TOM-Pruning: Target-aware Output Manifold for LLM Pruning

Junchen Hao[1,2] **Weikang Meng**[1,2] **Yingjian Li**[2,*] **Zheng Zhang**[1,2,*]

[1]Harbin Institute of Technology, Shenzhen, China
[2]Pengcheng Laboratory, Shenzhen, China

Abstract

Post-training pruning compresses pretrained large language models (LLMs) by removing redundant weights without full retraining. However, existing pruning criteria mainly rely on parameter statistics or generic reconstruction errors, with limited characterization of the local geometry of pruning-induced output perturbations. In this paper, we rethink LLM pruning from *a local manifold perspective* and reveal that representative methods (e.g., SparseGPT and Wanda) implicitly assume an isotropic Euclidean output metric, which treats equally sized perturbations uniformly across directions and thus overlooks target-dependent sensitivity. To overcome this structural limitation, we propose **Target-Aware Output Manifold Pruning (TOM-Pruning)**, which equips the output perturbation space with a target-aware local Riemannian metric, assigning direction-dependent geometric costs based on the perturbation's alignment with the target response. By mathematically pulling this metric back to the parameter space, we derive a basic pruning sensitivity score that inherently captures a cosine-based input–output channel affinity. To resolve the limited discriminability of this raw affinity in high-dimensional spaces, we further introduce a Competitive-Hubness Channel Affinity Mechanism to reconstruct a highly distinguishable final pruning criterion. Extensive experiments across multiple LLM families and model scales demonstrate that our manifold-driven approach generally outperforms existing no-weight-update pruning baselines under various sparsity levels and pruning patterns.

1 Introduction

Large language models (LLMs) have become central to modern natural language processing [53; 4; 6; 51; 42]. With billions of parameters, they deliver strong performance across language understanding, generation, and reasoning tasks. However, their scale incurs substantial memory, storage, and inference costs, hindering efficient deployment. To improve model efficiency, existing work explores norm–direction linear attention [33], polarity-aware linear attention [34], dynamic multi-subspace low-rank adaptation [21], and latent reasoning [22]. Alongside these approaches, compression techniques such as quantization [10; 15; 58; 55; 29] and pruning [24; 19; 14; 49] have been widely explored. This paper studies unstructured post-training pruning, which sparsifies pretrained LLMs using a small calibration set without full retraining.

Post-training pruning seeks to sparsify a pretrained dense LLM while maintaining its behavior. Compared with structured pruning that removes channels, attention heads, or layers [32; 2], unstructured pruning eliminates weights, enabling finer sparsity and stronger capacity preservation at high compression ratios. Existing LLM pruning methods focus on importance estimation for sparse mask construction. SparseGPT [14] exploits second-order reconstruction information, while Wanda [49]

*Corresponding authors: Zheng Zhang and Yingjian Li. Email: darrenzz219@gmail.com.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

combines weight magnitudes with input activation norms for efficient activation-aware pruning. More recent methods refine this paradigm through relative importance estimation [62], outlier-aware sparsity allocation [59], global optimization [3], and learnable mask modeling [12].

However, existing pruning metrics still estimate connection importance from weight statistics, input activations, or Euclidean reconstruction errors. Even reconstruction-based methods usually measure layer-output discrepancy with an isotropic Euclidean metric in the output perturbation space, assigning the same cost to different local perturbation directions. This geometry-agnostic assumption may miss pruning-induced variations that are sensitive to the target output response. Therefore, connection importance should be assessed not only by weight magnitude or activation strength, but also by the target-aware local geometry of the induced output perturbation.

To address this limitation, we rethink LLM post-training pruning through metric-induced local manifold modeling. Rather than assigning connection importance directly from weight or activation statistics, we model the output perturbation space with a local Riemannian metric, allowing different perturbation directions around the dense layer output to incur different geometric costs. By pulling this metric back through the linear layer mapping, pruning a single connection can be formulated as a constrained minimum-compensation problem in the parameter perturbation space. This view gives pruning sensitivity a geometric interpretation: the minimum local Riemannian length required to compensate for removing a connection.

Building on this perspective, we propose Target-Aware Output Manifold Pruning (TOM-Pruning), a post-training pruning method that evaluates pruning-induced perturbations with a target-aware local Riemannian metric. Unlike isotropic Euclidean metrics, the proposed metric incorporates the target output response into the output perturbation space, making pruning costs sensitive to target-related perturbation directions. When pulled back to the parameter perturbation space, this metric naturally yields a cosine-based input–output channel affinity term in the basic pruning score. However, this raw affinity is less discriminative in high-dimensional feature spaces, where input and output channel responses are often nearly orthogonal. We therefore introduce the Competitive-Hubness Channel Affinity Mechanism to enhance affinity separation, improve the identification of high-affinity input–output connections, and derive the final pruning criterion.

We evaluate TOM-Pruning on LLaMA, LLaMA-2, LLaMA-3, and Qwen3.5 under 40%–70% unstructured sparsity and 2:4/4:8 semi-structured sparsity. TOM-Pruning generally improves no-weight-update pruning baselines, with larger gains at high sparsity ratios. At 70% sparsity, it reduces Wanda's perplexity on LLaMA-3.2-1B by 26.85% and improves average zero-shot accuracy on LLaMA-3-8B with an 11.22% relative gain. It also preserves sparse-model quality under semi-structured sparsity while retaining hardware-friendly acceleration. These results highlight its practical applicability to high-sparsity and semi-structured LLM pruning.

2 Related Work

2.1 Unstructured Post-training Pruning for LLMs

Pruning has been widely studied as an effective technique for reducing the storage and computation costs of neural networks, from early second-order saliency criteria to modern deep compression pipelines [24; 19; 17; 18; 13]. For large language models, post-training pruning has attracted increasing attention because it can compress pretrained models with only a small set of calibration samples and without costly full retraining. According to pruning granularity, existing LLM pruning methods can be roughly divided into structured pruning, semi-structured pruning, and unstructured pruning. Structured pruning removes channels, heads, layers, or blocks for direct dense acceleration, but usually requires recovery training to compensate the removed structures [23; 32]. Semi-structured pruning adopts hardware-friendly sparse patterns such as N:M sparsity to balance acceleration and flexibility [12; 16]. Compared with these strategies, unstructured pruning introduces finer-grained weight-level sparsity and usually preserves model performance better under high sparsity ratios [17; 14; 49]. Since retraining LLMs is expensive, recent pruning studies increasingly adopt the post-training setting and rely only on calibration data [14; 49; 62; 59]. SparseGPT [14] proposes a one-shot post-training pruning framework that uses Hessian-based reconstruction metrics for weight removal and compensation. Wanda [49] designs an activation-aware pruning score by combining weight magnitudes with input activation norms, avoiding both retraining and weight updates. RIA [62] refines activation-aware pruning by jointly considering relative importance and activations for post-training

sparse LLMs. OWL [59] adjusts layer-wise sparsity ratios according to the distribution of activation outliers. SparseLLM [3] formulates global pruning as an optimization problem and decomposes it into tractable subproblems with auxiliary variables. ALPS [35] uses operator splitting and preconditioned conjugate-gradient refinement for one-shot LLM pruning. SAFE and SAFE+ [25] seek sparse, flat solutions through augmented-Lagrangian optimization and generalized projection. ELSA [26] adopts surrogate-free ADMM optimization to achieve high LLM sparsity. DSA [27] searches for sparsity allocation functions that map element-wise scores to layer-wise pruning ratios. ALS [28] estimates inter-layer dependency through activation correlation and allocates sparsity according to layer sensitivity. AlphaPruning [31] uses heavy-tailed self-regularization theory to determine layer-wise sparsity in a theoretical manner. ATP [20] simplifies layer-wise sparsity allocation by reducing multiple sparsity rates to a single arithmetic progression hyperparameter. DLP [5] dynamically assigns layer-wise pruning ratios according to weight and activation characteristics. These methods demonstrate the effectiveness of post-training pruning for LLM compression, but most of them still rely on weight statistics, input activations, or reconstruction errors to estimate pruning sensitivity.

2.2 Importance Estimation for Layer-wise Pruning

The design of pruning metrics is crucial for unstructured layer-wise pruning, since it directly determines which connections are removed and which parameters should be retained. Existing methods usually estimate weight importance from magnitude, activation strength, reconstruction error, gradient information, searched saliency functions, or sparse mask optimization. Magnitude pruning removes weights with small absolute values and serves as a simple but widely used baseline [17]. Pruner-Zero [11] evolves symbolic pruning metrics automatically instead of relying on manually designed saliency scores. BESA [57] parameterizes local sparsity objectives with learnable combinations of candidate pruning rates and minimizes block-wise reconstruction error. MaskLLM [12] incorporates Gumbel Softmax to learn semi-structured pruning masks in a differentiable manner. Search-LLM [48] identifies effective sub-networks through mask mutation under a training-free search setting. OATS [61] performs outlier-aware pruning through sparse and low-rank decomposition, aiming to preserve important outlier components. D^{2}Prune [56] estimates pruning sensitivity through dual Taylor expansion and attention distribution awareness. LLM-Pruner [32], SliceGPT [2], KPrune [43], and DISP-LLM [16] further explore gradient-based, row-column deletion, knowledge-preserving, and dimension-independent pruning criteria for pretrained language models. Although these methods improve pruning metrics from different perspectives, most of them mainly measure importance from the parameter side or the input-activation side. Even reconstruction-based methods usually evaluate output discrepancy under an isotropic Euclidean metric, where perturbations along different output directions are assigned the same geometric cost.

