Brain-Inspired Exploration of Functional Networks and Key Neurons in Large Language Models

Anonymous ACL submission

Abstract

In recent years, the rapid advancement of large language models (LLMs) in natural language processing has sparked significant interest among researchers to understand their mechanisms and functional characteristics. Although prior studies have attempted to explain LLM functionalities by identifying and interpreting specific neurons, these efforts mostly focus on individual neuron contributions, neglecting the fact that human brain functions are realized through intricate interaction networks. Inspired by research on functional brain networks (FBNs) in the field of neuroscience, we utilize similar methodologies estabilished in FBN analysis to explore the "functional networks" within LLMs in this study. Experimental results highlight that, much like the human brain, LLMs exhibit certain functional networks that recur frequently during their operation. Further investigation reveals that these functional networks are indispensable for LLM performance. Inhibiting key functional networks severely impairs the model's capabilities. Conversely, amplifying the activity of neurons within these networks can enhance either the model's overall performance or its performance on specific tasks. This suggests that these functional networks are strongly associated with either specific tasks or the overall performance of the LLM. Our study provides novel insights into the interpretation of LLMs.

1 Introduction

In recent years, large language models (LLMs) have become a focal point of research in the field of artificial intelligence (AI) due to their remarkable capabilities in natural language processing (Zhao et al., 2024a, 2023; Liu et al., 2023b; Wang et al., 2024; Liu et al., 2025). However, these models are often considered "black boxes", with insufficient understanding of their internal mechanisms. Establishing methods to explain and understand LLMs is essential both for improving model transparency and trustworthiness and for establishing a foundation to develop more efficient and reliable AI systems.

One research direction on the mechanistic interpretability of LLMs focuses on the functional role of individual neurons (Yu and Ananiadou, 2024a; Dai et al., 2022; Yu and Ananiadou, 2024b; Niu et al., 2024; Chen et al., 2024). Prior studies have shown the specific functional roles of LLM neurons (AlKhamissi et al., 2024; Wang et al., 2022). For example, some neurons may specialize in processing linguistic structures, while others might be responsible for reasoning tasks (Huo et al., 2024; Zhao et al., 2024b). Removal of certain neurons leads to a significant degradation in model performance. In addition, by manipulating certain neurons (for example, amplifying their output signals (Song et al., 2024; Duan et al., 2025)), it is possible to enhance an LLM's performance on specific tasks. This suggests that some neurons are strongly associated with particular tasks.

Methods for identifying key neurons within LLM can be categorized into several categories. These include analyzing the gradients of neurons to evaluate their impact on model predictions (Sundararajan et al., 2017; Lundstrom et al., 2022), employing causal tracing techniques to uncover the causal relationships that influence model behavior (Nikankin et al., 2024), and conducting statistical analyzes of activated neurons to measure their information content and variability (AlKhamissi et al., 2024; Song et al., 2024; Tang et al., 2024). These approaches provide valuable tools for understanding and explaining LLMs, offering deeper insight into their inner mechanisms.

However, the function of an individual neuron in human brain is much more complex than it might initially seem. Neurons in human brain often form functional networks through their interactions and connectivity, collaboratively working to perform

higher-level cognitive tasks (Smith et al., 2009; Bullmore and Sporns, 2009). The role of a neuron therefore extends beyond its individual activation patterns and is shaped by its cooperation with other neurons within these networks (Bullmore and Sporns, 2009; Liu et al., 2024b,a). In this context, current key neuron identification studies largely overlook the functional network perspective and fail to consider the coordinated roles of neurons. As a result, these limitations have hindered a deeper understanding of neuronal function, neglecting the insights offered by neuroscience research on functional brain networks (FBNs) (Hassabis et al., 2017; Vilas et al.).

In this study, we draw inspiration from neuroscience to investigate whether LLMs contain functional networks similar to those found in the human brain. By recognizing the similarities between functional magnetic resonance imaging (fMRI) (Matthews and Jezzard, 2004; Logothetis, 2008) signals and the output signals of neurons in LLM, we hypothesize that the techniques used in fMRI analysis could be adapted to analyze LLM neurons. Specifically, we treat the neuron outputs from the multilayer perception (MLP) layers of LLMs as analogous to fMRI signals and applied Independent Component Analysis (ICA) (Hyvärinen and Oja, 2000; Beckmann et al., 2005; Varoquaux et al., 2010b) to decompose these neuron outputs into multiple functional networks.

Our experiments on extensive datasets confirmed the existence of functional networks within LLMs, conceptually mimicking deriving FBNs from fMRI data (Mensch et al., 2016; Varoquaux et al., 2010a; Liu et al., 2023a; He et al., 2023; Ge et al., 2020; Lv et al., 2015). Some functional networks exhibit highly consistent spatial organization across diverse inputs and play a critical role in model functionality. Inhibition of specific key networks (typically comprising fewer than 2% of the model's neurons) significantly degrades performance, while amplifying these critical networks can improve the performance of the model in a specific task or overall.

Our contributions are summarized as follows:

1. Inspired by functional brain network analysis in neuroscience, we have introduced an analytical framework based on Independent Component Analysis (ICA) to explore "functional networks" within LLMs. This approach moves beyond treating activations as undifferentiated representations and instead reveals structured, distributed subnetworks that consistently co-activate across inputs.

2. We have demonstrated that LLMs exhibit functional networks, which share a notable similarity with the human brain in that both demonstrate stable functional patterns.

3. We have demonstrated that neurons within these functional networks are strongly associated with specific tasks and are essential to maintain the functionality of LLMs.

2 Preliminaries

Transformer, MLP layer: LLMs utilized in this study are based on the transformer architecture (Vaswani, 2017), specifically employing a decoder-only configuration (Radford et al., 2018, 2019; Brown et al., 2020; Yang et al., 2024b; GLM et al., 2024). In this configuration, each transformer decoder consists of two primary components: a multi-head self-attention module and an MLP module. Typically, an MLP module consists of two fully connected layers. The first layer increases the dimensionality, often to four times the original dimension, followed by a non-linear activation function. The second layer then reduces the dimensionality back to its original size. In our study, we focus on the neurons located in the final MLP layer of each decoder module within the model. Given an input vector $\mathbf{x} \in \mathbb{R}^{d_{\text{model}}}$, the MLP module can be represented as follows:

$$\text{MLP}(\mathbf{x}) = \mathbf{W}_2 \cdot \sigma(\mathbf{W}_1 \cdot \mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2 \quad (1)$$

$\mathbf{W}$ is the weight matrix of the linear transformation, $\mathbf{b}$ is the bias vector of the linear transformation, σ is a non-linear activation function. The neuron outputs used in this paper are the outputs of the $\text{MLP}(\mathbf{x})$. The output of neurons in the MLP layer is also the output of the last component of the Transformer block, which can better reflect the information processing results of the entire module, possessing stronger integrity and representativeness.

Functional Brain Networks, FBNs: FBNs refer to collections of brain regions that are co-activated during specific tasks or while at rest (Dong et al., 2020). FMRI is a non-invasive technique used to measure Blood-Oxygen Level Dependent (BOLD) signals, which reflect neuronal activity indirectly (Matthews and Jezzard, 2004; Logothetis, 2008). The intensity of voxel values in fMRI signals indirectly reflects neuronal activity

