Analyzing the Rapid Generalization of SFT via the Perspective of Attention Head Activation Patterns

Yang Zhao[♠*], **Li Du**[♡*], **Xiao Ding**[♠†], **Kai Xiong**[♠], **Ting Liu**[♠] **and Bing Qin**[♠]
[♠]Research Center for Social Computing and Interactive Robotics,
Harbin Institute of Technology, China
[♡]Beijing Academy of Artificial Intelligence, Beijing, China
{yangzhao, xding, kxiong, tliu, qinb}@ir.hit.edu.cn
duli@baai.ac.cn

Abstract

LLMs' performance on complex tasks is still unsatisfactory. A key issue is that presently LLMs learn in a data-driven schema, while the instructions about these complex tasks are both scarce and hard to collect or construct. On the contrary, a prominent phenomenon is that LLMs can learn rather fast on simpler tasks with adequate prior knowledge captured during pretraining stage. Thus, if the prerequisite and mechanism of such rapid generalization could be elucidated, it could enhance the efficiency and effectiveness of the LLM's ability to learn complex tasks. Thus, in this paper, we employ a gradient-based method, to dissect the process that the SFT process adapts LLMs to downstream tasks via the perspective of attention patterns. We find that: (1) LLMs selectively activate task-specific attention heads during SFT; (2) activation patterns for complex tasks are combinations of basic task patterns; and (3) changes in a few parameters can significantly impact activation patterns after SFT on a small number of samples. Based on these insights, experiments are conducted to actually enhance the efficiency and effectiveness of SFT. The source code for this work is publicly available at: https://github.com/zy125413/SFT_AP.

1 Introduction

The Supervised Fine-tuning Process (SFT) is essential for optimizing Large Language Models (LLMs) to effectively complete downstream tasks (Zhang et al., 2023). Leveraging the extensive prior knowledge acquired during pretraining, fine-tuning LLMs on instruction sets enables them to perform well across various tasks (Dong et al., 2023; Xia et al., 2024). However, in more complex tasks like mathematical reasoning, LLM performance remains unsatisfactory (Fourrier et al., 2024; Contributors, 2023). One major reason for such performance limitation is that these complex tasks often require simultaneously using multiple types of knowledge and skills, and thus, more instructions are often required to adapt LLMs to these tasks (Kaplan et al., 2020). However, the more complex the task, the more challenging it is to collect and construct relevant instruction data (Zhang et al., 2023; Minaee et al., 2024). This in turn limits the ability of LLMs to improve performance on these tasks.

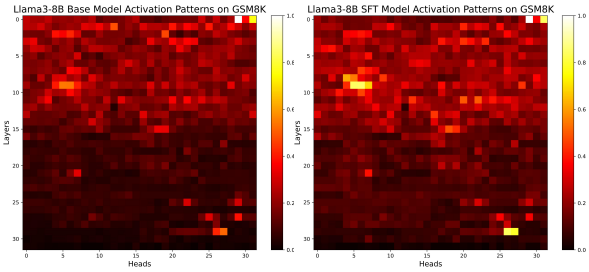

Figure 1: Visualization of activation pattern changes in Llama3-8B on the test set before and after SFT with the GSM8K training set.

In contrast, on most basic tasks, LLMs demonstrate prominent rapid generalization (Zhang et al., 2023). By training on just a few thousand instructions, LLMs can learn to complete various tasks (Xia et al., 2024). Therefore, understanding the mechanisms and conditions that enable LLMs' rapid learning and generalization could strongly guide their adaptation to complex tasks and potentially enhance their efficiency.

To address this issue, we propose to analyze the prerequisite and mechanism of such rapid task adaption during SFT from the perspective of *activation patterns of attention heads* using a *gradient-based* method. Previous studies show that attention heads serve as basic functional units in transformer-based models (Voita et al., 2019; Clark et al., 2019; Hao et al., 2021; Wang et al., 2023). These attention heads could capture different types of information and model various relationships for completing

[*]These authors contributed equally to this work.
[†]Corresponding Author.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 16980–16992

different tasks. Therefore, analyzing how LLMs learn to utilize these basic function units to solve tasks during the SFT process could shed light on the mechanisms behind instruction learning and generalization. If an attention head influences outputs in certain tasks, it is heuristically considered "*activated*". The composition of activations across different attention heads forms the *activation pattern* of the LLMs (Figure 1), indicating how the model's functional units are integrated to solve a specific task.

To determine if an attention head influences the output, we use a gradient-based method as gradients naturally measure the sensitivity and impact of output changes with respect to input features (Wang et al., 2024a). By calculating the expected value of the gradient on the attention score matrix, we can quantify the impact of each attention head on model performance before and after SFT and explains how SFT adapts LLMs to downstream tasks.

From the perspective of attention head activation patterns, we find the critical role of the change of activation pattern in fast learning and generalization during the SFT process: **(1) LLMs learn to complete tasks by selectively activating attention heads during the SFT process**. After SFT, the LLMs activate certain tasks-specific attention heads, and the differences in activation patterns between tasks become more pronounced; **(2) LLMs complete tasks by learning to integrate basic skills**.This finding suggests that complex task-solving in LLMs may be decomposed into a series of simpler subtasks, offering a modular perspective on how LLMs can be more effectively fine-tuned for intricate problems; **(3) Changes in a few parameters can significantly impact activation patterns after SFT on a small number of samples**. The model's activation patterns exhibit significant changes compared to the base model in the early stage of the SFT process with rather limited optimization steps, showing that minimal parameter adjustments can significantly alter activation patterns.

