RAP: Fast Feedforward Rendering-Free Attribute-Guided Primitive Importance Score Prediction for Efficient 3D Gaussian Splatting Processing

Kaifa Yang[1] Qi Yang[2*] Yiling Xu[1*] Zhu Li[2]
[1]Shanghai Jiao Tong University [2]University of Missouri–Kansas City
{sekiroyyy, yl.xu}@sjtu.edu.cn {qiyang, lizhu}@umkc.edu
*Corresponding authors

Abstract

3D Gaussian Splatting (3DGS) has emerged as a leading technology for high-quality 3D scene reconstruction. However, the iterative refinement and densification process leads to the generation of a large number of primitives, each contributing to the reconstruction to a substantially different extent. Estimating primitive importance is thus crucial, both for removing redundancy during reconstruction and for enabling efficient compression and transmission. Existing methods typically rely on rendering-based analyses, where each primitive is evaluated through its contribution across multiple camera viewpoints. However, such methods are 1) sensitive to the number and selection of views; 2) rely on specialized differentiable rasterizers; and 3) have long calculation times that grow linearly with view count, making them difficult to integrate as plug-and-play modules, as well as resulting in limited scalability and generalization. To address these issues, we propose RAP — a fast feedforward Rendering-free Attribute-guided method for efficient importance score Prediction in 3DGS. RAP infers primitive significance directly from intrinsic Gaussian attributes and local neighborhood statistics, avoiding any rendering-based or visibility-dependent computations. A compact MLP is trained to predict per-primitive importance scores using a combination of rendering loss, pruning-aware loss, and significance distribution regularization loss. After being trained on a small set of scenes, RAP generalizes effectively to unseen data and can be seamlessly integrated into reconstruction, compression, and transmission pipelines, providing a unified and efficient pruning solution. Our code is publicly available at: https://github.com/yyyykf/RAP

1. Introduction

3D Gaussian Splatting (3DGS) [15] has recently emerged as a powerful explicit representation for novel view synthesis, achieving high-fidelity reconstruction with real-time rendering efficiency. Despite its strengths, 3DGS typically relies on millions of Gaussian primitives to achieve high-fidelity rendering, which imposes a substantial burden on storage and memory. However, these primitives contribute to rendering quality in a highly unbalanced manner: only a fraction of these primitives is valuable for rendering quality, while a large portion is redundant due to either the sub-optimal densification process or incomplete training. Therefore, the primitive importance scores are proposed and employed to select and prioritize primitives for differentiated processing, e.g., pruning [6, 9], compression [4, 5, 20, 21, 26, 29], level-of-detail rendering, and transmission [24, 25, 33]. For example, prior works such as LightGaussian [6] and PUP-3DGS [9] demonstrate that pruning guided by importance scores can significantly reduce the number of primitives without sacrificing fidelity. Consequently, it is crucial to propose an importance estimation method that is accurate, robust, and plug-and-play, serving as a modular component in multiple practical applications of 3DGS.

Existing efforts on primitive importance estimation can be broadly categorized into three groups: attribute-based, rendering-based, and learning-based methods. 1) Attribute-based heuristics [12, 15] adopt simple rules, such as splitting large primitives during optimization or discarding primitives with opacity below a threshold. While lightweight and independent of camera viewpoints, these strategies ignore the complex blending interactions among overlapping primitives and fail to reflect the true contributions to rendering quality. 2) Rendering-based approaches [6, 9, 23, 28] estimate primitive significance by projecting Gaussians onto multiple views to measure their 2D footprints or by evaluating perturbation sensitivity through reconstruction-error gradients. These methods offer higher accuracy but are inherently view-dependent, sensitive to the number and choice of views, and computationally expensive, as the time cost grows linearly with view count. Moreover, they require specialized rasterization, limiting

their modularity and scalability. 3) Learning-based approaches [3, 5, 20, 26] jointly optimize a learnable score mask alongside scene reconstruction. Although they can implicitly learn primitive significance, the scores are tightly coupled with specific reconstruction frameworks and thus lack universality. Moreover, once the scene is modified, the precomputed scores become invalid and cannot be reused. These limitations motivate us to revisit intrinsic attributes as potential signals of primitive importance. Unlike view-dependent or framework-specific strategies, such indicators are lightweight, naturally modular, and generalizable, though their reliability requires further exploration.

Our key observation, illustrated in Fig. 1, is that redundant primitives often exhibit abnormal attributes. For instance, primitives with extremely small scales or very low opacity usually contribute little to rendering quality. Isolated spatial outliers tend to be visually insignificant, though in some cases they may correspond to valid background primitives. Similarly, densification can spawn invisible points with inconsistent colors or near-zero SH coefficients, as they receive little gradient during optimization. However, no single attribute provides a reliable criterion for importance estimation, as each captures only a limited aspect of information. Building on this insight, we propose **RAP**, a **R**endering-free **A**ttribute-guided primitive importance score **P**rediction framework for estimating primitive importance directly from intrinsic attributes and local neighborhood statistics. Unlike rendering-based or learning-based joint optimization methods, RAP does not require view-dependent computations or per-scene retraining, and also achieves faster calculation with a feedforward style.

Specifically, we extract a 15-dimensional per-point feature vector that captures both global absolute values and local relative statistics, including distance, color, scale, volume, and opacity. To model the interactions among these features, we adopt a lightweight MLP that predicts an importance score for each primitive. The network is trained under the supervision of three complementary losses designed to balance fidelity, compactness, and stability: 1) a rendering loss enforces that the pruned model preserves visual fidelity with respect to the ground truth views; 2) A pruning-aware loss prevents the network from trivially assigning high importance to all primitives. This is achieved by regularizing the average predicted score toward a small predefined target, thereby encouraging the network to discard as many primitives as possible and naturally counteracting the rendering loss; 3) A significance distribution regularization ensures that the predicted scores are well separated and distributed from low to high, making downstream tasks, such as pruning decisions, more stable and flexible. RAP is trained once on a small set of scenes and generalizes well to unseen datasets. Its attribute-based design allows fast, feedforward, plug-and-play importance score prediction without additional scene-specific optimization. Rendering is only required during the network training, after training, rendering is not necessary for score prediction.

Our contributions are summarized as follows:

- We propose RAP, a rendering-free attribute-guided primitive importance score prediction framework that estimates primitive importance directly from intrinsic attributes and local neighborhood statistics.
- We design a set of importance-related per-point features, including average KNN distance, color anisotropy, etc., which together form a compact and discriminative representation of primitive characteristics. A unified learning framework that based on a lightweight MLP with three tailored loss functions is proposed. These losses jointly constrain the model training, guiding it to produce stable and separable importance score distributions that are friendly to downstream tasks.
- We conduct extensive experiments across diverse datasets and tasks, showing that RAP generalizes well to unseen scenes and consistently improves multiple downstream performance.