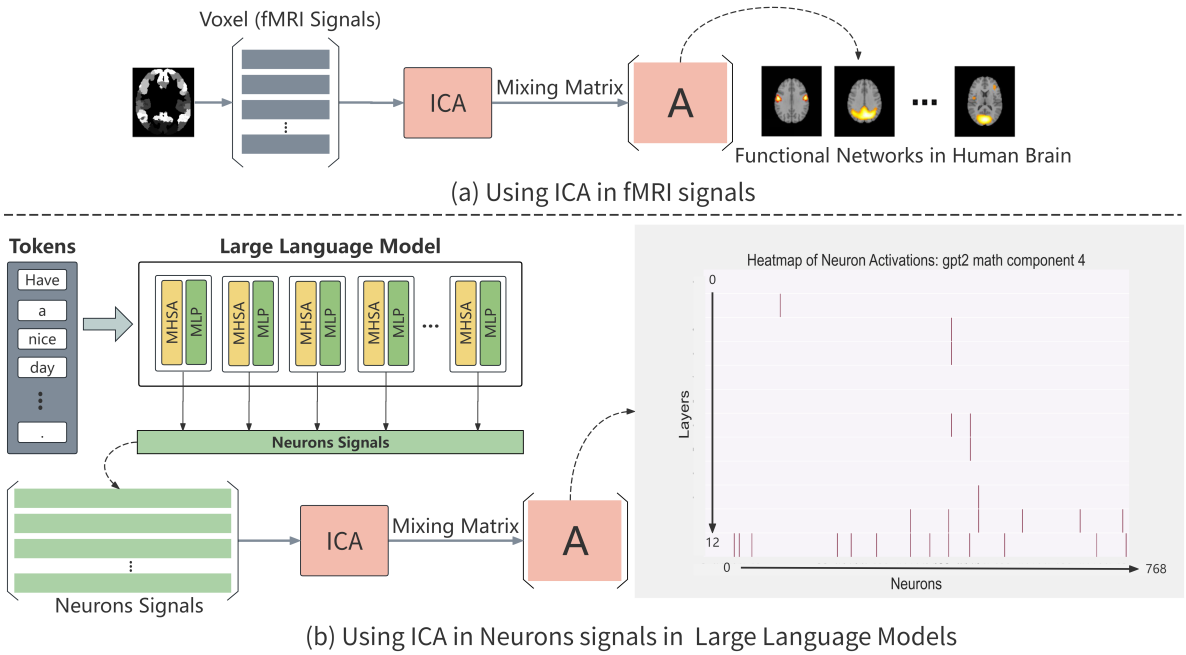

Figure 1: The framework of brain inspired exploration of functional networks within LLMs. (a) Identifying functional brain networks from whole-brain fMRI signals via ICA. (b) Identifying functional networks from responses of artificial neurons within LLMs.

by capturing variations in local blood oxygen levels due to neural metabolism. Neuroscientific research hypothesizes that the observed BOLD signals in fMRI signals are likely the result of multiple independent functional networks working together. Essentially, these BOLD signals can be considered as linear combinations of several source signals, each representing a distinct functional network.

Independent Component Analysis, ICA: ICA is a powerful data-driven technique used to extract source signals that are as statistically independent as possible from a mixed signal. In the field of neuroscience, ICA is widely applied to fMRI data to uncover underlying FBNs (Hyvärinen and Oja, 2000). It disentangles mixed fMRI signals into several independent components, where each component represents a distinct functional network (Varoquaux et al., 2010b). Each extracted independent component is associated with a spatial map that illustrates which brain regions contribute to that component. These regions typically display synchronized activities, indicating their coordinated involvement in specific neural processes.

The objective of ICA is to recover the source signals $\mathbf{S}$ from the observed signals $\mathbf{X}$. Suppose that we have n observed signals $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$, which are linear mixtures of m independent source signals $[\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_m]$. The relationship between the observed signals $\mathbf{X}$ and the source signals $\mathbf{S}$ can be expressed as:

$$\mathbf{X} = \mathbf{AS} \tag{2}$$

where $\mathbf{A}$ is the mixing matrix that describes how the source signals are combined to produce the observed signals. Each row of the linear mixing matrix represents the spatial pattern of the corresponding functional network. In this study, the functional networks derived from LLMs neuron signals refer to the rows of the linear mixing matrix $\mathbf{A}$, which indicate the set of neurons that are consistently co-activated under different conditions.

3 Method

3.1 Datasets and Models

Five pre-trained LLM models, including GPT2 (Radford et al., 2019), Qwen2.5-3B-Instruct (Yang et al., 2024a), Qwen2.5-7B-Instruct (Yang et al., 2024a), ChatGLM3-6B-base (GLM et al., 2024) were used to validate the proposed method. The

consistency and variations of the functional networks within these LLM models when processing various external stimuli were assessed by exposing them to text samples from each of four datasets, including the AGNEWS dataset (Zhang et al., 2015), encyclopedia entries from Wikitext2 (Merity et al., 2016), mathematical texts from MathQA (Amini et al., 2019), and code snippets from the CodeNet dataset (Puri et al., 2021). In addition, some benchmark datasets including SQUAD (Rajpurkar et al., 2016), GLUE (Wang, 2018), and AGNEWS (Zhang et al., 2015) were used to test the performance of the LLM models after editing the key functional networks identified in LLMs.

3.2 Artificial Neurons and Neuron Signals within LLMs

There is a certain similarity between fMRI signals and the neuron signals in LLMs. fMRI signals reflect the activity of biological neurons in the brain when subjected to external stimuli or under resting state, while the neuron signals of artificial neurons in LLMs represent the internal state changes of the model as it processes input information. Both fMRI signals and the neuron signals in LLMs exhibit temporal characteristics. In fMRI, different stimuli trigger activity changes in different regions of the brain at different time points. Similarly, in LLMs, different inputs causes variations in neuron output at different time steps. This temporal characteristic allows us to analyze these signals dynamically, looking for coordinated activities and regularities within them.

To extract functional networks from LLMs, the first step is to define artificial neurons (ANs) in LLMs, analogous to voxels in fMRI data. In this study, we utilize the neurons in the last layer of the MLP module in each Transformer blocks (Fig. 1). Afterwards, the neuron signal of an AN, analogous to the fMRI signal of a voxel, is collected as the responses of the AN subjected to text inputs.

3.3 Identifying Functional Networks within LLMs via ICA

Both fMRI signals and neuron signals in LLMs can be viewed as mixtures of multiple independent components. In the brain, different neural activity patterns combine to form complex fMRI signals. In LLMs, interactions among different functional modules result in the output signals being a superposition of multiple independent components. ICA is specifically designed for such signal-mixing scenarios and can effectively separate these independent components. In fMRI analysis, the independent components separated by ICA correspond to different functional networks in the brain (Fig. 1(a)), such as the visual network or auditory network. Likewise, in LLMs, the independent components separated by ICA may correspond to different functional modules within the model (Fig. 1(b)). Therefore, ICA can help reveal the functional organization inside LLMs.

3.4 Validation and Evaluation

3.4.1 Consistency of Functional Networks within LLMs

Identifying FBNs in fMRI data typically involves performing ICA on individual-level and group-level. In individual level analysis, ICA is applied to single-subject fMRI data. In contrast, in group level analysis, ICA is performed on data from multiple subjects to derive common brain networks shared by multiple subjects. Individual-level ICA captures subject-specific functional networks that are either spontaneous or task-evoked, offering high individuality but suffering from unstable component number and interpretation across subjects. Group-level ICA, in contrast, identifies reproducible, shared functional networks across subjects, providing greater stability and comparability. We use group-level ICA to extract stable, interpretable, and comparable functional network templates.

Following the similar strategy, we randomly selected 100 samples from each dataset, analogous to select 100 subjects in group-level fMRI analysis. We then randomly selected another 100 samples from the same dataset for individual-level analysis. By comparing functional networks derived from individual-level and group-level ICA, we investigate whether similarly stable "functional networks" also exist in LLMs.

The functional networks derived from the group-level ICA analysis can be considered as templates representing a set of common functional networks. We then assessed the spatial similarity between these templates and the functional networks derived from the individual-level analysis to evaluate the consistency and variations of functional networks in LLMs. Here, we adopted intersection over union (IoU), , which is commonly used in neuroscience studies, as a metric to measure the spatial similarity between two functional networks $N^{(1)}$ and $N^{(2)}$. The IoU is defined as follows:

$$IoU(N^{(1)}, N^{(2)}) = \frac{\sum_{i=1}^{n} |N_i^{(1)} \cap N_i^{(2)}|}{\sum_{i=1}^{n} |N_i^{(1)} \cup N_i^{(2)}|} \quad (3)$$

where n represents the number of neurons in the corresponding functional network. It is worth noting that, similar to comparing functional networks in neuroscience, calculating spatial similarities between functional networks in LLMs also requires applying a threshold to the mixing matrix $\mathbf{A}$ to filter out neurons with lower activation values. We set the threshold as 3.6, a value commonly used in functional brain network analysis.

According to our experimental findings and experience in functional brain network analysis, two networks appear reasonably similar when the IoU between them exceeds 0.2. Consequently, such networks are usually classified into the same category of functional networks. In this study, a template of LLM functional network extracted in group level analysis is considered consistent if its counterpart (IoU greater than 0.2) in individual level analysis can be reliably identified, representing a consistent functional network within an LLM.