We validate our findings by exploring their practical application in enhancing the effectiveness and efficiency of SFT, particularly for complex tasks and in scenarios where high-quality instructions are scarce. Specifically: (1) When complex task data is limited, by fine-tuning LLMs using instructions on the basic skills required for these tasks, the efficiency of instruction tuning significantly improves. (2) When high-quality domain data is private or unavailable, based on activation patterns, we can select relevant instructions from a large pool of publicly available data to approximate the effects of private data. These findings not only validate our understanding of rapid learning and generalization mechanisms but also provide a scalable framework for improving LLM performance in real-world, data-constrained scenarios, such as specialized industries or domains where data availability is limited.

2 Background and Related Work

One major factor limiting the learning efficiency of instruction learning of LLMs is the opacity of their internal mechanisms. Though pioneer studies recognized key features or parameters of LLM (Hao et al., 2021; Wang et al., 2023; Yang et al., 2023b), the mechanism of adapting LLMs towards downstream tasks during SFT is still largely unknown (He et al., 2024; Dong et al., 2023). In this paper, we aim to serve as a pioneer to explore these underlying mechanisms.

A critical issue is from which perspective should we start. Instead of focusing on the parameters of LLMs, we choose to investigate the *activation pattern of attention heads*, as:

(1) Previous studies have identified that an attention head can act as a basic functional unit in transformer-based models (Voita et al., 2019; Clark et al., 2019; Wang et al., 2023), with different attention heads serving distinct functions. For example, Voita et al. (2019) and Clark et al. (2019) found that attention heads at different layers focus on different parts-of-speeches, while Wang et al. (2023) observed that during in-context learning, different attention heads focus on different parts of the prompts.

(2) Additionally, prior research (Jin et al., 2024; Lee and Hwang, 2024) has suggested the task-specific activation of attention heads, as pruning certain attention heads would lead to task-specific performance degradation. Heuristically, since a task can be viewed as a combination of several basic tasks, completing it requires the coordination of multiple corresponding basic functional units, i.e., attention heads.

(3) In contrast, Previous studies (Yu et al., 2024; Fu et al., 2023) have suggested that parameter changes after SFT are limited. Furthermore, given the vast number of parameters in LLMs, interpreting the significance of these parameters and their

changes are highly challenging.

For clarity, we define attention heads that impact downstream tasks as *activated* and those that do not as *inactive*. The matrix representing the influence of attention heads on downstream tasks is called the *activation pattern*.

3 Methodology

During pretraining, LLMs learn to use different attention heads to capture various types of information (Voita et al., 2019; Khaki and Plataniotis, 2024). Heuristically, a complex task can often be viewed as a composition of several basic tasks. For instance, solving a mathematical problem using code, can be broken down into fundamental tasks like code generation and mathematical reasoning. Since each attention head may correspond to a specific function, an LLM can invoke multiple attention heads and integrate their functions to accomplish a complex task. Combining these observations leads to our core assumption: **during the SFT process, LLMs quickly adapt to downstream tasks by invoking different attention heads.** Thus, in this paper, we investigate the mechanism of fast generalization in the SFT process by analyzing changes in the activation patterns of attention heads.

3.1 Gradient-Based Analysis of Attention Head Activation Patterns

A key issue is determining whether an attention head is *activated* during a task. Inspired by Hao et al. (2021) and Wang et al. (2023), we measure this using model gradients, as they reflect the influence of input features or parameters on the model's output (Wang et al., 2024a). A larger gradient for a particular attention head suggests that the model's output is more sensitive to it, indicating a stronger influence on outputs.

Specifically, given an LLM with L layers and H attention heads per layer, and a dataset $\mathcal{T}$, the activation pattern $AP^{\mathcal{T}}$ can be represented as an $L \times H$ matrix, with each element representing the activation level reflecting the contribution of one specific attention head to a given task:

$$AP^{\mathcal{T}} = \{AL^{\mathcal{T}}_{l,h}\}_{l\in[1,\ldots,L],h\in[1,\ldots,H]}, \quad (1)$$

where each element $AL_{l,h}$ corresponds to the activation level of the h-th attention head in the l-th layer during the given task, which is measured as:

$$AL_{l,h} = \frac{1}{N}\sum_{i} \Gamma^{T}_{l,h} \frac{\partial L(x_i)}{\partial \Gamma_{l,h}}, \quad (2)$$

where N represents the size of $\mathcal{T}$, x_i is the ith instance in $\mathcal{T}$ and $L(x_i)$ is the loss value of the model for x_i. $\Gamma_{l,h}$ is the attention matrix of the h th attention head in the l th layer. $\frac{\partial L(x_i)}{\partial \Gamma_{l,h}}$ is the gradient of the h th attention head in the lth layer with respect to $L(x_i)$. By combining the absolute values of attention scores with the sensitivity of the loss to these scores, the total influence of a specific attention head on outputs and the task can be quantified. Thus, AP^{task}, composed of AL^{task}, reflects how each attention head contributes to task completion.

3.2 Analytical Framework

To validate our core assumption, we focus on three progressive issues: (1) How do activation patterns change during the SFT process for each task, and are these changes task-specific? In other words, can we grasp the characteristics of the task from the perspective of activation patterns? (2) What is the relationship between activation patterns in complex tasks and basic tasks? This reflects the prerequisites that may be necessary for learning complex tasks; and (3) on basic tasks, how many training samples are required to significantly change the activation patterns? The necessary sample size indicates the model's ability to achieve rapid generalization.

