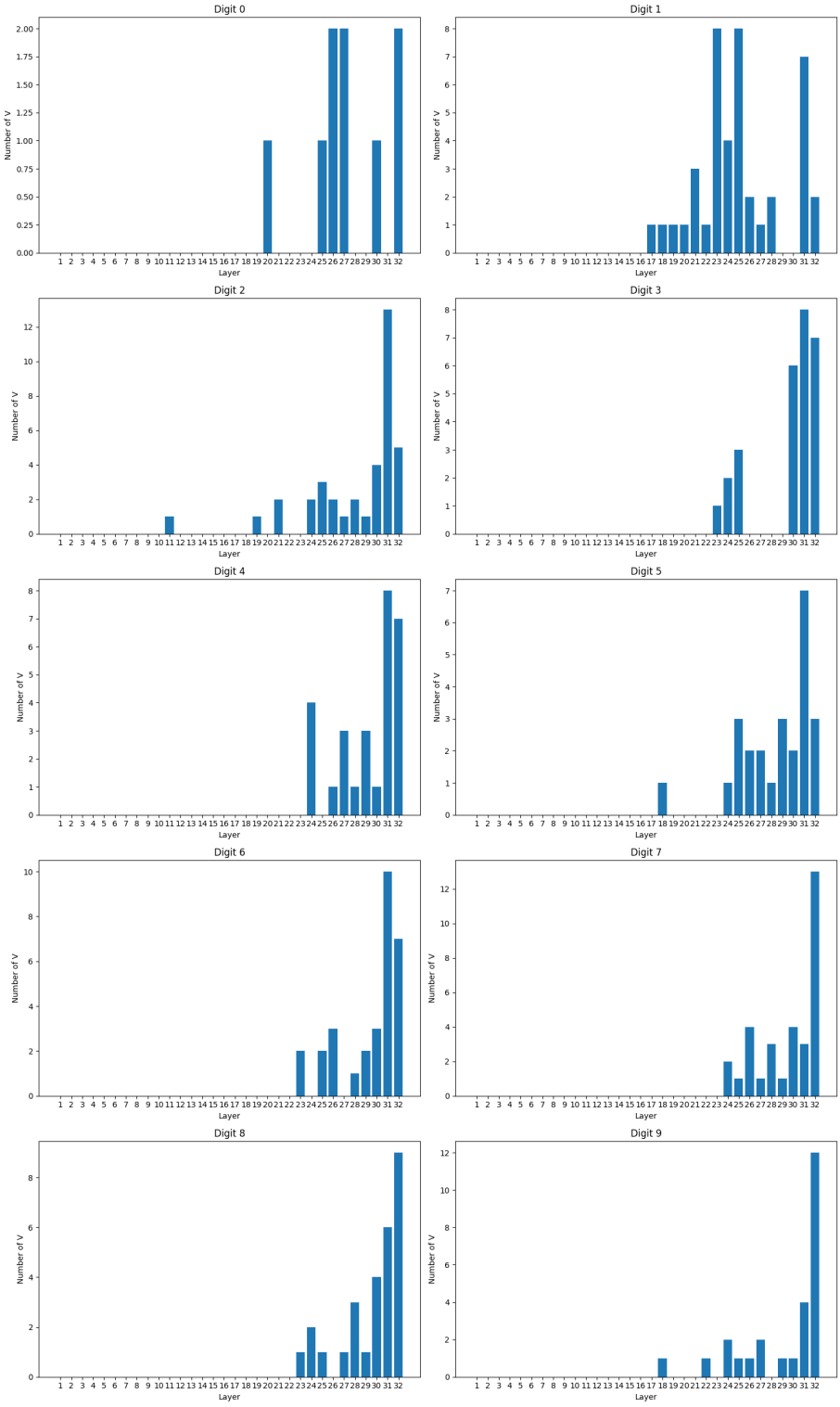


Figure 14: Total v_z that have encoded ten digits in Qwen1.5-7B (k=10, r=3)

Table 2: Defination of number related word

Digit	Related tokens
0	zero, zero(In Chinese), null, nil, zeros
1	one, one(In Chinese), January, Jan, 1, I, first, ten, uno, Uno, First, One
2	two, two(In Chinese), February, Feb, 2, II, second, twenty, duo, deux
3	three, three(In Chinese), March, Mar, 3, III, third, thirty, tri, Tres, triple, Three
4	four, four(In Chinese), April, Apr, 4, IV, fourth, forty, tetra, quatre, quad
5	five, five(In Chinese), May, 5, V, fifth, fifty, penta
6	six, six(In Chinese), June, Jun, 6, VI, sixth, sixty, hexa
7	seven, seven(In Chinese), July, Jul, 7, VII, seventh, seventy
8	eight, eight(In Chinese), August, Aug, 8, VIII, eighth, eighty, octa
9	nine, nine(In Chinese), September, Sep, 9, IX, ninth, ninety