3.4.2 Functional Networks Editing

We conducted functional network editing experiments to investigate how targeted manipulation of neurons influences model behavior. Specifically, we first identified functional networks using ICA, then directly edited the corresponding neurons during inference, either by inhibiting them (setting their activations to zero) or amplifying them (scaling up their outputs). First, we edit these functional networks and applied on the same task and dataset to assess their task-specific role. Second, we applied the same neuron edits across different tasks and datasets to evaluate whether these functional networks support general, cross-task capabilities.

4 Results

4.1 Consistent Functional Networks within LLMs

In the experiment, we derived 10 functional network templates in group-level analysis and 64 functional networks in each of the 100 individual-level analysis. For each group level template, we counted the number of its counterparts (see 3.4 for details) in all 100 individual-level analysis, as summarized in Tables 1, 6, 7, 8 for GPT-2, Qwen2.5-3B-Instruct, Qwen2.5-7B-Instruct and ChatGLM3-6B-base, respectively (We include the Qwen and GLM results in the appendix.).

Table 1: The number of the counterparts in 100 individual-level analysis for the functional network templates in group-level analysis in GPT-2.

TEMPLATES	NEWS	WIKI	MATH	CODE
1	139	1848	219	1151
2	236	1830	165	1198
3	243	47	5	59
4	316	1722	116	1230
5	307	1301	74	1209
6	416	120	54	1352
7	16	1080	88	1473
8	107	529	114	959
9	92	37	125	915
10	330	450	190	975

We observed that, across all tested LLMs, most group-level functional network templates have corresponding counterparts in individual-level analysis. Given that 100 individual models were analyzed, each with 64 functional networks (totaling 6,400 per dataset), the widespread presence of group-level templates in individual analyses indicates strong reproducibility. In GPT-2 (Table 1), match counts are exceptionally high—some templates appear thousands of times (e.g., Template 7 in Code: 1,473 matches). This reflects both the robustness of the group-level templates and the relative functional simplicity of smaller models, where fewer, more dominant networks emerge consistently across individuals.

In contrast, for larger models like Qwen2.5-3B-Instruct (Table 6), Qwen2.5-7B-Instruct (Table 7), and GLM (Tables 8), the number of matches per template is generally lower, but this is not due to instability. Instead, as model scale increases, the total number of distinct functional networks grows, leading to greater functional diversity and specialization. Consequently, individual networks are more distributed across a larger repertoire, diluting the frequency of any single template. Nevertheless, key templates still emerge consistently across individuals, such as Template 7 in Qwen2.5-3B (Wiki: 89; Code: 62) and Template 10 in Qwen2.5-7B (Code: 119; Math: 191), demonstrating that stable, shared functional organizations persist even in larger models.

We next evaluate whether the same functional networks emerge across different tasks in LLMs, aiming to identify shared, task-invariant functional

Table 2: Spatial similarity between functional network templates in group-level analysis in Qwen2.5-7B-Instruct.

Template 1	Template 2	IoU
News Component 3	Code Component 2	0.8605
News Component 6	Code Component 3	0.7018
News Component 7	Code Component 2	0.7959
Code Component 9	Wiki Component 7	0.8837
Code Component 2	Math Component 1	0.8723
Code Component 2	Math Component 2	0.9091
Code Component 2	Math Component 3	0.8043
Code Component 2	Math Component 5	0.9070
Code Component 9	Math Component 10	0.9091
News Component 6	Wiki Component 7	0.7255
News Component 3	Math Component 1	0.7872
News Component 3	Math Component 2	0.8605
News Component 3	Math Component 3	0.8372
News Component 3	Math Component 5	0.9024
News Component 6	Math Component 10	0.6852
Wiki Component 7	Math Component 10	0.9302

structures. Table 2 shows the spatial similarity between group-level functional network templates in Qwen2.5-7B-Instruct across News, Wiki, Math and Code tasks.

The results reveal strong cross-task consistency in specific functional networks. These high IoU values indicate that certain functional networks are not task-specific but instead reappear across diverse domains. This parallels findings in human neuroscience, where core brain networks like the default mode network (Raichle and Snyder, 2007; Raichle, 2015) activate across a wide range of cognitive tasks.

4.2 Functional Networks Editing Experiments

4.2.1 Inhibiting Functional Networks

We first adopt the neuron-lesion experiment, we selectively deactivate functional networks within the LLMs to assess the effects of functional network removal on overall performance and functional behavior.

For Qwen2.5 and ChatGLM, we used lm eval framework and carefully crafted prompts to evaluate their performance in a zero-shot setting. In our assessments, we used accuracy, F1-score, Matthews Correlation Coefficient (MCC) and word perplexity as the performance metric.

Table 3 presents the performance results of the Qwen2.5-7B-Instruct model on the SST-2 dataset after selectively inhibiting specific functional networks derived in group-level analysis. The results indicate that inhibiting these functional networks leads to varying degrees of performance degradation. Notably, inhibiting just a few dozens of key

Table 3: Performance of Qwen2.5-7B-Instruct with inhibited functional networks on the SST-2 Dataset. The first column represents each functional network and the number of neurons associated with it. The second column to forth column shows the accuracy in SST-2, CoLA and MRPC. These functional networks derived from the SST-2 dataset. The model's accuracy under normal conditions (without lesion) is 0.9358 (SST-2), 0.6913 (CoLA), 0.6765 (MRPC) and 0.8448 (RTE).

Networks (100352)	Accuracy			
	SST-2	CoLA	MRPC	RTE
$1 \rightarrow 41$	0.0963	0.1074	0.0221	0.0072
$2 \rightarrow 336$	0.8085	0.6913	0.3799	0.5271
$3 \rightarrow 356$	0.8131	0.6913	0.5907	0.5271
$4 \rightarrow 43$	0.8417	0.6913	0.5294	0.5271
$5 \rightarrow 239$	0.8016	0.6913	0.5196	0.4982
$6 \rightarrow 40$	0.0401	0.0662	0.1838	0.0036
$7 \rightarrow 41$	0.1456	0.0518	0.2672	0.0361
$8 \rightarrow 41$	0.1892	0.1850	0.2059	0.0144
$9 \rightarrow 53$	0.8360	0.6913	0.3922	0.5271
$10 \rightarrow 45$	0.8314	0.6913	0.3333	0.5271

neurons can significantly impair the model's performance. In contrast, random neuron inhibiting has a negligible impact on model performance, as shown in Table 10 (see in the appendix). It is seen that even when up to 15% neurons are randomly inhibiting, the performance drop remains minimal. This finding is consistent with previous studies that have also observed that random neuron inhibiting does not substantially affect model performance (AlKhamissi et al., 2024; Song et al., 2024). Taken together, our experimental results emphasize the critical role of specific functional networks identified through ICA. These networks appear to encode essential information that is vital for the model's capability.

We also observed that some functional networks identified on the SST-2 dataset are not only crucial for SST-2 but also play an important role in other tasks. Inhibiting these networks not only degrades the model's performance on SST-2 but also leads to a decline in performance on other tasks. In contrast, networks that are less important for SST-2 may still be critical for other tasks. This observation suggests that the model may contain shared underlying structures or patterns that are not task-specific but instead capture general linguistic features or principles.

Table 10 and Table 11 (see in the appendix) present the performance results of Qwen2.5-7B-Instruct and ChatGLM3-6B-base by inhibiting neurons within specific functional networks. These

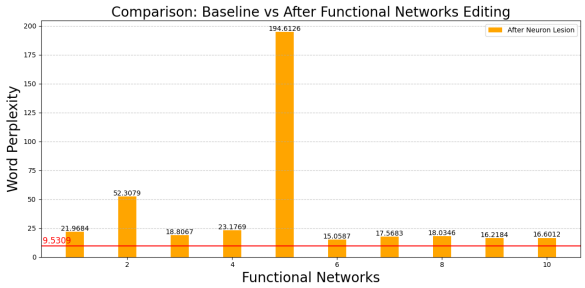

Figure 2: Perplexity results for Qwen2.5-7B-Instruct after inhibiting the neurons corresponding to each of 10 functional networks obtained from SST-2 dataset, respectively.

models rely on well-designed prompts to zero-shot generate appropriate text for various tasks. When functional networks are inhibited, the models' ability to generate coherent and task-relevant text is severely compromised.