3 Rethinking Post-training Pruning from a Manifold Perspective

3.1 Preliminaries

LLM Post-training Pruning: A common objective of Post-training Pruning (PTP) is to preserve the layer output after removing weights. For a linear layer with weight matrix $W \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}}}$ and calibration input $X \in \mathbb{R}^{N \times C_{\text{in}}}$, the dense output is $Y = XW^\top$. After pruning, the weight becomes W', and the output becomes $Y' = X(W')^\top$. Layer-wise PTP aims to find a sparse weight matrix or pruning mask that keeps the discrepancy between Y and Y' small under a sparsity constraint. Instead of fixing this discrepancy as a Euclidean reconstruction error, we denote it by a general local loss:

$$\Delta\mathcal{L}_{\text{loc}} = \mathcal{D}(Y, Y'). \tag{1}$$

where $\mathcal{D}(\cdot,\cdot)$ measures the output change induced by pruning. We focus on the local effect of removing a single connection. For the i-th output channel, the original output is $Y_{:,i} = XW_{i,:}^\top$. To evaluate the pruning cost of $w_{i,j}$, we introduce a compensatory perturbation $\delta_i \in \mathbb{R}^{C_{\text{in}}}$ on the i-th weight row, with $W'_{i,:} = W_{i,:} + \delta_i^\top$. The pruned output is $Y'_{:,i} = X(W_{i,:}^\top + \delta_i)$, and the induced output perturbation is $\Delta Y_{:,i} = Y'_{:,i} - Y_{:,i} = X\delta_i$. The channel-wise local loss is written as

$$\Delta\mathcal{L}_{\text{loc}}^{(i)} = \mathcal{D}_i\big(Y_{:,i}, Y'_{:,i}\big). \tag{2}$$

If $w_{i,j}$ is exactly removed, the perturbed weight must satisfy $e_j^\top \delta_i + w_{i,j} = 0$, where e_j is the j-th standard basis vector. Therefore, the pruning sensitivity of $w_{i,j}$ is defined as the minimum

channel-wise local loss under this deletion constraint:

$$S_{i,j} = \min_{\delta_i} \Delta\mathcal{L}_{\text{loc}}^{(i)}, \quad \text{s.t.} \quad e_j^\top \delta_i + w_{i,j} = 0. \tag{3}$$

A larger $S_{i,j}$ indicates a higher compensation cost for removing $w_{i,j}$, while a smaller value suggests that the induced output change can be more easily compensated. The next subsection specifies this loss with a quadratic approximation and interprets it as a metric-induced local manifold structure.

3.2 Metric-Induced Local Manifold Modeling for PTP

Different PTP methods mainly differ in how they define or approximate the local loss $\mathcal{D}(Y, Y')$. Many layer-wise methods use Euclidean output reconstruction error, assigning uniform cost to all output perturbation directions. This scalar view does not reveal how parameter perturbations induce output variations. To expose this relation, we view the output space as a local manifold and interpret $\mathcal{D}$ as a divergence-like discrepancy, whose curvature induces a metric-induced local manifold structure.

Building on the single-connection formulation in Section 3.1, perturbing the i-th weight row induces $\Delta Y_{:,i} = X\delta_i$, mapping parameter perturbations to output perturbations. For the i-th output channel, $Y_{:,i}$ and $Y'_{:,i} = Y_{:,i} + \Delta Y_{:,i}$ are two nearby points in the output perturbation space. We define $\phi_i(\Delta Y_{:,i}) = \mathcal{D}_i(Y_{:,i}, Y_{:,i} + \Delta Y_{:,i})$ and assume that ϕ_i is twice differentiable around $\Delta Y_{:,i} = 0$. Since the dense output is the reconstruction reference, $\phi_i(0) = 0$ and the first-order term vanishes at this local minimum. Thus, the leading variation is given by the second-order Taylor approximation:

$$\Delta\mathcal{L}_{\text{loc}}^{(i)} \approx \frac{1}{2}\Delta Y_{:,i}^\top M_i \Delta Y_{:,i}. \tag{4}$$

Here, $M_i = \nabla^2_{Y'_{:,i}} \mathcal{D}_i(Y_{:,i}, Y'_{:,i})|_{Y'_{:,i}=Y_{:,i}}$ denotes the local Hessian, or its symmetric positive-definite approximation, in the output perturbation space. When M_i is symmetric positive definite, this quadratic form defines a local inner product. Therefore, M_i acts as a local Riemannian metric tensor [46] and endows the output perturbation space with a metric-induced local manifold structure. Under this metric, the infinitesimal perturbation $\Delta Y_{:,i}$ is a tangent vector in the tangent space at the current output point, and Eq. (4) gives its squared local Riemannian length. The case $M_i = I$ recovers the Euclidean reconstruction error $\frac{1}{2}\|\Delta Y_{:,i}\|_2^2$ used in layer-wise reconstruction-based pruning [14; 49], while $M_i \neq I$ captures anisotropic local sensitivity through direction-dependent costs. Since pruning is optimized over δ_i, the output-space metric is pulled back through $\Delta Y_{:,i} = X\delta_i$:

$$\Delta\mathcal{L}_{\text{loc}}^{(i)} \approx \frac{1}{2}(X\delta_i)^\top M_i (X\delta_i) = \frac{1}{2}\delta_i^\top (X^\top M_i X)\delta_i. \tag{5}$$

Thus, the output-space metric induces the pullback matrix $X^\top M_i X$ in the parameter perturbation space [1]. By denoting the parameter-space local metric as G_i, the pruning cost can be written as

$$\Delta\mathcal{L}_{\text{loc}}^{(i)} = \frac{1}{2}\delta_i^\top G_i \delta_i. \tag{6}$$

When G_i is symmetric positive definite, it further defines a metric-induced local manifold structure in the parameter perturbation space. Therefore, pruning $w_{i,j}$ can be interpreted as finding the compensation perturbation with the minimum local Riemannian length under the deletion constraint:

$$\min_{\delta_i} \frac{1}{2}\delta_i^\top G_i \delta_i = \min_{\delta_i} \frac{1}{2}\|\delta_i\|_{G_i}^2 \quad \text{s.t.} \quad e_j^\top \delta_i + w_{i,j} = 0. \tag{7}$$

Solving Eq. (7) gives the closed-form optimal compensation perturbation; substituting this perturbation into the parameter-space quadratic approximation in Eq. (6) yields the corresponding exact pruning sensitivity:

$$\delta_i^* = -\frac{w_{i,j}}{[G_i^{-1}]_{j,j}} G_i^{-1} e_j. \tag{8}$$

$$S_{i,j}^{\text{exact}} = \frac{1}{2}(\delta_i^*)^\top G_i \delta_i^* = \frac{1}{2}\|\delta_i^*\|_{G_i}^2 = \frac{w_{i,j}^2}{2[G_i^{-1}]_{j,j}}. \tag{9}$$

Appendix A.1 derives the optimal compensation perturbation and exact pruning sensitivity. This sensitivity depends on both the weight magnitude $w_{i,j}^2$ and the compensation geometry encoded by $[G_i^{-1}]_{j,j}$. Thus, pruning methods can be characterized by their construction of the output-space metric M_i and approximation of the induced parameter-space metric G_i.

3.3 Revisiting SparseGPT and Wanda

Under the metric-induced local manifold framework in Section 3.2, SparseGPT [14] and Wanda [49] can be revisited as two approximations under the same Euclidean output metric. Both measure the dense–pruned output discrepancy with the standard Euclidean reconstruction error, implicitly setting $M_i = I$ and reducing the channel-wise local loss to $\Delta\mathcal{L}_{\text{loc}}^{(i)} \approx \frac{1}{2}\|\Delta Y_{:,i}\|_2^2$, where the output perturbation manifold is isotropic and all tangent directions around $Y_{:,i}$ share the same geometric cost. Pulling this metric back through $\Delta Y_{:,i} = X\delta_i$ gives the regularized pullback metric $\widetilde{G}_i = X^\top X + \epsilon I$. Thus, SparseGPT and Wanda share the same output-space geometry but differ in their use of $\widetilde{G}_i$: SparseGPT retains the full Gram matrix to model coupled compensation across parameter directions, whereas Wanda uses a diagonal approximation that keeps only input-channel activation strength for element-wise scoring. Detailed derivations and metric interpretations are provided in Appendix A.2.

4 Target-Aware Output-Manifold Pruning

In this section, we introduce Target-Aware Output-Manifold Pruning (TOM-Pruning). Building on the metric-induced local manifold modeling in Section 3.2, TOM-Pruning formulates pruning sensitivity from an output-manifold perspective, where pruning-induced variations are measured by a target-aware local Riemannian metric. This metric is then pulled back to the parameter perturbation space to evaluate the minimum compensation cost of removing each connection. As shown in Fig. 1, TOM-Pruning consists of two stages. The first stage derives a basic pruning score from the target-aware metric. The second stage reconstructs the input–output channel affinity term induced in the basic pruning score through the Competitive-Hubness Channel Affinity Mechanism, leading to the final pruning criterion.

4.1 Target-Aware Metric for Basic Pruning Score

As discussed in Section 3.3, existing post-training pruning methods such as SparseGPT [14] and Wanda [49] can be interpreted under an isotropic Euclidean metric in the output perturbation space, i.e., $M_i = I$. From the manifold perspective, this assumption endows the output perturbation space with a flat Euclidean local manifold structure. As illustrated on the left side of Fig. 1, the 3D local manifold appears as a sphere, and its 2D equal-distance sets are concentric circles. Such a structure assigns the same geometric weight to all tangent directions around the current output point $Y_{:,i}$, and therefore fails to encode the target-output-dependent sensitivity of different local perturbation directions. To make the output perturbation manifold aware of the target response, we construct a target-aware local Riemannian metric for the i-th output channel. Let $Y_{:,i} \in \mathbb{R}^N$ denote the output response vector of the i-th channel over calibration samples. The local Riemannian metric tensor on the output perturbation manifold is defined as

$$M_i = I + \lambda_m Y_{:,i} Y_{:,i}^\top, \quad \lambda_m \geq 0. \tag{10}$$

Here, λ_m controls the strength of target-aware modulation. The proposed target-aware local Riemannian metric augments the Euclidean metric with a low-rank term $Y_{:,i}Y_{:,i}^\top$ induced by the target output response, which penalizes the projection of the output perturbation $\Delta Y_{:,i}$ onto the target response direction. Since I is symmetric positive definite and $Y_{:,i}Y_{:,i}^\top$ is symmetric positive semi-definite, $M_i = I + \lambda_m Y_{:,i}Y_{:,i}^\top$ remains symmetric positive definite for $\lambda_m \geq 0$. Thus, M_i defines a valid local Riemannian metric tensor in the output perturbation space. Under this metric, the local manifold changes from an isotropic Euclidean structure to a target-aware anisotropic structure. As illustrated in Stage A of Fig. 1, the Euclidean metric yields a spherical local manifold with concentric equal-distance sets, whereas the target-aware metric deforms them into an ellipsoid with elliptical equal-distance sets. Therefore, tangent perturbations are no longer weighted uniformly, but according to their geometric relation to the target output response. Following the pullback principle