Table 3: Defination of sentiment related word

Sentiment	Related tokens
positive	positive, posit, Pos, pos, happy, favor, fav
negative	negative, neg, Neg, bad

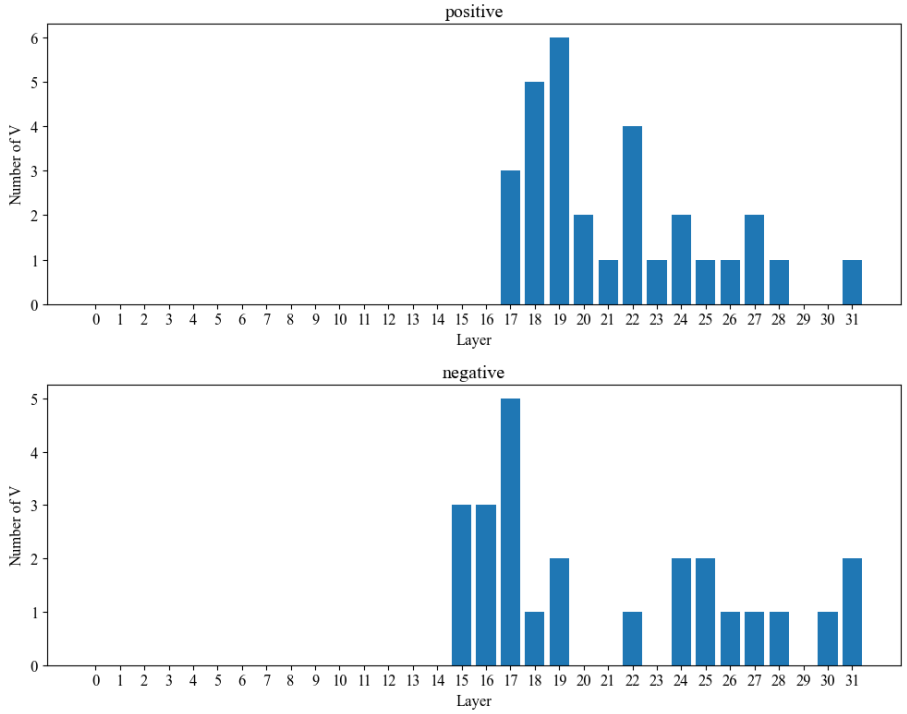


Figure 15: Total v_z that have encoded sentiment concepts in LLaMA3-7B (k=10, r=5)

Figure 16: t-SNE visualization of the last token hidden state across different layers, highlighting the classification of class 1 in blue and class 2 in red through layers 0, 11, 22, and 31.

D. Neurons activation and model output

LLaMA2, Qwen, and Mistal all use Gated-MLP (Dauphin et al., 2017) and SiLU activation function, leading to the existance of negative activation values. In the experiment, we found that considerable activations are in negative values. On the one hand, negative activation often means a decrease in $p(z)$ probability, and on the other hand, the output probability is strongly correlated with the sum of the absolute values of activations. This may indicate that the model does not simply activate all v to positive values to output the final answer. The model has learned a positive and negative distribution.

By observing Figure 17. The absolute sum of activation values reflects the probability of the model's output z to a relatively sensitive extent. The activation of random neurons remains at a relatively low level. The neuron activation of other digits remains at a higher level but lower than target neurons, which indicates that the model chooses to directly increase the value of the target instead of decreasing other neurons' activations.

When the model completes the addition task, other neurons related to numbers but not the answer will also be slightly activated. This may be due to the ambiguity of the selected v. For example, we found that the top-k word of some v selected in LLaMA2 includes both the number 9 and the number 8.

Interestingly, even randomly selected neurons exhibit a slight trend of change consistent with current neuronal activation, which may be related to other mechanisms in the model. Changes in input length may affect all neurons as a whole.

E. Additional information of commutative law and carry

In this section, we show more details about model learning commutative law and carry propagation.

Combine Figure 9 and Figure 18, model learns commutative law well when $z = 9, 6$, worst when $z = 7$.

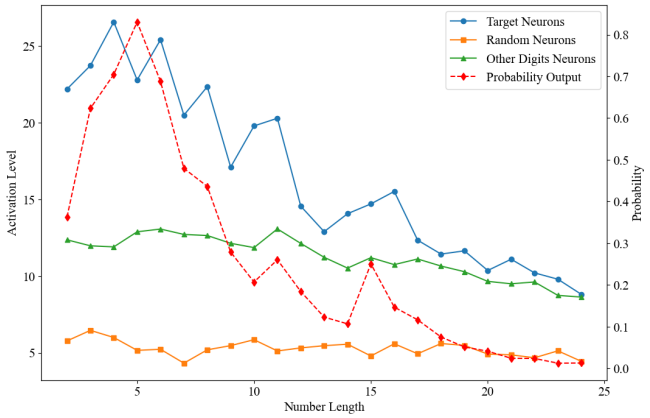
The attention head presents a stepped pattern(See Figure(19)), and the model moves the attention backward one bit for each bit calculated.