2. Related work

2.1. Primitive Importance Score Prediction

Existing approaches for estimating Gaussian primitive importance can be broadly categorized into attribute-based, rendering-based, and learning-based methods.

Attribute-based methods adopt simple heuristics such as Gaussian volume and opacity, which were used in 3DGS [15] and ADC-GS [12] to detect redundant points. These methods are computationally efficient and easy to integrate, but they fail to reliably capture a Gaussian's actual contribution to rendering quality, especially when multiple Gaussians overlap or interact in complex ways.

Rendering-based methods derive importance by measuring each Gaussian's contribution to rendered images, either through visibility or via reconstruction error sensitivity. LightGaussian [6] computes importance as the 2D projected area multiplied by absolute opacity and further refined by the 3D Gaussian volume. PRoGS [33] and EAGLES [8] replace absolute opacity with blending opacity obtained during rasterization, which better reflects accumulated visibility but lacks volume refinement. MesonGS [28] and HGSC [11] combine blending opacity with volume, effectively integrating the strengths of both strategies for more stable scoring. These strategies provide a closer approximation to perceptual importance but come with notable limitations. They rely on custom Gaussian rasterizers, and their computation time increase proportionally with the number of images used for score calculation. In addition, their performance is sensitive to the number and distribution

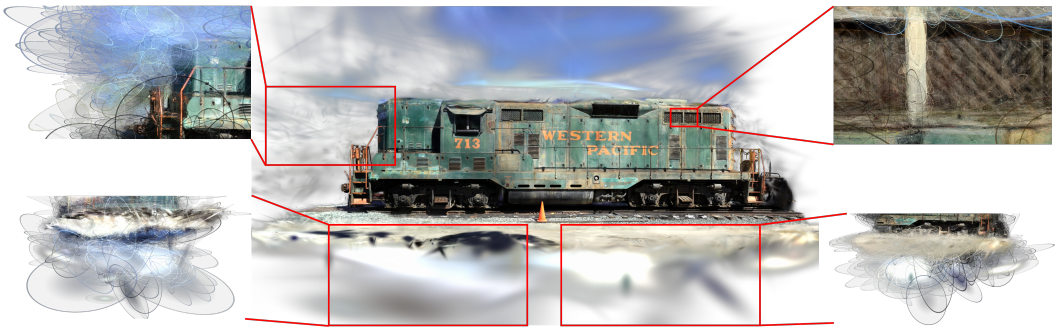


Figure 1. Visualization of redundant and low-importance Gaussians in the *Train* scene from Tanks&Temples.

of views, as shown in the Supplementary Material (Sec. 7). Gradient-based variants estimate significance from reconstruction error sensitivity. C3DGS [23] uses backward gradients, selecting the maximum SH coefficient channel for color and the maximum eigenvalue of the rotation or scaling matrix for geometry, while PUP-3DGS [9] extends this idea by computing perturbation sensitivity from the Hessian of reconstruction errors. While gradient signals can effectively indicate importance in well-optimized regions, they often produce unstable scores in poorly reconstructed areas, where large gradients arise from fitting errors rather than true contribution.

Learning-based approaches incorporate joint optimization of a per-Gaussian mask into the reconstruction pipeline, enabling data-driven selection of effective primitives [3, 5, 17, 20, 26]. The mask implicitly encodes point significance, but these methods are tightly coupled with the reconstruction process, require lengthy optimization, and lack reusability. Moreover, once the Gaussian set changes (e.g., after pruning or editing [2, 27, 31]), the learned importance distribution becomes invalid, limiting generalization and transferability.

Motivated by these limitations, we propose RAP, a lightweight and rendering-free framework for efficient and generalizable Gaussian importance estimation.

2.2. Applications of Primitive Importance Score

The concept of Gaussian primitive importance has been extensively explored in 3D reconstruction, where importance-guided strategies are employed to eliminate redundancy while preserving scene fidelity. Prune-and-refine approaches [6, 9, 30] typically remove low-importance primitives before refining the remaining set, while LP-3DGS [32] further learns to determine a favorable pruning ratio.

Alternatively, Mini-Splatting [7] selects representative Gaussians through sampling instead of pruning, mitigating potential artifacts from aggressive removal. More recently, TAMING-3DGS [22] utilizes importance scores to drive targeted densification, enhancing reconstruction efficiency under hardware and resource constraints.

Gaussian primitive importance has also become a key factor in compression and coding frameworks. Threshold-based pruning, adopted in MesonGS [28] and EAGLES [8], discards low-importance Gaussians prior to encoding to reduce data size. Beyond simple thresholding, C3DGS [23] employs sensitivity-aware clustering to group Gaussians according to their visual contribution, thereby preserving perceptually important structures while improving compression efficiency. Another direction focuses on learnable masking, with methods [3, 5, 20, 26] jointly optimizing mask values during reconstruction to guide the selection of effective primitives. This idea, first introduced in Compact-3DGS [17], has been extended in compression-oriented frameworks to enable adaptive importance modeling.

3. Preliminary

To design an attribute-guided pruning framework, it is essential to understand how intrinsic Gaussian parameters relate to their perceptual and structural significance. Each Gaussian primitive $\mathcal{G}_i = \{x, y, z, \text{SH}_{0\text{-}47}, o, s_0, s_1, s_2, r_0, r_1, r_2, r_3\}$ encodes its spatial position, view-dependent color, opacity, scale, and rotation. We observe that the relative importance of Gaussians can be largely inferred from their attributes and local neighborhood relations.

Four observations reveal the primitive importance: (1) Existing rendering-based methods already provide evidence that opacity and volume are central to significance estimation: by weighting projected areas with these factors, they approximate each Gaussian's contribution to blending and occlusion. Consistent with this, primitives with **small scales or low opacity** generally have limited visual influence compared to dominant neighbors, as shown in the top-right region of Fig. 1. (2) Spatial isolation provides another cue for importance estimation. Gaussians are typically aligned with the underlying geometry, while those with **abnormally large distances to their k nearest neighbors** tend to have lower importance, as illustrated by the floating points in the top-left region of Fig. 1. (3) Under-optimization offers the third evidence for importance prediction. During densification, only Gaussians that remain

consistently visible to the training cameras receive stable updates, while those with insufficient visibility fail to converge. Such Gaussians often exhibit an **abnormal appearance**: some display inconsistent or random colors compared with their neighbors, as seen in the clutter beneath the train (bottom-left and bottom-right of Fig. 1); others appear as nearly uniform single-color blobs because their higher-order spherical harmonics remain close to zero, leaving only the low-order terms to dominate the color. Both cases indicate insufficient optimization and, thus, low perceptual importance. (4) Finally, both global and local magnitude statistics provide complementary signals for importance estimation—**local relative magnitudes capture occlusion and redundancy, while global magnitudes balance visibility across the scene**.

