
GLASS: Global-Local Aggregation for Inference-time Sparsification of LLMs

Amirmohsen Sattarifard* Sepehr Lavasani* Kunlin Zhang Amirhossein Rajabpour
Huawei Technologies Canada Co., Ltd.

Hanlin Xu Fengyu Sun
Huawei Technologies Ltd.

Negar Hassanpour† Chao Gao†
Huawei Technologies Canada Co., Ltd.

Abstract

Inference-time sparsification is a promising path to deploy large language models (LLMs) on resource-constrained devices, yet existing training-free methods typically estimate feedforward network (FFN) neuron importance from the input prompt alone. We show this prompt-only signal is often unreliable, especially for short prompts and long-form decoding, leading to inaccurate masks and degraded generation fidelity. We propose GLASS, a plug-and-play, training-free framework that stabilizes dynamic FFN pruning by aggregating two complementary views of neuron criticality: local prompt-specific activations and a global model-intrinsic prior. GLASS fuses global and local signals via rank aggregation, yielding robust critical-neuron selection even when the prompt is short. We interpret GLASS as the maximum-a-posteriori consensus ranking under a permutation-based probabilistic model, providing a principled foundation for its weighted rank-aggregation rule. We apply GLASS to a diverse set of open-source LLMs, and show that it yields substantial improvements over prior training-free baselines in the challenging short-prompt, long-generation scenarios, achieving up to 45.10% lower perplexity and 25.73% lower KL divergence, while delivering significant on-device decoding speedup.

1 Introduction

Large Language Models (LLMs) have achieved strong performance across a wide range of language tasks [48, 49, 52, 21, 4], motivating their deployment on diverse consumer and enterprise devices [3, 45, 55]. Yet, their large parameter counts translate into substantial memory traffic and compute demand, making low-latency inference on consumer-grade hardware difficult [55]. In practice, some form of pruning is often required to reduce latency and memory footprint while preserving quality.

Depending on how weights are removed or skipped, pruning methods are commonly categorized as *static* or *dynamic*. Static pruning [23, 32, 54] permanently deletes a subset of parameters (one-shot or iteratively), typically followed by training/fine-tuning to recover performance. After deployment, the sparsity pattern is fixed and does not depend on the input. In contrast, dynamic pruning [45, 55, 3] conditionally skips computation based on the current input, so different requests may activate different subsets of the model.

A straightforward dynamic approach is to train a *predictor* that estimates layer-wise activation patterns and prunes computation accordingly. While effective, predictor-based methods incur (i) an extra training stage, (ii) inference-time overhead, and (iii) potentially higher memory-transfer overhead due

*Equal contribution

†Equal advising

to frequent (un)loading of neurons. As a result, *training-free* dynamic pruning (i.e., without training an auxiliary predictor and without modifying the LLM weights) is often preferred in deployment settings.

In LLMs, training-free FFN pruning methods [13, 31] estimate a prompt-derived sparsity mask and reuse it during decoding, relying on the *flocking phenomenon* [31]. However, with short prompts or long responses, activation patterns may drift from the prefill mask. Updating masks online [62, 58, 57, 46, 38, 44, 25, 56] can mitigate this drift, but reduces the aggressively pruned portion of decoding and limits end-to-end speedups [29]. It also sacrifices a key edge-deployment benefit of static masks: keeping a fixed compact FFN subset resident in fast memory and avoiding repeated I/O.

We address this reliability-efficiency tension with **GLASS**, a simple global-local FFN sparsification framework. GLASS augments prompt-derived neuron importance with an independent *global* prior, computed once offline via NPS without task-specific data or model training. Using Null-Prompt Stimulation (NPS; see Sec. 3.3), GLASS estimates either neuron activation magnitude or performance impact, yielding **A-GLASS** and **I-GLASS**, respectively. At inference, it fuses this global prior with local prompt-specific activations to produce masks robust to short prompts and long generations. An overview is shown in Fig. 2, and Fig. 1 shows that global-local aggregation improves alignment with the oracle active-neuron mask.

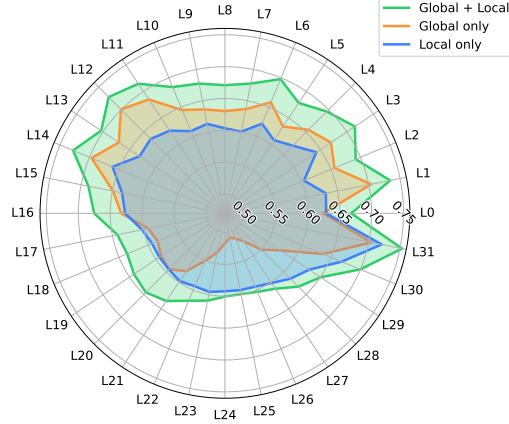


Figure 1: Jaccard similarity ($\uparrow$) between critical neuron sets identified by the Global-Local Aggregation (green), Global (orange), and Local (blue) methods using the oracle set as reference, across various layers on Llama 3 8B. Aggregating global and local information allows GLASS to more faithfully identify the oracle critical neuron set (see Sec. 4.3 and App. C.1 for details).

Contributions. We make four contributions: (i) we identify a previously unrecognized failure mode of prompt-only training-free FFN sparsification: short-prompt masks drift from decoding-time activations, hurting long-form generation; (ii) we introduce **GLASS**, a plug-and-play, training-free inference-time pruning framework that fuses *local* prompt evidence with a *global* model-intrinsic prior, yielding reliable FFN masks under short prompts and long generations; (iii) we provide a probabilistic consensus-ranking interpretation of GLASS, showing that under a Mallows-type model with squared Spearman rank distance, its weighted rank aggregation is the MAP estimator; and (iv) we achieve up to **45.10%** lower perplexity and **25.73%** lower KL divergence than the strongest training-free baseline at 50% sparsity, plus substantial on-device decoding phase speedup (up to $\sim 11\times$ on Samsung Galaxy S25 Ultra when memory residency dominates).

2 Assumptions and Objectives

2.1 Model Structure Assumption

We consider a modern Transformer [53] model composed by stacking of multiple layers each consisting of an attention block and a FFN block. Each FFN block adopts a gated structure:

$$\begin{aligned} \mathbf{z}_u &= \mathbf{x}W_{\text{up}} + \mathbf{b}_{\text{up}}, & \mathbf{a}_u &= \phi_u(\mathbf{z}_u) \\ \mathbf{z}_g &= \mathbf{x}W_{\text{gate}} + \mathbf{b}_{\text{gate}}, & \mathbf{a}_g &= \phi_g(\mathbf{z}_g) \\ \mathbf{h} &= \mathbf{a}_u \odot \mathbf{a}_g, & \mathbf{y} &= \mathbf{h}W_{\text{down}} + \mathbf{b}_{\text{down}}, \end{aligned} \quad (1)$$

Here, $\mathbf{x} \in \mathbb{R}^d$ is the input token embedding. The FFN expands the representation to a higher-dimensional space of width m , performs elementwise gating, and projects back to $\mathbb{R}^d$:

- $W_{\text{up}}, W_{\text{gate}} \in \mathbb{R}^{d \times m}$ are the expansion matrices.

- $W_{\text{down}} \in \mathbb{R}^{m \times d}$ is the projection matrix.
- $\mathbf{b}_{\text{up}}, \mathbf{b}_{\text{gate}} \in \mathbb{R}^m$, and $\mathbf{b}_{\text{down}} \in \mathbb{R}^d$ are the bias terms.
- $\phi_u(\cdot)$ is an activation function and $\phi_g(\cdot)$ is a sigmoid-like gating function.
- $\mathbf{h} \in \mathbb{R}^m$ is the FFN hidden unit vector. Its j -th element is denoted by h_j

Associated weights of a hidden unit. Each hidden unit $h_j, j \in \{1, \dots, m\}$, is associated with the j -th up-projection column ($W_{\text{up}}^{(:,j)}$), the optional j -th gate-projection column ($W_{\text{gate}}^{(:,j)}$), and the j -th down-projection row ($W_{\text{down}}^{(j,:)}$). For each output dimension $k \in \{1, \dots, d\}$, the FFN output can be written as

$$y_k = \sum_{j=1}^m \phi_u(\mathbf{x}W_{\text{up}}^{(:,j)} + b_{\text{up}}^{(j)}) \cdot g_j \cdot W_{\text{down}}^{(j,k)} + b_{\text{down}}^{(k)}, \quad k \in \{1, \dots, d\}, \quad (2)$$

where

$$g_j = \begin{cases} \phi_g(\mathbf{x}W_{\text{gate}}^{(:,j)} + b_{\text{gate}}^{(j)}), & \text{if a gate branch is used,} \\ 1, & \text{otherwise.} \end{cases} \quad (3)$$

Eq. (2) expresses the FFN output as an additive decomposition over the hidden dimension j . Sparsifying the FFN corresponds to removing or masking a subset of hidden units j , thereby eliminating their corresponding summands in the FFN output.

2.2 Dynamic Activation for FFN

The objective is to dynamically prune a subset of FFN hidden units (neurons) at inference-time, while minimizing degradation in model quality with respect to the base (unpruned) model. For each FFN block, given a budget $k < m$, we aim to select a subset of k hidden units (referred to as *critical neurons*). This can be represented as a binary mask with k non-zero entries. In the decoding phase, with the mask, model inference becomes faster due to reduced computation for each FFN.

Computation and Storage Overhead. The computation and storage overhead of dynamic FFN activation is small, requiring only a 1D binary mask of size m for each FFN layer. This overhead is almost negligible in comparison to the whole LLM.