We conducted ablation of the attention head to verify the specific effect of attention and found that when we performed zero value ablation, the model canceled the calculation of carry or borrow.

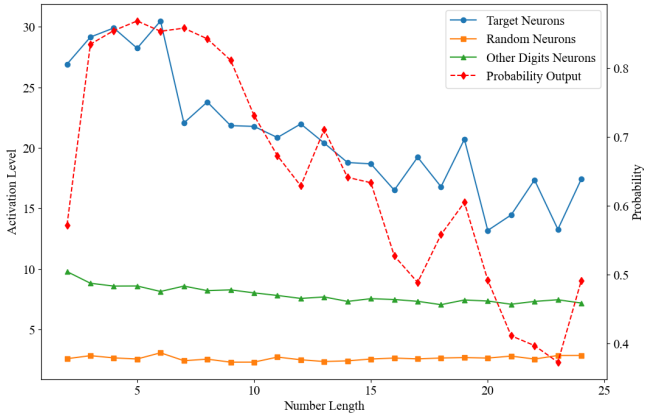
The problem is that attention does not dynamically change with the calculation results. when $xi + yi = 9$ attention does not keep tracking the next digits, the model simply increases the probability of adding a carry.

F. Compute resources

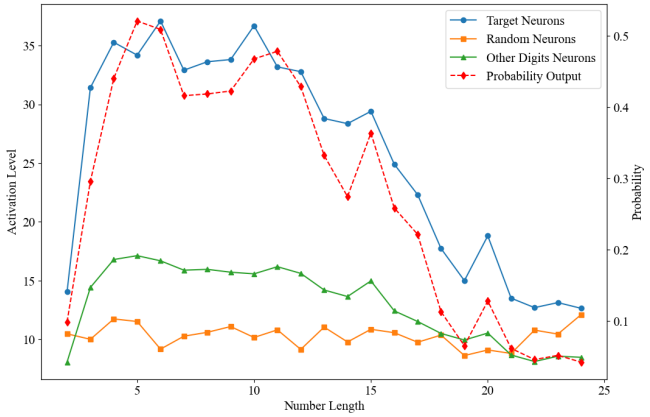
All experiments can be completed within 5 hours using the A40 graphics card.



Neurons activation and probability output $p(z)$ on Qwen1.5-7B



Neurons activation and probability output $p(z)$ on Mistral-7B



Neurons activation and probability output $p(z)$ on Qwen-7B

Figure 17: The trend of activation and probability changes in numbers of different lengths with double y-axis: top image represents Qwen1.5-7B, middle image represents Mistral-7B, and bottom image represents Qwen-7B.

