Improving Generalization in LLM Structured Pruning via Function-Aware Neuron Grouping

Tao Yu[1 2], **Yongqi An**[1 2], **Kuan Zhu**[1], **Guibo Zhu**[1 2 3], **Ming Tang**[1 2], **Jinqiao Wang**[1 2 3*]

[1]Foundation Model Research Center, Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[3]Wuhan AI Research, Wuhan, China
yutao2022@ia.ac.cn, {yongqi.an, kuan.zhu, gbzhu, tangm, jqwang}@nlpr.ia.ac.cn

Abstract

Large Language Models (LLMs) demonstrate impressive performance across natural language tasks but incur substantial computational and storage costs due to their scale. Post-training structured pruning offers an efficient solution. However, when few-shot calibration sets fail to adequately reflect the pretraining data distribution, existing methods exhibit limited generalization to downstream tasks. To address this issue, we propose Function-Aware Neuron Grouping (FANG), a post-training pruning framework that alleviates calibration bias by identifying and preserving neurons critical to specific function. FANG groups neurons with similar function based on the type of semantic context they process and prunes each group independently. During importance estimation within each group, tokens that strongly correlate with the functional role of the neuron group are given higher weighting. Additionally, FANG also preserves neurons that contribute across multiple context types. To achieve a better trade-off between sparsity and performance, it allocates sparsity to each block adaptively based on its functional complexity. Experiments show that FANG improves downstream accuracy while preserving language modeling performance. It achieves the state-of-the-art (SOTA) results when combined with FLAP and OBC, two representative pruning methods. Specifically, FANG outperforms FLAP and OBC by 1.5%–8.5% in average accuracy under 30% and 40% sparsity.

Introduction

In recent years, Large Language Models (LLMs) (Brown et al. 2020; Touvron et al. 2023a) have shown remarkable capabilities in generating high-quality text and tackling a wide range of downstream tasks. Despite their powerful performance, these models typically contain vast numbers of parameters, resulting in increased computational and storage demands that hinder efficient deployments (Xu et al. 2024). Structured pruning has emerged as an effective solution to these challenges by removing redundant neurons or attention heads (Xia et al. 2023; Kurtić, Frantar, and Alistarh 2023), thereby reducing both computation and memory costs.

Given the massive scale of LLMs, traditional prune-and-retrain methods are often computationally prohibitive (Xia et al. 2023; Tang et al. 2025). A more practical alternative is post-training pruning without fine-tuning (An et al. 2024; Lin et al. 2024; Wei et al. 2024; Ling, Wang, and Liu 2024), which eliminates redundant parameters in a one-shot manner while employing numerical compensation to recover performance. Specifically, these methods compute importance metrics for neurons based on a calibration set, rank the neurons, and remove those with the lowest importance. The calibration set is typically sampled from pretraining data to preserve modeling ability on the pretraining task, which is commonly measured by perplexity. It also provides a certain degree of generalization, as reflected in zero-shot accuracy on downstream tasks. In practice, these methods remain effective even with a limited number of calibration samples.

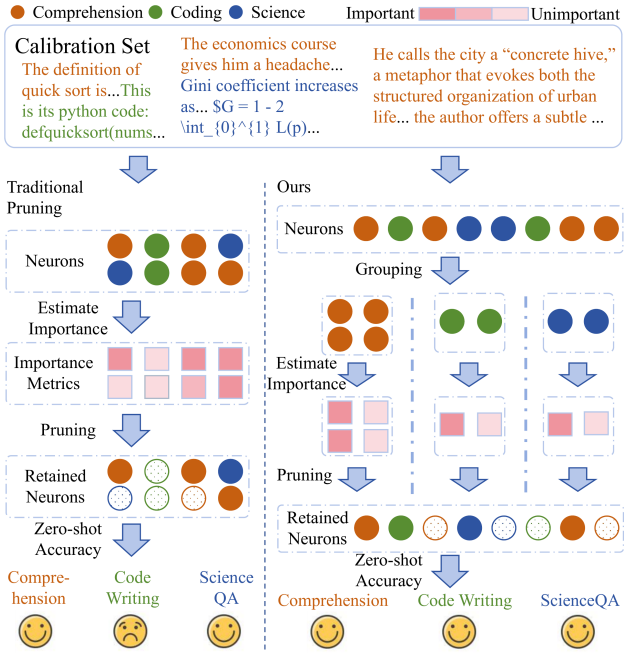

Figure 1: Compared to traditional post-training pruning, our method mitigates calibration set distribution mismatch by pruning neurons based on functional grouping, reducing the risk of misestimating important neurons and improving generalization.

*Corresponding author.

However, despite its effectiveness, the small size of the calibration set makes the model prone to overfitting. When the calibration data fails to adequately reflect the distribution of the pretraining corpus, this overfitting can lead to degraded generalization (JAISWAL et al. 2024). In particular, the importance of functionally critical neurons may be misestimated, resulting in their erroneous removal and irreversible accuracy loss on downstream tasks, as shown in Fig. 1. This raises a key question: ***Can we enhance generalization ability while preserving the efficiency of pruning and maintaining performance on the pretraining task?***

To address this question, we draw inspiration from interpretability research (Gurnee et al. 2023; Cao, Sanh, and Rush 2021; Dai et al. 2022; De Cao et al. 2020), which suggests that LLMs exhibit functional specialization—analogous to the human brain—where different neurons are responsible for processing distinct types of contextual information. Building on this insight, we propose ***Function-Aware Neuron Grouping (FANG)***, a post-training structured pruning framework designed to retain functional diversity and improve generalization.