3 GLASS: Global-Local Aggregation for FFN Sparsification

Although it is sensible to add a global measurement for assisting neuron pruning, how to compute such a metric is still unclear. A suitable definition of global importance (noted as M^g) is also unclear. In GLASS, we propose two definitions for model-intrinsic neuron importance. The first one is defined by measuring activation magnitude (noted as A^g), and the second is by measuring performance impact (I^g). For both definitions, given a model $\mathcal{M}$, we propose a null prompt stimulation (NPS) algorithm to reliably compute each metric. NPS not only eliminates the need for offline datasets, but also produces more accurate model-intrinsic importance estimates. After introducing A^g, I^g , and NPS, we describe in detail how GLASS aggregates them with local importance learned during the prefill phase.

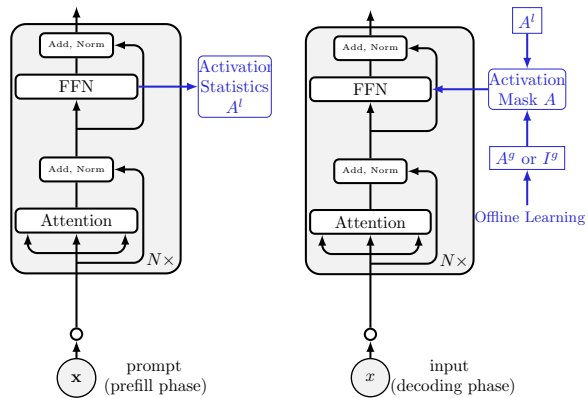


Figure 2: Overview of GLASS. We (left) learn a robust mask by fusing local (prompt) evidence with a global prior from NPS, and (right) apply the resulting mask for FFN activation pruning during generation.

3.1 Activation Magnitude A^g as Global Importance

Similar to learning importance scores at prefill, we can use average activation on a global stimulation set as global importance. For each FFN unit j , we can thus define its *global activation magnitude* as:

$$A_j^g := \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[|\hat{h}_j(\mathbf{x})| \right], \quad (4)$$

where $\hat{\mathbf{h}}(\mathbf{x}) = \mathbf{h}(\mathbf{x}) / (\|\mathbf{h}(\mathbf{x})\|_2 + \epsilon)$ denotes the ℓ_2 -normalized activation vector for token $\mathbf{x}$. The ℓ_2 normalization is to ensure that activations are comparable across tokens and layers. The resulting A_j^g serves as a model-intrinsic importance score for unit j , and will be used for critical neuron selection.

3.2 Impact I^g as Global Importance

Alternatively, we propose to define the global importance of a neuron by considering how it impacts a given loss, echoing classical Taylor pruning (e.g., [15, 35]). Let $f_\theta(\mathbf{h}(\mathbf{x}))$ be the network output, $\mathbf{y}$ the target token (label), and $\mathcal{L}(f_\theta(\mathbf{h}(\mathbf{x})), \mathbf{y})$ the cross-entropy loss of the model as a function of the activations of the current FFN layer $\mathbf{h}(x) = (h_1(x), \dots, h_m(x))$. Ablating neuron j sets $h_j \mapsto 0$, i.e. $\Delta h_j = -h_j$. Applying a first-order Taylor expansion of $\mathcal{L}$ at $\mathbf{h}$ yields

$$\Delta \mathcal{L}_j \approx \nabla_{\mathbf{h}} \mathcal{L}(\mathbf{h}) \cdot \Delta \mathbf{h} = \frac{\partial \mathcal{L}}{\partial h_j} \Delta h_j = -h_j \delta_j, \quad (5)$$

where $\delta_j := \partial \mathcal{L} / \partial h_j$. Because only the magnitude of $\Delta \mathcal{L}_j$ in Eq. (5) matters for pruning, we define the (neuron-wise) *impact score*

$$I_j^g = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[|h_j(\mathbf{x}) \delta_j(\mathbf{x})| \right]. \quad (6)$$

Neurons with larger I_j^g are expected to incur a larger local disturbance in loss when removed. As in standard Taylor pruning, this score is a first-order saliency proxy. Since GLASS removes multiple neurons simultaneously, I_j^g should be interpreted as a ranking heuristic rather than an exact estimate of the full-mask loss change. The resulting I_j^g serves as another model-intrinsic importance score and will be used for critical neuron selection.

3.3 Null Prompt Stimulation to Compute I^g and A^g

To compute global activations or neural impacts, one approach is to run the model on a dataset and collect the respective statistics [24]. However, standard corpora like Wikipedia can bias the statistics toward specific linguistic patterns. We instead propose Null Prompt Stimulation (NPS): generating text from the model itself using a null prompt (e.g., empty string or BOS token) under typical sampling settings (implementation details in App. B.3). Importantly, NPS is not merely “using no dataset”; by sampling from the model’s own predictive distribution given minimal conditioning, it directly probes the model’s intrinsic neuron utilization, eliminating any external corpus bias. Sec. 4.4 shows that NPS-derived priors consistently outperform WikiText-derived priors at every density level (see Tab. 3). For I-GLASS, gradients δ_j are computed by teacher-forcing the generated NPS sequence and using each self-generated next token as the pseudo-label; full details are in App. B.3.

3.4 Fusion of Global and Local Importance

Assume each FFN layer contains m neurons. For each neuron $j \in [m]$, let A_j^l denote the *expected activation magnitude* of FFN unit j over the prompt tokens, and let M_j^g denote its *global importance* obtained either from activation magnitude A^g or impact I^g through NPS: $A_j^l, M_j^g \in \mathbb{R}_{\geq 0}$. Directly fusing these heterogeneous quantities can be problematic because their scales, distributions, and semantics differ. This is particularly acute when fusing an activation-based local signal with an

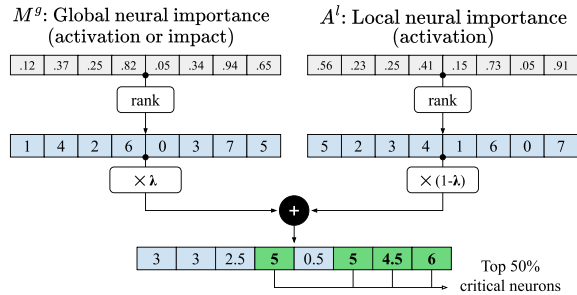


Figure 3: Rank aggregation in GLASS.

impact-based global signal, which live on an entirely different scales. We therefore first convert both signals into rank space.

Specifically, let $\text{rank}_\uparrow$ assign rank 1 to the smallest value and rank m to the largest value, and define $R_j^{(l)} = \text{rank}_\uparrow(A^l)_j$, $R_j^{(g)} = \text{rank}_\uparrow(M^g)_j$. Thus, smaller importance scores receive smaller ranks, and larger $R_j^{(l)}$ or $R_j^{(g)}$ indicates higher importance, with rank m corresponding to the most important neuron under each signal.³ Ranking makes the local and global signals comparable while preserving their ordinal preferences, and makes the fusion invariant to monotone transformations of either importance measure.

We now give a probabilistic interpretation of GLASS as a maximum-a-posteriori (MAP) consensus ranking under a permutation-based model. Let π denote the latent consensus permutation over neurons in a layer, and let $\pi^{(l)}$ and $\pi^{(g)}$ denote the observed local and global permutations induced by sorting the local and global importance scores, respectively. Here, π is an ordered list of neurons from least important to most important. Let $r(\pi) \in \{1, \dots, m\}^m$ denote the corresponding rank vector, where the j -th entry gives the rank position assigned to neuron j ; rank 1 corresponds to the least important neuron and rank m corresponds to the most important neuron. Thus, larger rank values indicate higher importance.

Following Mallows-type distance-based ranking models [33, 16], we model the two observed permutations as conditionally independent noisy observations of π using squared Spearman rank distance [12]:

$$P(\pi^{(l)} \mid \pi) \propto \exp\left(-\beta_l \|r(\pi^{(l)}) - r(\pi)\|_2^2\right), \quad P(\pi^{(g)} \mid \pi) \propto \exp\left(-\beta_g \|r(\pi^{(g)}) - r(\pi)\|_2^2\right).$$

The MAP⁴ consensus permutation is therefore

$$\pi^* = \arg \min_{\pi} \beta_l \|r(\pi^{(l)}) - r(\pi)\|_2^2 + \beta_g \|r(\pi^{(g)}) - r(\pi)\|_2^2.$$

Note that we treat the local and global rankings as conditionally independent given π . This conditional-independence assumption is a simplifying modeling approximation: both rankings are derived from the same model and need not be statistically independent in a strict sense. Its role is to yield a tractable additive consensus objective. Empirically, our Local-Only, Global-Only, and Global-Local ablations in Sec. 4.3 support that the two rankings contain complementary information rather than one signal simply dominating the other.

As shown in App. A, because every feasible rank vector is a permutation of $(1, \dots, m)$, the squared-norm $\|r(\pi)\|_2^2$ is constant across all candidate π , so the MAP solution reduces (via the rearrangement inequality) to assigning larger consensus ranks to larger values of $s_j = \beta_l R_j^{(l)} + \beta_g R_j^{(g)}$. Equivalently, the neurons are ordered from most important to least important by sorting s_j in descending order. Normalizing by the positive constant $\beta_l + \beta_g$ does not change the induced ranking. With $\lambda = \frac{\beta_g}{\beta_l + \beta_g}$, we obtain the GLASS score

$$\text{GLASS}_j = (1 - \lambda)R_j^{(l)} + \lambda R_j^{(g)}. \quad (7)$$

This is a weighted Borda-style rank aggregation rule [10]: each neuron receives a weighted sum of its local and global ranks, and larger GLASS_j indicates higher consensus importance. In practice, GLASS does not require the full permutation; it retains the k neurons with the largest scores under this induced consensus ordering.