First, we evaluated the changes in activation patterns across tasks before and after SFT. We observed that more attention heads are activated after SFT, and these activations are task-specific, suggesting that attention heads are selectively activated during SFT. Next, we explored the relationship between activation patterns in complex and basic tasks, finding that the patterns of several basic tasks effectively approximate those of complex tasks. This led us to conclude that activation patterns in complex tasks are combinations of basic task patterns. Finally, dynamic observations during SFT showed that even small datasets can significantly alter activation patterns.

To further validate our conclusions, we applied these insights to enhance SFT effectiveness on complex tasks. The resulting improvements further confirmed our theory. The following sections provide a detailed analysis and conclusions.

4 SFT Adapts to Downstream Tasks by Modifying Activation Patterns

This section uses gradient-based analysis to examine the effect of SFT on transformer attention head activation patterns. We identify three key insights: (1) After SFT, LLMs accomplish tasks by selectively activating specific attention heads. (2) Changes in activation patterns for compound tasks can be interpreted as a combination of those in more basic tasks. (3) During SFT, activation patterns exhibit swift changes, even small datasets can cause significant changes.

4.1 Attention Heads are Selectively Activated

In this section, we analyze changes in attention head activation patterns before and after SFT across various tasks and explore the relationships between these changes.

4.1.1 Analytical Method

To analyze changes in model activation patterns before and after SFT for each task, we used three metrics: Gini Coefficient, Coefficient of Variation (CV), and Kurtosis. The Gini Coefficient measures distribution inequality, with values above 0.4 indicating a concentration of activation in a few attention heads. The Coefficient of Variation reflects relative dispersion, with values over 1 signaling substantial variability across attention heads. Kurtosis describes the distribution's shape, where values above 3 suggest heavy tails and sharp peaks, indicating the presence of extreme values

4.1.2 Experiments Setup

We conducted experiments on models with varying performance levels, including Llama3-8B (Grattafiori et al., 2024), Gemma-7B (Team et al., 2024), and OPT-6.7B (Zhang et al., 2022), across a range of tasks: mathematical reasoning (MATH (Hendrycks et al., 2021b), GSM8K (Cobbe et al., 2021)), coding (Code Search Net (Husain et al., 2019), SGSM (Christ et al., 2024)), and natural language processing and reasoning (HellaSwag (Zellers et al., 2019), Winogrande (Sakaguchi et al., 2021), ARC (Clark et al., 2018)).

4.1.3 Results

As shown in Table 1, before SFT, base models exhibited significant outliers (Kurtosis > 3), uneven distribution (Gini > 0.4), and high variability (CV > 1) in activation patterns, suggesting that only a few attention heads contributed to the tasks. After SFT, activation patterns became more uniform, with decreases in Gini, CV, and Kurtosis, indicating that SFT adapts LLMs by increasing activation levels across attention heads. However, activation values remain skewed, showing that a few heads still dominate.

Model	State	Gini	CV	Kurtosis
Llama3-8B	Before SFT	0.50	1.19	95.37
	After SFT	0.33	0.71	39.55
Gemma-7B	Before SFT	0.48	1.71	240.99
	After SFT	0.38	1.23	130.82
OPT-6.7B	Before SFT	0.42	1.06	50.67
	After SFT	0.38	0.82	24.23

Table 1: Statistics on the distribution of activation patterns for different LLMs. Experiments were conducted on tasks such as Code Search Net, GSM8k, MATH, SGSM, ARC, HellaSwag, and Winogrande.

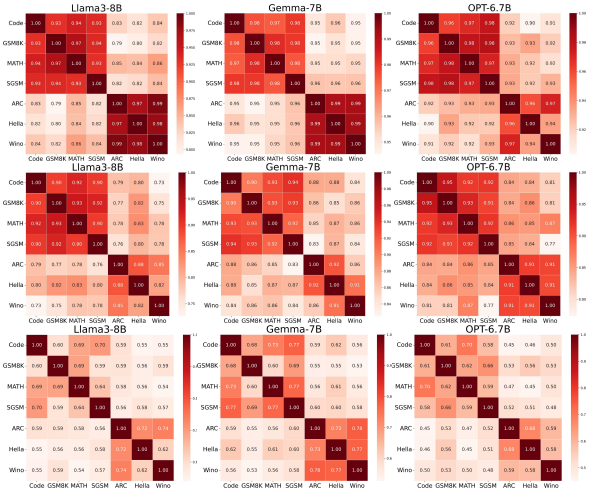

Figure 2: The correlation coefficients of the activation pattern change rates for the Llama3-8B, Gemma-7B, and OPT-6.7B models on tasks before SFT, after SFT, and during the SFT process (corresponding to the top, middle, and bottom sections, respectively).

Figure 2 shows the similarity in activation pattern change rates across tasks for three models before and after SFT, measured by correlation coefficients. After SFT, tasks grouped into two categories—math/code and text reasoning—aligning with human understanding of task relationships. This suggests that attention patterns reflect task characteristics and specificity. Post-SFT, stronger task specificity is observed, as indicated by decreased correlation coefficients between tasks, implying reduced similarity.