FANG consists of three core components: (1) A ***function-aware pruning strategy***, in which neurons are grouped by their functional roles and pruned independently. During importance estimation within each group, tokens most semantically aligned with the group's function are assigned greater weight. (2) A ***shared neuron group retention*** mechanism that identifies and preserves neurons contributing across multiple context types. (3) An ***adaptive sparsity allocation*** scheme that assigns lower sparsity to functionally complex blocks based on a complexity metric. These designs retain both specialized and general capacities, support more balanced pruning, and improve generalization across various downstream tasks.

Fig. 2 illustrates the overall framework of our proposed method. The main contributions of this work are summarized as follows:

- We analyze the pruning process and identify the failure to distinguish among neurons based on the types of contextual information they process as a key factor limiting generalization.
- We propose Function-Aware Neuron Grouping (FANG), a post-training structured pruning method that integrates function-aware pruning strategy, shared neuron group retention mechanism, and adaptive sparsity allocation. This is the first method to explicitly consider functional specialization during pruning to enhance generalization.
- Extensive experiments demonstrate that our method can be effectively combined with classical pruning approaches such as FLAP and OBC, leading to 1.5%–8.5% improvements in downstream accuracy while maintaining low perplexity, thereby achieving state-of-the-art (SOTA) performance.

Related Works

Structured Pruning

Pruning refers to the removal of redundant parameters in a model to reduce its storage and computational costs. Based on the granularity of the pruned units, pruning methods can be broadly categorized into unstructured pruning and structured pruning.

Unstructured pruning removes individual weights (Frantar and Alistarh 2022, 2023; Sun et al. 2024; Yin et al. 2024), resulting in sparse weight matrices. While it often preserves model performance better than structured pruning at the same sparsity level, its practical speedup typically relies on specialized hardware accelerators. In contrast, structured pruning removes entire rows, columns, or even full layers of the weight matrix (An et al. 2024; Kurtić, Frantar, and Alistarh 2023; Men et al. 2024; Ashkboos et al. 2024). This leads to reduced channel or layer counts, enabling actual inference acceleration on general GPUs.

This work focuses on structured pruning, with the goal of mitigating performance degradation on downstream tasks after pruning.

Post-training Pruning without Fine-tuning

Structured pruning for LLMs is typically implemented in two ways: training-based pruning and training-free pruning.

Training-based pruning can achieve higher compression but requires costly retraining, resulting in substantial computational and storage overhead during this stage (Tang et al. 2025; Xia et al. 2023). In contrast, training-free pruning—typically implemented in a post-training manner—is more efficient. It typically adopts a layer-wise pruning strategy, where pruning is performed independently at each layer: unimportant parameters are identified and removed based on predefined importance metrics, followed by numerical compensation to minimize the reconstruction error of that layer's output (Kurtić, Frantar, and Alistarh 2023; Ashkboos et al. 2024; Ling, Wang, and Liu 2024). This enables the model to recover without further retraining and requires only a small calibration set. Methods such as FLAP (An et al. 2024), ModeGPT (Lin et al. 2024), and SoBP (Wei et al. 2024) follow this paradigm.

Our study focuses on the post-training pruning paradigm without fine-tuning. Although this approach only requires a small calibration set, the scarcity of data may give rise to overfitting, which frequently leads to the degraded generalization ability. This has become a critical challenge that this research endeavors to tackle.

Preliminary

Pruning of Transformer

In the architecture of LLM, each Transformer (Vaswani et al. 2017) block consists of two components: a multi-head attention (MHA) module and a feed-forward network (FFN). In each layer l, MHA pruning is typically performed at the head level by removing the entire projection matrices $W_q^{l,i}$, $W_k^{l,i}$ and $W_v^{l,i} \in \mathbb{R}^{d_H \times d}$ for the pruned head i, along with the corresponding columns in $W_o^l \in \mathbb{R}^{d \times d}$. d denotes the embedding dimension, d_H is the hidden dimension of each attention head. Given a pruning mask $M_H^l \in \{0,1\}^{N_h}$, where 1 indicates a pruned head and 0 indicates a retained head, and N_h denotes the total number of attention heads in a layer.