In addition to evaluating zero-shot classification performance, we also assessed the language modeling capability of LLMs with inhibited functional networks using the Wikitext dataset, measured by word-level perplexity.

We conducted experiments on the Qwen2.5-7B-Instruct model, using the same 10 functional network templates previously derived from the SST-2 dataset. The results are shown in the Figure 2. After inhibiting different functional networks, the model's language modeling performance degraded to varying degrees, with functional network templates 2 and 5 causing the largest increases in perplexity.

4.2.2 Amplifying Functional Networks

To further investigate the functional roles of key neurons within these functional networks, we performed functional networks editing experiments by amplifying their activations. Specifically, we tested both additive and multiplicative editing strategies and found that multiplicative editing yielded the most stable and consistent effects. Our experiments show that scaling neuron activations by a factor of 1.02 yields the best results. Therefore, all experiments presented in this section use 1.02 as the multiplicative editing factor. Results of ablation studies on additive editing and other multiplicative factors can be found in the appendix.

We first verify whether the neurons identified by our method are indeed more critical. The results of amplifying the top 5% of functional network neurons in ChatGLM3-6B-base are shown in the Table 4. Our method selects the top 5% of neurons with the highest weights across 10 functional networks and amplifies their activations. As shown in Table 4, this leads to performance improvements on several tasks, and the degree of enhancement achieved by our method exceeds that of other approaches (LLM Localization (AlKhamissi et al., 2024)). Moreover, it can be observed that our method improves performance on the majority of tasks. Figure 3 shows the results of applying multiplicative editing to the neurons of one functional network. The evaluation metrics used are accuracy and F1-score. After editing, the average score is 0.7457, compared to 0.7451 for the unedited (normal) model.

Notably, editing this functional network leads to improved performance on SST-2. Among our 10 functional networks, only two enhance SST-2 performance, and these two exhibit relatively higher spatial similarity (reaching 0.1514) compared to all other pairs, whose maximum spatial similarity is only 0.0851.

We reasonably hypothesize that the neurons

Table 4: Performance of ChatGLM3-6B-base after amplifying top 5% Neurons.

Task / Metric	Normal	ICA	LLM Localization
CoLA (MCC)	0.1497	**0.1580**	0.1577
MNLI	0.5636	**0.5638**	0.5633
MRPC (F1)	0.8415	**0.8436**	**0.8436**
QNLI	0.8223	**0.8223**	0.8217
QQP (F1)	0.8065	0.8062	0.8064
RTE	0.7581	**0.7617**	0.7581
SST-2	0.9553	0.9553	0.9553
WNLI	0.6479	0.6338	0.6338
Average	0.6931	0.6931	0.6925
Wikitext (PPL)	10.1177	10.1168	10.1181

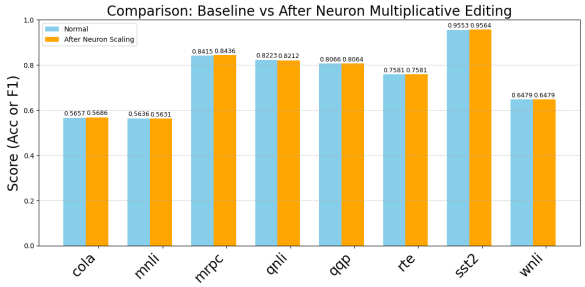

Figure 3: Zero-shot results for ChatGLM3-6B-base after amplifying the neurons corresponding to a functional network obtained from SST-2 dataset.

shared by these two functional networks are directly involved in the sentiment analysis task on SST-2. To verify this, we took the union of the two networks and amplified their activations, which successfully improved SST-2 performance. Furthermore, even when we amplified only the non-overlapping (network-specific) neurons (those not shared between the two networks) performance on SST-2 still increased. And amplifying neurons in the other functional networks does not improve SST-2 performance. This demonstrates that the neurons in both decomposed functional networks are meaningfully associated with sentiment analysis, rather than being redundant or irrelevant.

4.3 Do Different Methods Find the Same Important Neurons?

We further evaluated the consistency of important neurons identified by different methods by computing the spatial similarity (IoU) between the neuron sets derived by our ICA-based approach and those from other methods. For each method, we extracted the top 5% of neurons ranked by importance and calculated the IoU between these binary neuron masks.

Table 5: Spatial similarity between the important neurons derived by our ICA-base approach and LLM Localization (AlKhamissi et al., 2024).

Methods	ICA vs LLM Localization
Spatial Similarity (IoU)	0.0220

Table 5 shows the spatial similarity between the sets of important neurons identified by our method and those from LLM Localization. It can be observed that the two methods yield neuron sets with almost no spatial overlap, indicating that the important neurons they identify are largely distinct in location. Combined with the results in Table 4 and Table 9 (see in the appendix), our method identifies neurons that have a significantly greater impact on model performance than LLM Localization. This strongly supports the neuroscientific insight that functionally relevant computation in neural systems arises not from isolated neurons but from co-activated ensembles–that is, groups of neurons working together in coordinated patterns.

5 Discussion and Conclusion

This study introduces a framework for analyzing LLMs by adapting analytical methods originally developed in cognitive neuroscience to identify functional brain networks from neuroimaging data. Specifically, we apply group-level ICA to neuron activations in LLMs, treating them as high-dimensional signals analogous to fMRI time series.

Our data-driven approach reveals reproducible ensembles of co-activating neurons within the model's MLP layers. These ensembles form stable, task-general functional units: their collective activity persists across diverse inputs, and targeted perturbation of these groups leads to measurable degradation in representation quality. This suggests that LLMs rely on structured, coordinated functional networks for encoding semantic information, offering a system-level perspective that goes beyond isolated neuron interpretations.

Our work demonstrates that methods from cognitive neuroscience can be effectively repurposed to locate structured, task-relevant neuron ensembles in LLMs. These functional networks are not arbitrary collections of units; rather, they behave as integrated circuits, with constituent neurons jointly encoding specific semantic or computational functions. This coordinated activity persists across tasks and examples, suggesting that information processing in LLMs relies on the collective behavior of these ensembles. Recognizing this collaborative structure provides a more meaningful lens for interpreting model internals than isolated neuron analyses or heuristic importance metrics.

Although our study focuses on analysis rather than engineering applications, the clear functional role of these neuron ensembles opens a plausible avenue for future work, such as using functional networks as a guide for model compression and model fingerprint. For instance, preserving neurons within critical functional groups while simplifying or removing others could offer a more principled basis for pruning than current heuristics. The unique functional activity of neurons within functional networks, along with the functional connectivity between these networks, holds promise as a model fingerprint (similar to the conception "brain fingerprints" in neuroscience (Finn et al., 2015)) and can serve as an effective mechanism for protecting the intellectual property of large language models. Ultimately, this work underscores how cross-disciplinary frameworks, borrowing rigorously from neuroscience, can yield actionable insights into the inner mechanisms of LLMs.

Limitations

Our investigation focused solely on the MLP layers. Future work could extend this approach to other components within these models, such as attention mechanisms or embedding layers. By broadening the scope of analysis to include these additional modules, we can gain a more comprehensive understanding of how functional networks manifest in different components of LLMs.

The algorithm used in this research is ICA, which has various derivatives. These variants can improve model's performance based on the unique characteristics of the dataset. In addition to ICA, other approaches such as dictionary learning and deep learning–based autoencoders can also be employed to decompose the internal representations of LLMs into functional networks. Consequently, there is potential for the development of novel algorithms tailored specifically to the properties of LLMs.

References

Badr AlKhamissi, Greta Tuckute, Antoine Bosselut, and Martin Schrimpf. 2024. The llm language network: A neuroscientific approach for identifying causally task-relevant units. *arXiv preprint arXiv:2411.02280*.

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

Christian F Beckmann, Marilena DeLuca, Joseph T Devlin, and Stephen M Smith. 2005. Investigations into resting-state connectivity using independent component analysis. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 360(1457):1001–1013.

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

Ed Bullmore and Olaf Sporns. 2009. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature reviews neuroscience*, 10(3):186–198.

Yuheng Chen, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. 2024. Journey to the center of the knowledge neurons: Discoveries of language-independent knowledge neurons and degenerate knowledge neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

Qinglin Dong, Ning Qiang, Jinglei Lv, Xiang Li, Tianming Liu, and Quanzheng Li. 2020. Discovering functional brain networks with 3d residual autoencoder (resae). In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2020: 23rd International Conference, Lima, Peru, October 4–8, 2020, Proceedings, Part VII 23*, pages 498–507. Springer.