These observations motivate the construction of a compact feature representation that embeds intrinsic Gaussian attributes with local neighborhood statistics. Building on this representation, we model the relationship between these features and Gaussian importance. Since explicitly handcrafting this mapping is challenging, we adopt a lightweight MLP to learn it automatically, as described in Sec. 4.

4. Proposed Method

Our proposed RAP framework consists of two stages. First, we perform importance-aware feature extraction, where each Gaussian is represented by a set of importance-related features derived from intrinsic attributes and local neighborhood statistics. These features are then normalized to ensure consistency across different scenes. Second, we employ a learning-based importance prediction module, in which a lightweight MLP maps the extracted features to importance scores. Three carefully designed loss functions are proposed to make sure the predicted importance scores are stably distributed from 0 to 1, resulting in flexible, effective, and strong generalization across datasets and downstream GS processing tasks.

4.1. Importance-aware Feature Extraction

We construct a compact 15-dimensional feature vector for each Gaussian primitive by combining intrinsic geometric and appearance attributes with normalized statistics, which consist of three steps: feature computation, feature normalization, and feature concatenation.

•**Feature computation.** For a Gaussian set $\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_N\}$, each primitive $\mathcal{G}_i$ is represented by a set of raw features,

$$\mathbf{F}_i^{raw} = \{d_i, A_i, s_{0i}, s_{1i}, s_{2i}, V_i, o_i, C_i\} \in R^{1\times 8}, \quad (1)$$

where d_i is the average K-NN distance, A_i is the color anisotropy, $s_{0,i}, s_{1,i}, s_{2,i}$ are the sorted Gaussian scales, V_i is the Gaussian volume, o_i is the opacity, and C_i is the DC color. These quantities are computed as follows.

- Average K-NN distance: Spatial isolation is measured as

$$d_i = \frac{1}{K} \sum_{j\in\mathcal{N}_K(i)} \|\mathbf{p}_i - \mathbf{p}_j\|_2, \quad (2)$$

  where $\mathbf{p}_i \in \mathbb{R}^3$ denotes the spatial position of Gaussian $\mathcal{G}_i$, and $\mathcal{N}_K(i)$ is the set of its K nearest neighbors in Euclidean space.
- Color anisotropy: To capture view-dependent appearance variation, we randomly sample M directions $\mathbf{v}_m$ and compute the corresponding RGB colors $\mathbf{c}_i(\mathbf{v}_m)$. The anisotropy is defined as the channel-wise standard deviation across directions:

$$A_i = \frac{1}{3} \sum_{c\in\{R,G,B\}} \sigma_m\big(c_i^c(\mathbf{v}_m)\big), \quad (3)$$

  where σ_m denotes the standard deviation over sampled directions. A higher A_i indicates a stronger view-dependent color change.
- Scales and volume: The scales are sorted to ensure rotation invariance: $s_{0,i} \le s_{1,i} \le s_{2,i}$. The Gaussian volume is then computed as

$$V_i = s_{0,i} \times s_{1,i} \times s_{2,i}. \quad (4)$$

- Opacity and DC color: Opacity o_i reflects the blending contribution. The DC color C_i is derived from the zeroth-order SH coefficients and averaged over the three RGB channels.

•**Feature Normalization.** To eliminate scale bias and enhance comparability across Gaussians, each raw feature $f_i \in \mathbf{F}_i^{raw}$ is standardized using both global and local statistics. Global normalization $f_i^{(G)}$ applies a scene-wide z-score to provide a consistent reference across the entire scene, where $\mu^{(G)}$ and $\sigma^{(G)}$ denote the global mean and standard deviation across all Gaussians. Local normalization $f_i^{(L)}$ further computes a K-NN based z-score to emphasize local contrast:

$$f_i^{(G)} = \frac{f_i - \mu^{(G)}}{\sigma^{(G)}}, \quad f_i^{(L)} = \frac{f_i - \mu_i^{(L)}}{\sigma_i^{(L)}}, \quad (5)$$

where $\mu_i^{(L)}$ and $\sigma_i^{(L)}$ are the local mean and standard deviation over the neighbor set $\mathcal{N}_K(i)$. For the DC color C_i, only local normalization is applied, as color deviation is typically a consequence of local densification while global normalization would be less meaningful due to natural scene characteristics.

Since z-score normalization centers features around zero with unit variance, the majority of values are expected to fall within $[-3, 3]$ according to the empirical three-sigma rule. In practice, however, extreme outliers may still occur. To enhance robustness, all features are clipped to a fixed percentile range and subsequently linearly rescaled to $[0, 1]$.

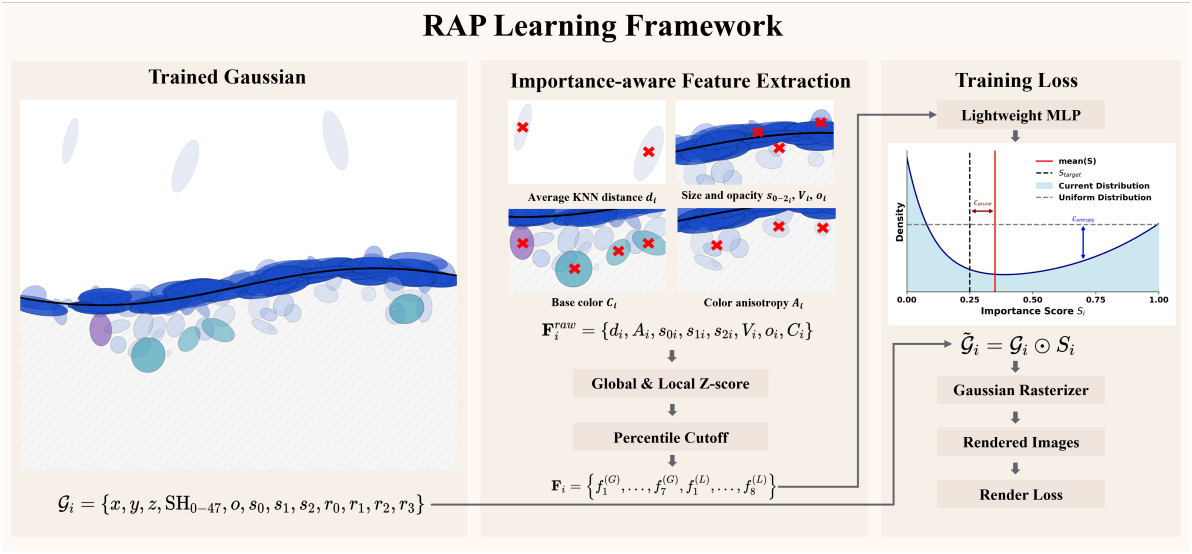


Figure 2. RAP learning framework.