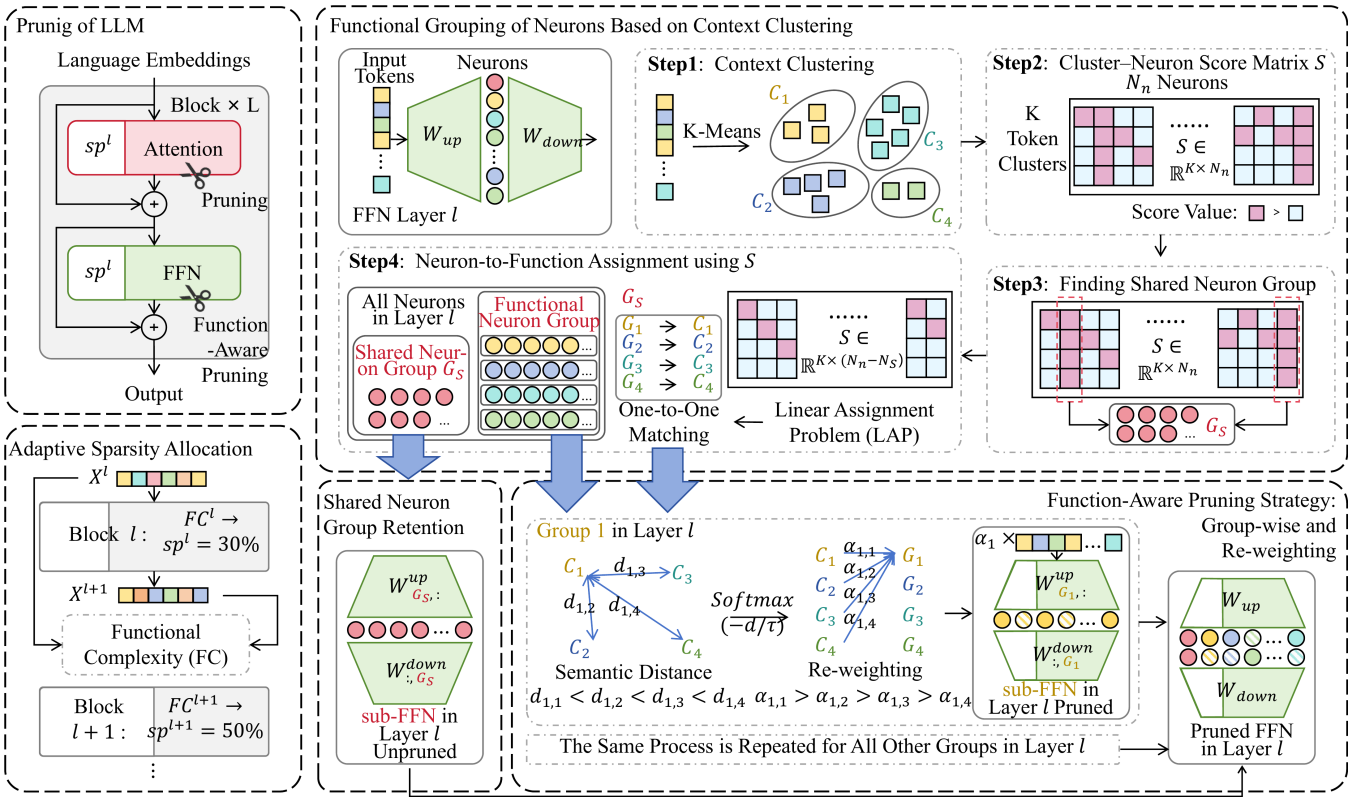

Figure 2: Overview of Function-Aware Neuron Grouping (FANG). For attention heads, we adopt baseline pruning methods such as OBC and FLAP. For FFN layers, a function-aware pruning strategy is applied.

The output of the pruned MHA module is formulated in Eq. 1 and 2. Attn refers to the self-attention operation, and $\circ$ represents element-wise multiplication.

$$X_{\text{head}}^{l,i} = (1 - M_H^{l,i}) \circ \text{Attn}\left(X^{l-1}, W_{q,k,v}^{l,i}\right), \quad (1)$$

$$X_{\text{MHA}}^{l} = W_o^l \,\text{Concat}\left(X_{\text{head}}^{l,1}, \ldots, X_{\text{head}}^{l,N_h}\right). \quad (2)$$

In the FFN module, pruning is usually conducted at the neuron level by removing the corresponding rows in $W_{up} \in \mathbb{R}^{d \times N_n}$ and columns in $W_{down} \in \mathbb{R}^{N_n \times d}$. N_n denotes the number of neurons in a layer, which is also the dimension of the hidden features in the FFN. Given a pruning mask $M_N^l \in \{0,1\}^{N_n}$, the output of the pruned MHA module is formulated in Eq. 3, where σ denotes activation function.

$$X_{\text{FFN}}^{l} = W_{down}^{l}(1 - M_N^l) \circ \sigma\left(W_{up}^{l}(X_{\text{MHA}}^{l} + X^{l-1})\right). \quad (3)$$

Layer-wise Pruning

Layer-wise pruning is a kind of classical method in post-training pruning. Given a linear layer l with weight matrix $W^l \in \mathbb{R}^{C_{out} \times C_{in}}$ and input features $X^l \in \mathbb{R}^{C_{in} \times L}$, the goal of layer-wise pruning is to obtain a sparse weight matrix $\widehat{W}$ that minimizes the reconstruction error of the output, as defined in the following equation:

$$\arg\min_{\widehat{W}^l} \|W^l X^l - \widehat{W}^l X^l\|_2^2 \quad \text{s.t} \quad sp(\widehat{W}^l) \geq sp^l, \quad (4)$$

where $sp(\widehat{W}^l)$ denotes the sparsity of the pruned weights and sp is the target sparsity. This formulation is typically applied to the W_o and W_{down} layers to identify removable input channels, while in the W_{qkv} and W_{up} layers, the corresponding output channels are directly removed according to the pruning decisions made in W_o and W_{down}.

A common solution is to define an importance metric for each input channel, remove those with the lowest importance, and apply numerical compensation to reduce reconstruction error. Structured OBC (Kurtić, Frantar, and Alistarh 2023) and FLAP (An et al. 2024) are representative methods of this approach.

Method

The proposed method adopts a function-aware pruning strategy by grouping neurons according to their functional roles and pruning them independently. It enhances robustness by preserving a shared neuron group and adaptively allocating layer-wise sparsity based on functional complexity.

Functional Grouping of Neurons Based on Context Clustering

The proposed method begins by identifying functional neuron groups, which serve as the basis for subsequent pruning. Following prior work, we define functional roles as the model's capacity to process distinct types of semantic context (Liu et al. 2023; Qu et al. 2024). This gives rise to two

key challenges: (1) how to identify different context types from the model's learned representations, and (2) how to determine which neurons primarily contribute to each type.

To address these challenges, we propose a clustering-based strategy to distinguish semantic contexts and a score-based assignment mechanism to associate neurons with their corresponding functional groups.

Context Clustering. We cluster input tokens to distinguish different types of contextual information. Specifically, we extract the input to each FFN layer, defined as $X^l_{\text{MHA}} + X^{l-1}$ in Eq. 3, and apply the K-Means algorithm to partition them into K clusters $\{C^l_1, C^l_2, \ldots, C^l_K\}$. To reduce computational complexity, we first apply Principal Component Analysis (PCA) to reduce the dimensionality of the token representations, preserving the most informative components for clustering.

