effect is mediated by node-to-node connections between individual neurons or features. Recent works have used path patching to explain neural networks in terms of circuits (Olah et al., 2023), identified for different capabilities including indirect object identification (Wang et al., 2023a), greater-than computation (Hanna et al., 2023), and mapping answer text to answer labels (Lieberum et al., 2023).

Interpretability for Mathematical Tasks. Mathematical ability has long been a subject of interest in natural language processing (Kushman et al., 2014; Huang et al., 2016; Wang et al., 2017; Thawani et al., 2021). Some studies have investigated the mathematical abilities of LLMs (Frieder et al., 2023; Saxton et al., 2019; Nogueira et al., 2021; Qian et al., 2023; Imani et al., 2023; Romera-Paredes et al., 2024), but they mainly focus on explaining *what* these models can do rather than *how* they do it. In contrast, some other studies have dived deeper into the LLM structure without treating LLM as an inscrutable black box. Stolfo et al. (2023) identified the key attention *layers* relating to arithmetic questions, but lacking in-depth explanation and validation of the key layers' behaviors. Wu et al. (2023) scaled the methods from causal abstraction to understand how Alpaca (7B) (Taori et al., 2023) follows the instruction in comparing two numbers. (Hanna et al., 2023) provided a causal explanation about how GPT2-small (0.1B) (Radford et al., 2019) implements the "greater-than" task, but only reveal simple phenomena limited by the small size of model and the lack of diversity in the dataset.

Fine-tune LLMs for Mathematical Tasks. Numerous studies improve the mathematical reasoning ability of LLMs by aggregating various sampled reasoning paths during either fine-tuning or inference. Cobbe et al. (2021) train and devise a reasoning path verifier to select the correct results during inference. Wang et al. (2023b) propose to sample various reasoning paths during inference and then derive the final result by majority voting on the answers or through verifiers (Li et al., 2023). Uesato et al. (2022) explore to use of reinforcement learning methods for improving the mathematical reasoning abilities of LLMs. Several works apply the idea of rejection sampling along with other techniques to filter the diverse sampled reasoning paths for fine-tuning data augmentation (Huang et al., 2022; Zelikman et al., 2022; Ni et al., 2023). There also exist related works (Panigrahi et al., 2023) that locate key parameters to update for better task-specific ability. Panigrahi et al. (2023) locates a minuscule subset of parameters from an already fine-tuned model onto a pre-trained model without further tuning. The selection process for this subset is via optimizing the task-related objective function with L1 norm ensuring the sparsity of the subset. In our work, we locate the task-related parameters of pre-trained model via measuring the *causal effect* of each component, then *precisely fine-tune* the key components for mathematical tasks.

3. Preliminary

Large Language Models (LLMs). The LLMs utilized in this work comprise LLaMA2-7B and LLaMA2-13B (Touvron et al., 2023a). These are pre-trained language models freely available from HuggingFace[2]. All of these models are decoder-only transformers equipped with multi-head attention (MHA) and a single MLP in one transformer layer. For example, LLaMA2-7B consists of 32 transformer layers and 32 attention heads in MHA for each layer.

Transformer Architecture. The input to the transformer is a combination of position and token embeddings in $\mathbb{R}^{N\times d}$, where N is the number of tokens in the input and d is the model dimension. Following the definitions in (Elhage et al., 2021), the input embedding serves as the initial value for the *residual stream*, which is read from and written to by all attention heads and MLPs. Focusing on individual heads, the j-th head in the i-th layer is parametrized by four matrices: $W_Q^{i,j}$, $W_K^{i,j}$, $W_V^{i,j} \in \mathbb{R}^{d\times\frac{d}{H}}$, and $W_O^{i,j} \in \mathbb{R}^{\frac{d}{H}\times d}$. To simplify these parameters, we can express them as low-rank matrices in $\mathbb{R}^{d\times d}$: $W_{OV}^{i,j} = W_O^{i,j}W_V^{i,j}$ and $W_{QK}^{i,j} = W_Q^{i,j}(W_K^{i,j})^T$. The QK matrix is used to compute the attention pattern $A_{i,j} \in \mathbb{R}^{N\times N}$ for head (i, j), while the OV matrix determines the information written into the residual stream. At the end of the forward pass, a layer norm is applied before the unembed matrix W_U projects the residual stream into logits.

Task and Dataset. We focus on classic and widely encountered mathematical operations, *e.g.*, addition, subtraction, multiplication, division. Taking addition as an example, the arithmetic logic of addition ({A} + {B} = {C}) might naturally appear in sentences. Taking inspiration from the sentence styles and forms present in mathematical benchmarks of GSM8K (Cobbe et al., 2021) and SVAMP (Patel et al., 2021), we create a dataset for the addition task containing $10,000$ samples based on 36 templates with random single-token names, objects, and numbers. To assess the performance of LLMs on the calculation task, we measure the prediction probability of the {C} token. The average probability of correct predictions across the models was 82%. In this study, we select the samples that the language models are able to predict correctly. We denote the sentences generated by this procedure as reference data using the notation of X_{r}. For the templates and sentences, please refer to Figure 8 and Figure 10 in Appendix A.

Moreover, to meet the demand for perturbing component activation, we create another dataset comprising counterfactual sentences without the inclusion of calculation logic, using the notation of X_{c}. The samples are generated following two core principles: (1) maintaining the grammatical structures derived from the X_{r} templates; (2) substituting

[2]https://huggingface.co/