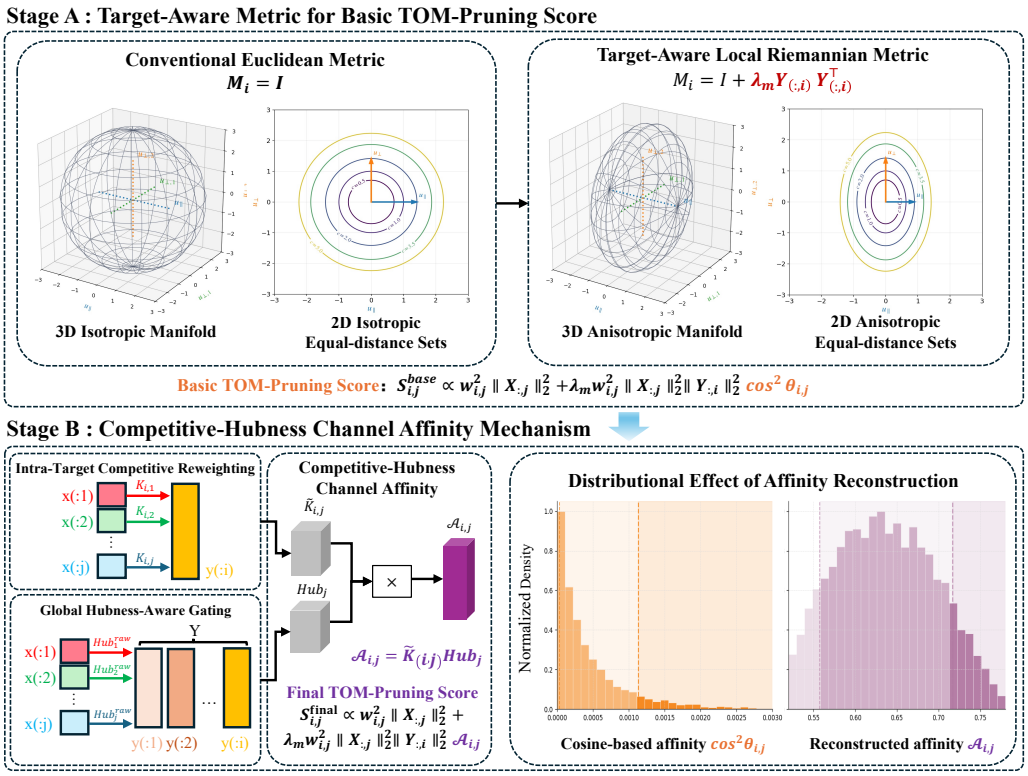

Figure 1: Overall framework of TOM-Pruning. Stage A derives the basic TOM-Pruning with the cosine-based channel affinity by pulling a target-aware local Riemannian metric from the output perturbation space to the parameter perturbation space. Stage B reconstructs $\cos^2 \theta_{i,j}$ into $\mathcal{A}_{i,j}$ through the Competitive-Hubness Channel Affinity Mechanism for the final pruning criterion.

in Section 3.2, we pull this metric back to the parameter perturbation space through $\Delta Y_{:,i} = X\delta_i$, obtaining the regularized pullback metric:

$$\widetilde{G}_i^{\text{pull}} = X^\top M_i X + \epsilon I = X^\top X + \lambda_m X^\top Y_{:,i} Y_{:,i}^\top X + \epsilon I. \tag{11}$$

Here, ϵI is a stabilization term with $\epsilon > 0$ for numerical robustness and positive definiteness. Since M_i is symmetric positive definite, $X^\top M_i X$ is symmetric positive semi-definite, and thus $\widetilde{G}_i^{\text{pull}} = X^\top M_i X + \epsilon I$ is symmetric positive definite. This regularized pullback metric defines a local inner product over parameter perturbations, endowing the parameter perturbation space with a metric-induced local manifold structure for pruning compensation. Using the full pullback metric requires matrix inversion or dense second-order coupling computation, which is expensive for large LLM layers. We therefore follow Wanda [49] and adopt a diagonal approximation that retains only the self-response term of each input channel. The diagonal pullback metric is written as

$$\widetilde{G}_{i,\text{diag}}^{\text{pull}} = \text{diag}\left(\|X_{:,1}\|_{M_i}^2 + \epsilon,\ \|X_{:,2}\|_{M_i}^2 + \epsilon,\ \ldots,\ \|X_{:,C_{\text{in}}}\|_{M_i}^2 + \epsilon \right). \tag{12}$$

Substituting this diagonal pullback metric into the connection-level sensitivity form in Section 3.2, and removing constants and the small stabilization term that do not affect ranking, yields the basic TOM-Pruning score:

$$S_{i,j}^{\text{base}} \propto w_{i,j}^2 \|X_{:,j}\|_2^2 + \lambda_m w_{i,j}^2 \|X_{:,j}\|_2^2 \|Y_{:,i}\|_2^2 \cos^2 \theta_{i,j}. \tag{13}$$

where $\theta_{i,j}$ denotes the angle between $X_{:,j}$ and $Y_{:,i}$. The second target-aware term in Eq. (13) originates from the pullback expansion of the target-aware local Riemannian metric. Specifically, the inner product $X_{:,j}^\top Y_{:,i}$ induces the input–output channel affinity term $\cos^2 \theta_{i,j}$, which measures the local association between the input channel activation $X_{:,j}$ and the target output response $Y_{:,i}$.

Stronger alignment therefore indicates a higher pruning cost, since removing $w_{i,j}$ is more likely to disturb the target output direction. Appendix A.3 provides the detailed derivation. The next subsection reconstructs this channel affinity term through the Competitive-Hubness Channel Affinity Mechanism to obtain the final pruning criterion.

4.2 Competitive-Hubness Channel Affinity Mechanism

The basic TOM-Pruning score in Section 4.1 contains $\cos^2\theta_{i,j}$ as the input–output channel affinity term, where $\theta_{i,j}$ measures the angular relation between $X_{:,j}$ and $Y_{:,i}$. However, in high-dimensional feature spaces, most input–output channel pairs are nearly orthogonal, causing $|\cos\theta_{i,j}|$ to concentrate in a narrow low-value region. As shown in Fig. 1, $\cos^2\theta_{i,j}$ exhibits limited effective support and a sharp accumulation near zero, indicating lower overall affinity resolution. Its high-affinity region is also weakly separated from the main density region, reducing the identifiability of high-affinity connections. To address this limitation, we reconstruct the input–output channel affinity term as

$$\mathcal{A}_{i,j} = \widetilde{K}_{i,j}\mathrm{Hub}_j, \tag{14}$$

where $\widetilde{K}_{i,j}$ denotes the competitively reweighted local affinity of $X_{:,j}$ with respect to the target output channel $Y_{:,i}$, and Hub_j denotes the global hubness-aware factor of $X_{:,j}$ across the layer. As illustrated by the distribution comparison in Fig. 1, $\mathcal{A}_{i,j}$ exhibits a broader main distribution and a more visible high-affinity region than $\cos^2\theta_{i,j}$, indicating that the reconstructed affinity improves the overall affinity discriminability and the identifiability of high-affinity connections. Specifically, We first reparameterize the absolute angular association strength with a nonlinear affinity kernel:

$$K_{i,j} = \exp\left(-\frac{(1-|\cos\theta_{i,j}|)^2}{2\sigma^2}\right), \quad K_{i,j}\in(0,1], \tag{15}$$

where σ is the kernel bandwidth. This nonlinear affinity kernel maps $|\cos\theta_{i,j}|$ to a scale-sensitive local affinity response, enlarging effective affinity gaps controlled by σ while preserving the ranking order. For a fixed target output response $Y_{:,i}$, however, the affinity of $X_{:,j}$ should be evaluated relative to all candidate input channels $\{X_{:,k}\}_{k=1}^{C_{\mathrm{in}}}$, since they jointly contribute to the same target response. Therefore, as shown in the intra-target competitive reweighting branch of Stage B in Fig. 1, we further perform intra-target competitive reweighting:

$$\mathrm{Share}_{i,j} = \frac{K_{i,j}}{\sum_{k=1}^{C_{\mathrm{in}}} K_{i,k}+\epsilon}, \qquad \widetilde{K}_{i,j} = K_{i,j}\mathrm{Share}_{i,j} = \frac{K_{i,j}^2}{\sum_{k=1}^{C_{\mathrm{in}}} K_{i,k}+\epsilon}. \tag{16}$$

The denominator defines the intra-target competition over input channels, while the squared numerator amplifies strongly associated candidates. Thus, $\widetilde{K}_{i,j}$ emphasizes dominant input–output channel pairs and suppresses weak ones. Since $X_{:,j}$ contributes to all output channels in $Y = XW^\top$, its importance should also reflect its cumulative role in the overall output structure. We define the local affinity contribution of connection (i,j) and the raw hubness of input channel j as

$$E_{i,j}^{\mathrm{local}} = \|Y_{:,i}\|_2^2\widetilde{K}_{i,j}, \qquad \mathrm{Hub}_j^{\mathrm{raw}} = \sum_{i=1}^{C_{\mathrm{out}}} E_{i,j}^{\mathrm{local}}. \tag{17}$$

As illustrated by the global hubness-aware branch in Stage B of Fig. 1, the raw hubness is further smoothed and normalized to obtain the global hubness-aware factor:

$$\mathrm{Hub}_j = \frac{\log\left(1+\mathrm{Hub}_j^{\mathrm{raw}}\right)}{\frac{1}{C_{\mathrm{in}}}\sum_{k=1}^{C_{\mathrm{in}}}\log\left(1+\mathrm{Hub}_k^{\mathrm{raw}}\right)+\epsilon}. \tag{18}$$

This factor measures the normalized global contribution of $X_{:,j}$ relative to the average input-channel contribution, with $\mathrm{Hub}_j > 1$ and $\mathrm{Hub}_j < 1$ indicating above-average and below-average layer-wise roles, respectively. The reconstructed channel affinity $\mathcal{A}_{i,j} = \widetilde{K}_{i,j}\mathrm{Hub}_j$ acts as a local–global soft gate, favoring connections that are locally competitive for the target output response and globally important in the layer-wise output structure. Thus, $\mathcal{A}_{i,j}$ replaces the cosine-based channel affinity term in the basic TOM-Pruning score, giving the final pruning criterion:

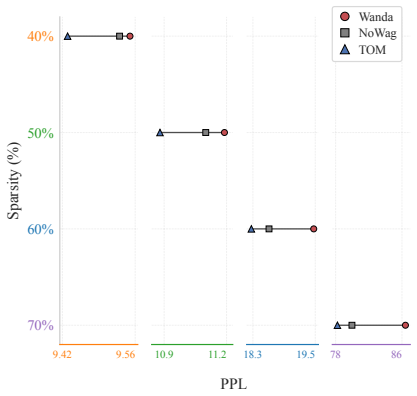

Figure 2: WikiText-2 PPL on Qwen3.5-4B across sparsity levels (lower is better). Each sparsity level uses a locally expanded PPL scale.