Cluster–Neuron Score Matrix S. Once token clusters are obtained, we quantify the contribution of each neuron to each cluster using a Taylor expansion-based sensitivity score (Molchanov et al. 2016, 2019; LeCun, Denker, and Solla 1989). For the k-th token cluster and the j-th neuron, the score is computed as:

$$s^l_{k,j} = \frac{1}{N_{C_k}} \sum_{i \in C_k} \left| h^l_{i,j} \frac{\partial \mathcal{L}}{\partial h^l_{i,j}} \right|, \tag{5}$$

where $i \in C_k$ denotes the tokens belonging to the k-the cluster, and N_{C_k} is the number of such tokens. $\mathcal{L}$ is the loss function of the pretraining task, and h^l denotes the intermediate hidden features in the FFN layer l. The resulting score matrix $S^l \in \mathbb{R}^{K \times N_n}$ captures the relevance of each of the N_n neurons to each token cluster.

Neuron-to-Function Assignment using S. We formulate the assignment of neurons to functional groups as a Linear Assignment Problem (LAP). Following the solution to LAP as in (Qu et al. 2024), K neuron groups $\{G^l_1, G^l_2, \ldots, G^l_K\}$ are obtained for layer l, where each group G^l_k consists of an equal number of neurons most relevant to cluster C^l_k. These groups are treated as functionally specialized units, each responsible for processing a specific type of context.

Function-Aware Pruning Strategy

The core idea of the function-aware pruning strategy is to prune neurons independently within each functional neuron group and to compute group-specific importance metrics that place greater emphasis on the tokens each group is responsible for processing.

Group-Wise Pruning. Given the neuron groups $\{G^l_1, G^l_2, \ldots, G^l_K\}$, we partition the weight matrix W^l_{down} of each FFN layer accordingly. Each sub-matrix $W^l_{:,G_k}$ is then pruned independently using importance metrics specific to group G^l_k.

Token-Aware Neuron Importance Reweighting. To ensure that each group's pruning process focuses on relevant context, we introduce a reweighting mechanism during importance estimation. Token clusters that are semantically closer to the cluster associated with a neuron group are assigned higher weights α. To quantify the semantic relevance between token clusters, we compute the pairwise L_2 distances $d^l_{k,j}$ between cluster centers, which are measured at the input to W^l_{down}, resulting in a distance matrix $D^l \in \mathbb{R}^{K \times K}$. For each token cluster k, the relevance weights $\alpha^l_k \in \mathbb{R}^K$ are computed via a softmax over the negative distances:

$$\alpha^l_k = \text{softmax}\left(-\frac{d^l_k}{\tau}\right), \tag{6}$$

where τ is a temperature hyperparameter that controls the sharpness of the distribution.

Group-Wise Reweighted Pruning Objective. Incorporating group-wise structure and semantic relevance, the standard layer-wise pruning objective (Eq. 4) is reformulated as:

$$\begin{aligned} &\arg\min_{\widehat{W}^l} \sum_{k,j} \alpha^l_{k,j} \left\| W^l_{:,G_k} X^l_{G_k,C_j} - \widehat{W}^l_{:,G_k} X^l_{G_k,C_j} \right\|^2_2 \\ &\text{s.t. } sp(\widehat{W}^l_{:,G_k}) \geq sp_{G^l_k}, \forall G^l_k, \end{aligned} \tag{7}$$

where $W_{:,G_k}$ refers to the columns of the weight matrix corresponding to neurons in G_k, and X_{G_k,C_j} refers to the sub-matrix of input features indexed by neurons in G_k and tokens in C_j. To address the above objective, OBC (Kurtić, Frantar, and Alistarh 2023) or FLAP (An et al. 2024) can be applied independently to each group. The neuron importance and compensation within a group are computed with consideration of $\alpha^l_{k,j}$.

Enhancing Robustness and Performance

To enhance the robustness and performance of the function-aware pruning strategy, two limitations must be addressed. First, some neurons contribute to multiple context types and should not be rigidly assigned to a single functional group; pruning such shared neurons may lead to the loss of general-purpose capacity. Second, functional neuron groups vary in complexity, and applying a uniform sparsity ratio may cause unnecessary accuracy loss in more critical groups.

To mitigate these issues, we introduce a shared neuron group retention mechanism and an adaptive sparsity allocation strategy that assigns block-wise sparsity based on the aggregated functional complexity within each block.

Shared Neuron Group Retention. To preserve general-purpose capacity, we construct one shared neuron group per layer using the cluster–neuron score matrix S. The group size equals that of each functional group, denoted as $m = N_n/(K+1)$. For each token cluster, we select the top-m scoring neurons. Neurons selected by multiple clusters are ranked by selection frequency, and the top-m ones are retained to form the shared group. These neurons are exempted from pruning, ensuring that widely useful representations are preserved.