When M_j^g represents global activation magnitude (Eq. (4)), we refer to the resulting method as A-GLASS; when it represents neural impact (Eq. (6)), we refer to it as I-GLASS. We set $\lambda = 0.5$ by default, corresponding to equal reliability of local and global evidence. This balances input-specific but short-prompt-sensitive local ranks with stable but input-agnostic global ranks, preserving the plug-and-play, training-free nature of GLASS. Our sensitivity analysis in App. C.2 shows that performance varies smoothly with λ and remains strong near $\lambda = 0.5$ on representative models.

³If exact ties occur before rank conversion, we use stable deterministic tie-breaking by neuron index to obtain a valid permutation. In practice, exact ties are rare because A^l , A^g , and I^g are floating-point averages over many tokens. If a tie occurs at the top- k boundary, the same deterministic rule is used to ensure reproducible neuron selection.

⁴For this probabilistic interpretation, we assume a uniform prior over the latent consensus permutation π , so MAP estimation reduces to maximizing the likelihood.

Table 1: Classification tasks (0-shot unnormalized accuracy) and short-generation at 50% FF sparsity.

		Classification						Short-Generation		CoQA F1 / EM	QASPER F1
		HellaSwag	PIQA	COPA	ARC-e	ARC-c	BoolQ	XSum Rouge-1 / 2 / L	CNN/DailyMail Rouge-1 / 2 / L		
Gemma 7B	I-GLASS	60.59	80.25	88.00	81.69	49.83	81.56	25.74 / 7.69 / 20.75	18.97 / 4.78 / 17.31	78.20 / 64.67	67.13
	GRIFFIN	60.62	80.03	88.00	81.99	50.09	81.56	25.36 / 7.67 / 20.48	18.26 / 4.74 / 16.75	78.27 / 64.05	67.13
Llama2 7B	I-GLASS	57.14	78.07	86.00	76.30	43.52	77.74	27.18 / 9.06 / 22.62	10.07 / 0.13 / 9.55	77.35 / 63.88	67.31
	GRIFFIN	57.14	78.07	86.00	76.30	43.52	77.74	27.18 / 9.06 / 22.62	10.07 / 0.13 / 9.55	77.35 / 63.88	67.31
Mistral 7B	I-GLASS	61.01	80.47	92.00	79.76	50.77	80.18	27.27 / 8.70 / 22.37	0.59 / 0.06 / 0.55	80.53 / 67.70	72.73
	GRIFFIN	60.97	80.69	92.00	79.71	50.09	80.37	27.03 / 8.95 / 22.39	1.03 / 0.20 / 0.92	80.39 / 66.93	72.73
OPT 6.7B	I-GLASS	50.45	75.90	80.00	63.93	30.72	65.93	19.95 / 4.54 / 16.47	12.27 / 0.69 / 11.59	68.42 / 54.85	64.54
	GRIFFIN	50.49	75.73	80.00	63.89	30.63	65.41	21.18 / 5.36 / 17.68	12.99 / 1.05 / 12.25	69.03 / 55.00	64.54
ReLU-Llama2 7B	I-GLASS	53.95	77.42	88.00	74.62	39.25	77.98	24.66 / 7.45 / 20.36	21.15 / 6.83 / 19.48	78.49 / 66.73	60.35
	GRIFFIN	53.95	77.42	88.00	74.62	39.25	77.98	24.66 / 7.45 / 20.36	21.15 / 6.83 / 19.48	78.49 / 66.73	60.35

Table 2: Perplexity (PPL) and top-100 KL Divergence (KLD) for GRIFFIN compared to our A/I-GLASS (NPS). The Imp% columns represent the improvement percentage over GRIFFIN.

Model	Metric	GRIFFIN	A-GLASS	→ Imp%	I-GLASS	→ Imp%
Gemma 7B	PPL	3.7014 (0.0316)	3.5635 (0.0368)	3.73%	3.3982 (0.0310)	8.19%
	KLD	0.6453 (0.0033)	0.5998 (0.0037)	7.05%	0.5661 (0.0036)	12.27%
Gemma 2 9B	PPL	4.0143 (0.0262)	3.5105 (0.0217)	12.55%	3.4962 (0.0218)	12.91%
	KLD	0.6970 (0.0026)	0.5877 (0.0025)	15.68%	0.5821 (0.0026)	16.48%
Gemma 2 27B	PPL	2.7729 (0.0190)	2.4860 (0.0165)	10.35%	2.4268 (0.0155)	12.48%
	KLD	0.4338 (0.0024)	0.3515 (0.0025)	18.97%	0.3349 (0.0024)	22.80%
Gemma 3n E2B	PPL	23.1611 (0.2938)	12.7155 (0.1806)	45.10%	14.4240 (0.2159)	37.72%
	KLD	1.9983 (0.0063)	1.4841 (0.0068)	25.73%	1.5677 (0.0072)	21.55%
Gemma 3n E4B	PPL	8.5984 (0.0920)	5.9075 (0.0707)	31.30%	6.3256 (0.0809)	26.43%
	KLD	1.3106 (0.0053)	0.9796 (0.0061)	25.26%	1.0190 (0.0066)	22.25%
Llama 3 8B	PPL	5.2523 (0.0434)	4.0374 (0.0324)	23.13%	4.0295 (0.0331)	23.28%
	KLD	0.9426 (0.0047)	0.7407 (0.0044)	21.42%	0.7354 (0.0046)	21.98%
Mistral 7B	PPL	5.0059 (0.0407)	4.4786 (0.0384)	10.53%	4.4860 (0.0387)	10.39%
	KLD	0.8774 (0.0041)	0.7615 (0.0045)	13.21%	0.7593 (0.0046)	13.46%
Qwen 2.5 7B	PPL	5.0840 (0.0378)	5.5823 (0.0581)	-9.80%	5.5424 (0.0593)	-9.02%
	KLD	0.6772 (0.0028)	0.6880 (0.0039)	-1.59%	0.6684 (0.0040)	1.30%
Qwen 2.5 14B	PPL	5.4330 (0.0882)	5.5823 (0.0677)	-2.75%	5.0279 (0.0701)	7.46%
	KLD	0.6557 (0.0048)	0.5947 (0.0048)	9.30%	0.5680 (0.0050)	13.38%
Phi 3 14B (medium)	PPL	3.8981 (0.0328)	4.1384 (0.0283)	-6.16%	3.7364 (0.0237)	4.15%
	KLD	0.6763 (0.0034)	0.6928 (0.0035)	-2.44%	0.6188 (0.0031)	8.50%

4 Experiments

Following prior work, we evaluate GLASS on both Classification (C) and Short-form Generation (SG) benchmarks. To fully show the merits of GLASS, we also introduce and evaluate on a Long-form Generation (LG) benchmark consisting of brief prompts and lengthy outputs. This makes LG ideal for comparing sparsification quality in long-generation scenarios.

Evaluation Datasets. C datasets include **HellaSwag** [60], **PIQA** [5], **COPA** [41], **ARC-E/C** [8], and **BoolQ** [7]. For SG, we use **XSum** [37], **CNN/DailyMail** [42, 19], **CoQA** [40], and **QASPER** [9]. Details of the C and SG datasets are provided in the App. B.1.

For the LG benchmark, we use **Alpaca** [47], a self-instruction dataset generated with GPT-3. Alpaca is suitable because it spans diverse tasks, pairs short prompts with long outputs, and is widely used for reproducible LLM evaluation. To focus on long generation, we keep samples with ground-truth outputs over 100 tokens and exclude cases where any studied base model generates fewer than 20 tokens, yielding 3,602 samples.

Evaluation Models. For C and SG tasks, we evaluate five publicly available LLMs in the 6-7 billion parameter range: **Gemma 7B** [49], **Llama 2 7B** [52], **Mistral 7B** [21], **OPT 6.7B** [61], and

Relu-LlaMA 7B [51], following GRIFFIN and TDA. These models vary in architecture, training data, and optimization strategies, providing a representative set of contemporary foundation models.

For the LG task, noticing that most models above are not capable of generating cohesive long responses, we use the instruction-tuned pretrained models listed as follows: **Gemma 7B** [49], **Gemma 2 9B & 27B** [50], **Mistral 7B** [21], **Qwen 2.5 7B & 14B** [39], **Phi 3 14B (medium)** [1], **Llama 3 8B** [14], and **Gemma 3n E2B & E4B**. In particular, Gemma 3n models adopt MatFormer strategy [11] in training, making them inherently sparse and possibly more suitable for the application of inference time sparsification algorithms like GLASS.

Evaluation Metrics. For evaluation of the C and SG tasks, following prior work, we use a set of evaluation metrics including classification accuracy, F1-Score, exact match, and ROUGE-1/2/L.

For the LG benchmark, ROUGE or exact match can be misleading because small next-token distribution shifts compound over long generations. We therefore measure deviation at each step using Perplexity (PPL) and Kullback-Leibler Divergence (KLD) (metrics details in App. B.2) over next-token vocabulary distributions, conditioned on the dense model’s generation. Both are *deviation-from-dense* metrics: the dense model has $KLD=0$ by construction, while PPL measures how well the sparsified model preserves the dense model’s generated trajectory.

4.1 Performance Comparison on C and SG Tasks

On Classification and Short-form Generation tasks, GLASS performs comparably to GRIFFIN, suggesting that global priors are most beneficial in the more challenging short-prompt/long-generation regime. The largest improvement of GLASS over GRIFFIN was from Mistral 7B on the ARC-C benchmark, with a gain of 0.68%. See Tab. 1 for details.