Xufeng Duan, Xinyu Zhou, Bei Xiao, and Zhenguang Cai. 2025. Unveiling language competence neurons: A psycholinguistic approach to model interpretability. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 10148–10157.

Emily S Finn, Xilin Shen, Dustin Scheinost, Monica D Rosenberg, Jessica Huang, Marvin M Chun, Xenophon Papademetris, and R Todd Constable. 2015. Functional connectome fingerprinting: identifying individuals using patterns of brain connectivity. *Nature neuroscience*, 18(11):1664–1671.

Bao Ge, Huan Wang, Panpan Wang, Yin Tian, Xin Zhang, and Tianming Liu. 2020. Discovering and characterizing dynamic functional brain networks in task fmri. *Brain Imaging and Behavior*, 14:1660–1673.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, and 1 others. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*.

Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. 2017. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245–258.

Mengshen He, Xiangyu Hou, Enjie Ge, Zhenwei Wang, Zili Kang, Ning Qiang, Xin Zhang, and Bao Ge. 2023. Multi-head attention-based masked sequence model for mapping functional brain networks. *Frontiers in Neuroscience*, 17:1183145.

Jiahao Huo, Yibo Yan, Boren Hu, Yutao Yue, and Xuming Hu. 2024. MMNeuron: Discovering neuron-level domain-specific interpretation in multimodal large language model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6801–6816, Miami, Florida, USA. Association for Computational Linguistics.

Aapo Hyvärinen and Erkki Oja. 2000. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430.

Yiheng Liu, Enjie Ge, Mengshen He, Zhengliang Liu, Shijie Zhao, Xintao Hu, Ning Qiang, Dajiang Zhu, Tianming Liu, and Bao Ge. 2024a. Mapping dynamic spatial patterns of brain function with spatial-wise attention. *Journal of Neural Engineering*, 21(2):026005.

Yiheng Liu, Enjie Ge, Zili Kang, Ning Qiang, Tianming Liu, and Bao Ge. 2024b. Spatial-temporal convolutional attention for discovering and characterizing functional brain networks in task fmri. *NeuroImage*, 287:120519.

Yiheng Liu, Enjie Ge, Ning Qiang, Tianming Liu, and Bao Ge. 2023a. Spatial-temporal convolutional attention for mapping functional brain networks. In *2023 IEEE 20th International Symposium on Biomedical Imaging (ISBI)*, pages 1–4. IEEE.

Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, and 1 others. 2023b. Summary of chatgpt-related research and perspective towards the future of large language models. *Meta-Radiology*, page 100017.

Yiheng Liu, Hao He, Tianle Han, Xu Zhang, Mengyuan Liu, Jiaming Tian, Yutong Zhang, Jiaqi Wang, Xiaohui Gao, Tianyang Zhong, Yi Pan, Shaochen Xu, Zihao Wu, Zhengliang Liu, Xin Zhang, Shu Zhang, Xintao Hu, Tuo Zhang, Ning Qiang, and 2 others. 2025. Understanding llms: A comprehensive overview from training to inference. *Neurocomputing*, 620:129190.

Nikos K Logothetis. 2008. What we can do and what we cannot do with fmri. *Nature*, 453(7197):869–878.

Daniel D Lundstrom, Tianjian Huang, and Meisam Razaviyayn. 2022. A rigorous study of integrated gradients method and extensions to internal neuron attributions. In *International Conference on Machine Learning*, pages 14485–14508. PMLR.

Jinglei Lv, Xi Jiang, Xiang Li, Dajiang Zhu, Hanbo Chen, Tuo Zhang, Shu Zhang, Xintao Hu, Junwei Han, Heng Huang, and 1 others. 2015. Sparse representation of whole-brain fmri signals for identification of functional networks. *Medical image analysis*, 20(1):112–134.

Paul M Matthews and Peter Jezzard. 2004. Functional magnetic resonance imaging. *Journal of Neurology, Neurosurgery & Psychiatry*, 75(1):6–12.

Arthur Mensch, Gaël Varoquaux, and Bertrand Thirion. 2016. Compressed online dictionary learning for fast resting-state fmri decomposition. In *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, pages 1282–1285. IEEE.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Yaniv Nikankin, Anja Reusch, Aaron Mueller, and Yonatan Belinkov. 2024. Arithmetic without algorithms: Language models solve math with a bag of heuristics. *arXiv preprint arXiv:2410.21272*.

Jingcheng Niu, Andrew Liu, Zining Zhu, and Gerald Penn. 2024. What does the knowledge neuron thesis have to do with knowledge? In *The Twelfth International Conference on Learning Representations*.

Ruchir Puri, David S. Kung, Geert Janssen, Wei Zhang, Giacomo Domeniconi, Vladimir Zolotov, Julian Dolby, Jie Chen, Mihir R. Choudhury, Lindsey Decker, Veronika Thost, Luca Buratti, Saurabh Pujar, Shyam Ramji, Ulrich Finkler, Susan Malaika, and Frederick Reiss. 2021. Codenet: A large-scale AI for code dataset for learning a diversity of coding tasks. In *NeurIPS Datasets and Benchmarks*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, and 1 others. 2018. Improving language understanding by generative pre-training. *OpenAI*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Marcus E Raichle. 2015. The brain's default mode network. *Annual review of neuroscience*, 38(1):433–447.

Marcus E Raichle and Abraham Z Snyder. 2007. A default mode of brain function: a brief history of an evolving idea. *Neuroimage*, 37(4):1083–1090.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Stephen M Smith, Peter T Fox, Karla L Miller, David C Glahn, P Mickle Fox, Clare E Mackay, Nicola Filippini, Kate E Watkins, Roberto Toro, Angela R Laird, and 1 others. 2009. Correspondence of the brain's functional architecture during activation and rest. *Proceedings of the national academy of sciences*, 106(31):13040–13045.

Ran Song, Shizhu He, Shuting Jiang, Yantuan Xian, Shengxiang Gao, Kang Liu, and Zhengtao Yu. 2024. Does large language model contain task-specific neurons? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7101–7113.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR.

Tianyi Tang, Wenyang Luo, Haoyang Huang, Dongdong Zhang, Xiaolei Wang, Xin Zhao, Furu Wei, and Ji-Rong Wen. 2024. Language-specific neurons: The key to multilingual capabilities in large language models. *arXiv preprint arXiv:2402.16438*.

Gaël Varoquaux, Merlin Keller, Jean-Baptiste Poline, Philippe Ciuciu, and Bertrand Thirion. 2010a. Ica-based sparse features recovery from fmri datasets. In *2010 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pages 1177–1180. IEEE.

Gaël Varoquaux, Sepideh Sadaghiani, Philippe Pinel, Andreas Kleinschmidt, Jean-Baptiste Poline, and Bertrand Thirion. 2010b. A group model for stable multi-subject ica on fmri datasets. *Neuroimage*, 51(1):288–299.

A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Martina G Vilas, Federico Adolfi, David Poeppel, and Gemma Roig. Position: An inner interpretability framework for ai inspired by lessons from cognitive neuroscience. In *Forty-first International Conference on Machine Learning*.

Alex Wang. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Jiaqi Wang, Enze Shi, Huawen Hu, Chong Ma, Yiheng Liu, Xuhui Wang, Yincheng Yao, Xuan Liu, Bao Ge, and Shu Zhang. 2024. Large language models for robotics: Opportunities, challenges, and perspectives. *Journal of Automation and Intelligence*.

Xiaozhi Wang, Kaiyue Wen, Zhengyan Zhang, Lei Hou, Zhiyuan Liu, and Juanzi Li. 2022. Finding skill neurons in pre-trained transformer-based language models. In *Proceedings of EMNLP*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 40 others. 2024a. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024b. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Zeping Yu and Sophia Ananiadou. 2024a. Interpreting arithmetic mechanism in large language models through comparative neuron analysis. *arXiv preprint arXiv:2409.14144*.