•**Feature Concatenation.** Each Gaussian $\mathcal{G}_i$ is ultimately represented as

$$\mathbf{F}_i = \{f_1^{(G)}, \ldots, f_7^{(G)}, f_1^{(L)}, \ldots, f_8^{(L)}\}, \tag{6}$$

yielding a 15-dimensional vector (7 global + 8 local) that compactly encodes both intrinsic attributes and normalized neighborhood statistics. This representation provides a discriminative yet lightweight basis for importance prediction.

4.2. Learning Framework and Optimization

The features introduced in Sec. 4.1 capture diverse cues correlated with Gaussian primitive importance, but their interactions are highly nonlinear and difficult to model with handcrafted rules. To address this, we employ a lightweight MLP that learns to map the 15-dimensional feature vector $\mathbf{F}_i$ to an importance score. The network takes the normalized feature vector $\mathbf{F}_i$ as input and produces a single-dimensional output, which is passed through a sigmoid activation to yield an importance score $S_i \in [0, 1]$. The architecture is intentionally kept lightweight, avoiding convolutions and graph operations, which enables scalability to millions of Gaussians and ensures efficient inference.

Given a GS scene, we expect the importance scores to be smoothly distributed from 0 to 1. The importance score of each primitive can be explicitly quantified by the influence of pruning this primitive from the scene. Therefore, we adopt three complementary loss functions, ensuring stable and robust predictions by simulating pruning during the training:

•**Rendering Loss.** The first is a rendering loss, which ensures that the rendering quality after pruning based on importance scores is as high as possible. For differentiability, each Gaussian's opacity and scales are softly reweighted by its predicted importance score S_i:

$$\tilde{o}_i = o_i S_i, \quad \tilde{\mathbf{s}}_i = \mathbf{s}_i S_i, \quad \tilde{\mathcal{G}} = \{\tilde{o}_i, \tilde{\mathbf{s}}_i, \text{others}\}. \tag{7}$$

The modified set $\tilde{\mathcal{G}}$ is then rendered through the differentiable rasterizer $\mathcal{R}(\cdot)$. The loss adopts the standard 3DGS formulation:

$$\begin{aligned}\mathcal{L}_{\text{render}} = (1 - \lambda_{\text{dssim}})\mathcal{L}_1(\mathcal{R}(\tilde{\mathcal{G}}), I_{\text{gt}}) \\ + \lambda_{\text{dssim}}\, \mathcal{L}_{\text{D-SSIM}}(\mathcal{R}(\tilde{\mathcal{G}}), I_{\text{gt}}).\end{aligned} \tag{8}$$

Here, I_{gt} is the ground-truth reference image. This objective encourages the network to suppress redundant primitives while maintaining perceptual consistency with the ground-truth views.

•**Pruning-Aware Loss.** The second loss is a pruning-aware loss, designed to prevent trivial solutions. Without additional constraints and only rendering loss, the network could simply assign high importance to all Gaussians. To address this, we regularize the mean predicted score by penalizing deviations from a predefined target:

$$\mathcal{L}_{\text{prune}} = \left(\text{mean}(S_i) - S_{\text{target}}\right)^2, \tag{9}$$

where S_{target} specifies a target mean score that explicitly controls the pruning level. This objective drives the network to suppress redundant primitives in a manner that counterbalances the rendering loss, ensuring that pruning plays a constructive role in model efficiency without sacrificing rendering fidelity.

•**Distribution Regularization.** The third component is a distribution regularization that encourages smooth and diverse importance scores. Entropy acts as an intrinsic measure of prediction dispersion; by maximizing entropy, the model avoids collapsing into trivial binary outcomes,

thus enabling flexible and adaptive pruning under varying thresholds. To approximate entropy in a differentiable manner, we construct a soft histogram with B bins using Gaussian kernels, and compute a normalized entropy as

$$\text{EntropyNorm}(S) = -\frac{1}{\log B}\sum_{k=1}^{B}\tilde{p}_k \log(\tilde{p}_k + \epsilon), \quad (10)$$

where $\tilde{p}_k$ denotes the soft bin occupancy. In our case, we expect a larger entropy, the distribution regularization is thus defined as

$$\mathcal{L}_{\text{entropy}} = 1 - \text{EntropyNorm}(S). \quad (11)$$

This objective promotes a well-spread distribution of scores within $[0, 1]$, enabling pruning thresholds can be flexibly adjusted.

Overall Loss. The overall training objective combines the three components in a weighted sum:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{render}}\mathcal{L}_{\text{render}} + \lambda_{\text{prune}}\mathcal{L}_{\text{prune}} + \lambda_{\text{entropy}}\mathcal{L}_{\text{entropy}}. \quad (12)$$

Here, λ_{render}, λ_{prune}, and λ_{entropy} balance reconstruction fidelity, pruning strength, and score distribution regularization, respectively.

Model Training. In each epoch, one view is randomly sampled from one scene to promote generalization across diverse content. During training, pruning is simulated in a differentiable manner by softly reweighting Gaussians with their predicted scores, allowing the network to learn how removal influences rendering quality. At inference time, no rendering is performed: the normalized feature vector $\mathbf{F}_i$ is fed through the lightweight MLP once to obtain the importance score S_i, after which pruning is applied by a fixed threshold or a percentile-based ratio. This deployment protocol requires neither scene-specific retraining nor additional rendering, enabling fast, plug-and-play pruning on unseen datasets.

5. Experiments

5.1. Implementation Details

The MLP of RAP consists of three hidden layers with widths of 32, 32, and 16 neurons, respectively. We set $\lambda_{\text{dssim}} = 0.2$, $\lambda_{\text{render}} = 1.0$, $\lambda_{\text{prune}} = 1.0$, and $\lambda_{\text{entropy}} = 0.25$ for the loss terms. Feature extraction uses $K = 128$ nearest neighbors for local statistics, $B = 250$ bins for the entropy approximation, and a Gaussian kernel width of $\sigma = 0.01$. The MLP is trained for 15,000 iterations on 10 randomly selected scenes from the DL3DV-10K [19] dataset.