Considering that the activation levels increase after SFT, we deduce that SFT adapts models to spe-

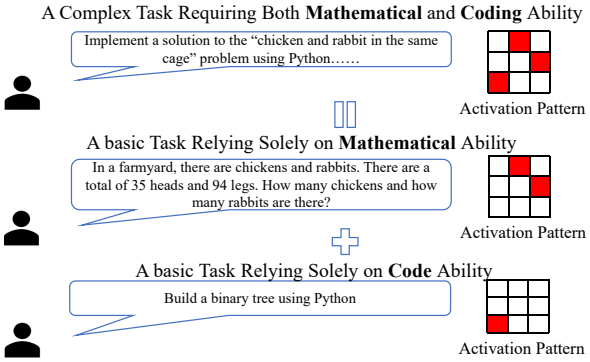

Figure 3: The two tasks above are basic tasks that each rely on a single skill, while the one below is a complex task that relies on both coding and mathematical skills.

cific tasks by selectively enhancing task-relevant activation patterns.

4.2 Activation Patterns on Complex Tasks are Combinations of Basic Tasks

This section focuses on the relationship between activation pattern changes in basic and complex tasks. Understanding this relationship is key to equipping models with the necessary prerequisites for rapid learning of complex tasks.

4.2.1 Analytical Method

In this paper, we define *complex tasks* as those that can be decomposed into basic skills. For example, as shown in Figure 3, solving the "chicken and rabbit" problem requires mathematical reasoning, while constructing a binary tree requires Python coding—both of which are basic tasks. However, solving the "chicken and rabbit" problem using Python code is a complex task, as it integrates both mathematical reasoning and coding skills.

The actual relationship between activation patterns in basic tasks and complex tasks can be quite intricate. without loss of generality, in our study, we employ a linear function to model the relationship between the activation patterns of basic and complex tasks. Specifically, we take the changes in activation patterns for complex tasks as the dependent variable, and employ the changes in activation patterns for multiple simple tasks as independent variables, using the least squares method to fit the regression coefficients.

$$\Delta AP^{complex} = \sum_{i=1}^{n} \alpha_i \Delta AP^{basic_i} + \epsilon, \quad (3)$$

where α_i is a regression coefficient representing the contribution of each basic task i to the change

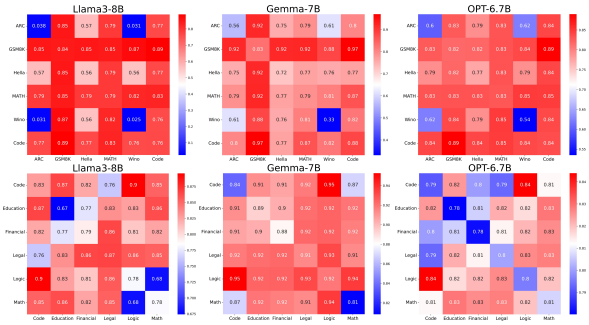

Figure 4: **Top**: The least squares method was used to fit the activation pattern changes of traditional NLP tasks in SFT to the activation pattern changes in SGSM. A higher R^2 indicates a better fit, with Code Search Net and GSM8k showing the highest R^2 values. **Bottom**: The least squares method was used to fit the activation pattern changes of SFT instruction data to those of tasks requiring both "logical reasoning" skills and "programming and software development" instructions. The combination of "logical reasoning" skills and "programming" instructions achieved the highest R^2 value.

in activation patterns for the complex task, n is the total number of basic tasks, and ϵ is the error term. Subsequently, we employ the R-squared (R^2) value to measure the goodness of fit, which represents the proportion of the variance in the dependent variable that can be explained by the independent variables.

4.2.2 Experiments Setup

In this study, we conducted two groups of experiments: (1) We take MATH, GSM8K, Code Search Net, HellaSwag, Winogrande, and ARC as the basic tasks, and select the SGSM task, which requires coding ability and elementary math skills, as the complex task. (2) We conduct experiments on the Infinity Instruct dataset (Zhao et al., 2024), which provides each instruction tags describing the skills and knowledge required to complete that instruction, so that we can select both basic and complex task-related instructions. Specifically, we selected instructions requiring only a single skill, including 'mathematics', 'logical reasoning', 'programming and software development', 'education and consulting', 'financial and business knowledge', and 'legal knowledge', to form into the basic task data set. Next, we selected instructions requiring both 'logical reasoning' and 'programming/software development' skills as the complex tasks for SFT and calculated their activation patterns.

4.2.3 Result

The upper part of Figure 4 shows the R^2 values obtained by fitting the activation patterns of vari-

ous traditional NLP tasks using the least squares method, with the activation pattern changes of the SGSM task as the dependent variable. Among these tasks, Code Search Net and GSM8K yield the highest R^2 value of 0.97. This indicates that (1) the activation patterns of complex tasks can be well-fitted by the linear combination of activation patterns from simpler tasks; (2) SGSM is a task that involves solving elementary math problems using code, requiring a combination of elementary math and coding skills. In fact, the changes in SGSM's activation patterns can be well-fitted by the activation pattern changes from the elementary math task GSM8K and the coding task dataset Code Search Net.

A similar phenomenon can be observed in the results based on Infinity Instruct. The lower part of Figure 4 shows the R^2 values obtained by fitting the activation patterns of various instruction-based NLP tasks using the least squares method, with the activation pattern changes of the instruction tasks that require both logical reasoning and programming/software development skills as the dependent variable. Among these tasks, the instructions that rely solely on logical reasoning and those that rely solely on programming/software development yield the highest R^2 value of 0.95.