Adaptive Sparsity Allocation. Inspired by ShortGPT (Men et al. 2024), we define the Functional Complexity (FC) of each block as the degree of change between its input

Model		LLaMA1-7B		LLaMA2-7B		LLaMA2-13B		LLaMA2-70B	
Sparsity	Method	PPL↓	Avg↑	PPL↓	Avg↑	PPL↓	Avg↑	PPL↓	Avg↑
0%	Dense	5.68	66.05	5.47	66.69	4.88	69.24	3.32	73.61
20%	SliceGPT (ICLR 2024)	6.99	56.16	6.84	54.25	6.06	56.78	4.46	69.60
	SoBP (EMNLP 2024)	6.78	62.19	6.53	63.27	5.62	67.73	**3.88**	71.24
	SVD-LLM (ICLR 2025)	7.89	56.25	8.38	48.70	6.66	58.98	4.66	68.14
	FLAP (AAAI 2024)	6.89	61.19	7.16	56.62	6.31	61.55	4.12	71.60
	F-FANG (Ours)	6.70	**63.21**	6.63	**63.29**	5.82	**67.91**	3.98	72.11
	OBC (NeurIPS 2023)	6.61	61.53	**6.30**	62.27	**5.53**	66.59	3.95	70.54
	O-FANG (Ours)	**6.43**	62.65	6.31	62.88	5.57	67.06	3.97	**72.24**
30%	SliceGPT (Ashkboos et al. 2024)	8.69	46.90	8.64	46.70	7.44	50.10	5.41	61.61
	SoBP (Wei et al. 2024)	7.57	59.61	7.58	59.15	6.27	**66.82**	**4.36**	70.30
	SVD-LLM (Wang et al. 2024)	9.52	51.39	10.66	44.77	8.00	53.16	5.44	64.65
	FLAP (An et al. 2024)	8.23	57.30	8.85	50.91	7.57	57.27	4.82	69.68
	F-FANG (Ours)	7.94	**60.34**	7.78	59.44	6.53	65.50	4.46	71.13
	OBC (Kurtić, Frantar, and Alistarh 2023)	7.38	57.56	7.34	57.88	7.72	51.56	4.40	67.94
	O-FANG (Ours)	**7.23**	59.50	**7.23**	**59.83**	**6.17**	64.21	4.46	**71.34**
40%	SliceGPT (Ashkboos et al. 2024)	15.94	39.64	12.80	41.47	10.60	44.31	7.08	52.00
	SoBP (Wei et al. 2024)	9.09	**56.10**	9.28	**56.06**	7.39	60.86	4.96	68.58
	SVD-LLM (Wang et al. 2024)	13.85	42.95	16.14	40.47	10.79	45.40	6.83	58.05
	FLAP (An et al. 2024)	10.16	52.45	11.49	48.70	9.07	53.18	6.24	67.96
	F-FANG (Ours)	10.31	53.82	10.34	50.94	7.84	58.62	5.06	**69.30**
	OBC (Kurtić, Frantar, and Alistarh 2023)	8.85	52.49	9.13	52.47	11.67	48.80	**4.87**	67.23
	O-FANG (Ours)	**8.56**	55.16	**8.67**	54.96	**7.04**	**61.16**	5.01	68.77

Table 1: Comparison of Structured Pruning and Low-Rank Decomposition Methods on LLaMA Models. **Bold** indicates the best result, underline denotes the second-best. O-FANG and F-FANG denote the combinations of our method with OBC and FLAP, respectively.

and output representations, measured by cosine similarity. A lower similarity indicates higher complexity, suggesting that the block performs more substantial transformations. Formally, the FC of block l is computed as:

$$FC^l = 1 - \mathbb{E}_{X,t} \frac{X_{:,t}^{l\top} X_{:,t}^{l+1}}{\left\|X_{:,t}^{l}\right\|_2 \left\|X_{:,t}^{l+1}\right\|_2}, \tag{8}$$

where $X_{:,t}^{l}$ denotes the representation of token t at the input of block l. Following OWL (Yin et al. 2024), we set the block-wise sparsity $sp^l \propto 1 - FC^l$, and linearly scale all sp^l values to fall within $[0.5sp, 1.5sp]$, where sp denotes the target sparsity.

Experiments

Experimental Settings

Implementation Details. In the proposed method, neuron functional grouping, the Function-Aware Pruning Strategy, and shared neuron group retention are applied only to FFN pruning, while adaptive sparsity allocation is used for both attention head and FFN pruning. Each layer is divided into seven functional groups and one shared group. For context clustering, input token representations are reduced to 64 dimensions via PCA, where the chosen dimensionality balances clustering efficiency and accuracy. In Function-Aware Pruning, τ controls reweighting, with model-specific values (e.g., $\tau = 9$ for LLaMA2-7B and $\tau = 7$ for LLaMA3.1-8B across all sparsity levels). We explore τ values in the range of 3 to 11 with a step size of 2 to select optimal values. The calibration set is sampled from WikiText2 (Merity et al. 2016) training data, which is the most commonly used calibration dataset in the model compression literature (An et al. 2024; Wang et al. 2024; Wei et al. 2024). All steps use 128 samples with a sequence length of 2048.

Evaluation. We conduct experiments on LLaMA-1 (Touvron et al. 2023a), LLaMA-2 (Touvron et al. 2023b), LLaMA-3.1 (Grattafiori et al. 2024), and Qwen-2.5 (Qwen et al. 2025), covering model sizes from 7B to 70B and sparsity levels from 20% to 40%. Perplexity is evaluated on the WikiText2 (Merity et al. 2016) test set. Downstream performance is measured by average accuracy across standard benchmarks, including ARC-c, ARC-e (Mihaylov et al. 2018a), WinoGrande (Sakaguchi et al. 2021), BoolQ (Clark et al. 2019), HellaSwag (Zellers et al. 2019), OpenBookQA (Mihaylov et al. 2018b), and PIQA (Bisk et al. 2020). These tasks follow standard evaluation practices in model compression. Additionally, we use the MMLU (Hendrycks et al. 2021) benchmark to more comprehensively assess model capabilities.

