several crucial words responsible for the calculation logic with irrelevant words. For example, the sentence from X_{r} like "42 plus 34 is equal to _" is replaced to the counterfactual one "42 *nothing* 34 is equal to _". In this way, it allows for a direct reflection of the model's impact on the arithmetic calculation tasks, rather than being influenced by the sentence structure or syntax.

4. Method

Our goal is to interpret the LLMs in a way that is human-understandable, thus enabling targeted modification of models through precise SFT. This section delves into the "identify-analyze-finetune" methodology. First, in Section 4.1, we describe the process of identifying and validating key components within LLMs. Then in Section 4.2, we examine the inherent patterns of these pivotal components to decode their behavior and distinct features. Finally, in Section 4.3, we introduce a strategy of precise SFT that fine-tunes these influential components to enhance the proficiency in calculation.

4.1. Key Components Identification.

The computation of the LLM can be reorganized as a directed acyclic graph (DAG) (Wang et al., 2023a). In the graph, each node is a computation component, including attention heads, MLP layers, residual connections, and each edge represents the data flow that the output of the previous node will be transposed to the input of the later node. Please refer to Appendix B for more details. To unravel the underlying cause of the model's predicted answer, we employ the causal intervention technique known as *path patching* (Goldowsky-Dill et al., 2023; Wang et al., 2023a). By perturbing targeted activation with counterfactual data X_{c} and freezing others with reference data X_{r}, the comparison on output logits is employed to measure the counterfactual effect. The whole process is illustrated in Algorithm 1. In this work, we scan through all nodes $\mathcal{N}$ one by one, and measure the changes in the output logit of ground-truth token $\{\mathtt{C}\}$, recoding in $E_{\mathcal{N}}$. Notably, since the residual operations and MLPs compute each token separately (Elhage et al., 2021), patching the head output at the END position (*i.e.*, the last token in the input sentence) is enough to measure the effects on the next token prediction.

Explanations for model behavior can easily be misleading or non-rigorous (Bolukbasi et al., 2021; Wiegreffe & Pinter, 2019). To address this issue, we further assess the importance of the identified heads/MLPs, while also confirming the insignificance of others. For this purpose, we employ a knockout technique called *mean ablation* (Wang et al., 2023a) to deactivate the individual heads/MLPs and observe their impact on model performance. Specifically, we replace their activation with average activation across counterfactual data X_{c} to remove the task-related information. By observing changes in model performance, we can verify the roles of these key heads/MLPs.

Algorithm 1 Identifying Key Components

Input: Set $\mathbf{\Omega}$ of reference and counterfactual sample pairs $(X_{\mathrm{r}}, X_{\mathrm{c}})$, model $\mathcal{M}$ with nodes $\mathcal{N}$.
Output: Causal effects for $\mathcal{N}$: $E_{\mathcal{N}}$.
for $(X_r^{(i)}, X_c^{(i)})$ in $\mathbf{\Omega}$ **do**
  Compute all activations A_r, A_c on $(X_r^{(i)}, X_c^{(i)})$
  for n in $\mathcal{N}$ **do**
    $A'_r(n) \leftarrow A_c(n)$; ▷ replace output in A_r by A_c
    $A'_r(k) \leftarrow A_r(k), \forall k \in [1, \cdots, |\mathcal{N}|], k \neq n.$
    $logit_o \leftarrow \mathcal{M}(X_r^{(i)}, A_r)$ ▷ get original logits
    $logit_p \leftarrow \mathcal{M}(X_r^{(i)}, A'_r)$ ▷ get patched logits
    $s_n^{(i)} \leftarrow \frac{logit_p - logit_o}{logit_o}$ ▷ causal effect
  end for
end for
Return: $\overline{s_n} = \frac{\sum_{i=1}^{|\mathbf{\Omega}|} s_n^{(i)}}{|\mathbf{\Omega}|}$ ▷ averaged effect w.r.t. samples

4.2. Pattern Analysis.

To make the identified heads/MLPs accessible to human understanding, we conduct a deeper analysis of their operational "behaviors". For attention heads, we examine the attention pattern $A_{i,j} \in \mathbb{R}^{N \times N}$ to comprehend which tokens are prioritized. N is the number of input tokens. Specifically, we begin by gathering the respective attention patterns $A_{i,j}$ on reference data X_r of the key heads. We extract the last row of $A_{i,j}$ for each sample, analyzing the attention scores $A_{i,j}^{END} \in \mathbb{R}^{1 \times N}$ between the Query token at the END position and each Key token, and obtaining the averaged scores w.r.t. samples. Generally, the type of token with the highest attention score represents the characteristics of the head, such as numbers, math symbols, etc.

For MLPs, we use the unembedding matrix as the probing to measure the content of token, especially numerical tokens, contained in MLPs' inputs and outputs. Prior studies, such as those reported in (Elhage et al., 2021), have illustrated that the MLP layer initially receives its input from the residual stream (*i.e.*, MLP_{in}), subsequently adding its output back into that stream (*i.e.*, MLP_{out}). Let W_U represent the unembedding matrix, and $W_U[*]$ denote the unembedding vector corresponding to a specific token. We calculate the cosine similarity between MLP_{in}, MLP_{out} and $W_U[\{\mathtt{A}\}], W_U[\{\mathtt{B}\}], W_U[\{\mathtt{C}\}]$ to reflect the information the MLP receives and generates. To isolate the specific contribution of MLP to specific numerical tokens, we further evaluate the subtraction of outputs and inputs of MLP, *i.e.*, $\frac{MLP_{out} - MLP_{in}}{||MLP_{out} - MLP_{in}||} \cdot \frac{W_U[\{\mathtt{A}\}]}{||W_U[\{\mathtt{A}\}]||}$. Research in (Geva et al., 2022) presents that each MLP layer's output token repre-