Zeping Yu and Sophia Ananiadou. 2024b. Neuron-level knowledge attribution in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3267–3280.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2024a. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, 15(2):1–38.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Yiran Zhao, Wenxuan Zhang, Guizhen Chen, Kenji Kawaguchi, and Lidong Bing. 2024b. How do large language models handle multilingualism? In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

A Appendix

A.1 ICA

FastICA (Hyvärinen and Oja, 2000) is an efficient algorithm for implementing ICA and is the method used in this paper to derive FBNs from LLMs. The FastICA algorithm can be described as follows:

Pre-whitening: The signals are first centered (zero mean) and whitened to remove any linear correlations between the variables and have unit variance.

The whitened signals $\mathbf{Z}$ can be represented as:

$$\mathbf{Z} = \mathbf{E}^{-1/2}\mathbf{V}^{-1}(\mathbf{X} - \mathbb{E}[\mathbf{X}]) \tag{4}$$

where $\mathbf{V}$ and $\mathbf{E}$ are the eigenvectors and eigenvalues of the covariance matrix $\mathbf{\Sigma}$ of $\mathbf{X}$.

Finding Independent Components: For each independent component $\mathbf{w}_i$, we maximize the following objective function:

$$J(\mathbf{w}) = [\mathbb{E}\{G(\mathbf{w}^T\mathbf{z})\}] - \frac{1}{2}\mathbb{E}\{(\mathbf{w}^T\mathbf{z})^2\} \tag{5}$$

where G is a non-linear function used to approximate negentropy, a_1 is a constant usually $a_1 \in [1, 2]$, Common choices for G include:

$$G(u) = \log\cosh(a_1 u) \tag{6}$$

$$G(u) = -\exp(-u^2/2) \tag{7}$$

To find the optimal $\mathbf{w}$, FastICA uses fixed-point iteration:

$$\mathbf{w}_{\text{new}} = \mathbb{E}\{\mathbf{z}g(\mathbf{w}^T\mathbf{z})\} - \mathbb{E}\{g'(\mathbf{w}^T\mathbf{z})\}\mathbf{w} \tag{8}$$

where g is the derivative of G. After each iteration, normalize $\mathbf{w}$:

$$\mathbf{w} \leftarrow \frac{\mathbf{w}}{\|\mathbf{w}\|} \tag{9}$$

If multiple independent components need to be extracted, perform orthogonalization to ensure that the weight vectors remain orthogonal:

$$\mathbf{W} \leftarrow (\mathbf{W}\mathbf{W}^T)^{-1/2}\mathbf{W} \tag{10}$$

where $\mathbf{W}$ is the matrix that contains all weight vectors as columns. Repeat the iteration until the change in the weight vectors falls below a predefined threshold, indicating convergence. Once the demixing matrix $\mathbf{W}$ is obtained, the source signals $\mathbf{S}$ can be estimated from the whitened data $\mathbf{Z}$:

$$\hat{\mathbf{S}} = \mathbf{W}\mathbf{Z} \tag{11}$$

Finally, the mixing matrix $\mathbf{A}$, which represents the spatial patterns of the functional networks, can be obtained as the inverse of the matrix $\mathbf{W}$. Considering the whitening transformation applied to the signals, the mixing matrix can be computed as:

$$\mathbf{A} = \mathbf{V}\mathbf{E}^{1/2}\mathbf{W}^{-1} \tag{12}$$

In visualizations and experimental comparisons, the mixing matrix $\mathbf{A}$, representing the final derived functional networks, typically undergoes thresholding. This process involves setting a threshold to filter out lower values, ensuring that only the regions with significant activation are retained. This approach helps in focusing on the most relevant activations and improving the clarity of the results.

The CanICA (Canonical Independent Component Analysis) (Varoquaux et al., 2010b,a) algorithm is a spatial independent component analysis (sICA) method used for brain imaging data. The subsequent section presents a detailed mathematical description, concentrating on PCA preprocessing and the application of FastICA for spatial component extraction.

Assume we have an observation data matrix $X \in \mathbb{R}^{n\times t}$, where n is the number of sensors or voxels (spatial dimension), and t is the number of time points. Each column represents the observation at one time point, and each row represents the observations of one sensor or voxel across all time points.

In CanICA, Principal Component Analysis (PCA) is first applied to reduce the dimensionality of the data. PCA can be implemented via Singular Value Decomposition (SVD):

$$X = U\Sigma V^T$$

where:

- $U \in \mathbb{R}^{n\times n}$ is the left singular vector matrix.
- $\Sigma \in \mathbb{R}^{n\times t}$ is the singular value matrix, with singular values on its diagonal.
- $V \in \mathbb{R}^{t\times t}$ is the right singular vector matrix.

We select the top k singular vectors corresponding to the largest singular values to construct the reduced data matrix X_{reduced}:

$$X_{\text{reduced}} = U_k\Sigma_k$$

where U_k and Σ_k are the first k columns of U and the first k rows of Σ, respectively.

FastICA is then applied to the reduced data. Unlike traditional ICA, the mixing matrix A here does not represent the functional networks. In spatial ICA $S \in \mathbb{R}^{k\times t}$, which is the source signal matrix that represents the functional networks that are to be extracted. The model can be expressed as:

$$X_{\text{reduced}} = AS$$

where $A \in \mathbb{R}^{n\times k}$ is the mixing matrix, indicating how spatial components combine to form the observed data.

In practice, the reduced data X_{reduced} is fed into the FastICA algorithm for further processing.

The extracted spatial components are stored in the source signal matrix S, with each row vector representing an independent spatial pattern.

A.2 Stable Functional Networks within LLM

The experiments in Section 4.1 are shown in Table 6, 7 and 8.

A.3 Distribution of Neurons in Functional Networks

In Figure 4, we present several group-level functional networks derived from different datasets. These functional networks involve only a small fraction of total neurons, ranging from less than 0.1% to around 2%, consistent with the well-established principle in neuroscience that functional brain networks also activate a sparse subset of regions (Dong et al., 2020; Liu et al., 2023a) (see

Table 6: The number of the counterparts in 100 individual-level analysis for the functional network templates in group-level analysis in Qwen2.5-3B-Instruct.

TEMPLATES	NEWS	WIKI	MATH	CODE
1	0	57	0	4
2	7	51	0	62
3	0	44	5	80
4	11	53	6	21
5	51	17	0	12
6	19	20	0	26
7	74	89	48	62
8	8	10	9	61
9	14	0	0	54
10	56	88	7	60

Table 7: The number of the counterparts in 100 individual-level analysis for the functional network templates in group-level analysis in Qwen2.5-7B-Instruct.

TEMPLATES	NEWS	WIKI	MATH	CODE
1	12	71	121	60
2	4	10	115	60
3	89	25	139	64
4	0	2	0	131
5	19	2	139	73
6	67	1	0	61
7	131	62	22	75
8	23	7	4	5
9	31	87	7	68
10	74	19	191	119

Table 8: The number of the counterparts in 100 individual-level analysis for the functional network templates in group-level analysis in ChatGLM3-6B-base.

TEMPLATES	NEWS	WIKI	MATH	CODE
1	0	47	0	50
2	27	5	8	2
3	4	10	0	0
4	8	88	0	38
5	0	33	6	61
6	27	75	13	202
7	3	0	12	11
8	26	0	1	0
9	12	16	0	27
10	0	51	0	171

Table 9: Performance of ChatGLM3-6B-base after inhibiting top 5% Neurons.

TASK	NORMAL	ICA	LLM LOCALIZATION
COLA (MCC)	0.1497	-0.0179	0.0702
MNLI	0.5636	0.3533	0.4034
MRPC (F1)	0.8415	0.0676	0.7470
QNLI	0.8223	0.5318	0.5535
QQP (F1)	0.8065	0.5334	0.4244
RTE	0.7581	0.5632	0.6570
SST-2	0.9553	0.5952	0.7787
WNLI	0.6479	0.4648	0.4225
AVERAGE	0.6931	0.3864	0.5071
WIKITEXT (PPL)	10.1177	40.4245	14.6199

Fig. 1(a) for an illustration of human functional networks). This sparsity aligns with our observation that LLMs, like the human brain, rely on highly selective neural activation. Moreover, Figure 4 visually demonstrates that similar functional network structures emerge across diverse input data types, supporting our claim that stable, reusable functional units exist in LLMs.

A.4 Functional Networks Editing Experiments

The neuron-leison experiments results are shown in Table 10 and 11. Table 9 shows the results of experiments on ChatGLM3-6B-base using the lm-eval framework, comparing our method (where the top 5% most important neurons are inhibited) with the LLM Localization (AlKhamissi et al., 2024) approach. It can be observed that, when inhibiting the same number of neurons, our method causes a greater drop in model performance, indicating that the neurons identified by our approach are more critical to the model's functionality.