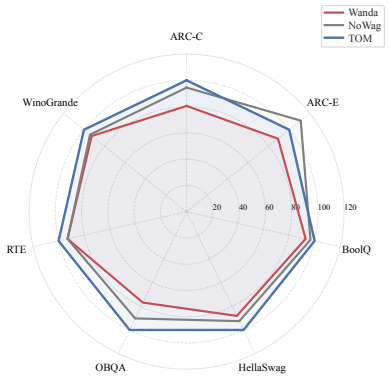

Figure 3: TOM-normalized zero-shot profile on LLaMA-3-8B at 70% sparsity. Larger radar profiles indicate stronger task preservation.

$$S_{i,j}^{\text{final}} \propto w_{i,j}^2 \|X_{:,j}\|_2^2 + \lambda_m w_{i,j}^2 \|X_{:,j}\|_2^2 \|Y_{:,i}\|_2^2 \widetilde{K}_{i,j} \text{Hub}_j. \tag{19}$$

The final pruning criterion combines weight magnitude, input activation strength, and a more discriminative reconstructed affinity term. Section 5.3 ablates each affinity component.

5 Experiments

5.1 Experimental Setup

We evaluate TOM-Pruning across multiple representative LLM families, including LLaMA [51], LLaMA-2 [52], LLaMA-3 series [50; 37; 38; 39], and Qwen3.5. All checkpoints are loaded from public releases using Hugging Face Transformers [54], and all experiments are implemented in PyTorch [44] with FP16 precision on NVIDIA A100 40GB GPUs. Following standard post-training pruning protocols, we prune only the linear layers in Transformer blocks while keeping the token embedding layer and language modeling head unchanged. For calibration, we use 128 randomly sampled C4 examples [45] with random seed 0, and cap the sequence length at 4096 tokens for LLaMA-3-series and Qwen3.5 models to reduce memory overhead. We compare TOM-Pruning with representative post-training pruning methods, including Wanda [49], SparseGPT [14], and NoWag [30]. We evaluate unstructured sparsity ratios of 0.4, 0.5, 0.6, and 0.7, together with 2:4 and 4:8 semi-structured patterns [41]. Inference efficiency is measured by layer-wise latency under 2:4 semi-structured sparsity. Section 5.3 ablates the Competitive-Hubness Channel Affinity Mechanism. All methods use the same calibration data, precision, implementation, and evaluation protocol.

5.2 Main Results

Language Modeling. Table 1 reports WikiText-2 perplexity [36] on pruned LLaMA-3 and Qwen3.5 series models across parameter scales, where lower values indicate better performance. SparseGPT [14] is included as a weight-update compensation reference, while Wanda [49], NoWag [30], and TOM-Pruning are no-weight-update pruning methods. TOM-Pruning consistently achieves lower PPL than Wanda across model families and sparsity settings, showing that target-aware output sensitivity and the Competitive-Hubness Channel Affinity Mechanism better preserve language modeling behavior. It also compares favorably with NoWag and obtains the best no-weight-update results in most settings, especially under high sparsity. At 70% sparsity, TOM-Pruning reduces Wanda's PPL from 492.58 to 360.32 on LLaMA-3.2-1B and from 194.62 to 178.29 on Qwen3.5-2B. Although SparseGPT remains competitive due to its weight-update compensation, TOM-Pruning improves pruning quality without modifying the remaining weights. Appendix B reports LLaMA/LLaMA-2 results, while Fig. 2 visualizes PPL on Qwen3.5-4B across sparsity.

Zero-shot Performance. Table 2 reports zero-shot accuracy of 70% sparse LLaMA-3 and Qwen3.5 series models on seven downstream tasks, with average accuracy presented in the last column. TOM-

Table 1: Perplexity on WikiText-2 for pruned LLaMA-3 and Qwen3.5 series models. Lower values indicate better performance; bold marks the best no-weight-update pruning method. SparseGPT* performs weight-update compensation and is included as a reference.

Sparsity	Method	LLaMA-3			Qwen3.5			
		3.0-8B	3.1-8B	3.2-1B	2B	4B	9B	27B
0%	Dense	5.75	5.84	9.06	11.07	8.89	8.21	6.72
40%	SparseGPT [14]* (ICML23)	7.07	7.07	12.20	13.09	9.39	7.98	6.80
	Wanda [49] (ICLR24)	7.00	7.03	12.90	12.72	9.55	8.09	6.65
	NoWag [30] (COLM25)	6.93	6.97	12.65	12.86	9.53	8.17	6.84
	TOM-Pruning	**6.91**	**6.94**	**12.62**	**12.58**	**9.43**	**7.99**	**6.57**
50%	SparseGPT [14]* (ICML23)	8.92	8.82	19.26	17.16	10.56	8.68	6.99
	Wanda [49] (ICLR24)	9.37	9.24	22.03	16.47	11.19	8.78	7.05
	NoWag [30] (COLM25)	**8.74**	**8.67**	22.19	16.70	11.10	8.77	7.18
	TOM-Pruning	9.15	9.02	**20.59**	**16.12**	**10.88**	**8.67**	**6.96**
60%	SparseGPT [14]* (ICML23)	14.75	14.96	42.91	32.41	14.76	11.71	8.24
	Wanda [49] (ICLR24)	22.45	21.12	77.52	36.72	19.47	13.79	8.33
	NoWag [30] (COLM25)	21.60	20.74	116.14	37.24	18.61	**12.69**	8.28
	TOM-Pruning	**21.12**	**20.61**	**70.21**	**36.03**	**18.27**	13.22	**8.20**
70%	SparseGPT [14]* (ICML23)	44.40	40.48	127.83	103.94	40.90	24.23	13.65
	Wanda [49] (ICLR24)	109.72	97.63	492.58	194.62	86.42	56.08	17.69
	NoWag [30] (COLM25)	679.32	403.83	177004.27	187.96	79.97	**43.78**	16.93
	TOM-Pruning	**97.96**$_{-11.76}$	**87.34**$_{-10.29}$	**360.32**$_{-132.26}$	**178.29**$_{-9.67}$	**78.22**$_{-1.75}$	50.34$_{+6.56}$	**16.26**$_{-0.67}$
2:4	SparseGPT [14]* (ICML23)	15.04	15.10	33.23	23.64	15.91	12.69	8.44
	Wanda [49] (ICLR24)	23.19	21.36	76.38	40.47	21.33	14.61	8.67
	NoWag [30] (COLM25)	24.46	22.21	79.08	**39.44**	21.75	14.59	8.60
	TOM-Pruning	**21.70**$_{-1.49}$	**20.68**$_{-0.68}$	**67.53**$_{-8.85}$	40.33$_{+0.89}$	**21.06**$_{-0.27}$	**14.47**$_{-0.12}$	**8.56**$_{-0.04}$
4:8	SparseGPT [14]* (ICML23)	11.64	11.56	23.76	18.76	12.63	10.24	7.56
	Wanda [49] (ICLR24)	13.82	13.02	41.90	25.31	15.72	10.77	7.78
	NoWag [30] (COLM25)	13.33	12.95	37.83	**24.51**	15.67	10.71	7.82
	TOM-Pruning	**13.27**$_{-0.06}$	**12.51**$_{-0.44}$	**37.16**$_{-0.67}$	25.11$_{+0.60}$	**15.53**$_{-0.14}$	**10.64**$_{-0.07}$	**7.52**$_{-0.26}$

Pruning consistently improves average zero-shot performance over Wanda across all evaluated models, for instance from 35.11% to 39.05% on LLaMA-3-8B and from 35.42% to 39.22% on LLaMA-3.1-8B, demonstrating that gains from target-aware pruning translate to downstream generalization. These improvements indicate that the target-aware local Riemannian metric and the Competitive-Hubness Channel Affinity Mechanism better preserve task-relevant connections than purely activation-based criteria. Figure 3 visualizes task-wise performance in a radar plot, comparing TOM-Pruning with Wanda and NoWag across tasks and highlighting its improvements over Wanda. Appendix C provides the full set of average zero-shot accuracies across LLaMA, LLaMA-2, LLaMA-3, and Qwen3.5 models under various sparsity settings.

Inference Speedup. Following prior pruning studies [14; 49], we use the 2:4 semi-structured setting supported by NVIDIA sparse Tensor Cores [41] under the setup in Section 5.1. We evaluate the layer-wise inference efficiency of TOM-Pruning on the 7 sparse linear layers (q/k/v/o, up/gate, and down projections) in LLaMA-7B and LLaMA-2-7B. Table 3 reports the latency of each layer. For LLaMA-7B, TOM-Pruning reduces per-layer latency from 1.56–2.71 ms to 0.96–1.62 ms, yielding 1.63–1.69× speedup. Similarly, LLaMA-2-7B achieves 1.60–1.67× acceleration. These results show that the layer-wise computation of sparse linear layers is efficiently supported, and the overhead introduced by TOM's pruning score computation and channel affinity reconstruction is minimal.