We evaluate RAP on the same three benchmarks as the original 3DGS [15], including the full set of scenes from Mip-NeRF360 [1] (five outdoor and four indoor scenes), two scenes from DeepBlending [10], and two scenes from Tanks&Temples [16]. All scenes are trained following the standard 3DGS protocol, using $7/8$ of the views for training and the remaining $1/8$ for testing.

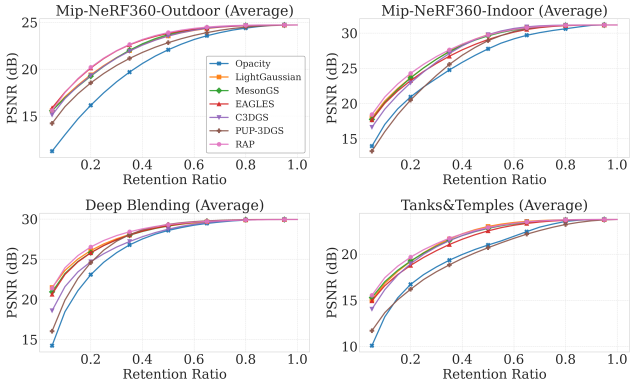


Figure 3. PSNR vs. retention ratio curves.

We compare RAP with multiple baselines, including a simple opacity-thresholding heuristic, visibility-based approaches such as LightGaussian [6], MesonGS [28], and EAGLES [8], and gradient-based approaches such as C3DGS [23] and PUP-3DGS [9]. For rendering-based baselines, importance scores are computed using the training views to ensure a fair comparison under identical conditions.

5.2. Post-hoc Pruning on Trained 3DGS

We perform post-hoc pruning experiments on pretrained 3DGS representations by retaining a fixed percentage (from 5% to 95%) of the Gaussians with the highest predicted importance scores.

Performance is evaluated by plotting retention-ratio vs. reconstruction-quality curves (PSNR, SSIM, LPIPS) in Fig. 3. Due to space constraints, we report only the average PSNR across the Mip-NeRF360-Outdoor, Mip-NeRF360-Indoor, Deep Blending, and Tanks&Temples datasets in the main paper; per-scene results and the full set of SSIM and LPIPS metrics are provided in the supplementary material.

Considering that pruning can be used as part of GS compression, to further quantify pruning efficiency, we compute the BD-Rate (Bjøntegaard Delta Bitrate) inspired by compression research of each method relative to the opacity-based baseline, using the average per-scene storage size as the rate axis. This metric provides a compact assessment of rate-distortion behavior across datasets—a lower BD-Rate indicates superior reconstruction quality under comparable storage budgets. The summarized results are presented in Table 1.

As shown in Fig. 3 and Table 1, our method consistently outperforms prior approaches across all datasets and pruning ratios. While the performance gap is relatively small under light pruning (e.g., retention >50%), it becomes increasingly pronounced as pruning becomes more aggres-

Table 1. BD-Rate (%) comparison of pruning-based Gaussian Splatting methods relative to the opacity baseline. Lower BD-Rate indicates better rate–distortion efficiency

	LightGS	MesonGS	EAGLES	C3DGS	PUP-3DGS	RAP
Mip-Outdoor	**-35.21**	-34.89	**-41.28**	-33.77	-22.54	**-42.63**
Mip-Indoor	**-31.15**	**-30.34**	-24.98	-27.63	-8.70	**-33.90**
Deep Blending	**-30.72**	-28.84	**-29.87**	-16.28	-19.46	**-36.76**
Tanks&Temples	**-37.98**	**-36.12**	-30.01	-34.37	7.06	**-40.11**

Table 2. Importance score computation time (seconds).

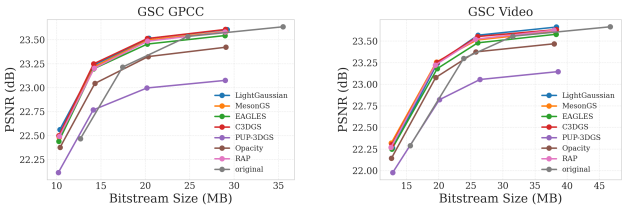
Dataset	Opacity	LightGS	MesonGS	EAGLES	C3DGS	PUP-3DGS	RAP
Mip-Outdoor	**3.73**	15.86	42.40	42.96	**11.69**	20.95	15.64
Mip-Indoor	**1.27**	22.71	14.84	14.97	15.96	21.22	**5.72**
Deep Blending	**2.99**	20.20	31.30	31.65	13.62	20.28	**11.64**
Tanks&Temples	**2.53**	18.62	17.53	13.92	9.22	20.50	**6.66**

sive. At a pruning ratio of 60%, RAP achieves up to a 0.5 dB PSNR gain over competing methods. The superiority is further reflected in BD-Rate reductions, with improvements such as -42.63% on the Mip-NeRF360-Outdoor dataset.

We also observe a notable difference in robustness between visibility-based and gradient-based methods. The former (e.g., LightGaussian, MesonGS, EAGLES) exhibit more consistent performance across datasets, while the latter (e.g., C3DGS, PUP-3DGS) show larger variance. For instance, C3DGS performs comparably on Mip-NeRF360 but suffers over 1 dB drop on Deep Blending. On Tanks&Temples, PUP-3DGS even underperforms the opacity baseline, with a positive BD-Rate of +7.06%, indicating poor generalization across diverse scenes.

Lastly, we find that when the importance estimation function is sufficiently accurate, up to 40% of Gaussians can be pruned with negligible degradation in rendering quality. Even at a 60% pruning rate, the average PSNR drop remains within 2 dB. These results suggest that reliable importance scoring can enable highly compact representations without significantly compromising visual fidelity.

Table 2 reports the computation time for importance score prediction (including point loading and metric computation). RAP is among the fastest methods across datasets: it ranks second (behind the trivial opacity baseline) on Mip-NeRF360-Indoor, Deep Blending, and Tanks&Temples, and third on Mip-NeRF360-Outdoor where C3DGS is slightly faster (11.69 s vs. 15.64 s for RAP). Despite this single exception, RAP consistently outpaces all visibility-based approaches (LightGaussian, MesonGS, EAGLES) by a wide margin, and is generally faster than gradient-based methods (C3DGS, PUP-3DGS). This advantage stems from RAP's feature-driven, rendering-free design, which avoids per-view rasterization and back-propagation; consequently, runtime scales with the number of primitives rather than with the number of training views, enabling efficient deployment across diverse scenes.