This shows that the activation patterns of complex tasks can be formed by combining the activation patterns of several simpler tasks, and this compositional relationship reflects the real-world connections between these tasks. Therefore, SFT adapts LLMs to complex tasks by integrating the attention heads corresponding to the simpler tasks.

Furthermore, the weighted sum of activation patterns from models fine-tunned on simple tasks resembles that on complex tasks. It suggest that learned enough from basic tasks, its state will approximate that of a model directly fine-tuned with instructions related to complex tasks. This provides guidance on how to establish the necessary prerequisites for complex task performance.

4.3 Small Data Can Change Activation Patterns

In this section, we find that SFT can quickly adapt LLMs to certain basic tasks by adjusting the activation level of attention heads with rather limited instances. This suggests that with sufficient prior knowledge, the model can quickly adapt to a task even with a small amount of data.

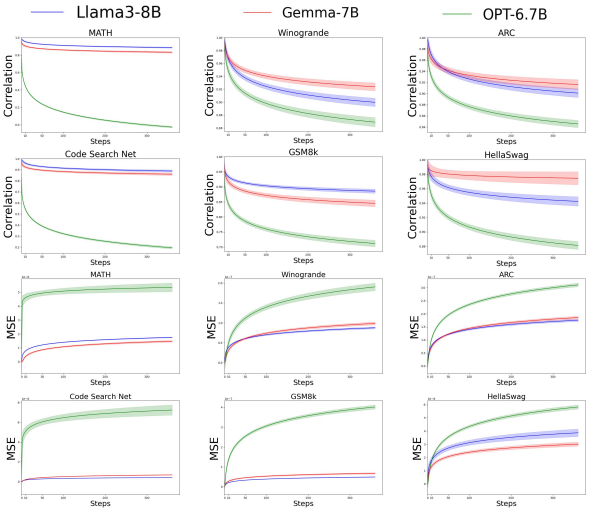

Figure 5: Tracking changes in correlation coefficient and MSE activation patterns of the Llama3-8B, Gemma-7B, and OPT-6.7B models during fine-tuning on datasets including Code Search Net, GSM8k, MATH, SGSM, ARC, HellaSwag, and Winogrande.

4.3.1 Analytical Method

To analyze changes in instruction activation patterns during SFT, we saved checkpoints throughout the process for different tasks. We measured the similarity and distance between the activation patterns at different checkpoints using two metrics: Correlation Coefficient and Mean Squared Error (MSE). Specifically, we calculated the Correlation and MSE between the activation patterns of the model at the i-th step of SFT on a given task and the activation patterns of the base model on the same task. The Correlation reflects the consistency in activation pattern changes, while MSE captures the magnitude of these changes.

4.3.2 Experiments and Results

As shown in Figure 5, activation patterns change rapidly in the initial steps of SFT. This suggests that fine-tuning with small datasets can significantly reshape a model's attention patterns and alter its performance. Previous research (Xia et al., 2024) supports this, indicating that rapid learning during SFT is driven by swift changes in attention patterns.

Moreover, the extent of changes in activation patterns correlates with improvements in model performance. As shown in Figure 5, the OPT-6.7B model shows significant changes in activation patterns across most tasks, particularly for complex tasks like MATH. The shifts in activation patterns post-fine-tuning are more pronounced compared to those in the Llama3-8B and Gemma-7B mod-

els. This suggests that the initial activation patterns of OPT-6.7B may not be well-suited for solving complex problems like MATH, and thus need more instances to approach the convergence.

By analyzing models with varying capabilities, we find that for simpler tasks, fewer steps, samples, and less SFT data are needed for stronger LLMs to approach convergence and adapt to the task. In contrast, weaker models require more data due to having less prior knowledge from pre-training. This suggests that the foundation of rapid generalization is sufficient prior knowledge. Consequently, with enough prior knowledge, a small number of samples can enable rapid learning of complex tasks. Given that complex tasks can be decomposed into simpler ones, this raises the question: **Once a model has sufficient prior knowledge, can it be fine-tuned with just a small number of instructions to rapidly learn complex tasks?**

5 Deduction and Applications

We further validate our conclusions by testing their applicability in enhancing the effectiveness of SFT. Based on our observations of activation patterns, we focus on two scenarios: (1) Can we improve SFT on complex tasks by providing LLMs with prior knowledge of the basic tasks that make up the complex task? (2) Can we identify relevant data from a large candidate pool to approximate the effect of a target dataset, particularly when the target dataset is unavailable?

5.1 Scenario 1: Combining Data of Basic Tasks to Promote the Learning of Complex Task

Our analysis indicates that LLMs learn to complete complex tasks by leveraging a combination of their fundamental capabilities. A natural question arises: for a complex task, can we enhance the efficiency and effectiveness of the learning process by equipping LLMs with relevant basic capabilities? We conduct experiments on these settings: (1) Aiming at adapting LLMs to a complex task with a limited number of instructions given; (2) A large number of instructions about various basic tasks are available; (3) The relationship between the complex task and basic tasks are unknown.

Since the relationships between the complex task and basic tasks are unknown, we first estimate these relationships using the regression function described in Equation 3. Intuitively, the regression coefficients α_i describe the proportion that the complex task can be "composed " of a basic task i. Based on the regression coefficients, we create a preliminary instruction set by combining instructions from basic tasks.

$$\text{Dataset}^{pre} = \{N \times \alpha_i / \sum_i \alpha_i\}_{i=0}^{|B|} \quad (4)$$

where $|B|$ is the number of candidate basic tasks, N is the predefined size of Dataset^{pre}. During experiment, LLM is first trained on Dataset^{pre}, then finetuned on the target task dataset Dataset$^{\text{complex}}$.