5.3 Ablation Study

We ablate the components of the Competitive-Hubness Channel Affinity Mechanism introduced in Section 4.2. The full TOM-Pruning criterion uses the reconstructed input–output channel affinity $\mathcal{A}_{i,j} = \widetilde{K}_{i,j}\mathrm{Hub}_j$, combining competitively reweighted local affinity with global hubness-aware gating. Table 4 reports results on LLaMA-7B at 70% unstructured sparsity under the same calibration and evaluation protocol. The raw $\cos^2\theta_{i,j}$ affinity gives high PPL, indicating limited discriminability in high-dimensional channel spaces. The nonlinear kernel $K_{i,j}$ improves the local affinity response, and intra-target competitive reweighting $\widetilde{K}_{i,j}$ further reduces PPL from 211.94 to 74.37 by comparing $X_{:,j}$ against all input channels for the same target output response $Y_{:,i}$. The hubness factor Hub_j provides global input-channel cues, while the full affinity $\mathcal{A}_{i,j} = \widetilde{K}_{i,j}\mathrm{Hub}_j$ achieves the lowest PPL, confirming the complementarity of local competitiveness and global hubness.

Table 2: Zero-shot accuracy at 70% sparsity on LLaMA-3 and Qwen3.5 series models. Higher values indicate better performance; bold marks the best no-weight-update pruning method. SparseGPT* is a weight-update pruning method and included as reference.

Model	Method	Downstream Tasks							Avg.
		ARC-C	ARC-E	BoolQ	HellaS.	OBQA	RTE	WinoG.	
LLaMA-3-8B	Dense	51.88	80.81	82.32	60.66	33.80	70.04	73.48	64.71
	SparseGPT [14]* (ICML23)	22.10	45.62	64.34	32.84	15.80	52.71	56.04	41.35
	Wanda [49] (ICLR24)	16.21	33.29	54.16	27.53	12.60	52.71	49.25	35.11
	NoWag [30] (COLM25)	19.03	**41.54**	56.36	28.92	14.80	52.71	50.12	37.64
	TOM-Pruning (Ours)	**20.14**	37.37	**58.26**	**31.25**	**16.40**	**56.68**	**53.28**	**39.05** +1.41
LLaMA-3.1-8B	Dense	52.05	82.24	83.09	60.70	34.00	71.12	74.59	65.40
	SparseGPT [14]* (ICML23)	22.44	46.89	63.67	33.60	16.20	52.71	54.78	41.47
	Wanda [49] (ICLR24)	17.92	33.16	53.91	27.76	12.40	52.71	50.04	35.42
	NoWag [30] (COLM25)	18.77	**42.30**	**60.00**	29.14	13.60	52.71	53.35	38.55
	TOM-Pruning (Ours)	**21.59**	37.21	57.92	**31.53**	**16.20**	**56.32**	**53.75**	**39.22** +0.67
LLaMA-3.2-1B	Dense	31.40	66.29	63.73	48.15	27.80	56.32	60.85	50.65
	SparseGPT [14]* (ICML23)	19.45	36.45	61.41	28.04	12.60	51.99	49.09	37.00
	Wanda [49] (ICLR24)	19.11	29.92	38.17	26.54	10.20	52.71	50.36	32.43
	NoWag [30] (COLM25)	19.03	29.55	**53.46**	26.52	**14.00**	52.71	49.49	34.96
	TOM-Pruning (Ours)	**21.93**	**32.91**	41.31	**29.31**	13.00	**55.60**	**53.12**	**35.31** +0.35
Qwen3.5-2B	Dense	38.14	70.54	71.96	46.30	26.40	64.26	63.22	54.40
	SparseGPT [14]* (ICML23)	19.80	35.14	61.44	28.26	14.40	52.71	51.85	37.66
	Wanda [49] (ICLR24)	19.03	30.35	58.72	27.25	14.40	53.07	49.33	36.02
	NoWag [30] (COLM25)	18.69	**34.09**	53.76	27.60	14.20	52.71	49.01	35.72
	TOM-Pruning (Ours)	**20.48**	32.03	**60.46**	**28.73**	**16.20**	**54.87**	**50.83**	**37.66** +1.64
Qwen3.5-4B	Dense	51.71	81.48	86.30	54.34	29.00	73.65	70.48	63.85
	SparseGPT [14]* (ICML23)	22.44	46.51	62.35	31.50	16.00	52.71	53.20	40.67
	Wanda [49] (ICLR24)	17.92	37.92	62.17	28.87	12.40	52.71	49.80	37.40
	NoWag [30] (COLM25)	18.94	**41.67**	62.20	29.76	14.20	53.79	49.88	38.63
	TOM-Pruning (Ours)	**20.22**	40.45	**64.74**	**31.19**	**15.00**	**55.23**	**52.09**	**39.85** +1.22
Qwen3.5-9B	Dense	54.52	81.40	89.20	58.31	32.60	81.95	73.09	67.30
	SparseGPT [14]* (ICML23)	25.68	53.62	63.06	35.36	17.00	53.79	59.59	44.01
	Wanda [49] (ICLR24)	19.62	43.98	62.17	30.67	14.20	52.71	52.17	39.36
	NoWag [30] (COLM25)	**23.38**	**51.26**	62.48	**33.01**	15.20	52.71	**55.49**	**41.93**
	TOM-Pruning (Ours)	21.93	46.51	**64.74**	32.99	**16.80**	**55.23**	54.46	41.81 -0.12

Table 3: Layer-wise latency (ms) of sparse linear layers under 2:4 semi-structured sparsity. Speedup is relative to the dense.

Model	Module	Dense	TOM 2:4	Speedup
LLaMA-7B	q/k/v/o	1.56 ms	0.96 ms	1.63×
	up/gate	2.71 ms	1.62 ms	1.67×
	down	2.68 ms	1.59 ms	1.69×
LLaMA2-7B	q/k/v/o	1.52 ms	0.95 ms	1.60×
	up/gate	2.65 ms	1.59 ms	1.67×
	down	2.60 ms	1.57 ms	1.66×

Table 4: Ablation of the Competitive-Hubness Channel Affinity Mechanism on LLaMA-7B at 70% sparsity. Prune time is measured in seconds.

Affinity Term	PPL	Prune Time (s)
$\cos^2\theta_{i,j}$	5448.09	3.80
$K_{i,j}$	211.94	3.80
$\widetilde{K}_{i,j}$	74.37	3.82
Hub_j	82.58	6.77
$\mathcal{A}_{i,j}$	**70.31**	5.05

6 Conclusion

This paper introduced Target-Aware Output Manifold Pruning (TOM-Pruning), a no-weight-update pruning method for large language models. TOM-Pruning first constructs a target-aware local Riemannian metric to measure pruning-induced output perturbations by their relation to the target response. The pullback of this metric induces an input–output channel affinity term, which is further refined by the Competitive-Hubness Channel Affinity Mechanism to derive the final pruning criterion. Experiments across LLM families and sparsity settings show that TOM-Pruning improves sparse model quality while remaining compatible with semi-structured sparse inference.

Acknowledgments and Disclosure of Funding

This work was supported in part by the National Natural Science Foundation of China under Grant 62372132. This work was supported in part by the National Natural Science Foundation of China under Grant 62606255. This work was supported by the Key Research Project of Pengcheng Laboratory (PCL2025A14). The authors would also like to thank the Huawei Ascend Cloud Ecological Development Project for providing high-performance Ascend 910 processors. The authors declare no competing interests.

References

[1] Georgios Arvanitidis, Miguel González Duque, Alison Pouplin, Dimitris Kalatzis, and Søren Hauberg. Pulling back information geometry. *CoRR*, abs/2106.05367, 2021.

[2] Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari Do Nascimento, Torsten Hoefler, and James Hensman. Slicegpt: Compress large language models by deleting rows and columns. In *Proc. International Conference on Learning Representations (ICLR)*, 2024.

[3] Guangji Bai, Yijiang Li, Chen Ling, Kibaek Kim, and Liang Zhao. Sparsellm: Towards global pruning of pre-trained language models. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, et al. Language models are few-shot learners. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[5] Yuli Chen, Bo Cheng, Jiale Han, Yingying Zhang, Yingting Li, and Shuhao Zhang. DLP: dynamic layerwise pruning in large language models. In *Proc. International Conference on Machine Learning (ICML)*, 2025.

[6] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24:240:1–240:113, 2023.

[7] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *CoRR*, abs/1905.10044, 2019.

[8] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457, 2018.

[9] Ido Dagan, Oren Glickman, and Bernardo Magnini. The PASCAL recognising textual entailment challenge. In *Proc. PASCAL Machine Learning Challenges Workshop (MLCW)*, pages 177–190, 2005.

[10] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm.int8(): 8-bit matrix multiplication for transformers at scale. *CoRR*, abs/2208.07339, 2022.

[11] Peijie Dong, Lujun Li, Zhenheng Tang, Xiang Liu, Xinglin Pan, Qiang Wang, and Xiaowen Chu. Pruner-zero: Evolving symbolic pruning metric from scratch for large language models. In *Proc. International Conference on Machine Learning (ICML)*, pages 11346–11374, 2024.

[12] Gongfan Fang, Hongxu Yin, Saurav Muralidharan, Greg Heinrich, Jeff Pool, Jan Kautz, Pavlo Molchanov, and Xinchao Wang. Maskllm: Learnable semi-structured sparsity for large language models. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[13] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *Proc. International Conference on Learning Representations (ICLR)*, 2019.

[14] Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *Proc. International Conference on Machine Learning (ICML)*, pages 10323–10337, 2023.

[15] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: accurate post-training quantization for generative pre-trained transformers. *CoRR*, abs/2210.17323, 2022.

[16] Shangqian Gao, Chi-Heng Lin, Ting Hua, Zheng Tang, Yilin Shen, Hongxia Jin, and Yen-Chang Hsu. DISP-LLM: dimension-independent structural pruning for large language models. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[17] Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural networks. *CoRR*, abs/1506.02626, 2015.

[18] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *Proc. International Conference on Learning Representations (ICLR)*, 2016.

[19] Babak Hassibi and David G. Stork. Second order derivatives for network pruning: Optimal brain surgeon. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 164–171, 1992.

[20] Weizhong Huang, Yuxin Zhang, Xiawu Zheng, Fei Chao, and Rongrong Ji. Determining layer-wise sparsity for large language models through a theoretical perspective. In *Proc. International Conference on Machine Learning (ICML)*, 2025.

[21] Cong Jiang, Fangzhi Zhu, Xiaowei Chen, Junxiong Zhu, Bo Zheng, Yifeng Wang, and Zheng Zhang. Dmlora: Dynamic multi-subspace low-rank adaptation. In *Companion Proceedings of the ACM on Web Conference*, pages 1038–1042, 2025.