Main Results

Perplexity and Zero-Shot Accuracy. Tab. 1 compares different methods on LLaMA models in terms of perplexity and average zero-shot accuracy across 7 downstream tasks. Our method exhibits strong adaptability and can be effectively integrated with advanced pruning strategies

Sparsity	Method	BoolQ	HellaS.	WinoG.	ARC-e	ARC-c	OBQA	PIQA
0%	Dense	77.74	76.02	68.98	74.58	46.25	44.20	79.05
30%	FLAP (AAAI 2024)	44.71	56.58	61.72	53.83	31.23	37.00	71.27
	SliceGPT (ICLR 2024)	38.32	49.09	60.69	49.58	31.74	32.60	64.91
	SoBP (EMNLP 2024)	**71.19**	67.27	66.22	59.81	37.63	38.40	73.50
	LLM surgeon (ICLR 2024)	-	60.72	61.09	63.09	36.69	-	**73.56**
	ModeGPT (ICLR 2025)	-	63.26	67.32	63.26	38.73	-	70.40
	OBC (NeurIPS 2023)	64.68	64.34	66.06	62.88	36.43	38.20	72.58
	O-FANG (Ours)	63.06	**67.33**	**67.32**	**67.34**	**39.93**	**40.40**	73.45
40%	SliceGPT (Ashkboos et al. 2024)	-	34.80	53.43	36.49	24.57	-	54.90
	LLM surgeon (van der Ouderaa et al. 2024)	-	48.04	54.38	52.31	30.29	-	**69.26**
	ModeGPT (Lin et al. 2024)	-	53.01	61.96	49.45	30.03	-	64.96
	OBC (Kurtić, Frantar, and Alistarh 2023)	**63.00**	52.04	60.62	55.68	31.14	37.60	67.19
	O-FANG (Ours)	61.19	**58.72**	**64.48**	**58.16**	**33.11**	**40.20**	68.82

Table 2: Zero-Shot Accuracy of Compressed LLaMA2-7B on Downstream Tasks.

such as OBC and FLAP, resulting in O-FANG and F-FANG that further enhance overall performance. It consistently achieves lower perplexity while improving accuracy, performing comparably to or better than accuracy-leading methods such as SoBP. In most settings, perplexity remains lower, with only minor increases in a few cases. Overall, it delivers the best overall performance, consistently ranking among the top across both metrics.

Tab. 2 reports the zero-shot accuracy on downstream tasks for the compressed LLaMA2-7B model using different pruning methods. O-FANG improves accuracy by 1%–5% over OBC on nearly all tasks, with only a slight drop on BoolQ. It also outperforms other advanced methods such as LLM surgeon and ModeGPT. These results demonstrate the robustness and generalization advantages of the proposed approach.

Results on More Advanced LLMs and benchmark. We further evaluate our method on Qwen-2.5 and LLaMA-3.1 models. Results show that our approach consistently improves the downstream accuracy of OBC by a significant margin, even on these more advanced LLMs. This demonstrates the broad applicability of the proposed method.

The MMLU (Hendrycks et al. 2021) dataset is designed to evaluate the capabilities of LLMs across a wide range of language understanding tasks. It serves as a robust benchmark for assessing the generalization ability. As shown in Tab 4, integrating our method improves the baseline's zero-shot accuracy on MMLU, indicating enhanced generalization performance.

Model	Sparsity	Method	PPL↓	Avg↑
LLaMA-3.1 8B	20%	OBC	9.17	56.88
		O-FANG	**8.31**	**64.53**
Qwen2.5 7B	40%	OBC	14.04	48.82
		O-FANG	**11.72**	**52.14**

Table 3: Performance of Pruned LLaMA-3.1-8B and Qwen2.5-7B models.

Model	Sparsity	Method	Acc↑
LLaMA2-7B	30%	OBC	24.68
		O-FANG	**26.25**
LLaMA2-13B	30%	OBC	38.08
		O-FANG	**42.37**

Table 4: Zero-shot Accuracies of compressed models on the MMLU benchmark.

Ablation Experiments

Ablation Study on Key Components. The ablation results in Tab. 5 highlight the contribution of each module to overall performance. Experiments are conducted on the LLaMA2-7B model. Adaptive Sparsity Allocation (ASA) achieves a better balance between accuracy and sparsity, improving accuracy by approximately 1.3% over the baseline. Incorporating Shared Neuron Group Retention (SNGR) further improves accuracy by around 0.5%, as it preserves neurons that are important for handling diverse context types. Applying Function-Aware Pruning (FAP) provides an additional gain of 0.2%-0.6%, reflecting improved generalization. The full method, combining all modules, achieves the highest accuracy at both sparsity levels, confirming their complementary benefits.

ASA	SNGR	FAP	30%	40%
			57.88	52.47
✓			59.17 (+1.29)	53.90(+1.43)
✓	✓		59.64 (+0.47)	54.38 (+0.48)
✓	✓	✓	**59.83** (+0.19)	**54.96** (+0.58)

Table 5: Ablation study on LLaMA2-7B under 30% and 40% sparsity. The three abbreviations refer to Adaptive Sparsity Allocation (ASA), Shared Neuron Group Retention (SNGR), and Function-Aware Pruning (FAP). Average zero-shot average accuracy of 7 downstream tasks is reported.

Grouping and Re-weighting Method. Tab. 6 compares our proposed neuron grouping method with a random grouping baseline, where neurons in each layer are randomly assigned to groups and one group is randomly selected as the shared group. The results show that our method more effectively clusters neurons with similar functions, enabling the function-aware pruning strategy to work as intended.

Tab. 6 compares several reweighting strategies. Reverse assigns higher weights to less semantically related token clusters; Uniform uses equal weights; Only-Matched uses only the assigned token cluster for each group. The average accuracy follows: Ours $>$ Uniform $>$ Reverse $>$ Only-Matched. These results demonstrate that the proposed reweighting strategy effectively helps each group focus on neurons most relevant to its function. The low accuracy of Only-Matched is due to limited token coverage during pruning.