The results of additive editing and other multiplicative factors in ChatGLM3-6B-base are shown in Figure 5, 6 and 7. If values greater than 1 (e.g., between 1.5 and 3) are added to the neurons of functional networks in ChatGLM3-6B-base, the model's word perplexity can surge to tens of thousands or higher. It can be observed that neither additive nor multiplicative editing should use excessively large values, as this severely degrades model performance. Moreover, multiplicative editing yields more stable results, making it the preferred approach for neuron editing.

We present an example where amplifying the neuronal signals of a functional network improves the LLM's performance on the SST-2 dataset. As shown in Figure 8, this intervention corrects a prediction that would otherwise be incorrect under normal (unedited) model behavior—turning an originally wrong output into the correct one.

A.5 Neuron Probing Experiment

We investigate whether inhibiting functional networks impairs the LLM's feature representation capability, not just its text generation. Specifically, after inhibiting certain functional networks, we observe degraded generation quality (e.g., incorrect

Figure 4: Example functional networks that are consistent across various types of input samples in group level analysis. Each row represents the neurons in an MLP layer where activated neurons are highlighted.

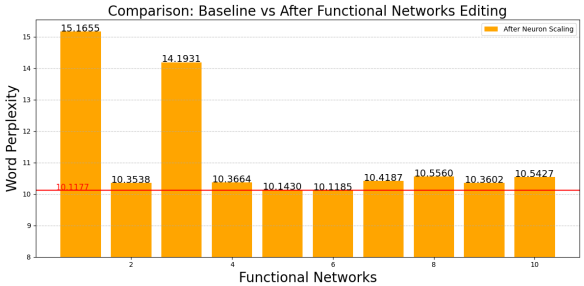

Figure 5: Results perplexity of additive editing for 10 functional networks in ChatGLM3-6B-base. Adding value is set as 1.

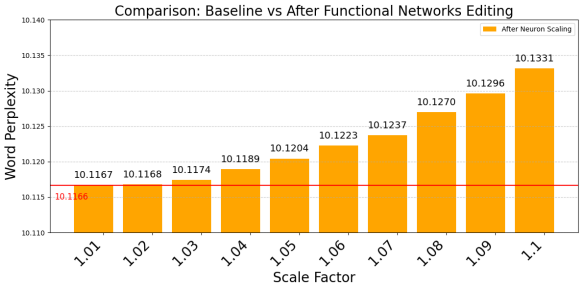

Figure 6: Results perplexity of different multiplicative factors editing in ChatGLM3-6B-base.

answers or gibberish). However, this could stem from disruption in the output head or decoding process rather than damaged internal representations.

To isolate the effect on representation, we conduct a linear probing experiment: we freeze the LLM (with inhibiting functional networks), extract features from its last layer, and train a simple linear classifier (e.g., logistic regression) on these features for a downstream task. If classification perfor-

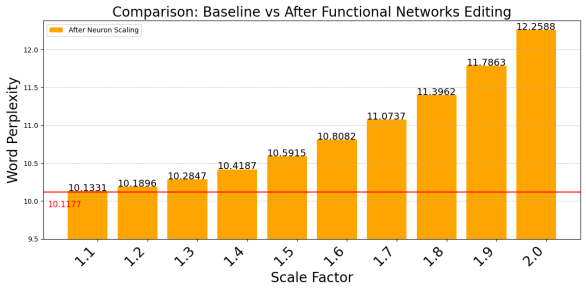

Figure 7: Results perplexity of different multiplicative factors editing in ChatGLM3-6B-base.

Table 10: Performance of Qwen2.5-7B-Instruct after inhibiting functional networks. First Column: Dataset. Second Column: Model performance under normal conditions (without inhibiting). Third Column: Model performance after inhibiting 15% of the neurons randomly. Fourth Column: Model performance after inhibiting the neurons belonging to 10 specific functional networks.

DATASETS	NORMAL	MASKED 15% (15053)	MASKED 10 (361-2217)
COLA	0.7641	0.7143	0.0000
MRPC	0.6765	0.3873	0.1691
SST-2	0.9358	0.9094	0.0000
MNLI	0.7243	0.6052	0.0000
QNLI	0.8060	0.5629	0.0000
QQP	0.8388	0.7367	0.0000
RTE	0.8448	0.8339	0.0000
WNLI	0.8169	0.7746	0.0000

Table 11: Performance of ChatGLM3-6B-base after inhibiting Functional Networks.

DATASETS	NORMAL	MASKED 15% (17203)	MASKED 10
COLA	0.6893	0.6913	0.0000
MRPC	0.7843	0.8015	0.0000
SST-2	0.9553	0.9346	0.0000
MNLI	0.5634	0.4832	0.0000
QNLI	0.8223	0.7633	0.0000
QQP	0.8548	0.8392	0.6335
WNLI	0.6479	0.5352	0.5493
RTE	0.7581	0.7040	0.4188
AG NEWS	0.9128	0.8961	0.0000
SQUAD	0.9021	0.8864	0.0000

Model: ChatGLM3
Dataset: SST2

Sample: but it still jingles in the pocket
Label: Positive
Before Amplifying: Negative
After Amplifying : Positive

Figure 8: Correcting an SST-2 prediction by amplifying a functional network.

mance drops significantly compared to the normal model, it indicates that the semantic content of the representations has been compromised, not merely the generation pipeline. This provides evidence that the inhibiting functional networks are essential for maintaining meaningful feature encoding.

To assess whether inhibiting functional networks impairs the feature representation capability of LLMs—distinct from mere disruption of text generation—we conduct linear probing experiments on the last-layer activations. As shown in Tables 12 and 13, we evaluate two models—ChatGLM3-6B-base and Qwen2.5-7B—under three conditions: (1) Normal (no inhibiting), (2) Masked 15% (randomly inhibiting 15% of neurons), and (3) Masked 10 (inhibiting 10 identified functional networks, corresponding to neuron sets ranging from 957 to 3,354 units).

The results reveal a clear pattern. Random inhibiting of 15% of neurons has minimal impact on downstream classification performance across all datasets, just like the results in neuron-leision experiments. In many cases, such as SST-2 for ChatGLM3-6B-base or MRPC for Qwen2.5-7B, performance even slightly improves, likely due to incidental removal of noisy or redundant units. This suggests that general neuron loss does not significantly degrade representational quality.

In stark contrast, inhibiting just 10 specific functional networks leads to substantial performance drops, confirming that these networks carry essential semantic information. For example, on the MNLI dataset, Qwen2.5-7B's accuracy plummets from 0.87 (Normal) to 0.4425 (inhibiting 10), a 49% relative decline, while ChatGLM3-6B-base drops from 0.86 to 0.705. Similar trends appear across QNLI, MRPC, and QQP, with classification accuracy under Masked 10 consistently falling to the 0.63–0.70 range, despite remaining above chance. This demonstrates that while the model's feature extractor is not completely disabled, its ability to encode task-relevant semantics is significantly compromised.

Although, Qwen2.5-7B slightly stronger than ChatGLM3-6B-base in the Normal condition, exhibits greater sensitivity to functional network inhibiting, especially on complex tasks like MNLI

Table 12: Classification Performance of ChatGLM3-6B-base Using Logistic Regression on the Last Layer Features

DATASETS	NORMAL	MASKED 15% (17203)	MASKED 10 (1827-3354)
QNLI	0.885	0.88	0.63
COLA	0.8038	0.7895	0.6938
MRPC	0.8049	0.7683	0.6707
MNLI	0.86	0.865	0.705
QQP	0.825	0.85	0.6825
SST-2	0.9314	0.9314	0.8514

Table 13: Classification Performance of Qwen2.5-7B Using Logistic Regression on the Last Layer Features

DATASETS	NORMAL	MASKED 15% (15052)	MASKED 10 (957-2035)
QNLI	0.9	0.87	0.635
COLA	0.8038	0.7464	0.6746
MRPC	0.7805	0.8171	0.6463
MNLI	0.87	0.84	0.4425
QQP	0.83	0.8225	0.6975
SST-2	0.9257	0.9486	0.7486

and COLA. This implies that Qwen2.5-7B may depend more heavily on these specific functional networks for high-level reasoning, possibly reflecting architectural or training differences between the two model families.