[22] Cong Jiang, Xiaofeng Zhang, Fangzhi Zhu, Xiaowei Chen, Junxiong Zhu, and Zheng Zhang. Rethinking LLM reasoning: From explicit trajectories to latent representations. In *Proc. International Conference on Learning Representations (ICLR)*, 2026.

[23] François Lagunas, Ella Charlaix, Victor Sanh, and Alexander M. Rush. Block pruning for faster transformers. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 10619–10629, 2021.

[24] Yann LeCun, John S. Denker, and Sara A. Solla. Optimal brain damage. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 598–605, 1989.

[25] Dongyeop Lee, Kwanhee Lee, Jinseok Chung, and Namhoon Lee. SAFE: finding sparse and flat minima to improve pruning. In *Proc. International Conference on Machine Learning (ICML)*, 2025.

[26] Kwanhee Lee, Hyeondo Jang, Dongyeop Lee, Dan Alistarh, and Namhoon Lee. The unseen frontier: Pushing the limits of LLM sparsity with surrogate-free ADMM. *CoRR*, abs/2510.01650, 2025.

[27] Lujun Li, Peijie Dong, Zhenheng Tang, Xiang Liu, Qiang Wang, Wenhan Luo, Wei Xue, Qifeng Liu, Xiaowen Chu, and Yike Guo. Discovering sparsity allocation for layer-wise pruning of large language models. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[28] Wei Li, Lujun Li, Mark G. Lee, and Shengjie Sun. Adaptive layer sparsity for large language models via activation correlation assessment. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[29] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Guangxuan Xiao, and Song Han. AWQ: activation-aware weight quantization for on-device LLM compression and acceleration. *GetMobile: Mobile Computing and Communications*, 28(4):12–17, 2024.

[30] Lawrence Liu, Inesh Chakrabarti, Yixiao Li, Mengdi Wang, Tuo Zhao, and Lin F. Yang. Nowag: A unified framework for shape preserving compression of large language models. *CoRR*, abs/2504.14569, 2025.

[31] Haiquan Lu, Yefan Zhou, Shiwei Liu, Zhangyang Wang, Michael W. Mahoney, and Yaoqing Yang. Alphapruning: Using heavy-tailed self regularization theory for improved layer-wise pruning of large language models. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[32] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

[33] Weikang Meng, Yadan Luo, Liangyu Huo, Yaowei Wang, Xin Li, and Zheng Zhang. Nalaformer: Norm-aware linear attention for transformer models. *CoRR*, abs/2506.21137, 2025.

[34] Weikang Meng, Yadan Luo, Liangyu Huo, Yingjian Li, Yaowei Wang, Zheng Zhang, and Heng Tao Shen. Polaformer++: Polarity-aware linear attention with channel-wise spikiness. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(11):15412–15426, 2026.

[35] Xiang Meng, Kayhan Behdin, Haoyue Wang, and Rahul Mazumder. ALPS: improved optimization for highly sparse one-shot pruning for large language models. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[36] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *Proc. International Conference on Learning Representations (ICLR)*, 2017.

[37] Meta AI. Introducing meta llama 3: The most capable openly available llm to date. https://ai.meta.com/blog/meta-llama-3/, 2024.

[38] Meta AI. Introducing llama 3.1: Our most capable models to date. https://ai.meta.com/blog/meta-llama-3-1/, 2024.

[39] Meta AI. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/, 2024.

[40] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2381–2391, 2018.

[41] Asit K. Mishra, Jorge Albericio Latorre, Jeff Pool, Darko Stosic, Dusan Stosic, Ganesh Venkatesh, Chong Yu, and Paulius Micikevicius. Accelerating sparse deep neural networks. *CoRR*, abs/2104.08378, 2021.

[42] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

[43] Seungcheol Park, Hojun Choi, and U Kang. Accurate retraining-free pruning for pretrained encoder-based language models. In *Proc. International Conference on Learning Representations (ICLR)*, 2024.

[44] Adam Paszke, Sam Gross, Francisco Massa, et al. Pytorch: An imperative style, high-performance deep learning library. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 8024–8035, 2019.

[45] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:140:1–140:67, 2020.

[46] Bernhard Riemann. *On the hypotheses which lie at the bases of geometry*. 2016.

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

[48] Xuan Shen, Pu Zhao, Yifan Gong, Zhenglun Kong, Zheng Zhan, Yushu Wu, Ming Lin, Chao Wu, Xue Lin, and Yanzhi Wang. Search for efficient large language models. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[49] Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A simple and effective pruning approach for large language models. In *Proc. International Conference on Learning Representations (ICLR)*, 2024.

[50] Llama Team. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024.

[51] Hugo Touvron, Thibaut Lavril, Gautier Izacard, et al. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023.

[52] Hugo Touvron, Louis Martin, Kevin Stone, et al. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023.

[53] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 5998–6008, 2017.

[54] Thomas Wolf, Lysandre Debut, Victor Sanh, et al. Transformers: State-of-the-art natural language processing. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 38–45, 2020.

[55] Guangxuan Xiao, Ji Lin, Mickaël Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *Proc. International Conference on Machine Learning (ICML)*, pages 38087–38099, 2023.

[56] Lang Xiong, Ning Liu, Ao Ren, Yuheng Bai, Haining Fang, Binyan Zhang, Zhe Jiang, Yujuan Tan, and Duo Liu. D2 prune: Sparsifying large language models via dual taylor expansion and attention distribution awareness. In *Proc. Association for the Advancement of Artificial Intelligence (AAAI)*, pages 27171–27179, 2026.

[57] Peng Xu, Wenqi Shao, Mengzhao Chen, Shitao Tang, Kaipeng Zhang, Peng Gao, Fengwei An, Yu Qiao, and Ping Luo. BESA: pruning large language models with blockwise parameter-efficient sparsity allocation. In *Proc. International Conference on Learning Representations (ICLR)*, 2024.

[58] Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[59] Lu Yin, You Wu, Zhenyu Zhang, Cheng-Yu Hsieh, Yaqing Wang, Yiling Jia, Gen Li, Ajay Kumar Jaiswal, Mykola Pechenizkiy, Yi Liang, Michael Bendersky, Zhangyang Wang, and Shiwei Liu. Outlier weighed layerwise sparsity (OWL): A missing secret sauce for pruning llms to high sparsity. In *Proc. International Conference on Machine Learning (ICML)*, pages 57101–57115, 2024.

[60] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4791–4800, 2019.

[61] Stephen Zhang and Vardan Papyan. OATS: outlier-aware pruning through sparse and low rank decomposition. In *Proc. International Conference on Learning Representations (ICLR)*, 2025.

[62] Yingtao Zhang, Haoli Bai, Haokun Lin, Jialin Zhao, Lu Hou, and Carlo Vittorio Cannistraci. Plug-and-play: An efficient post-training pruning method for large language models. In *Proc. International Conference on Learning Representations (ICLR)*, 2024.

A Supplementary Derivations

- A.1 **Derivation of Optimal Compensation and Exact Pruning Sensitivity.**
- A.2 **Euclidean Pullback Interpretation of SparseGPT and Wanda.**
- A.3 **Derivation of the Basic TOM-Pruning Score.**

A.1 Derivation of Optimal Compensation and Exact Pruning Sensitivity

We derive the closed-form solution of the constrained pruning problem in Eq. (7). For a fixed output channel i and a candidate weight $w_{i,j}$, the problem seeks a compensatory perturbation δ_i with the minimum parameter-space quadratic cost while enforcing exact deletion of $w_{i,j}$. Since G_i is assumed to be symmetric positive definite, the objective is strictly convex and G_i^{-1} exists, so the constrained problem admits a unique optimum. Introducing a Lagrange multiplier λ, we define

$$\mathcal{J}(\delta_i, \lambda) = \frac{1}{2}\delta_i^\top G_i \delta_i + \lambda(e_j^\top \delta_i + w_{i,j}), \tag{A.1}$$

where the first term is the quadratic approximation of the local pruning cost in the parameter perturbation space, and the second term enforces the deletion constraint $e_j^\top \delta_i = -w_{i,j}$.

The stationary conditions are obtained by taking derivatives with respect to δ_i and λ:

$$\frac{\partial \mathcal{J}}{\partial \delta_i} = G_i\delta_i + \lambda e_j = 0, \qquad \frac{\partial \mathcal{J}}{\partial \lambda} = e_j^\top \delta_i + w_{i,j} = 0. \tag{A.2}$$

The first condition gives

$$\delta_i = -\lambda G_i^{-1} e_j, \tag{A.3}$$

which shows that the optimal perturbation lies along the inverse-metric direction $G_i^{-1}e_j$. Substituting Eq. (A.3) into the deletion constraint yields

$$-\lambda e_j^\top G_i^{-1} e_j + w_{i,j} = 0. \tag{A.4}$$

Since $e_j^\top G_i^{-1} e_j$ equals the j-th diagonal entry of G_i^{-1}, we obtain

$$\lambda = \frac{w_{i,j}}{e_j^\top G_i^{-1} e_j} = \frac{w_{i,j}}{[G_i^{-1}]_{j,j}}. \tag{A.5}$$

Substituting Eq. (A.5) back into Eq. (A.3) gives the optimal compensation perturbation:

$$\delta_i^* = -\frac{w_{i,j}}{[G_i^{-1}]_{j,j}} G_i^{-1} e_j. \tag{A.6}$$

This expression shows that the compensation direction is determined by the inverse pullback metric G_i^{-1}, while the perturbation scale is controlled by the removed weight $w_{i,j}$ and the diagonal inverse-metric term $[G_i^{-1}]_{j,j}$. A larger $[G_i^{-1}]_{j,j}$ allows the deletion constraint to be satisfied with a smaller quadratic cost, indicating stronger local compensability along the j-th parameter direction.