(a) GSC GPCC (b) GSC Video

Figure 4. R-D curves for GSC with 20% pruning

5.3. Pruning-in-the-Loop Training

To evaluate the potential of integrating pruning directly into the 3DGS generation process, we modify the pipeline to periodically remove redundant Gaussians during optimization. This experiment is motivated by the observation in Sec. 5.2, where pruning up to 40% of Gaussians post-training leads to negligible quality degradation.

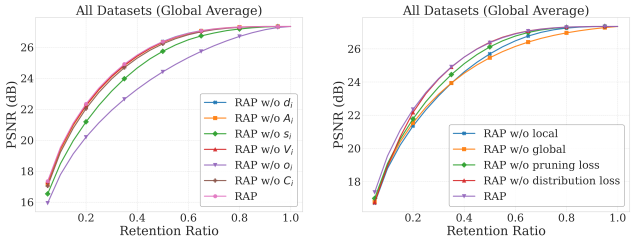
Specifically, we embed the pruning strategy into the densification phase: every **1500** training iterations, we apply a pruning operation to remove the bottom **40%** of Gaussians ranked by the importance score predicted by each method are removed. To assess the effectiveness of this integration, we compare the final reconstructions against the standard 3DGS training baseline (without pruning) using four metrics: PSNR, SSIM, LPIPS, and storage size (MB).

The results, summarized in Table 3, show that integrating pruning introduces negligible quality degradation while achieving substantial model size reduction. The reconstructed Gaussian sets are only about one-third to one-fifth the size of the original models, with similar performance across methods. The proposed RAP reports higher PSNRs than vanilla GS on MipNeRF360 Outdoor, Deep Blending, and Tanks & Temples, while other approaches generally give a PSNR drop of 0.3–0.5 dB. It indicates that the proposed RAP can facilitate the optimization process via accurately removing redundant primitives, as well as resulting in a better convergence direction. We found that all the pruning methods demonstrate a PSNR decrease on MipNeRF360 Indoor, and an obviously larger drop (around 1.5 dB) is observed with PUP-3DGS and EAGLES, revealing that this dataset is more challenging than the others.

On the other hand, directly using opacity as the pruning score also yields competitive PSNR results—ranking third on MipNeRF360-Outdoor and second on Deep Blending and Tanks&Temples. This can be explained by the densification behavior: when pruning is sub-optimal, the densification process tends to over-generate new Gaussians to compensate, resulting in more primitives but still competitive reconstruction quality, which can be found by the model size of "Opacity".

Table 3. Reconstruction quality and size comparison under integrated pruning (40% pruning every 1500 iterations).

Method	MipNeRF360 Outdoor				MipNeRF360 Indoor				Deep Blending				Tanks & Temples			
	PSNR ↑	SSIM ↑	LPIPS ↓	Size ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Size ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Size ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Size ↓
3DGS	24.688	0.729	0.238	926.895	31.076	0.925	0.186	299.202	29.817	0.907	0.239	586.064	23.734	0.853	0.169	372.036
Opacity	24.550	0.718	0.275	313.657	**30.604**	**0.912**	0.216	72.176	**29.899**	**0.910**	**0.248**	149.873	**23.674**	**0.842**	**0.200**	118.414
LightGaussian	**24.676**	**0.728**	**0.255**	300.907	**30.777**	**0.917**	**0.205**	71.676	**29.839**	**0.908**	**0.249**	155.207	**23.634**	**0.845**	**0.191**	118.048
MesonGS	24.554	**0.723**	**0.256**	297.255	30.173	**0.914**	**0.207**	71.587	29.638	0.904	0.254	154.137	23.541	**0.839**	**0.197**	115.922
EAGLES	24.520	0.722	**0.256**	**295.899**	29.801	0.911	**0.210**	71.356	29.551	0.904	0.254	154.352	23.212	0.834	0.202	**113.500**
C3DGS	**24.638**	**0.723**	0.262	303.820	30.271	**0.912**	0.212	**67.821**	29.739	0.905	0.258	**139.431**	23.421	0.836	0.205	115.941
PUP-3DGS	24.328	0.714	**0.261**	**281.821**	29.422	0.909	0.212	**70.056**	29.699	**0.908**	**0.248**	**145.294**	22.365	0.816	0.215	**107.662**
RAP	**24.709**	**0.726**	0.263	**285.067**	**30.696**	0.911	0.218	**67.804**	**29.913**	**0.909**	**0.252**	**147.250**	**23.774**	**0.842**	0.201	**113.236**



(a) Individual features (b) Normalization and loss

Figure 5. Ablation studies of the proposed feature and loss.

5.4. Integration with MPEG GSC

To evaluate whether importance-guided pruning benefits downstream compression, we integrate the pruning step as a pre-processing module into both branches of MPEG Gaussian Splat Coding (GSC) [14]: the G-PCC–based [13] and the video-based pipeline [18]. For each scene, 20% of the Gaussians are removed according to the predicted importance scores, and the remaining primitives are encoded using identical codec settings at four rate points. Rate–distortion (R-D) curves are reported in Fig. 4. Importance-guided pruning consistently improves coding efficiency for both branches. Most pruning methods achieve **15–20%** BD-Rate gains. RAP remains in the top performance band, indicating strong generalization under diverse scene conditions. In contrast, PUP-3DGS and simple opacity thresholding exhibit unstable behavior across datasets.

Although the G-PCC–based pipeline attains higher PSNR at lower bitrates than the video-based one, both benefit substantially from pre-pruning. Overall, these results show that effective importance-guided pruning produces compact yet encoder-friendly Gaussian sets and significantly boosts GSC coding efficiency regardless of the codec design.

5.5. Ablation Studies

To better understand the behavior and design choices of RAP, we conduct comprehensive ablation studies feature effectiveness and loss formulation. We perform a series of experiments to analyze two aspects: (1) the effectiveness of the proposed feature design and normalization strategy; and (2) the contribution of each loss term in our RAP training.

•**Effectiveness of feature design.** As formulated in Equation 1, RAP integrates geometric and appearance descriptors $\{d_i, A_i, s_{0,i}, s_{1,i}, s_{2,i}, V_i, o_i, C_i\}$ together with their local and global normalized counterparts. To assess the contribution of each component, we perform two ablation experiments.

Fig. 5a evaluates the contribution of each feature by removing one attribute at a time while keeping all others unchanged. Opacity o_i proves to be the most crucial cue—its removal leads to about 1-2 dB PSNR drop at the same pruning ratio. Gaussian scales $\{s_{0,i}, s_{1,i}, s_{2,i}\}$ follow as the next most influential features, causing around 0.5–1 dB degradation when excluded. Other features, including color anisotropy A_i and average KNN distance d_i, provide smaller yet consistent benefits, particularly at aggressive pruning ratios, indicating their complementary role.