4.2 Performance Comparison on Long Generation

Tab. 2 shows that both A-GLASS and I-GLASS improve over GRIFFIN across most models and metrics, with I-GLASS usually giving the strongest gains. For example, I-GLASS reduces PPL/KLD by 12.91%/16.48% on Gemma 2 9B and by 12.48%/22.80% on Gemma 2 27B. These results indicate that GLASS improves both next-token uncertainty and distributional fidelity across diverse model scales and architectures.

MatFormer-style models. Largest gains appear on Gemma 3n family, which uses a MatFormer-style architecture designed for efficient on-device deployment [11]. Since MatFormer supports extracting smaller submodels from a larger trained model, these results suggest that GLASS is especially effective when structured sparsification is naturally supported. On Gemma 3n E4B, A-GLASS/I-GLASS improve PPL by 31.30%/26.43% and KLD by 25.26%/22.25%; on Gemma 3n E2B, the gains reach 45.10%/37.72% in PPL and 25.73%/21.55% in KLD.

Qwen 2.5 7B exception. The main exception is Qwen 2.5 7B, where both variants slightly degrade PPL. We hypothesize this is due to a language mismatch: under NPS, the model tends to default to Chinese generation, while Alpaca is English, making the global prior partially misaligned with the evaluation distribution. Nevertheless, I-GLASS still improves KLD by 1.30%, suggesting that the global prior retains some distributional value despite this mismatch.

4.3 How Global Aggregation Enhances Local-Only Sparsification?

To evaluate the effectiveness of aggregating global and local activation statistics, we run the following controlled experiment (details deferred to App. C.1).

Oracle-overlap analysis. To examine the mechanism behind GLASS, we perform a controlled oracle-overlap diagnostic on Llama 3 8B. We use two disjoint WikiText corpora to avoid evaluating on the same samples used to estimate A^g : one corpus estimates global activation statistics, and the other provides a post hoc active-neuron oracle reference for overlap evaluation. At 50% sparsity, Global-Local ($A^l + A^g$) produces masks that are substantially closer to this reference than either Local-Only (A^l) or Global-Only (A^g) alone. Specifically, as shown in Fig. 1 and Tab. 5, mean Jaccard increases from 0.527 (Local-Only) and 0.510 (Global-Only) to 0.604, respectively, with lower layer-wise variance. These results provide direct evidence, within this diagnostic setting,

Table 3: KLD results across Gemma 7B, Llama 3 8B, and Mistral 7B. Due to space limitation, GRIFFIN is denoted as GRFN and GLASS is denoted as GLS.

Density (%)	Gemma 7B					Llama 3 8B					Mistral 7B				
	GRFN	A-GLS (Wiki)	A-GLS (NPS)	I-GLS (Wiki)	I-GLS (NPS)	GRFN	A-GLS (Wiki)	A-GLS (NPS)	I-GLS (Wiki)	I-GLS (NPS)	GRFN	A-GLS (Wiki)	A-GLS (NPS)	I-GLS (Wiki)	I-GLS (NPS)
90	0.0578	0.0529	0.0392	0.0467	0.0347	0.0760	0.0690	0.0534	0.0627	0.0505	0.0965	0.0876	0.0694	0.0807	0.0648
80	0.1453	0.1458	0.1127	0.1324	0.1025	0.1904	0.1795	0.1438	0.1699	0.1369	0.2170	0.2098	0.1755	0.1985	0.1669
70	0.2659	0.2782	0.2243	0.2593	0.2076	0.3298	0.3237	0.2739	0.3168	0.2644	0.3638	0.3593	0.3124	0.3460	0.3031
60	0.4266	0.4563	0.3799	0.4359	0.3552	0.5207	0.5099	0.4492	0.5022	0.4434	0.5650	0.5532	0.4971	0.5419	0.4906
50	0.6453	0.7022	0.5998	0.6825	0.5661	0.9426	0.7748	0.7407	0.7646	0.7354	0.8774	0.8316	0.7615	0.8300	0.7593
40	0.9735	1.1184	0.9610	1.1091	0.9107	1.3079	1.1942	1.0902	1.1801	1.0702	1.3697	1.2614	1.1570	1.2795	1.1575
30	1.6049	1.8326	1.6001	1.8205	1.5179	2.0843	1.7930	1.6937	1.7797	1.6746	2.1444	1.8603	1.7323	1.9065	1.7429
20	3.2728	3.5411	3.1478	3.5111	2.9906	3.3302	2.8017	2.7080	2.7759	2.6770	3.3806	2.8508	2.6695	2.9881	2.7417
10	8.3718	8.5014	7.7850	8.1909	7.4167	5.1898	5.1234	4.3153	5.0352	4.4492	5.2786	5.2224	4.3848	5.1669	4.5143

that local and global signals are complementary: local activations capture input-specific salience, while global statistics stabilize the mask by preserving neurons that are consistently important across contexts.

End-to-end PPL ablation. Tab. 6 in App. C.1 shows that better neuron selection translates to generation quality: Global-Local achieves the best PPL on all three models, improving over Local-Only by 8.19%, 23.28%, and 10.39% on Gemma 7B, Llama 3 8B, and Mistral 7B, respectively. Together, these results show that GLASS benefits from synergistically fusing stable global importance with prompt-specific local evidence.

4.4 Does NPS Produce More Reliable I^g and A^g for GLASS?

We evaluate whether NPS yields better global importance estimates by comparing it with WikiText-based global statistics across activation densities from 90% to 10% on Gemma 7B, Llama 3 8B, and Mistral 7B (Tab. 3). NPS-based GLASS consistently outperforms WikiText-based variants, indicating that NPS provides more reliable I^g and A^g . The absolute KLD reductions over GRIFFIN are generally more pronounced in the low-density regime, suggesting that the global prior is especially useful under aggressive sparsification, where local-only masks are more likely to miss important neurons. For example, at 10% density on Llama 3 8B, I-GLASS (NPS) reduces KLD from GRIFFIN’s 5.1898 to 4.4492, a 14.27% improvement.

4.5 On-Device Runtime Performance

We evaluate GLASS with 50% FFN static masking applied during the decoding phase on a Samsung Galaxy S25 Ultra, using Qwen3 4B, Llama3 8B, and Gemma 7B. We run all on-device experiments using the LiteRT-LM library [27] and convert models using the LiteRT-Torch [28]. All experiments are averaged over 30 runs with a fixed prefill length of 1024 tokens for Qwen3 4B and Llama3 8B, and 64 for Gemma 7B. GLASS achieves an average decoding speedup of 20% for Qwen3 4B when generating 256 tokens. For Llama3 8B, the corresponding improvement is 42% relative to the dense baseline. In the Gemma 7B case, the 50% FFN reduction enables full RAM residency on device, yielding $\sim 11\times$ decoding speedup. App. C.3 presents the details and runtime comparisons.

5 Related Works

Inference-time sparsification of LLMs. Inference-time sparsification accelerates LLM decoding by exploiting the inherent sparsity of FFN activations, complementing traditional static pruning. Existing methods can be broadly grouped into training-based and training-free approaches. Training-based methods such as DejaVu [30], ShadowLLM [2], and PowerInfer [45, 55] rely on learned predictors or auxiliary models to decide which components to execute at inference time. In contrast, training-free methods avoid additional model training. GRIFFIN [13] summarizes FFN activation magnitudes during prefill and reuses the resulting mask during decoding. TDA [31] uses prefill activations to set layer-wise activation thresholds, while TEAL [29] greedily optimizes layer-wise sparsity levels. CATS [24] instead estimates thresholds from offline activation statistics. GLASS is closest to GRIFFIN in that it is training-free and uses a fixed prefill-time mask, but differs by augmenting prompt-local activation evidence with a global model-intrinsic prior.

Static versus dynamic masks for edge deployment. Beyond the training-free/training-based distinction, another key deployment axis is whether masks are *static* or *dynamic*. GLASS and GRIFFIN compute a fixed prefill-time mask and reuse it during decoding, avoiding per-token mask recomputation and enabling a compact subset of FFN weights to remain resident in fast memory. In contrast, DejaVu and TEAL use dynamic masks, which introduce per-token overhead and frequent weight movement, making edge deployment more challenging.

DejaVu trains per-layer predictors and recomputes active neurons at every decoding step, adding predictor overhead and preventing fixed weight residency. TEAL addresses an orthogonal problem: it determines *how much* to prune per layer, whereas GLASS determines *which* neurons to keep under a given sparsity budget. Thus, TEAL’s layer-wise allocation could in principle be combined with GLASS’s neuron selection. GRIFFIN is the most direct baseline because it is both training-free and static, but it relies only on prompt-derived local evidence; GLASS improves robustness by fusing this signal with a global prior.

Neuron Importance Estimation. The pursuit of identifying and ranking neuron importance has been a core focus of neural network research since the field’s inception. Optimal Brain Damage (OBD) [23] and Optimal Brain Surgeon (OBS) [18] are among the earliest approaches to network pruning by estimating the impact of individual parameters on training loss. They use a second-order Taylor expansion of the loss with respect to *model weights*, and compute saliency scores using the diagonal (OBD) or full (OBS) Hessian of the loss. While theoretically grounded, these methods are computationally expensive and scale poorly with model size, limiting their applicability to modern large-scale networks. PerforatedCNNs [15] apply first-order Taylor expansion for structured pruning, focusing on reducing spatial redundancy in convolutional neural networks. Similarly, Molchanov et al. [35] apply the first-order Taylor criterion to prune convolutional channels. Molchanov et al. [36] generalizes this to second-order expansions, and Kwon et al. [22] introduce a Fisher information-based, training-free pruning framework for Transformers, enhancing inference efficiency.