The exact pruning sensitivity is obtained by substituting δ_i^* into the parameter-space quadratic approximation:

$$S_{i,j}^{\text{exact}} = \frac{1}{2}(\delta_i^*)^\top G_i \delta_i^*. \tag{A.7}$$

Using Eq. (A.6), we obtain

$$\begin{aligned} S_{i,j}^{\text{exact}} &= \frac{1}{2}\frac{w_{i,j}^2}{[G_i^{-1}]_{j,j}^2} e_j^\top G_i^{-1} G_i G_i^{-1} e_j \\ &= \frac{1}{2}\frac{w_{i,j}^2}{[G_i^{-1}]_{j,j}^2} e_j^\top G_i^{-1} e_j = \frac{w_{i,j}^2}{2[G_i^{-1}]_{j,j}}. \end{aligned} \tag{A.8}$$

Thus, the exact sensitivity is determined by the squared weight magnitude $w_{i,j}^2$ and the inverse-metric diagonal term $[G_i^{-1}]_{j,j}$. A larger $[G_i^{-1}]_{j,j}$ indicates that the perturbation along the j-th parameter direction is easier to compensate under the metric-induced local manifold structure defined by G_i, leading to a lower pruning cost. Conversely, a smaller value implies weaker compensability and higher pruning sensitivity.

A.2 Euclidean Pullback Interpretation of SparseGPT and Wanda

We interpret SparseGPT and Wanda under the metric-induced local manifold framework in Section 3.2. Both methods can be viewed as special cases that adopt the Euclidean output metric, i.e., $M_i = I$. Under this choice, the channel-wise local loss reduces to the standard Euclidean reconstruction error:

$$\Delta\mathcal{L}_{\text{loc}}^{(i)} \approx \frac{1}{2}\|\Delta Y_{:,i}\|_2^2. \tag{A.9}$$

Using the forward mapping $\Delta Y_{:,i} = X\delta_i$, this Euclidean output metric is pulled back to the parameter perturbation space as

$$\Delta\mathcal{L}_{\text{loc}}^{(i)} \approx \frac{1}{2}\delta_i^\top X^\top X\delta_i. \tag{A.10}$$

With a small regularization term for numerical stability and positive definiteness, the induced pullback metric becomes

$$\widetilde{G}_i = X^\top X + \epsilon I. \tag{A.11}$$

Therefore, SparseGPT and Wanda share the same output-space metric $M_i = I$, and their difference lies in how they use or approximate the regularized pullback metric $\widetilde{G}_i$.

SparseGPT preserves the full regularized Gram matrix $\widetilde{G}_i$. By substituting $G_i = \widetilde{G}_i$ into the exact sensitivity in Eq. (9), its connection-level pruning cost can be written as

$$S_{i,j}^{\text{SparseGPT}} = \frac{w_{i,j}^2}{2\left[\widetilde{G}_i^{-1}\right]_{j,j}} = \frac{w_{i,j}^2}{2\left[(X^\top X + \epsilon I)^{-1}\right]_{j,j}}. \tag{A.12}$$

This form shows that SparseGPT evaluates pruning sensitivity under the full pullback metric, where off-diagonal entries encode coupled compensation among parameter directions.

Wanda can be interpreted as a diagonal approximation of the same pullback metric. By discarding off-diagonal coupling terms, the metric becomes

$$\widetilde{G}_i^{\text{diag}} = \text{diag}\left(\|X_{:,1}\|_2^2 + \epsilon,\ \|X_{:,2}\|_2^2 + \epsilon,\ \ldots,\ \|X_{:,C_{\text{in}}}\|_2^2 + \epsilon\right). \tag{A.13}$$

Since $\widetilde{G}_i^{\text{diag}}$ is diagonal, we have

$$\left[\left(\widetilde{G}_i^{\text{diag}}\right)^{-1}\right]_{j,j} = \frac{1}{\|X_{:,j}\|_2^2 + \epsilon}. \tag{A.14}$$

Substituting Eq. (A.14) into Eq. (9) gives

$$S_{i,j}^{\text{diag}} = \frac{1}{2}w_{i,j}^2\left(\|X_{:,j}\|_2^2 + \epsilon\right). \tag{A.15}$$

For ranking, the constant factor and the small regularization term do not affect the ordering. Thus,

$$S_{i,j}^{\text{diag}} \propto w_{i,j}^2\|X_{:,j}\|_2^2. \tag{A.16}$$

Since the square function is monotonic over non-negative values, this ranking is equivalent to Wanda's activation-aware score:

$$S_{i,j}^{\text{Wanda}} \propto |w_{i,j}|\|X_{:,j}\|_2. \tag{A.17}$$

This interpretation shows that Wanda is not a heuristic criterion, but a diagonal activation-based approximation of the same Euclidean pullback metric used by SparseGPT. SparseGPT retains the full $\widetilde{G}_i$ to model coupled compensation, whereas Wanda keeps only its diagonal entries and reduces pruning to activation-aware ranking. Thus, their essential difference lies in the approximation of the induced parameter-space metric, rather than in the choice of the output-space metric.

A.3 Derivation of the Basic TOM-Pruning Score

We derive the basic TOM-Pruning score used in Eq. (13). The target-aware local Riemannian metric, its pullback form, and the diagonal approximation are defined in Section 4.1. Here, we focus on how the diagonal pullback metric leads to the connection-level ranking score.

Let $d_{i,j}$ denote the j-th diagonal entry of the diagonal pullback metric:

$$d_{i,j} = \left[\widetilde{G}^{\text{pull}}_{i,\text{diag}}\right]_{j,j}. \tag{A.18}$$

From Eq. (12), this entry can be expanded as

$$\begin{aligned} d_{i,j} &= \|X_{:,j}\|^2_{M_i} + \epsilon \\ &= X_{:,j}^\top \left(I + \lambda_m Y_{:,i} Y_{:,i}^\top\right) X_{:,j} + \epsilon \\ &= \|X_{:,j}\|_2^2 + \lambda_m \left(X_{:,j}^\top Y_{:,i}\right)^2 + \epsilon. \end{aligned} \tag{A.19}$$

Since $\widetilde{G}^{\text{pull}}_{i,\text{diag}}$ is diagonal, its inverse is also diagonal:

$$\left[\left(\widetilde{G}^{\text{pull}}_{i,\text{diag}}\right)^{-1}\right]_{j,j} = \frac{1}{d_{i,j}}. \tag{A.20}$$

Using the exact connection-level sensitivity in Eq. (9) and substituting $G_i = \widetilde{G}^{\text{pull}}_{i,\text{diag}}$, we obtain the diagonal-metric sensitivity:

$$\begin{aligned} S^{\text{diag}}_{i,j} &= \frac{w_{i,j}^2}{2\left[\left(\widetilde{G}^{\text{pull}}_{i,\text{diag}}\right)^{-1}\right]_{j,j}} \\ &= \frac{w_{i,j}^2}{2/d_{i,j}} = \frac{1}{2} w_{i,j}^2 d_{i,j}. \end{aligned} \tag{A.21}$$

This diagonal-metric sensitivity directly serves as the basis for the basic TOM-Pruning score. For ranking, the constant factor $1/2$ and the small stabilization term ϵ do not affect the ordering. Substituting the expanded form of $d_{i,j}$ from Eq. (A.19), we obtain

$$S^{\text{base}}_{i,j} \propto w_{i,j}^2 \left(\|X_{:,j}\|_2^2 + \lambda_m \left(X_{:,j}^\top Y_{:,i}\right)^2\right). \tag{A.22}$$

To expose the input–output channel affinity term, let $\theta_{i,j}$ denote the angle between $X_{:,j}$ and $Y_{:,i}$. Then

$$\left(X_{:,j}^\top Y_{:,i}\right)^2 = \|X_{:,j}\|_2^2 \|Y_{:,i}\|_2^2 \cos^2\theta_{i,j}. \tag{A.23}$$

Substituting Eq. (A.23) into Eq. (A.22) gives

$$S^{\text{base}}_{i,j} \propto w_{i,j}^2 \|X_{:,j}\|_2^2 + \lambda_m w_{i,j}^2 \|X_{:,j}\|_2^2 \|Y_{:,i}\|_2^2 \cos^2\theta_{i,j}. \tag{A.24}$$

This matches the basic TOM-Pruning score in Eq. (13). The first term retains the weight-magnitude and input-activation components, while the second term is induced by the pullback expansion of the target-aware local Riemannian metric. Specifically, the inner product $X_{:,j}^\top Y_{:,i}$ gives rise to the input–output channel affinity term $\cos^2\theta_{i,j}$, which measures the local association between the input channel activation and the target output response. A stronger alignment therefore indicates a higher pruning cost, since removing $w_{i,j}$ is more likely to disturb the target output direction.

B Additional Language Modeling Results

Table 5 reports WikiText-2 perplexity on LLaMA and LLaMA-2 series models across parameter scales. These results complement the main results on LLaMA-3 and Qwen3.5 series models in Table 1. Following the main text, SparseGPT is included as a weight-update compensation reference and is not considered when marking the best no-weight-update pruning method.

Overall, TOM-Pruning achieves the best no-weight-update performance in most settings. The advantage becomes more pronounced at high sparsity. At 70% sparsity, TOM-Pruning consistently improves over Wanda across all LLaMA and LLaMA-2 models, reducing the PPL of LLaMA-13B from 56.44 to 47.03 and that of LLaMA-2-13B from 43.54 to 38.94. TOM-Pruning also performs strongly under semi-structured sparsity, achieving the best no-weight-update results in most 2:4 and 4:8 settings. These results further support the effectiveness of TOM-Pruning as a no-weight-update pruning criterion across earlier LLaMA model families.

Table 5: Perplexity on WikiText-2 for pruned LLaMA and LLaMA-2 series models. Lower values indicate better performance; bold marks the best no-weight-update pruning method. SparseGPT* performs weight-update compensation and is included as a reference.