Experiment	Method	30%	40%
Grouping	Random	59.20	53.94
	Ours	**59.83**	**54.96**
Re-weight	Reverse	59.05	52.44
	Uniform	59.47	54.77
	Only-Matched	55.90	51.01
	Ours	**59.83**	**54.96**

Table 6: Ablation of Function-Aware Pruning Strategy: Grouping and Reweighting Methods. Average zero-shot average accuracy of 7 downstream tasks is reported and tested on LLaMA2-7B.

Adaptive Sparsity Allocation. Different strategies for allocating sparsity across blocks are compared in Tab. 7. We evaluate a Taylor-based method, which assigns sparsity based on output sensitivity, and our approach, which uses functional complexity. Experimental results show that our approach yields higher average accuracy, suggesting that functional complexity is a more effective criterion for sparsity allocation than output sensitivity.

Model	Sparsity	Taylor	FC (Ours)
LLaMA2-7B	30%	58.85	**59.83**
LLaMA2-13B	40%	52.17	**61.16**

Table 7: Effectiveness of Functional Complexity in Adaptive Sparsity Allocation. Average zero-shot average accuracy of 7 downstream tasks is reported.

Additional Experimental Analysis

Compression Time. Fig.3 presents the execution time analysis of algorithms on LLaMA2-7B. In our method, context clustering is the most time-consuming step, mitigated by PCA-based dimensionality reduction, which keeps the total time within one hour. Although less efficient than the fastest baseline, the runtime remains substantially lower than ModeGPT, indicating acceptable time cost. As shown in Tab.2, our method achieves higher accuracy than both baselines under this acceptable cost.

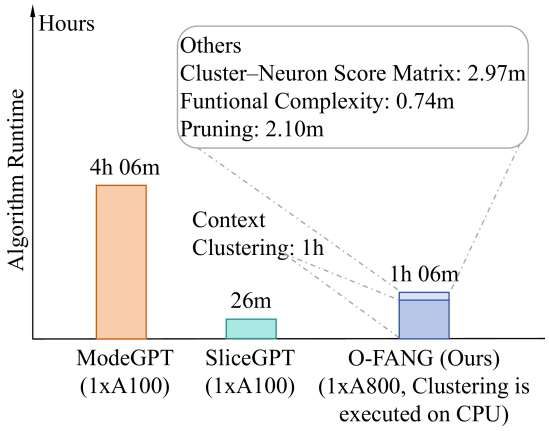

Figure 3: Execution Efficiency Comparison of Different Algorithms (LLaMA2-7B, 40% Sparsity).

Comparison with Downstream-Data-Based Calibration. Another way to improve downstream accuracy is to use downstream task data as the calibration set. Tab. 8 compares this approach with ours. The main difference is the calibration source: the baseline uses Alpaca (instruction-tuning data) (Taori et al. 2023), while our method uses WikiText (pretraining data) with function-aware pruning. All other settings, including sparsity allocation and the number of calibration tokens, are kept consistent. Results show that both approaches achieve improvements in downstream accuracy, but our method is more effective at preserving perplexity.

Settings	Method	Calibration	PPL ↓	Avg ↑
7B	OBC+ASA	Alpaca	10.24	60.35
30%	O-FANG	WikiText	7.23	59.83
13B	OBC+ASA	Alpaca	9.35	64.30
30%	O-FANG	WikiText	6.17	64.21

Table 8: Downstream Calibration vs. Function-Aware Pruning. Results are reported for OBC with Adaptive Sparsity Allocation (ASA) and O-FANG, evaluated on LLaMA2.

Conclusion

In this work, we propose Function-Aware Neuron Grouping (FANG), a post-training pruning framework inspired by the functional specialization observed in LLMs. By grouping neurons based on functional roles, reweighting importance scores, retaining shared neurons, and adaptively allocating sparsity, FANG improves generalization without compromising efficiency. Experiments across multiple models and sparsity levels show that FANG outperforms existing methods such as FLAP and OBC, achieving 1.5%-8.5% higher downstream accuracy while maintaining low perplexity.

Acknowledgements

This work was supported by the National Key R&D Program of China under Grant No. 2022ZD0160601, and National Natural Science Foundation of China (Grant No. 62276260, 62206290).

References

An, Y.; Zhao, X.; Yu, T.; Tang, M.; and Wang, J. 2024. Fluctuation-based adaptive structured pruning for large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 10865–10873.

Ashkboos, S.; Croci, M. L.; Nascimento, M. G. d.; Hoefler, T.; and Hensman, J. 2024. Slicegpt: Compress large language models by deleting rows and columns. *arXiv preprint arXiv:2401.15024*.

Bisk, Y.; Zellers, R.; Gao, J.; Choi, Y.; et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 7432–7439.

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

Cao, S.; Sanh, V.; and Rush, A. M. 2021. Low-Complexity Probing via Finding Subnetworks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 960–966.

Clark, C.; Lee, K.; Chang, M.-W.; Kwiatkowski, T.; Collins, M.; and Toutanova, K. 2019. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2924–2936.

Dai, D.; Dong, L.; Hao, Y.; Sui, Z.; Chang, B.; and Wei, F. 2022. Knowledge Neurons in Pretrained Transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 8493–8502.

De Cao, N.; Schlichtkrull, M. S.; Aziz, W.; and Titov, I. 2020. How do Decisions Emerge across Layers in Neural Models? Interpretation with Differentiable Masking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 3243–3255.

Frantar, E.; and Alistarh, D. 2022. Optimal brain compression: A framework for accurate post-training quantization and pruning. *Advances in Neural Information Processing Systems*, 35: 4475–4488.