Null Prompt Stimulation. Null or empty prompts have been used to probe LLM behavior [20, 26]. GLASS uses them for a different purpose: estimating model-intrinsic neuron importance. By letting the model generate its own stimulation data, NPS avoids reliance on a task dataset or external corpus, reducing potential corpus bias and preserving the plug-and-play nature of GLASS. To our knowledge, GLASS is the first method we are aware of that uses null-prompt stimulation for FFN neuron-importance estimation.

Conditional computation, sparse attention, and test-time adaptation. Conditional computation adaptively allocates computation based on the input. MoE models [43] route tokens to selected experts, SkipGPT [63] performs token-aware layer skipping, and MatFormer [11] trains nested submodels that can be extracted at inference time. These methods require architectural or training-time design, whereas GLASS is applied post hoc to pretrained LLMs. Sparse attention [6, 59] is orthogonal because it accelerates attention blocks rather than FFN layers, and can in principle be combined with GLASS. Test-time training adapts model weights at inference time; GLASS instead keeps weights fixed and adapts only the FFN sparsity mask using local and global neuron-importance estimates.

6 Conclusion

In this paper, we introduced two variants of a training-free method for inference-time sparsification of FFNs in LLMs. By fusing local prompt activations with global statistics, either activation magnitudes (A-GLASS) or impact scores (I-GLASS), our methods strike a balance between context sensitivity and model-intrinsic diversity without requiring offline predictor training or incurring run-time overhead. We also showed that this rank-aggregation rule is not ad hoc, but admits a probabilistic interpretation: under a Mallows-type model with squared Spearman distance, it corresponds to the MAP consensus ranking.

Extensive experiments across multiple tasks and models demonstrate that our methods outperform prior training-free approaches in the vast majority of cases, especially under short prompts and long generations. These results demonstrate that model-intrinsic global knowledge can significantly improve prompt-driven sparsification, unlocking reliable speedups even for the short-prompt long-

generation workloads typical of edge devices. Our work opens the door for efficient, adaptive inference in LLMs, which is particularly important for resource-constrained edge devices.

Limitations and Future Work. Building on the strong performance of GLASS, we identify a few opportunities to further improve its flexibility and effectiveness: (i) currently we apply a fixed sparsity level uniformly and share a single mixing coefficient λ across all layers; jointly optimizing the sparsity pattern and learning layer-specific λ values could lead to more efficient capacity allocation; (ii) each layer is pruned independently for simplicity and compatibility with standard sparsification frameworks, however, this may neglect potential cross-layer interactions that coordinated pruning strategies could leverage.

References

- [1] M. I. Abdin, S. Ade Jacobs, A. A. Awan, J. Aneja, A. Awadallah, H. Hassan Awadalla, N. Bach, A. Bahree, A. Bakhtiari, H. Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. Technical Report MSR-TR-2024-12, Microsoft, August 2024. URL <https://www.microsoft.com/en-us/research/publication/phi-3-technical-report-a-highly-capable-language-model-locally-on-your-phone/>.
- [2] Y. Akhauri, A. F. AbouElhamayed, J. Dotzel, Z. Zhang, A. M. Rush, S. Huda, and M. S. Abdelfattah. Shadowllm: Predictor-based contextual sparsity for large language models. *arXiv preprint arXiv:2406.16635*, 2024.
- [3] K. Alizadeh, S. I. Mirzadeh, D. Belenko, S. Khatamifard, M. Cho, C. C. Del Mundo, M. Rastegari, and M. Farajtabar. Llm in a flash: Efficient large language model inference with limited memory. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12562–12584, 2024.
- [4] A. Anthropic. The claude 3 model family: Opus, sonnet, haiku; 2024. *Claude-3 Model Card*, 2024.
- [5] Y. Bisk, R. Zellers, R. L. Bras, J. Gao, and Y. Choi. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.
- [6] R. Child. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.
- [7] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*, 2019.
- [8] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*, 2018.
- [9] P. Dasigi, K. Lo, I. Beltagy, A. Cohan, N. A. Smith, and M. Gardner. A dataset of information-seeking questions and answers anchored in research papers. *arXiv preprint arXiv:2105.03011*, 2021.
- [10] J. C. de Borda. Mémoire sur les élections au scrutin. *Histoire de l’Académie Royale des Sciences*, 1781.
- [11] F. Devvrit, S. Kudugunta, A. Kusupati, T. Dettmers, K. Chen, I. Dhillon, Y. Tsvetkov, H. Hajishirzi, S. Kakade, A. Farhadi, et al. Matformer: Nested transformer for elastic inference. *Advances in Neural Information Processing Systems*, 37:140535–140564, 2024.
- [12] P. Diaconis and R. L. Graham. Spearman’s footrule as a measure of disarray. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(2):262–268, 1977.
- [13] H. Dong, B. Chen, and Y. Chi. Prompt-prompted adaptive structured pruning for efficient llm generation. *arXiv preprint arXiv:2404.01365*, 2024.

- [14] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pages arXiv-2407, 2024.
- [15] M. Figurnov, A. Ibraimova, D. P. Vetrov, and P. Kohli. Perforatedcnns: Acceleration through elimination of redundant convolutions. *Advances in neural information processing systems*, 29, 2016.
- [16] M. A. Fligner and J. S. Verducci. Distance based ranking models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 48(3):359–369, 1986.
- [17] G. H. Hardy, J. E. Littlewood, and G. Pólya. *Inequalities*. Cambridge University Press, 2nd edition, 1952.
- [18] B. Hassibi, D. G. Stork, and G. J. Wolff. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, pages 293–299. IEEE, 1993.
- [19] K. M. Hermann, T. Kociský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701, 2015. URL <http://papers.nips.cc/paper/5945-teaching-machines-to-read-and-comprehend>.
- [20] W. Jeong, D. Kim, and T. Whangbo. Scope: Stochastic and counterbiased option placement for evaluating large language models. *arXiv preprint arXiv:2507.18182*, 2025.
- [21] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mistral 7b, 2023. URL <https://arxiv.org/abs/2310.06825>.
- [22] W. Kwon, S. Kim, M. W. Mahoney, J. Hassoun, K. Keutzer, and A. Gholami. A fast post-training pruning framework for transformers. *Advances in Neural Information Processing Systems*, 35: 24101–24116, 2022.
- [23] Y. LeCun, J. Denker, and S. Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.
- [24] J.-Y. Lee, D. Lee, G. Zhang, M. Tiwari, and A. Mirhoseini. CATS: Contextually-aware thresholding for sparsity in large language models. *arXiv preprint arXiv:2404.08763*, 2024.
- [25] M. Lee, A. Ramachandran, and T. Krishna. Recap: Training-free compensation for coarse activation channel pruning in compressed llms. In *Machine Learning for Computer Architecture and Systems*, 2025.
- [26] A. Leiding, R. Van Rooij, and E. Shutova. The language of prompting: What linguistic properties make a prompt successful? *arXiv preprint arXiv:2311.01967*, 2023.
- [27] LiteRT-LM. Litert-lm: A c++ library to efficiently run language models across edge platforms. <https://github.com/google-ai-edge/LiteRT-LM>.
- [28] LiteRT-Torch. Litert-torch: Supporting pytorch models with the google ai edge tflite runtime. <https://github.com/google-ai-edge/litert-torch>.
- [29] J. Liu, P. Ponnusamy, T. Cai, H. Guo, Y. Kim, and B. Athiwaratkun. Training-free activation sparsity in large language models. *arXiv preprint arXiv:2408.14690*, 2024.
- [30] Z. Liu, J. Wang, T. Dao, T. Zhou, B. Yuan, Z. Song, A. Shrivastava, C. Zhang, Y. Tian, C. Re, et al. Deja vu: Contextual sparsity for efficient llms at inference time. In *International Conference on Machine Learning*, pages 22137–22176. PMLR, 2023.
- [31] C. Ma, M. Huang, Y. Zhang, C. Wang, Y. Wang, L. Yu, C. Liu, and W. Lin. First activations matter: Training-free methods for dynamic activation in large language models. *AAAI*, 2025.
- [32] X. Ma, G. Fang, and X. Wang. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 2023.
- [33] C. L. Mallows. Non-null ranking models. I. *Biometrika*, 44(1/2):114–130, 1957.