Model	Base	SFT	Random	Ours
Llama-7B	28.68	31.78	33.82	**36.82**
Llama2-7B	29.07	36.43	34.51	**40.70**
Llama3-8B	50.39	52.33	51.16	**55.03**

Table 2: Performance on the Mathbench test set: "Base" refers to the base model results, "SFT" refers to results after fine-tuning with 100 Mathbench samples, and "Random" represents the average of five runs trained with a randomly mixed dataset of GSM8K and RefGPT data, followed by fine-tuning with 100 Mathbench samples.

5.1.1 Experiment Settings

To simulate scenarios where LLMs lack sufficient preliminary knowledge, we conducted experiments on the Llama series models, which have not been specifically optimized for Chinese corpora and exhibit relatively limited performance on Chinese-related tasks. In this section, we set the target complex task as MathBench (Liu et al., 2024), which requires the model to solve school mathematics questions using Chinese. The MathBench dataset 358 instances, and we use only 100 instances as Dataset$^{\text{complex}}$, the left part for testing. In addition to the basic tasks described in Section 4.2, we include RefGPT (Yang et al., 2023a) (a dataset of Chinese factual knowledge) as a basic task dataset. After the regression process, we keep only basic datasets with the Top 2 largest regression coefficients, so as to filter out the noises. The remaining basic tasks are GSM8K (English elementary math) and RefGPT.Experiments are conducted on models of similar size but varying capabilities: Llama-7B (Touvron et al., 2023a), Llama2-7B (Touvron et al., 2023b), and Llama3-8B (Grattafiori et al., 2024).

5.1.2 Analysis

As shown in Table 2, our approach consistently outperforms baseline models across all evaluations, including models fine-tuned solely on the Mathbench training set and those trained with a random mix of elementary mathematics and Chinese instruction data. Compared to models fine-tuned only on Mathbench, our method shows significant improvements in handling complex tasks through subtask training. Moreover, constructing datasets based on activation patterns improves the model's ability to leverage simple task data for complex tasks, surpassing models trained on randomly mixed subtasks. For Llama-7B, learning from randomly proportioned basic task data can still improve complex task performance, but for more powerful models like Llama2-7B and Llama3-8B, this approach may hinder performance. These findings suggest that guiding dataset construction using activation patterns from basic tasks significantly enhances the model's ability to learn and generalize to complex tasks, supporting our observation that complex tasks could be a combination of basic tasks, and once equipped with the necessary preliminary knowledge, the LLM could quickly adapt to the complex task.

5.2 Scenario 2: Selecting Relevant Data from Candidates

While various Domain-LLMs demonstrate strong capabilities, their instruction data is often unavailable. Considering the task-specificity of activation patterns, can we select relevant datasets from open instruction sets to approximate the performance of models trained on private domain datasets? Thus, we conduct experiments under the following settings: (1) only a small amount of developer-constructed pseudo-private data is available; (2) the complete dataset used to train the Domain-LLM is unknown;(3) the public data pool is large enough to contain data relevant to the target task.

5.2.1 Experiment Settings

We simulated LLM performance without target data using the multi-domain MMLU dataset, designating test set data from specific domains as private (target data) and treating the remaining as non-target data. We used validation and dev set data from the target domains as pseudo-private data, simulating small, manually created datasets that developers might construct when original data is unavailable. As concluded in Section 4.1, fine-tuning on pseudo-private data improves the model's task specificity in the target domain.

Model	Base	T100	R100	T300	R300	T500	R500
Llama-7B	30.00	31.66	30.33	36.15	32.81	**39.09**	34.68
Llama2-7B	35.08	39.05	37.06	40.51	37.62	**40.70**	37.95
Llama3-8B	56.75	57.39	56.19	**57.95**	56.37	57.51	56.19

Table 3: Average accuracy of the model on four tasks: mathematics, physics, chemistry, and biology. "Base" represents the baseline model's average accuracy on these tasks. "T100" indicates the accuracy after fine-tuning with 100 most similar data points based on activation patterns. "R100" represents the accuracy after fine-tuning with 100 randomly selected unrelated MMLU data points, with similar meanings for "T300", "R300", "T500", and "R500".

We used Equation 5 to identify the top m non-target data points from MMLU most similar to the pseudo-private data. These data points were then used for further training. We compared the performance of this approach with the base model and models trained on an equal amount of randomly selected non-target data.

To ensure robustness, we averaged the results across multiple domains—mathematics, physics, chemistry, and biology—treating them as target data.

$$\text{Dataset}^{app} = \text{Top}_m \, Corr(AP^{tar}, AP^{D_i}), \quad (5)$$

where Dataset^{app} represents the entire dataset, D_i is the i-th sample from D, AP^{D_i} is the activation pattern for the i-th sample after SFT, and $Corr(*, *)$ is the correlation coefficient.

5.2.2 Analysis

As shown in Table 3, models trained with data selected by activation pattern similarity consistently outperform those trained with randomly selected data. This suggests that using the similarity of activation patterns, we can effectively identify data that share similarities with the private data. These results support our hypothesis that attention heads carry task-specific information, enabling LLMs to adapt to various tasks. They also demonstrate the potential for identifying rare relevant data from large datasets to approximate private datasets.