•**Normalization and loss formulation.** Fig. 5b analyzes the effect of normalization and loss design. Removing either local or global normalization leads to a substantial PSNR drop (1.5–2 dB), confirming their complementary roles: local normalization enhances intra-region contrast to better separate redundant primitives, while global normalization ensures scene-level consistency and stabilizes feature magnitudes across datasets. Together, they support robust and generalizable importance estimation under diverse scene conditions.

The pruning-aware loss also plays a critical role. Removing it results in a moderate but consistent degradation (around 0.5 dB on average), as the network loses explicit control over the expected pruning ratio. As visualized in the supplementary material (Sec. 9), the absence of this loss causes the predicted score distribution to shift toward a higher mean, reflecting overly conservative pruning.

The distribution loss contributes smaller yet steady improvements, with a noticeable gain at low retention ratios (approximately 0.25 dB). This effect is most evident in dense and complex scenes, where enforcing a smooth score distribution helps differentiate subtle importance variations among overlapping primitives. Without this loss, the predicted scores collapse toward near-binary outputs around 0 and 1 (see Sec. 9), reducing the model's flexibility to support arbitrary pruning ratios.

6. Conclusion and Limitations

We presented RAP, a fast feedforward rendering-free and attribute-guided framework for estimating primitive importance in 3DGS. By leveraging intrinsic attributes and normalized neighborhood statistics, RAP avoids view-dependent rendering analysis while achieving strong generalization, efficient inference, and consistent improvements in pruning, reconstruction, and compression. Our current integration with downstream tasks relies on pruning a fixed global ratio based on predicted scores. A more principled coupling with reconstruction and compression remains open—particularly how to allocate different sampling densities across regions or how to enable hierarchical or region-adaptive coding within GSC. We leave these directions for future exploration.

References

[1] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, pages 5470–5479, 2022. 6

[2] Yiwen Chen, Zilong Chen, Chi Zhang, Feng Wang, Xiaofeng Yang, Yikai Wang, Zhongang Cai, Lei Yang, Huaping Liu, and Guosheng Lin. Gaussianeditor: Swift and controllable 3d editing with gaussian splatting. In *CVPR*, pages 21476–21485, 2024. 3

[3] Yihang Chen, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. Hac: Hash-grid assisted context for 3d gaussian splatting compression. In *ECCV*, pages 422–438. Springer, 2024. 2, 3

[4] Yihang Chen, Mengyao Li, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. Pcgs: Progressive compression of 3d gaussian splatting. *arXiv preprint arXiv:2503.08511*, 2025. 1

[5] Yihang Chen, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. Hac++: Towards 100x compression of 3d gaussian splatting. *arXiv preprint arXiv:2501.12255*, 2025. 1, 2, 3

[6] Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, Zhangyang Wang, et al. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. *NIPS*, 37:140138–140158, 2024. 1, 2, 3, 6

[7] Guangchi Fang and Bing Wang. Mini-splatting: Representing scenes with a constrained number of gaussians. In *ECCV*, pages 165–181. Springer, 2024. 3

[8] Sharath Girish, Kamal Gupta, and Abhinav Shrivastava. Eagles: Efficient accelerated 3d gaussians with lightweight encodings. In *ECCV*, pages 54–71. Springer, 2024. 2, 3, 6

[9] Alex Hanson, Allen Tu, Vasu Singla, Mayuka Jayawardhana, Matthias Zwicker, and Tom Goldstein. Pup 3d-gs: Principled uncertainty pruning for 3d gaussian splatting. In *CVPR*, pages 5949–5958, 2025. 1, 3, 6

[10] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (ToG)*, 37(6):1–15, 2018. 6

[11] He Huang, Wenjie Huang, Qi Yang, Yiling Xu, and Zhu Li. A hierarchical compression technique for 3d gaussian splatting compression. In *ICASSP*, pages 1–5, 2025. 2

[12] He Huang, Qi Yang, Mufan Liu, Yiling Xu, and Zhu Li. Adc-gs: Anchor-driven deformable and compressed gaussian splatting for dynamic scene reconstruction. In *IJCAI*, 2025. 1, 2

[13] ISO/IEC JTC 1/SC 29. Information technology — coded representation of immersive media — part 9: Geometry-based point cloud compression. Technical Report ISO/IEC 23090-9:2023, ISO/IEC, 2023. 8

[14] ISO/IEC JTC 1/SC 29/WG 07. Description of jee 6.3 on exploration of 3DGS representation and coding technologies, 2025. 8

[15] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 2, 6

[16] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36 (4):1–13, 2017. 6

[17] Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3d gaussian representation for radiance field. In *CVPR*, pages 21719–21728, 2024. 3

[18] Sicheng Li, Chengzhen Wu, Hao Li, Xiang Gao, Yiyi Liao, and Lu Yu. Gscodec studio: A modular framework for gaussian splat compression. *arXiv preprint arXiv:2506.01822*, 2025. 8

[19] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *CVPR*, pages 22160–22169, 2024. 6

[20] Yifei Liu, Zhihang Zhong, Yifan Zhan, Sheng Xu, and Xiao Sun. Maskgaussian: Adaptive 3d gaussian representation from probabilistic masks. In *CVPR*, pages 681–690, 2025. 1, 2, 3

[21] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *CVPR*, pages 20654–20664, 2024. 1

[22] Saswat Subhajyoti Mallick, Rahul Goel, Bernhard Kerbl, Markus Steinberger, Francisco Vicente Carrasco, and Fernando De La Torre. Taming 3dgs: High-quality radiance fields with limited resources. In *SIGGRAPH Asia*, pages 1–11, 2024. 3

[23] Simon Niedermayr, Josef Stumpfegger, and Rüdiger Westermann. Compressed 3d gaussian splatting for accelerated novel view synthesis. In *CVPR*, pages 10349–10358, 2024. 1, 3, 6

[24] Yunji Seo, Young Sun Choi, HyunSeung Son, and Youngjung Uh. Flod: Integrating flexible level of detail into 3d gaussian splatting for customizable rendering. *ACM Trans. Graph.*, 44(4), 2025. 1

[25] Yuang Shi, Géraldine Morin, Simone Gasparini, and Wei Tsang Ooi. Lapisgs: Layered progressive 3d gaussian

splatting for adaptive streaming. In *3DV*, pages 991–1000. IEEE, 2025. 1

[26] Yufei Wang, Zhihao Li, Lanqing Guo, Wenhan Yang, Alex Kot, and Bihan Wen. Contextgs: Compact 3d gaussian splatting with anchor level context model. *NIPS*, 37:51532–51551, 2024. 1, 2, 3