- [34] S. Merity, C. Xiong, J. Bradbury, and R. Socher. Pointer sentinel mixture models, 2016.
- [35] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz. Pruning convolutional neural networks for resource efficient inference. *International Conference on Learning Representations*, 2017.
- [36] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11264–11272, 2019.
- [37] S. Narayan, S. B. Cohen, and M. Lapata. Don’t give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *ArXiv*, abs/1808.08745, 2018.
- [38] J. Qi, C. Gao, Z. Ren, and Q. Chen. Deltallm: A training-free framework exploiting temporal sparsity for efficient edge llm inference. *arXiv preprint arXiv:2507.19608*, 2025.
- [39] Qwen, :, A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, H. Lin, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Lin, K. Dang, K. Lu, K. Bao, K. Yang, L. Yu, M. Li, M. Xue, P. Zhang, Q. Zhu, R. Men, R. Lin, T. Li, T. Tang, T. Xia, X. Ren, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Wan, Y. Liu, Z. Cui, Z. Zhang, and Z. Qiu. Qwen2.5 technical report, 2025. URL <https://arxiv.org/abs/2412.15115>.
- [40] S. Reddy, D. Chen, and C. D. Manning. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266, 2019. doi: 10.1162/tacl_a_00266. URL <https://aclanthology.org/Q19-1016>.
- [41] M. Roemmele, C. A. Bejan, and A. S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI spring symposium: logical formalizations of commonsense reasoning*, pages 90–95, 2011.
- [42] A. See, P. J. Liu, and C. D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1099. URL <https://www.aclweb.org/anthology/P17-1099>.
- [43] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.
- [44] J. Shin, H. Yang, and Y. Yi. Sparseinfer: Training-free prediction of activation sparsity for fast llm inference. In *2025 Design, Automation & Test in Europe Conference (DATE)*, pages 1–7. IEEE, 2025.
- [45] Y. Song, Z. Mi, H. Xie, and H. Chen. Powerinfer: Fast large language model serving with a consumer-grade gpu. In *Proceedings of the ACM SIGOPS 30th Symposium on Operating Systems Principles*, pages 590–606, 2024.
- [46] R. Storaí, J. Lee, and S.-w. Hwang. Smarter, not harder: Training-free adaptive computation for transformers. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 8147–8155, 2025.
- [47] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.
- [48] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv*, 2023.
- [49] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

- [50] G. Team, M. Riviere, S. Pathak, P. G. Sessa, C. Hardin, S. Bhupatiraju, L. Hussenot, T. Mesnard, B. Shahriari, A. Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.
- [51] S. Team. Sparse large language models with relu activation, 2023.
- [52] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- [53] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.
- [54] M. Xia, T. Gao, Z. Zeng, and D. Chen. Sheared llama: Accelerating language model pre-training via structured pruning. *arXiv preprint arXiv:2310.06694*, 2023.
- [55] Z. Xue, Y. Song, Z. Mi, X. Zheng, Y. Xia, and H. Chen. Powerinfer-2: Fast large language model inference on a smartphone. *arXiv preprint arXiv:2406.06282*, 2024.
- [56] Y. Yang, Y. Wang, C. Ma, L. Yu, E. Chersoni, and C.-R. Huang. Sparse brains are also adaptive brains: Cognitive-load-aware dynamic activation for llms. *arXiv preprint arXiv:2502.19078*, 2025.
- [57] R. Yin, Y. Li, D. Lee, and P. Panda. Duogpt: Training-free dual sparsity through activation-aware pruning in llms. *arXiv preprint arXiv:2506.20194*, 2025.
- [58] C. You, K. Wu, Z. Jia, L. Chen, S. Bhojanapalli, J. Guo, U. Evci, J. Wassenberg, P. Netrapalli, J. J. Willcock, et al. Spark transformer: Reactivating sparsity in ffn and attention. *arXiv preprint arXiv:2506.06644*, 2025.
- [59] J. Yuan, H. Gao, D. Dai, J. Luo, L. Zhao, Z. Zhang, Z. Xie, Y. Wei, L. Wang, Z. Xiao, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23078–23097, 2025.
- [60] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.
- [61] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, T. Mihaylov, M. Ott, S. Shleifer, K. Shuster, D. Simig, P. S. Koura, A. Sridhar, T. Wang, and L. Zettlemoyer. Opt: Open pre-trained transformer language models, 2022.
- [62] Z. Zhang, Z. Liu, Y. Tian, H. Khaitan, Z. Wang, and S. Li. R-sparse: Rank-aware activation sparsity for efficient llm inference. *arXiv preprint arXiv:2504.19449*, 2025.
- [63] A. Zhao, F. Ye, Y. Fan, J. Tong, J. Xiong, Z. Fei, H. Su, and X. Shen. Skipgpt: Each token is one of a kind. In *Forty-second International Conference on Machine Learning*.

A Permutation-Based Consensus View of GLASS

For completeness, we provide the full derivation of the MAP consensus rule used in Section 3.4. Let π denote the latent consensus permutation over m neurons. Let $\pi^{(l)}$ and $\pi^{(g)}$ denote the observed local and global permutations induced by sorting the local and global importance scores, respectively. The permutation $\pi = (\pi_1, \dots, \pi_m)$ lists neurons in ranked order from least important to most important, while the rank vector $r(\pi)$ records, for each neuron index j , the rank position assigned to j . Under this convention, rank 1 corresponds to the least important neuron and rank m corresponds to the most important neuron. Equivalently, $r(\pi)$ is the inverse permutation written in vector form. If exact ties occur in the local or global importance scores before rank conversion, we use stable deterministic tie-breaking by neuron index to obtain a valid permutation. In practice, exact ties are rare because A^l , A^g , and I^g are floating-point averages over many tokens. If a tie occurs at the top- k boundary, the same deterministic rule is used to ensure reproducible neuron selection.

We assume the following conditional ranking models:

$$P(\pi^{(l)} \mid \pi) = \frac{1}{Z_l} \exp\left(-\beta_l \|r(\pi^{(l)}) - r(\pi)\|_2^2\right), \quad (8)$$

$$P(\pi^{(g)} \mid \pi) = \frac{1}{Z_g} \exp\left(-\beta_g \|r(\pi^{(g)}) - r(\pi)\|_2^2\right), \quad (9)$$

where $\beta_l, \beta_g > 0$ are concentration parameters controlling the reliability of the local and global rankings. The squared rank distance $\|r(\sigma_1) - r(\sigma_2)\|_2^2 = \sum_j (R_j^{(\sigma_1)} - R_j^{(\sigma_2)})^2$ is the standard *squared Spearman rank distance* (Spearman's ρ -distance), and the resulting Mallows-type model is well-defined on the space of permutations [33, 12, 16]. We use squared Spearman distance because GLASS operates on rank-position vectors and because this distance penalizes the magnitude of rank displacement, yielding a closed-form MAP consensus ranking via weighted rank sums. For MAP estimation, the normalization constants do not affect the optimizer and can be dropped.

We also assume a uniform prior over the latent consensus permutation, $P(\pi) = 1/m!$. Therefore,

$$P(\pi \mid \pi^{(l)}, \pi^{(g)}) \propto P(\pi^{(l)}, \pi^{(g)} \mid \pi),$$

so MAP estimation is equivalent to maximizing the likelihood.

As a simplifying modeling assumption, we assume conditional independence of the two observed rankings given the latent consensus permutation:

$$P(\pi^{(l)}, \pi^{(g)} \mid \pi) = P(\pi^{(l)} \mid \pi) P(\pi^{(g)} \mid \pi). \quad (10)$$

Substituting the two likelihoods,

$$P(\pi^{(l)}, \pi^{(g)} \mid \pi) \propto \exp\left(-\beta_l \|r(\pi^{(l)}) - r(\pi)\|_2^2 - \beta_g \|r(\pi^{(g)}) - r(\pi)\|_2^2\right). \quad (11)$$

The MAP estimator is

$$\pi^* = \arg \max_{\pi} P(\pi^{(l)}, \pi^{(g)} \mid \pi). \quad (12)$$

Taking logarithms and dropping additive constants independent of π gives

$$\pi^* = \arg \min_{\pi} \beta_l \|r(\pi^{(l)}) - r(\pi)\|_2^2 + \beta_g \|r(\pi^{(g)}) - r(\pi)\|_2^2. \quad (13)$$

For shorthand, define

$$r := r(\pi), \quad r_l := r(\pi^{(l)}), \quad r_g := r(\pi^{(g)}). \quad (14)$$

Then Eq. (13) becomes

$$\pi^* = \arg \min_{r \in \mathcal{P}} \beta_l \|r - r_l\|_2^2 + \beta_g \|r - r_g\|_2^2, \quad (15)$$

where $\mathcal{P}$ denotes the set of all rank vectors corresponding to permutations of $(1, \dots, m)$.

Expanding the squared norms,

$$\|r - r_l\|_2^2 = \|r\|_2^2 - 2r^\top r_l + \|r_l\|_2^2, \quad (16)$$

$$\|r - r_g\|_2^2 = \|r\|_2^2 - 2r^\top r_g + \|r_g\|_2^2. \quad (17)$$

Therefore,

$$\beta_l \|r - r_l\|_2^2 + \beta_g \|r - r_g\|_2^2 = \beta_l (\|r\|_2^2 - 2r^\top r_l + \|r_l\|_2^2) + \beta_g (\|r\|_2^2 - 2r^\top r_g + \|r_g\|_2^2) \quad (18)$$

$$= (\beta_l + \beta_g) \|r\|_2^2 - 2r^\top (\beta_l r_l + \beta_g r_g) + \beta_l \|r_l\|_2^2 + \beta_g \|r_g\|_2^2. \quad (19)$$

The crucial observation is now: if r is a rank vector corresponding to some permutation, then it is always a permutation of $(1, 2, \dots, m)$. Hence

$$\|r\|_2^2 = \sum_{k=1}^m k^2 = \frac{m(m+1)(2m+1)}{6}, \quad (20)$$

which is constant over all feasible $r \in \mathcal{P}$. Moreover, since r_l and r_g are also rank vectors corresponding to permutations of $(1, \dots, m)$, they have the same squared norm:

$$\|r_l\|_2^2 = \|r_g\|_2^2 = \sum_{k=1}^m k^2.$$

Since r_l and r_g are fixed observations, all norm terms in Eq. (19) are constant with respect to the candidate consensus permutation. Therefore, the MAP problem is equivalent to

$$\pi^* = \arg \max_{r \in \mathcal{P}} r^\top (\beta_l r_l + \beta_g r_g). \quad (21)$$