Frantar, E.; and Alistarh, D. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, 10323–10337. PMLR.

Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Gurnee, W.; Nanda, N.; Pauly, M.; Harvey, K.; Troitskii, D.; and Bertsimas, D. 2023. Finding Neurons in a Haystack: Case Studies with Sparse Probing. *Transactions on Machine Learning Research*.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations*.

JAISWAL, A. K.; Gan, Z.; Du, X.; Zhang, B.; Wang, Z.; and Yang, Y. 2024. Compressing LLMs: The Truth is Rarely Pure and Never Simple. In *The Twelfth International Conference on Learning Representations*.

Kurtić, E.; Frantar, E.; and Alistarh, D. 2023. Ziplm: Inference-aware structured pruning of language models. *Advances in Neural Information Processing Systems*, 36: 65597–65617.

LeCun, Y.; Denker, J.; and Solla, S. 1989. Optimal brain damage. *Advances in neural information processing systems*, 2.

Lin, C.-H.; Gao, S.; Smith, J. S.; Patel, A.; Tuli, S.; Shen, Y.; Jin, H.; and Hsu, Y.-C. 2024. Modegpt: Modular decomposition for large language model compression. *arXiv preprint arXiv:2408.09632*.

Ling, G.; Wang, Z.; and Liu, Q. 2024. Slimgpt: Layer-wise structured pruning for large language models. *Advances in Neural Information Processing Systems*, 37: 107112–107137.

Liu, Z.; Wang, J.; Dao, T.; Zhou, T.; Yuan, B.; Song, Z.; Shrivastava, A.; Zhang, C.; Tian, Y.; Re, C.; et al. 2023. Deja vu: Contextual sparsity for efficient llms at inference time. In *International Conference on Machine Learning*, 22137–22176. PMLR.

Men, X.; Xu, M.; Zhang, Q.; Wang, B.; Lin, H.; Lu, Y.; Han, X.; and Chen, W. 2024. Shortgpt: Layers in large language models are more redundant than you expect. *arXiv preprint arXiv:2403.03853*.

Merity, S.; Xiong, C.; Bradbury, J.; and Socher, R. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Mihaylov, T.; Clark, P.; Khot, T.; and Sabharwal, A. 2018a. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*.

Mihaylov, T.; Clark, P.; Khot, T.; and Sabharwal, A. 2018b. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2381–2391.

Molchanov, P.; Mallya, A.; Tyree, S.; Frosio, I.; and Kautz, J. 2019. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11264–11272.

Molchanov, P.; Tyree, S.; Karras, T.; Aila, T.; and Kautz, J. 2016. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*.

Qu, X.; Dong, D.; Hu, X.; Zhu, T.; Sun, W.; and Cheng, Y. 2024. Llama-moe v2: Exploring sparsity of llama from perspective of mixture-of-experts with post-training. *arXiv preprint arXiv:2411.15708*.

Qwen; :; Yang, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Wei, H.; Lin, H.; Yang, J.; Tu, J.; Zhang, J.; Yang, J.; Yang, J.; Zhou, J.; Lin, J.; Dang, K.; Lu, K.; Bao, K.; Yang, K.; Yu, L.; Li, M.; Xue, M.; Zhang, P.; Zhu, Q.; Men, R.; Lin, R.; Li, T.; Tang, T.; Xia, T.; Ren, X.; Ren, X.; Fan, Y.; Su, Y.; Zhang, Y.; Wan, Y.; Liu, Y.; Cui, Z.; Zhang, Z.; and Qiu, Z. 2025. Qwen2.5 Technical Report. arXiv:2412.15115.

Sakaguchi, K.; Bras, R. L.; Bhagavatula, C.; and Choi, Y. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9): 99–106.

Sun, M.; Liu, Z.; Bair, A.; and Kolter, J. Z. 2024. A Simple and Effective Pruning Approach for Large Language Models. In *The Twelfth International Conference on Learning Representations*.

Tang, S.; Sieberling, O.; Kurtic, E.; Shen, Z.; and Alistarh, D. 2025. DarwinLM: Evolutionary Structured Pruning of Large Language Models. *arXiv preprint arXiv:2502.07780*.

Taori, R.; Gulrajani, I.; Zhang, T.; Dubois, Y.; Li, X.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6): 7.

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

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

van der Ouderaa, T. F.; Nagel, M.; Van Baalen, M.; and Blankevoort, T. 2024. The LLM Surgeon. In *The Twelfth International Conference on Learning Representations*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, X.; Zheng, Y.; Wan, Z.; and Zhang, M. 2024. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*.

Wei, J.; Lu, Q.; Jiang, N.; Li, S.; Xiang, J.; Chen, J.; and Liu, Y. 2024. Structured Optimal Brain Pruning for Large Language Models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 13991–14007.

Xia, M.; Gao, T.; Zeng, Z.; and Chen, D. 2023. Sheared llama: Accelerating language model pre-training via structured pruning. *arXiv preprint arXiv:2310.06694*.

Xu, M.; Yin, W.; Cai, D.; Yi, R.; Xu, D.; Wang, Q.; Wu, B.; Zhao, Y.; Yang, C.; Wang, S.; et al. 2024. A survey of resource-efficient llm and multimodal foundation models. *arXiv preprint arXiv:2401.08092*.

Yin, L.; Wu, Y.; Zhang, Z.; Hsieh, C.-Y.; Wang, Y.; Jia, Y.; Li, G.; JAISWAL, A. K.; Pechenizkiy, M.; Liang, Y.; et al. 2024. Outlier Weighed Layerwise Sparsity (OWL): A Missing Secret Sauce for Pruning LLMs to High Sparsity. In *Forty-first International Conference on Machine Learning*.

Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 4791–4800.