Sparsity	Method	LLaMA			LLaMA-2	
		7B	13B	30B	7B	13B
0%	Dense	5.67	5.09	4.10	5.12	4.57
40%	SparseGPT [14]* (ICML23)	6.31	5.56	4.71	5.72	5.03
	Wanda [49] (ICLR24)	6.38	5.51	4.66	5.67	5.01
	NoWag [30] (COLM25)	6.30	5.53	4.64	5.59	4.96
	TOM-Pruning (Ours)	**6.29**	**5.44**	**4.59**	**5.56**	**4.94**
50%	SparseGPT [14]* (ICML23)	7.19	6.21	5.32	6.53	5.63
	Wanda [49] (ICLR24)	7.25	6.15	5.25	6.46	5.58
	NoWag [30] (COLM25)	**7.12**	6.23	5.22	6.40	5.54
	TOM-Pruning (Ours)	7.16	**6.07**	**5.18**	**6.31**	**5.49**
60%	SparseGPT [14]* (ICML23)	10.30	8.46	6.69	9.62	7.82
	Wanda [49] (ICLR24)	10.71	8.76	6.56	10.04	7.93
	NoWag [30] (COLM25)	10.61	8.62	6.55	9.93	**7.66**
	TOM-Pruning (Ours)	**10.49**	**8.61**	**6.44**	**9.81**	7.78
70%	SparseGPT [14]* (ICML23)	26.32	18.93	12.63	24.13	18.03
	Wanda [49] (ICLR24)	82.67	56.44	17.30	74.33	43.54
	NoWag [30] (COLM25)	96.63	53.31	20.05	100.53	43.34
	TOM-Pruning (Ours)	**70.31**	**47.03**	**16.01**	**65.29**	**38.94**
2:4	SparseGPT [14]* (ICML23)	11.11	9.11	7.17	10.22	8.27
	Wanda [49] (ICLR24)	11.53	9.57	6.91	11.37	8.36
	NoWag [30] (COLM25)	11.65	9.53	6.98	11.28	8.44
	TOM-Pruning (Ours)	**11.39**	**9.45**	**6.82**	**11.12**	**8.18**
4:8	SparseGPT [14]* (ICML23)	8.56	7.42	6.12	7.92	6.59
	Wanda [49] (ICLR24)	8.57	7.40	5.98	8.07	6.55
	NoWag [30] (COLM25)	8.65	7.36	5.92	8.13	6.58
	TOM-Pruning (Ours)	**8.46**	**7.31**	**5.89**	**7.89**	**6.43**

C Additional Average Zero-shot Accuracy Results

Table 6 reports the average zero-shot accuracy across seven downstream tasks for LLaMA, LLaMA-2, LLaMA-3, and Qwen3.5 series models under different parameter scales and sparsity settings. The tasks include ARC-Challenge and ARC-Easy [8], BoolQ [7], HellaSwag [60], OpenBookQA [40], RTE [9], and WinoGrande [47]. These results complement the 70% sparsity task-wise accuracies reported in Table 2. Following the main text, SparseGPT is excluded from best-method highlighting since it updates the remaining weights for compensation.

Overall, TOM-Pruning achieves higher average zero-shot performance than Wanda across most no-weight-update settings, with the improvement particularly pronounced at high sparsity. For instance, at 70% sparsity, TOM-Pruning increases the average accuracy of LLaMA-13B from 39.04% to 42.89% and LLaMA-2-13B from 37.66% to 41.13%. These results demonstrate that TOM-Pruning preserves task-relevant connections more effectively, enhancing downstream generalization across model families, parameter scales, and pruning patterns.

Table 6: Average zero-shot accuracy on LLaMA, LLaMA-2, LLaMA-3, and Qwen3.5 series models across seven downstream tasks. Higher values indicate better performance; bold marks the best no-weight-update pruning method. SparseGPT* is a weight-update method and included as reference.

Sparsity	Method	LLaMA		LLaMA-2		LLaMA-3			Qwen3.5		
		7B	13B	7B	13B	3.0-8B	3.1-8B	3.2-1B	2B	4B	9B
0%	Dense	59.98	62.46	60.22	63.19	64.71	65.40	50.65	54.40	63.85	67.30
40%	SparseGPT [14]* (ICML23)	57.62	60.45	59.07	61.61	62.42	62.74	47.45	52.18	62.88	66.31
	Wanda [49] (ICLR24)	57.82	60.88	58.48	62.75	61.12	61.27	46.47	51.17	61.76	65.72
	NoWag [30] (COLM25)	58.21	60.19	58.59	62.40	61.79	61.52	47.31	50.88	61.23	65.70
	TOM-Pruning (Ours)	**59.09**	**62.15**	**59.75**	**64.02**	**62.24**	**62.41**	**47.48**	**52.44**	**62.96**	**66.85**
50%	SparseGPT [14]* (ICML23)	54.55	58.75	55.84	60.98	59.26	59.34	45.13	48.70	59.90	63.67
	Wanda [49] (ICLR24)	53.98	59.43	56.29	60.34	57.76	57.30	42.85	47.26	59.33	63.40
	NoWag [30] (COLM25)	**55.12**	58.75	55.69	60.27	57.76	57.65	42.77	48.18	59.57	63.52
	TOM-Pruning (Ours)	55.05	**60.51**	**57.36**	**61.41**	**58.93**	**58.47**	**43.98**	**48.33**	**60.51**	**64.57**
60%	SparseGPT [14]* (ICML23)	50.66	53.96	50.90	55.47	53.17	53.17	40.93	41.16	53.05	58.41
	Wanda [49] (ICLR24)	50.49	53.58	49.89	55.65	47.57	47.46	37.53	41.80	50.24	56.44
	NoWag [30] (COLM25)	51.13	53.74	50.10	56.62	**50.53**	**50.62**	37.09	41.98	50.50	55.80
	TOM-Pruning (Ours)	**51.81**	**54.91**	**51.21**	**57.15**	49.31	49.19	**39.14**	**43.11**	**51.97**	**58.17**
70%	SparseGPT [14]* (ICML23)	42.10	45.25	40.34	43.26	41.35	41.47	37.00	37.66	40.67	44.01
	Wanda [49] (ICLR24)	37.51	39.04	33.71	37.66	35.11	35.42	32.43	36.02	37.40	39.36
	NoWag [30] (COLM25)	**40.47**	**43.37**	37.09	**42.67**	37.64	38.55	34.96	35.72	38.63	**41.93**
	TOM-Pruning (Ours)	40.36	42.89	**37.82**	41.13	**39.05**	**39.22**	**35.31**	**37.66**	**39.85**	41.81
2:4	SparseGPT [14]* (ICML23)	49.43	53.06	51.10	56.08	50.96	50.70	41.29	43.17	52.01	56.19
	Wanda [49] (ICLR24)	48.35	52.17	48.80	53.97	46.42	46.51	37.21	41.68	50.08	54.71
	NoWag [30] (COLM25)	48.69	52.22	47.40	53.98	47.01	47.09	38.02	41.44	50.32	55.30
	TOM-Pruning (Ours)	**49.36**	**53.19**	**49.97**	**55.14**	**47.69**	**47.79**	**38.44**	**42.76**	**51.35**	**55.98**
4:8	SparseGPT [14]* (ICML23)	52.49	56.06	53.75	59.22	54.35	54.77	42.63	47.07	56.69	61.48
	Wanda [49] (ICLR24)	52.18	55.11	52.83	58.79	52.09	51.60	39.44	44.36	54.51	60.22
	NoWag [30] (COLM25)	52.15	55.22	51.89	58.32	52.29	51.50	38.75	45.07	54.28	58.94
	TOM-Pruning (Ours)	**53.41**	**56.34**	**54.06**	**60.02**	**53.16**	**52.68**	**40.71**	**45.59**	**55.58**	**61.29**

D Limitations

TOM-Pruning improves no-weight-update LLM pruning by constructing a target-aware local Riemannian metric in the output perturbation space and refining the induced input–output channel affinity through the Competitive-Hubness Channel Affinity Mechanism. Although the method is developed for post-training pruning, the target-aware output manifold perspective may also provide useful information for other compression tasks, such as quantization or structured pruning. Its broader applicability beyond the current pruning setting remains an open direction for future exploration. Moreover, while the proposed affinity reconstruction improves sparse-model quality, it introduces additional pruning-time overhead compared with simpler activation-based criteria. This cost occurs only during offline pruning and does not affect the inference structure of the compressed model. Improving the efficiency of input–output channel affinity estimation remains important future work.

E Experiments Compute Resources

All experiments were conducted on NVIDIA A100 40GB GPUs. TOM-Pruning is a no-weight-update post-training pruning method, and its main computational cost comes from calibration-based pruning score estimation and input–output channel affinity reconstruction. Following the experimental setup in Section 5.1, we used 128 randomly sampled C4 examples for calibration during pruning. Each pruning run was performed once for a given model, sparsity setting, and pruning pattern, and the required GPU resources mainly depended on the model scale. As a representative pruning-time measurement, the final TOM-Pruning criterion takes about 5.05 seconds per layer on LLaMA-7B, as reported in Table 4. The GPU allocation used for pruning and evaluation is summarized in Table 7.

Table 7: Compute Resources Used for Pruning and Evaluation.

Model Scale	Representative Checkpoints	GPUs Used
1B–4B	LLaMA-3.2-1B, Qwen3.5-2B, Qwen3.5-4B	1 × A100 40GB
7B–9B	LLaMA-7B, LLaMA-2-7B, LLaMA-3-8B, LLaMA-3.1-8B, Qwen3.5-9B	2 × A100 40GB
13B	LLaMA-13B, LLaMA-2-13B	2 × A100 40GB
27B–30B	Qwen3.5-27B, LLaMA-30B	4 × A100 40GB

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See Abstract and Introduction. They summarize the paper's motivation, scope, main contributions, and empirical findings.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations are discussed in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: See Sections 3.2, 3.3, and 4.1. The paper presents the relevant assumptions and derivations in the main text, with additional details provided in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Experiments. The method pipeline and experimental protocol are described with sufficient details to reproduce the main results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The main experimental setup and details are provided to support reproducibility at this stage. Additional materials will be provided in future updates.

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: See Experiments. The method pipeline and key experimental settings are described with sufficient details to understand and reproduce the reported results.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Error bars and statistical significance tests are not reported due to the computational cost of repeated evaluations across multiple LLM families and sparsity settings.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The information on the computer resources for experiments is provided in Appendix E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work complies with the NeurIPS Code of Ethics and uses only publicly available models and datasets, without human-subject data or private data collection.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This paper focuses on LLM post-training pruning for resource-efficient compression and has no obvious broader societal impact.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not introduce or release assets that pose high misuse risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Existing models, datasets, and software libraries are properly credited, with their licenses and terms of use explicitly mentioned and respected.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release any new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper includes no crowdsourcing or human-subject research.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper includes no crowdsourcing or human-subject research.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The proposed method is developed through analytic formulation and empirical evaluation, without using LLMs as core design components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.