6 Conclusion

This study investigated the mechanisms behind the rapid learning and generalization observed during SFT. We found that attention heads are selectively activated in a task-specific manner during SFT, and

that the activation patterns for complex tasks can be decomposed into combinations of basic task patterns. This offers a viable data substitution strategy when high-quality complex data is scarce.

Our experiments demonstrated that even small amounts of data can significantly alter activation patterns, validating the feasibility of (1) pretraining on simple tasks to equip large models with essential knowledge, and (2) fine-tuning with selected relevant data to enable rapid generalization. The improved effectiveness observed in these scenarios further supports the validity of our analytical approach and offers practical solutions to data challenges in complex and specialized tasks.

7 Limitations

While this study demonstrates the role of attention head activation patterns in rapid learning and generalization during the SFT process, we did not explore the specific impact of individual attention head activation levels on model performance in detail. Additionally, our method validation was primarily conducted on simpler tasks, and its applicability to more complex and real-world tasks remains to be fully evaluated. Future work will focus on finer-grained analysis of activation levels and further testing the effectiveness of our approach in more challenging and practical domains to enhance its performance in real-world applications.

8 Acknowledgements

The research in this article is supported by the New Generation Artificial Intelligence of China (2024YFE0203700), National Natural Science Foundation of China under Grants U22B2059 and 62176079.

References

Bryan R Christ, Jonathan Kropko, and Thomas Hartvigsen. 2024. Mathwell: Generating educational math word problems at scale. *arXiv preprint arXiv:2402.15861*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *Preprint*, arXiv:1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023. How abilities in large language models are affected by supervised fine-tuning data composition. *arXiv preprint arXiv:2310.05492*.

Clémentine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. 2024. Open llm leaderboard v2. https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard.

Zihao Fu, Haoran Yang, Anthony Man-Cho So, Wai Lam, Lidong Bing, and Nigel Collier. 2023. On the effectiveness of parameter-efficient fine-tuning. In *Proceedings of the AAAI conference on artificial intelligence*, pages 12799–12807.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2021. Self-attention attribution: Interpreting information interactions inside transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12963–12971.

Bingxiang He, Ning Ding, Cheng Qian, Jia Deng, Ganqu Cui, Lifan Yuan, Huan-ang Gao, Huimin Chen, Zhiyuan Liu, and Maosong Sun. 2024. Zero-shot generalization during instruction tuning: Insights from similarity and granularity. *arXiv preprint arXiv:2406.11721*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. 2019. Codesearchnet challenge: Evaluating the state of semantic code search. *arXiv preprint arXiv:1909.09436*.

Zhuoran Jin, Pengfei Cao, Hongbang Yuan, Yubo Chen, Jiexin Xu, Huaijun Li, Xiaojian Jiang, Kang Liu, and Jun Zhao. 2024. Cutting off the head ends the conflict: A mechanism for interpreting and mitigating knowledge conflicts in language models. *arXiv preprint arXiv:2402.18154*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Samir Khaki and Konstantinos N Plataniotis. 2024. The need for speed: Pruning transformers with one recipe. *arXiv preprint arXiv:2403.17921*.

Eunho Lee and Youngbae Hwang. 2024. Automatic channel pruning for multi-head attention. *arXiv preprint arXiv:2405.20867*.

Hongwei Liu, Zilong Zheng, Yuxuan Qiao, Haodong Duan, Zhiwei Fei, Fengzhe Zhou, Wenwei Zhang, Songyang Zhang, Dahua Lin, and Kai Chen. 2024. Mathbench: Evaluating the theory and application proficiency of llms with a hierarchical mathematics benchmark. *Preprint*, arXiv:2405.12209.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023. Label words are anchors: An information flow perspective for understanding in-context learning. *arXiv preprint arXiv:2305.14160*.

Yongjie Wang, Tong Zhang, Xu Guo, and Zhiqi Shen. 2024a. Gradient based feature attribution in explainable ai: A technical review. *arXiv preprint arXiv:2403.10415*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024b. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Preprint*, arXiv:2406.01574.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. Less: Selecting influential data for targeted instruction tuning. *arXiv preprint arXiv:2402.04333*.

Dongjie Yang, Ruifeng Yuan, YuanTao Fan, YiFei Yang, Zili Wang, Shusen Wang, and Hai Zhao. 2023a. Refgpt: Reference -> truthful & customized dialogues generation by gpts and for gpts. *Preprint*, arXiv:2305.14994.

Ruo Yang, Binghui Wang, and Mustafa Bilgic. 2023b. Idgi: A framework to eliminate explanation noise from integrated gradients. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23725–23734.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023. Scaling relationship on learning mathematical reasoning with large language models. *Preprint*, arXiv:2308.01825.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *Preprint*, arXiv:1905.07830.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Hanyu Zhao, Li Du, Yiming Ju, Chengwei Wu, and Tengfei Pan. 2024. Beyond iid: Optimizing instruction learning from the perspective of instruction interaction and dependency. *arXiv preprint arXiv:2409.07045*.

A Dataset Details

In this paper, we utilize a wide range of datasets related to LLMs, many of which are commonly used in prominent large model evaluation frameworks (Fourrier et al., 2024; Contributors, 2023).