Define the score vector

$$s := \beta_l r_l + \beta_g r_g. \quad (22)$$

Then Eq. (21) becomes

$$\pi^* = \arg \max_{r \in \mathcal{P}} r^\top s. \quad (23)$$

Since r must be a permutation of $(1, \dots, m)$, $r^\top s = \sum_{j=1}^m r_j s_j$ is a sum where the values $\{1, 2, \dots, m\}$ are distributed among the indices j , weighted by the fixed entries s_j . By the rearrangement inequality [17], this sum is maximized when the largest value m is paired with the index j^* such that s_{j^*} is maximal, the second-largest value $m-1$ is paired with the index achieving the second-largest s , and so on. Equivalently, the optimal permutation is obtained by sorting indices by s_j in descending order and assigning rank m to the index with the largest score, $m-1$ to the next, etc. Under our convention, larger rank values indicate higher importance. Therefore, the consensus ordering from most important to least important is obtained by sorting neurons in descending order of s_j . Writing

$$R_j^{(l)} := [r_l]_j, \quad R_j^{(g)} := [r_g]_j, \quad (24)$$

we have

$$s_j = \beta_l R_j^{(l)} + \beta_g R_j^{(g)}. \quad (25)$$

Finally, normalizing by $\beta_l + \beta_g$ does not change the induced ordering. Let

$$\lambda = \frac{\beta_g}{\beta_l + \beta_g}. \quad (26)$$

Then

$$\frac{s_j}{\beta_l + \beta_g} = (1 - \lambda) R_j^{(l)} + \lambda R_j^{(g)}. \quad (27)$$

Since multiplication by a positive constant preserves ordering, the same MAP permutation is obtained by sorting in descending order according to

$$(1 - \lambda) R_j^{(l)} + \lambda R_j^{(g)}. \quad (28)$$

Thus, under the convention where larger rank values indicate higher importance, the GLASS score arises as the MAP consensus score under this Mallows-type ranking model with squared Spearman rank distance. In practice, GLASS only needs the k neurons with the largest GLASS scores under this induced ranking, not the entire permutation.

Remark on uniqueness and ties. The argmax is unique whenever all s_j are distinct. If ties occur in the combined score s_j , multiple consensus permutations achieve the same MAP objective value, and any consistent tie-breaking rule recovers a valid optimum. This is separate from ties in the original local or global importance scores before rank conversion, which we handle by stable deterministic tie-breaking as described above.

B Additional Experiment Details

B.1 Benchmark Details

B.1.1 Classification Benchmark

HellaSwag [60] is a commonsense inference dataset where the task is to choose the most plausible continuation of a given context from four choices. **PIQA** [5] is a physical commonsense reasoning dataset focused on selecting the most feasible solution to everyday tasks involving physical interactions. **COPA** [41] is a causal reasoning dataset where models must choose the more plausible cause or effect of a given premise. **ARC-Easy/Challenge** [8] is a benchmark for grade-school science question answering, with Easy posing straightforward factual questions and Challenge requiring complex reasoning. **BoolQ** [7] is a question-answering dataset consisting of yes/no questions paired with paragraphs from Wikipedia, requiring models to determine the correct answer based on the context.

B.1.2 Short-form Generation Benchmark

XSum [37] is a dataset for extreme summarization, where the goal is to generate a single-sentence summary capturing the key information from a BBC news article. **CNN/DailyMail** [42, 19] is a large-scale dataset for abstractive summarization, consisting of news articles paired with multi-sentence highlights as summaries. **CoQA** [40] is a conversational question answering dataset where models answer a sequence of interconnected questions grounded in a passage. **QASPER** [9] is a dataset of question-answer pairs over scientific papers, designed for training and evaluating systems on long-context, evidence-based QA.

B.2 Details of Evaluation Metrics

B.2.1 Perplexity (PPL)

Perplexity (PPL) gauges how surprised a language model is when predicting the next token in a sequence. It is the exponentiated average negative log-likelihood per token:

$$\text{PPL} = \exp\left(-\frac{1}{N} \sum_{i=1}^N \log p_{\theta}(x_i)\right), \quad (29)$$

where x_i is the i -th token and $p_{\theta}(x_i)$ is the model’s output probability density under parameters θ . Lower values indicate better predictive performance.

We use the unpruned model’s generated response as the reference for computing PPL, which measures how unlikely the dense model’s generated trajectory is under the sparsified model. This protocol evaluates how well the sparsified model preserves the dense model’s chosen continuation. Values reported in the main tables therefore quantify how much harder the sparsified model finds the dense model’s tokens compared with the dense reference trajectory.

B.2.2 Kullback-Leibler Divergence (KLD)

The average token-level Kullback-Leibler Divergence (KLD) between the reference distribution P_i and the model distribution Q_i is

$$\text{KLD} = \frac{1}{N} \sum_{i=1}^N \sum_{v \in \mathcal{V}} P_i(v) \log \frac{P_i(v)}{Q_i(v)}, \quad (30)$$

where N is the sequence length, $v \in \mathcal{V}$ the vocabulary, P_i the reference token distribution, and $Q_i = \text{softmax}(\text{logits}_i)$ is the model-predicted distribution for the i -th position. When the sparsified

model coincides with the dense reference, $P_i = Q_i$ and $\text{KLD} = 0$ exactly, so the values reported in Tab. 2 directly quantify deviation from the dense baseline.

Since the vocabulary is very large, we report an approximate top-100 KLD computed over the 100 tokens with highest probability under the unpruned model. We restrict both distributions to this token set and renormalize them before computing KL. Empirically, this subset captures almost all the probability mass, with their summed probabilities effectively equal to 1 in our evaluated settings.

B.2.3 Classification Accuracy

Classification accuracy is the share of predictions that exactly match the ground-truth label:

$$\text{Accuracy} = \frac{1}{N} \sum_{i=1}^N \mathbf{1}\{\hat{y}_i = y_i\},$$

where N is the number of examples, y_i is the reference label, $\hat{y}_i$ the model’s label, and $\mathbf{1}\{\cdot\}$ the indicator function.

B.2.4 ROUGE-1 / 2 / L

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) compares a system output to one or more reference texts.

- **ROUGE- n** ($n = 1, 2$) measures n -gram recall:

$$\text{ROUGE-}n = \frac{\sum_{g \in \text{Ref}_n} \min(\text{Count}_{\text{Hyp}}(g), \text{Count}_{\text{Ref}}(g))}{\sum_{g \in \text{Ref}_n} \text{Count}_{\text{Ref}}(g)}.$$

- **ROUGE-L** uses the longest common subsequence (LCS) of length L_{LCS} :

$$\text{ROUGE-L} = \frac{(1 + \beta^2) R_{\text{LCS}} P_{\text{LCS}}}{\beta^2 P_{\text{LCS}} + R_{\text{LCS}}}$$

$$R_{\text{LCS}} = \frac{L_{\text{LCS}}}{|\text{Ref}|}, \quad P_{\text{LCS}} = \frac{L_{\text{LCS}}}{|\text{Hyp}|}.$$

B.2.5 Exact Match

Exact Match (EM) is the proportion of predictions whose normalized string (e.g., lower-cased, stripped of punctuation and articles) matches the reference answer exactly:

$$\text{EM} = \frac{1}{N} \sum_{i=1}^N \mathbf{1}\{\text{norm}(\hat{a}_i) = \text{norm}(a_i)\}.$$

B.2.6 F1-Score

In LLM evaluation (e.g., extractive QA or short-form generation), the F1-score is computed at the token level between the model’s answer $\hat{a}$ and the reference answer a .

After normalization (lower-casing, removing punctuation and articles) both answers are tokenized into multisets T_{pred} and T_{ref} . Let $C = |T_{\text{pred}} \cap T_{\text{ref}}|$ be the size of their multiset intersection. Then

$$P = \frac{C}{|T_{\text{pred}}|}, \quad R = \frac{C}{|T_{\text{ref}}|}, \quad F_1 = 2 \frac{P R}{P + R}.$$

For a dataset with N examples, the overall score is the mean of the per-example values:

$$F1_{\text{avg}} = \frac{1}{N} \sum_{i=1}^N F_1^{(i)}.$$

Table 4: Implementation details for A-GLASS and I-GLASS under NPS.

Item	A-GLASS	I-GLASS
NPS sequences	1000	1000
Max sequence length	1024	1024
Target for gradients	N/A	Self-generated next token
Passes	Forward only	Forward + backward
GPU	1×80GB	1×80GB
Precision	bf16	bf16
Stored statistic	Per-layer vector A^g	Per-layer vector I^g

B.3 Implementation Details

Tab. 4 summarizes the implementation details for the global-prior computation. For A-GLASS, we only collect normalized activation magnitudes during forward passes. For I-GLASS, we first generate NPS sequences from the dense model, then replay each generated sequence with teacher forcing and use the self-generated next token as the pseudo-label for the cross-entropy loss. We accumulate $|h_j(x)\delta_j(x)|$ over tokens and sequences to obtain one global impact vector per layer. These global vectors are stored once per model and reused for all downstream prompts.

NPS. Since some pretrained models tend to repeat the same token or sequence of tokens when given minimal conditioning, we applied NPS as follows to encourage diversity in generation: For the first 10 tokens, we set the temperature to 1.5 and enabled a bigram repetition penalty to maximize initial diversity. After that, we reduced the temperature to 1 and disabled the penalty to avoid generating out-of-distribution text. Here are the rest of the generation parameters: number of samples = 1000, sequence length = 1024, top-k = 20.