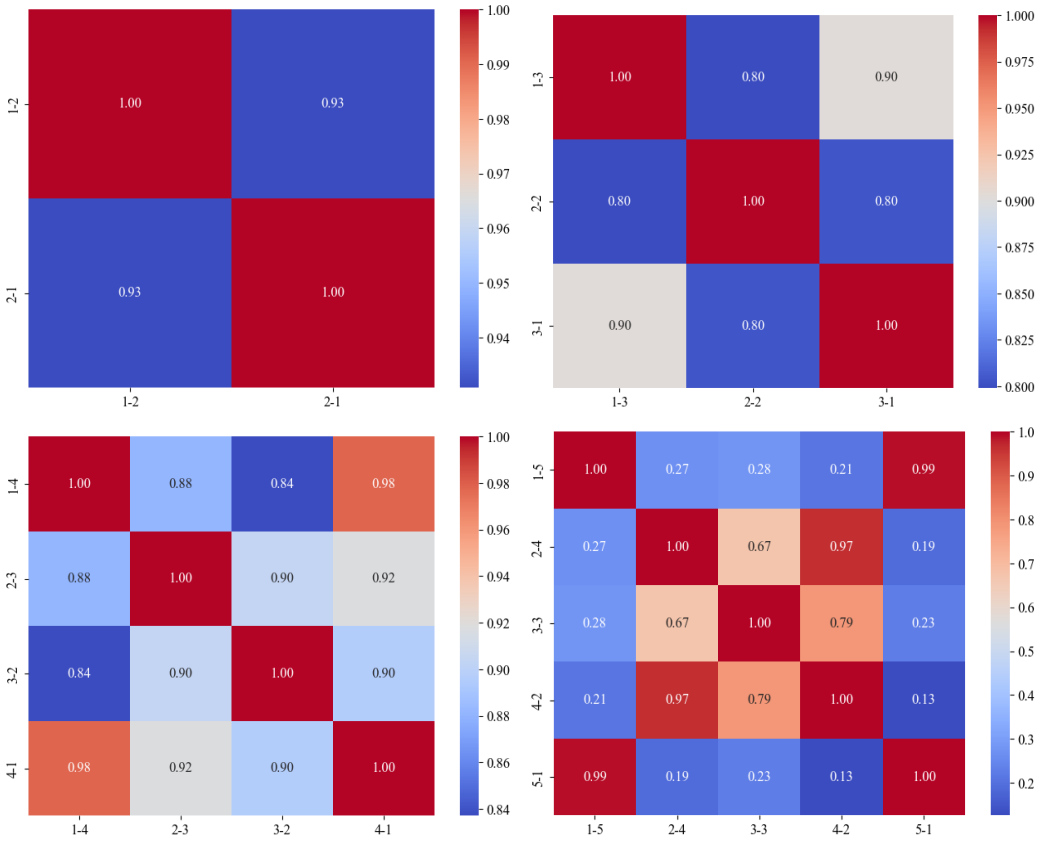


Figure 18: Cosine similarity of neuronal activation for digits 3, 4, 5, and 6, illustrating changes in activation patterns across different network layers. Each image represents the neuron activation and probability output $p(z)$ for different classes, averaged over 100 datasets.

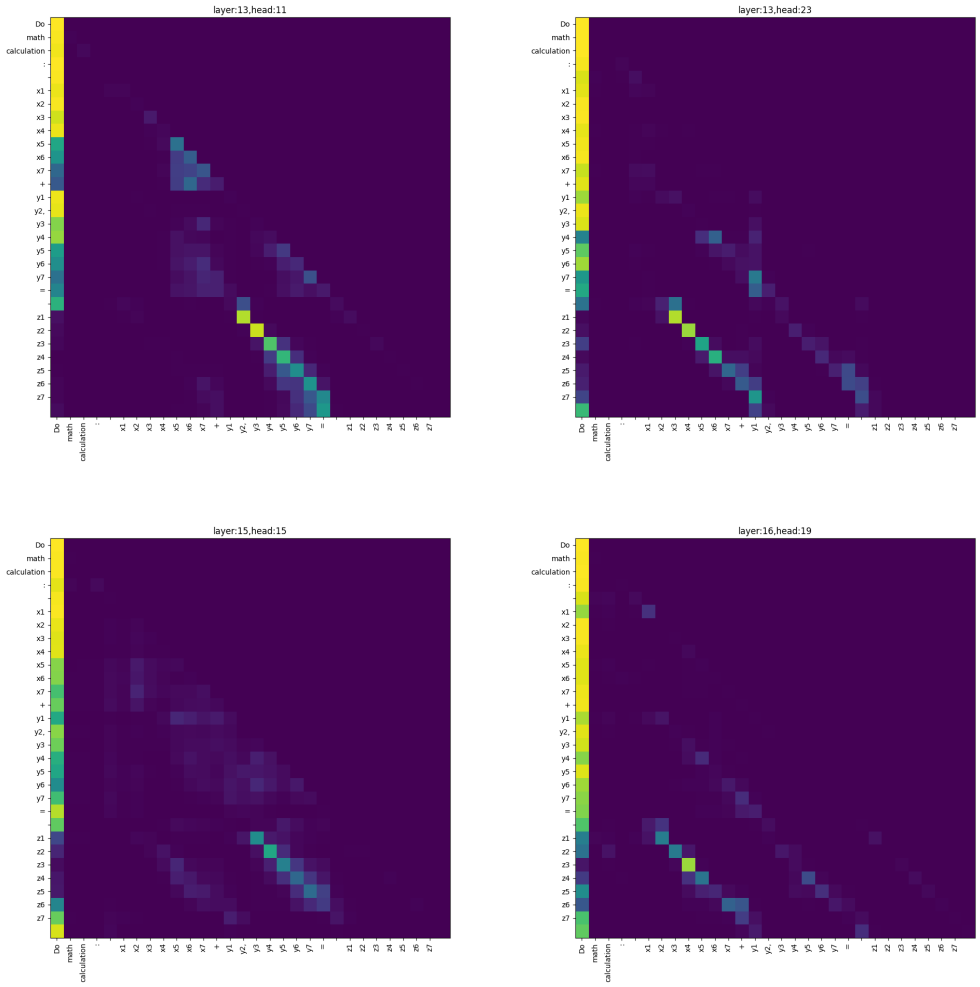


Figure 19: Visualization of various attention heads across different layers in LLaMA2-7B, demonstrating how each attention head contributes to controlling the carry in a neural network model.