- MATH (Hendrycks et al., 2021b): An advanced mathematical reasoning dataset.
- GSM8K (Cobbe et al., 2021): A dataset for elementary mathematical reasoning.
- Code Search Net (Husain et al., 2019): A large dataset of code sourced from GitHub; we specifically use the Python subset in this paper.
- SGSM (Christ et al., 2024): A dataset for solving math problems using code, requiring strong code generation and mathematical reasoning abilities.
- HellaSwag (Zellers et al., 2019): A dataset for natural language processing and reasoning tasks.
- Winogrande (Sakaguchi et al., 2021): Another dataset for natural language processing and reasoning tasks.
- ARC (Clark et al., 2018): A dataset for natural language processing and reasoning tasks.
- Infinity Instruct dataset (Zhao et al., 2024): A bilingual dialogue dataset; we use the English portion in this study. This dataset is annotated with the specific skills required to complete each dialogue, allowing us to determine which capabilities are needed for each task.
- MathBench (Liu et al., 2024): A dataset for evaluating mathematical performance in large models. We use the Chinese portions of the middle and primary subsets.
- RefGPT (Yang et al., 2023a): A Chinese dataset focused on factual knowledge.
- MMLU (Hendrycks et al., 2021a): A dataset containing data from 57 domains. Following the approach of Wang et al. (2024b), we use "college mathematics," "abstract algebra," "elementary mathematics," "high school statistics," and "high school mathematics" to represent math-related data; "college physics," "high school physics," and "conceptual physics" for physics-related data; "college chemistry" and "high school chemistry" for chemistry-related data; and "college biology" and "high school biology" for biology-related data.

B Training Details

- All experiments were conducted on an NVIDIA A100 GPU cluster.
- The training was performed using the DeepSpeed framework, following the hyperparameters outlined in Yuan et al. (2023). We set ZeRO to 2, used a batch size of 4, a maximum token length of 1024, a learning rate of 1e-6, and gradient accumulation set to 4. Full fine-tuning was conducted with bf16 precision.
- Given the varying sizes of the datasets, to ensure consistency within each experiment, the number of training samples was set to match the smallest dataset in the group, ensuring that the results were not influenced by the amount of data.
- To ensure fair evaluation, all assessments were conducted using the OpenCompass framework (Contributors, 2023), employing greedy search to eliminate randomness in the results.

C Equation Details

MSE represents the average distance between two matrices. The larger the MSE, the greater the difference between the two matrices.

$$\text{MSE}(\mathbf{AP}^{\text{task1}}, \mathbf{AP}^{\text{task2}}) = \frac{1}{L \times H} \sum_{l=1}^{L} \sum_{h=1}^{H} (\mathbf{AP}_{l,h}^{\text{task1}} - \mathbf{AP}_{l,h}^{\text{task2}})^2, \tag{6}$$

where $\mathbf{AP}^{\text{task1}}$ and $\mathbf{AP}^{\text{task2}}$ represent the activation pattern matrices for tasks 1 and 2, respectively, and L and H represent the number of layers and the number of heads per layer, respectively.

The Gini coefficient, which measures the inequality within the data of a matrix $\mathbf{AP}$, can be expressed as:

$$G(\mathbf{AP}) = \frac{\sum_{i=1}^{L} \sum_{j=1}^{H} \sum_{k=1}^{L} \sum_{l=1}^{H} |\mathbf{AP}_{ij} - \mathbf{AP}_{kl}|}{2LH \sum_{i=1}^{L} \sum_{j=1}^{H} \mathbf{AP}_{ij}} \tag{7}$$

Coefficient of Variation The Coefficient of Variation (CV), which measures the relative variability of the data in matrix $\mathbf{AP}$, is given by:

$$CV(\mathbf{AP}) = \frac{\sigma(\mathbf{AP})}{\mu(\mathbf{AP})}, \tag{8}$$

where $\sigma(\mathbf{AP})$ is the standard deviation of $\mathbf{AP}$, and $\mu(\mathbf{AP})$ is the mean of $\mathbf{AP}$.

The Kurtosis, which measures the peakedness of the data distribution in matrix $\mathbf{AP}$, can be expressed as:

$$K(\mathbf{AP}) = \frac{LH \sum_{i=1}^{L} \sum_{j=1}^{H} (\mathbf{AP}_{ij} - \mu(\mathbf{AP}))^4}{\left(\sum_{i=1}^{L} \sum_{j=1}^{H} (\mathbf{AP}_{ij} - \mu(\mathbf{AP}))^2 \right)^2}, \tag{9}$$

where $\mu(\mathbf{AP})$ is the mean of the matrix $\mathbf{AP}$.

These expressions can be used in a LaTeX document to calculate the Gini coefficient, Coefficient of Variation, and Kurtosis for the matrix $\mathbf{AP}$.

We use R^2 to analyze the task fitting performance:

$$R^2 = 1 - \frac{\sum_{l=1}^{L} \sum_{h=1}^{H} \left(\Delta AP_{lh}^{\text{complex}} - \Delta \hat{AP}_{lh}^{\text{complex}} \right)^2}{\sum_{l=1}^{L} \sum_{h=1}^{H} \left(\Delta AP_{lh}^{\text{complex}} - \overline{\Delta AP^{\text{complex}}} \right)^2} \tag{10}$$

The coefficient of determination, denoted as R^2, is a statistical measure that indicates how well the data fits a regression model. In this context, $\Delta AP_{lh}^{\text{complex}}$ represents the observed change in activation patterns at layer l and head h for a complex task, while $\Delta \hat{AP}_{lh}^{\text{complex}}$ denotes the corresponding predicted change. The numerator captures the sum of squared errors between the observed and predicted values, whereas the denominator reflects the total variance in the observed data. $\overline{\Delta AP^{\text{complex}}}$ represents the mean of the observed changes in activation patterns.