Compute. All experiments were performed on a single NVIDIA GPU with 80GB memory. I-GLASS is more expensive to compute offline than A-GLASS because it requires backward passes for gradient collection. However, this cost is incurred only once per model and is not paid during inference: the resulting global importance vectors are stored and reused for all prompts and deployments. At deployment time, both A-GLASS and I-GLASS only require rank fusion with local prefill statistics and application of the resulting FFN mask. We report the required passes and hardware rather than a single wall-clock precomputation time because offline runtime depends strongly on model size and implementation details.

C Additional Results and Discussions

C.1 How Global Aggregation Enhances Local-Only Sparsification?

We provide two controlled analyses to isolate the contribution of global-local aggregation: an oracle-overlap analysis, which tests *which* neurons are selected, and an end-to-end PPL ablation, which tests *how* the selected masks affect generation quality.

Oracle-overlap analysis. Using WikiText [34], we construct two disjoint corpora, each with 100 sequences of exactly 1024 tokens, formed by concatenating shorter documents. The first corpus is used to compute global activation statistics A^g . On the second corpus, we compare three 50%-sparsity variants: (i) **Local-Only**, which ranks neurons using only per-input activations A^l ; (ii) **Global-Only**, which uses only A^g ; and (iii) **Global-Local**, which aggregates A^g and A^l .

We evaluate each variant by its Jaccard similarity to an oracle critical-neuron set. Oracle critical neurons are the top-50% neurons by post-hoc decoding-time activation magnitude for each input. Although unavailable to practical dynamic pruning methods, this oracle provides a useful controlled reference. For sets A and B , the Jaccard similarity is

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}.$$

Table 5: Layer-aggregated summary of the oracle-overlap analysis in Fig. 1. We report Jaccard similarity to the oracle on Llama 3 8B, mean and std across all 32 layers, at 50% sparsity.

Variant	Mean Jaccard	Std
Local-Only	0.527	0.066
Global-Only	0.510	0.082
Global-Local (Ours)	0.604	0.048

Table 6: PPL on the Alpaca short-prompt/long-generation benchmark at 50% FFN sparsity. Numbers in parentheses are standard deviations across samples.

Variant	Gemma 7B	Llama 3 8B	Mistral 7B
Local-Only ($\lambda=0$; GRIFFIN)	3.7014 (0.0316)	5.2523 (0.0434)	5.0059 (0.0407)
Global-Only ($\lambda=1$; static global mask)	5.6611 (6.5708)	4.6872 (2.5860)	6.3699 (3.6886)
Global+Local ($\lambda=0.5$; I-GLASS)	3.3982 (0.0310)	4.0295 (0.0331)	4.4860 (0.0387)

On Llama 3 8B, Global-Local consistently better matches the oracle than either Local-Only or Global-Only. Aggregated across all 32 layers, Global-Local reaches a mean Jaccard of 0.604 ± 0.048 , compared to 0.527 ± 0.066 for Local-Only and 0.510 ± 0.082 for Global-Only. Thus, fusion improves both accuracy and consistency: it lifts the two individual signals by 14–18% and reduces layer-wise variance, indicating synergy rather than dominance by either signal.

End-to-end PPL ablation. We next test whether improved oracle overlap translates into generation quality on the Alpaca short-prompt/long-generation benchmark. Tab. 6 reports PPL at 50% sparsity for Gemma 7B, Llama 3 8B, and Mistral 7B.

Two conclusions follow. First, Global-Only is not a reliable standalone strategy: it has substantially higher variance than the prompt-conditioned methods, indicating that a static input-agnostic mask is unstable across diverse prompts. Second, Global+Local achieves the best mean PPL on all three models, improving over Local-Only by 8.19%, 23.28%, and 10.39%, respectively. Since Global-Only alone is often worse than Local-Only, the gains cannot be explained by the global prior alone; they arise from fusing stable model-intrinsic importance with prompt-specific evidence.

C.2 λ Sensitivity Sweep

We performed a fine-grained sweep over $\lambda \in [0, 1]$ with step size 0.05 on three representative instruction-tuned models (Gemma 7B, Llama 3 8B, Mistral 7B) at 50% sparsity on the Alpaca

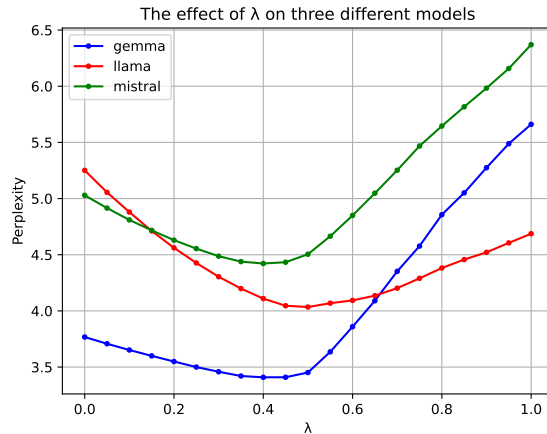


Figure 4: Sensitivity of PPL to the mixing weight λ at 50% sparsity. The minima cluster tightly around $\lambda = 0.5$ across architectures, supporting the use of an untuned default.

long-generation benchmark using I-GLASS with NPS. For each λ , we report mean PPL across the same 3,602 samples used in Tab. 2. Two findings stand out:

- The PPL landscape is *smooth and unimodal* for all three models, with no abrupt transitions. This indicates that the rank-aggregation rule degrades gracefully as λ moves away from its optimum.
- All three models attain their best PPL near $\lambda = 0.5$. The optima do not require model-specific tuning, supporting the principled choice of $\lambda = 0.5$ as the equal-reliability default ($\beta_l = \beta_g$ in the Mallows model of Sec. 3.4).

The two endpoints recover the natural baselines: $\lambda = 0$ corresponds to GRIFFIN (Local-Only) and $\lambda = 1$ corresponds to a static global mask (Global-Only). Both are *worse* than the fused $\lambda \approx 0.5$ on every model evaluated, in agreement with the Tab. 6 ablation in the main text. We provide the full sweep in Fig. 4.

C.3 Edge Device Experiments

Figs. 5a and 5b report the on-device runtime performance of Gemma 7B evaluated on a Samsung Galaxy S25 Ultra with 12 GB of RAM. Compared to Qwen3 4B and Llama3 8B (Figs. 5c and 5d), Gemma 7B has higher memory requirements, particularly during the decoding phase. As a result, the dense version of the model does not fit entirely in device memory, leading to significantly slower execution. For Gemma 7B, we therefore use a prefill length of 64 tokens and decoding lengths of 64 and 128 tokens, as larger settings (e.g., 1024 prefill tokens and 256 decode tokens) exceed the device’s memory constraints and cannot be executed reliably.

In contrast, the GLASS variant reduces the effective FFN size by 50% during decoding, which enables the model to reside fully in RAM. By reducing memory usage, GLASS shows a substantially larger performance gap relative to the dense variant compared to the other two models. Specifically, GLASS achieves an approximately $11\times$ decoding speedup over the dense Gemma 7B baseline, which is considerably higher than the improvements observed for Qwen3 4B and Llama3 8B. This case illustrates that on memory-constrained edge hardware, the I/O reduction enabled by structured FFN sparsity can be the dominant source of speedup, beyond the direct compute reduction.

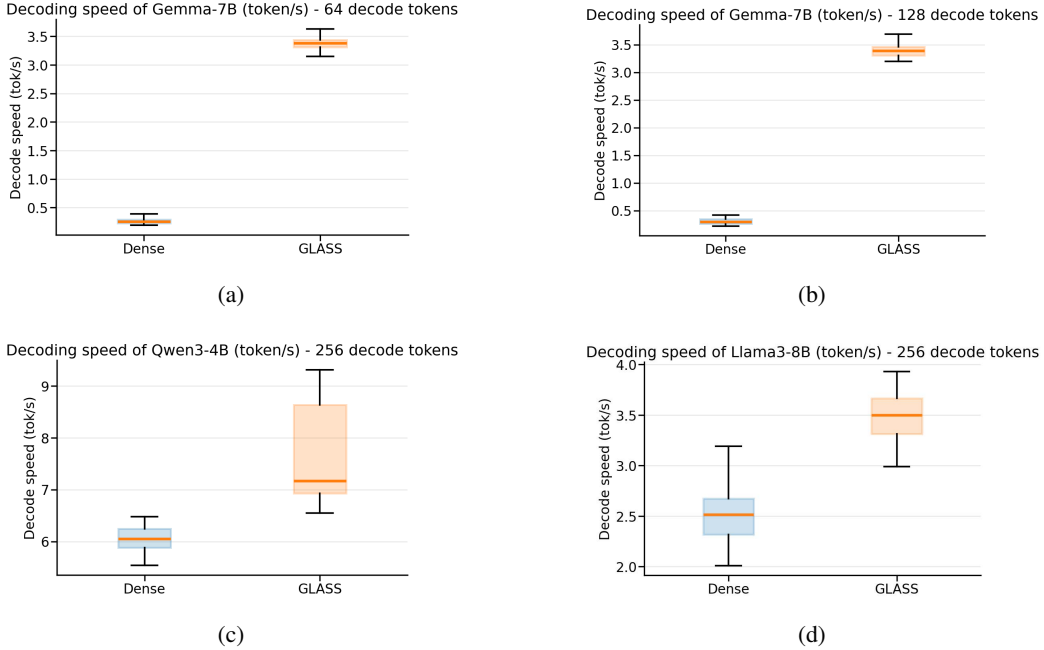


Figure 5: Wall-clock speedup of GLASS compared to the base model with 256 tokens generated during the decoding phase for the Qwen3 4B and Llama3 8B models (bottom row), and 64 and 128 generated tokens for the Gemma 7B model (top row).