Crucially, these probing results contrast sharply with the models' behavior in zero-shot generation, where inhibiting the same networks often leads to complete failure (e.g., incoherent outputs). The fact that linear probes still extract usable features while autoregressive generation fails. This suggests a dual impact: (1) functional networks are vital for constructing high-quality internal representations, and (2) their disruption disproportionately affects the generative decoding process.

These findings provide strong evidence that the identified functional networks are not incidental but core computational units that shape both representation and generation in LLMs.

A.6 Functional Interpretation of ICA Components

ICA decomposes the model's internal activations into a set of statistically independent components, each representing a distinct functional network. By design, each component is expected to capture a coherent and interpretable aspect of model computation, ideally corresponding to a specific linguistic or cognitive function.

To interpret these components, we feed an input sentence through the model, apply ICA to its hidden activations, and extract the resulting source signals $\mathbf{S}$. For a given component, the entries in its source signal are aligned with the input tokens and used as attribution weights. We then visualize these weights to highlight which tokens most strongly activate that functional network, offering insight into the component's potential role (e.g., attending to sentiment words, entities, or syntactic structures).

Figure 9, Figure 10, Figure 11, Figure 12, Figure 13, Figure 14, Figure 15, Figure 16, Figure 17 and Figure 18 show the example interpret results.

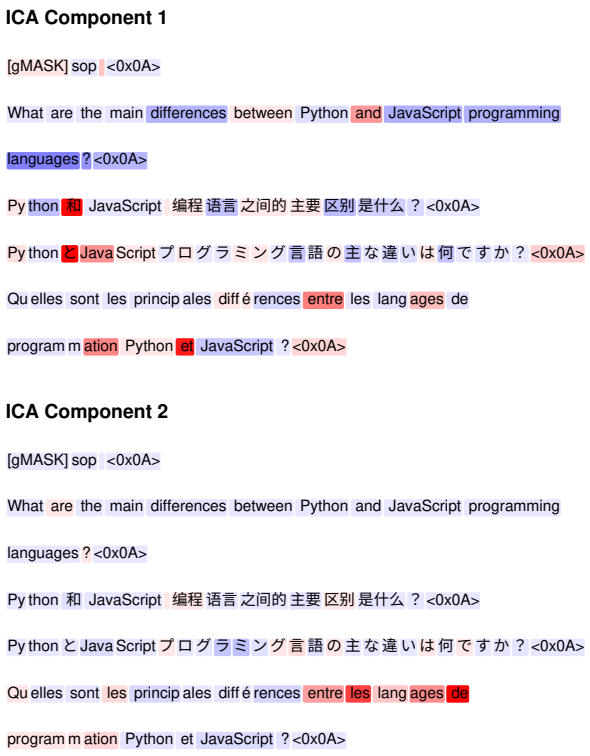

Figure 9: Interpreting ICA components by projecting their source signals back onto input tokens as attribution weights. High-weight tokens are visualized to reveal what linguistic features each functional network responds to.

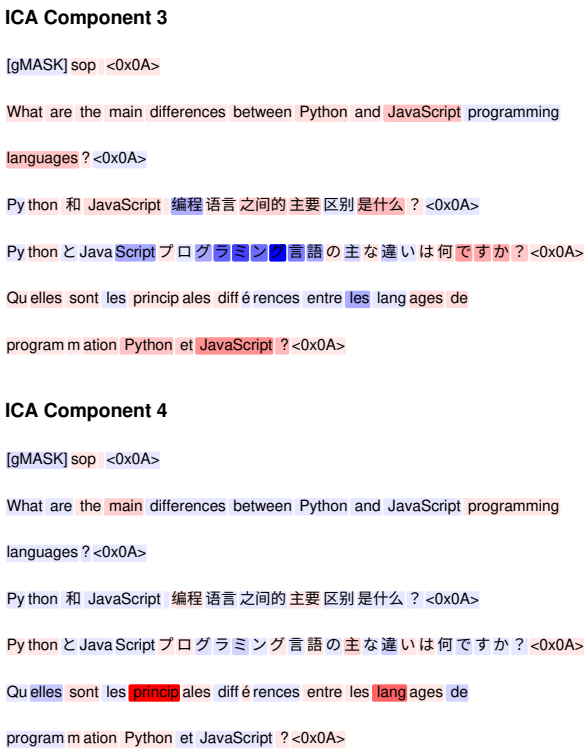

Figure 10: Interpreting ICA components by projecting their source signals back onto input tokens as attribution weights. High-weight tokens are visualized to reveal what linguistic features each functional network responds to.

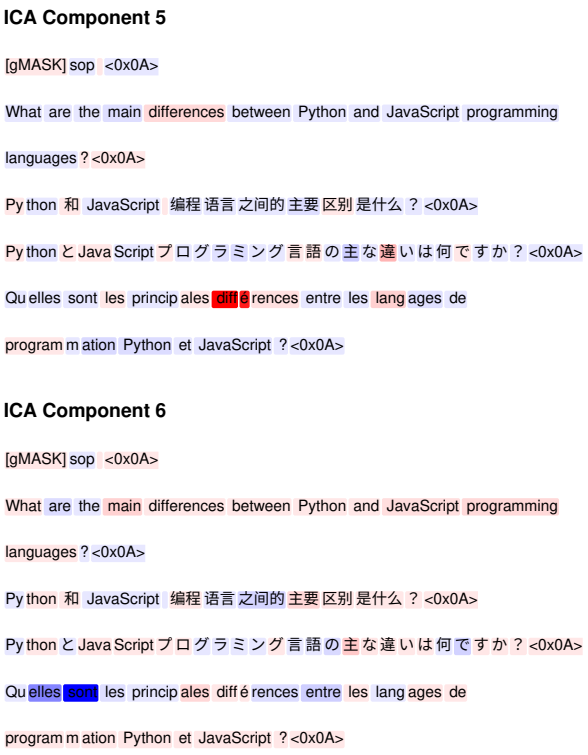

Figure 11: Interpreting ICA components by projecting their source signals back onto input tokens as attribution weights. High-weight tokens are visualized to reveal what linguistic features each functional network responds to.

ICA Component 7

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ? <0x0A>

ICA Component 8

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ? <0x0A>

Figure 12: Interpreting ICA components by projecting their source signals back onto input tokens as attribution weights. High-weight tokens are visualized to reveal what linguistic features each functional network responds to.

ICA Component 9

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ? <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ? <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ? <0x0A>

ICA Component 10

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ? <0x0A>

Figure 13: Interpreting ICA components by projecting their source signals back onto input tokens as attribution weights. High-weight tokens are visualized to reveal what linguistic features each functional network responds to.

ICA Component 11

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ？<0x0A>

ICA Component 12

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ？<0x0A>

Figure 14: Interpreting ICA components by projecting their source signals back onto input tokens as attribution weights. High-weight tokens are visualized to reveal what linguistic features each functional network responds to.

ICA Component 13

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ？<0x0A>

ICA Component 14

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ？<0x0A>

Figure 15: Interpreting ICA components by projecting their source signals back onto input tokens as attribution weights. High-weight tokens are visualized to reveal what linguistic features each functional network responds to.

ICA Component 15

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ? <0x0A>

ICA Component 16

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ? <0x0A>

Figure 16: Interpreting ICA components by projecting their source signals back onto input tokens as attribution weights. High-weight tokens are visualized to reveal what linguistic features each functional network responds to.

ICA Component 17

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ? <0x0A>

ICA Component 18

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ? <0x0A>

Figure 17: Interpreting ICA components by projecting their source signals back onto input tokens as attribution weights. High-weight tokens are visualized to reveal what linguistic features each functional network responds to.

ICA Component 19

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ? <0x0A>

ICA Component 20

[gMASK] sop <0x0A>

What are the main differences between Python and JavaScript programming

languages ? <0x0A>

Py thon 和 JavaScript 编程 语言 之间的 主要 区别 是什么 ？ <0x0A>

Py thon と Java Script プ ロ グ ラ ミ ン グ 言 語 の 主 な 違 い は 何 で す か ？ <0x0A>

Qu elles sont les princip ales diff é rences entre les lang ages de

program m ation Python et JavaScript ? <0x0A>

Figure 18: Interpreting ICA components by projecting their source signals back onto input tokens as attribution weights. High-weight tokens are visualized to reveal what linguistic features each functional network responds to.