[27] Yuxuan Wang, Xuanyu Yi, Zike Wu, Na Zhao, Long Chen, and Hanwang Zhang. View-consistent 3d editing with gaussian splatting. In *ECCV*, pages 404–420. Springer, 2024. 3

[28] Shuzhao Xie, Weixiang Zhang, Chen Tang, Yunpeng Bai, Rongwei Lu, Shijia Ge, and Zhi Wang. Mesongs: Post-training compression of 3d gaussians via efficient attribute transformation. In *ECCV*, pages 434–452. Springer, 2024. 1, 2, 3, 6

[29] Qi Yang, Kaifa Yang, Yuke Xing, Yiling Xu, and Zhu Li. A benchmark for gaussian splatting compression and quality assessment study. In *ACMMM Asia*, pages 1–8, 2024. 1

[30] Qi Yang, Le Yang, Geert Van Der Auwera, and Zhu Li. Hybridgs: High-efficiency gaussian splatting data compression using dual-channel sparse representation and point cloud encoder. In *ICML*, 2025. 3

[31] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. In *ECCV*, pages 162–179. Springer, 2024. 3

[32] Zhaoliang Zhang, Tianchen Song, Yongjae Lee, Li Yang, Cheng Peng, Rama Chellappa, and Deliang Fan. Lp-3dgs: Learning to prune 3d gaussian splatting. *NIPS*, 37:122434–122457, 2024. 3

[33] Brent Zoomers, Maarten Wijnants, Ivan Molenaers, Joni Vanherck, Jeroen Put, and Nick Michiels. Progs: Progressive rendering of gaussian splats. In *WACV*, pages 3118–3127. IEEE, 2025. 1, 2

RAP: Fast Feedforward Rendering-Free Attribute-Guided Primitive Importance Score Prediction for Efficient 3D Gaussian Splatting Processing

Supplementary Material

7. Robustness of rendering-based methods.

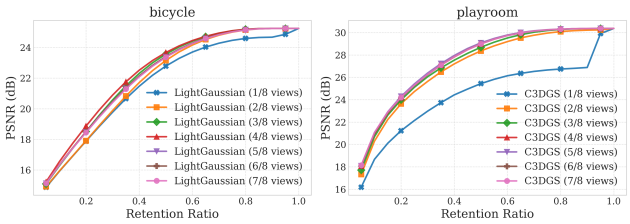


Figure 6. Robustness of rendering-based importance estimation to the number of training views used. Left: LightGaussian on the *bicycle* scene from Mip-NeRF360-Outdoor. Right: C3DGS on the *playroom* scene from Deep Blending. Each curve corresponds to a different subset of views ($1/8$–$7/8$) used for score computation.

As discussed in Section 5.1, rendering-based baselines estimate importance by projecting Gaussians from all available training views (i.e., $7/8$ of the total views). To analyze their sensitivity to the number of views, we uniformly subsample $1/8$, $2/8$, …, $7/8$ of the training views to recompute importance scores and evaluate post-hoc pruning results. As shown in Fig. 6, rendering-based methods exhibit notable instability when the number or distribution of input views changes. For instance, LightGaussian's pruning quality on the *bicycle* scene fluctuates by up to 1.5 dB PSNR across different view counts, while C3DGS on *playroom* suffers a 4 dB drop when using only $1/8$ of the views and still varies by about 1 dB across the remaining subsets. Besides, using more views does not always yield better scores: LightGaussian achieves its best performance with $4/8$ views, and C3DGS peaks at $5/8$. These results suggest that rendering-based importance estimation is highly dependent on the number and spatial distribution of selected views, whereas our RAP, relying solely on point-level features without rendering, is inherently view-agnostic and thus more robust.

8. Post-hoc Pruning: Additional Results

Per-scene results are shown in Fig. 9, RAP consistently achieves the highest PSNR on nearly all scenes, with the only exceptions being *bicycle* and *stump* from the Mip-NeRF360 Outdoor dataset, where EAGLES is approximately 0.3 dB better around the 60% pruning ratio. Visibility-based approaches such as LightGaussian, MesonGS, and EAGLES tend to perform slightly worse than RAP (typically within 0.3 dB). Although these methods differ in their use of opacity, blending opacity, or volume weighting, their overall behavior is largely comparable. Each achieves slightly better results on certain scenes and slightly worse on others, but none exhibits a consistent or systematic advantage across datasets.

In contrast, gradient-based methods (C3DGS, PUP-3DGS) are far less stable. They show substantial degradation on several scenes—*kitchen*, *playroom*, *truck*, and *train*—where their PSNR drops by 2–3 dB relative to RAP. At certain pruning ratios, their performance even falls below the naive opacity baseline (e.g., C3DGS on *playroom*, PUP-3DGS on *truck*).

For SSIM and LPIPS, RAP typically ranks second or third on most Mip-NeRF360 scenes, with EAGLES often obtaining the best perceptual scores. To better understand these differences, we visualize the retained primitives on the *garden* scene in Fig. 7. The comparison reveals distinct selection biases across methods:

- **Visibility-based methods (e.g., EAGLES).** These methods strongly favor primitives near the scene center. This arises because: (1) training views surround the central object, causing central primitives to accumulate many projected contributions while background primitives receive far fewer; and (2) projected area decreases with depth, causing distant primitives to appear smaller and thus receive lower scores. Consequently, foreground structures are well preserved, but background regions are severely underrepresented.
- **Gradient-based methods (e.g., C3DGS).** These approaches mainly retain high-frequency edges, while smooth surfaces lose most of their support. Under heavy pruning, this leads to incomplete geometry and strong artifacts.
- **RAP (ours).** RAP produces a more uniform selection across the entire scene, preserving both foreground and background content. This explains why RAP achieves the strongest PSNR across datasets—PSNR benefits from globally consistent coverage—whereas SSIM and LPIPS, which emphasize structural similarity, may sometimes favor visibility-based techniques.

9. Loss Function Analysis and Score Distribution Visualization

To better understand how the pruning-aware loss and the distribution regularization shape the predicted scores, Fig. 8 visualizes the score distributions produced by RAP and its ablated variants. Without the pruning-aware loss, the net-

(a) Original scene

(b) RAP (ours, 5% retained)

(c) EAGLES (visibility-based, 5% retained)

(d) C3DGS (gradient-based, 5% retained)

Figure 7. Pruning behavior of different importance estimators at 5% retention. EAGLES favors central regions, C3DGS keeps edges, while RAP produces a more uniform, structure-preserving subset.

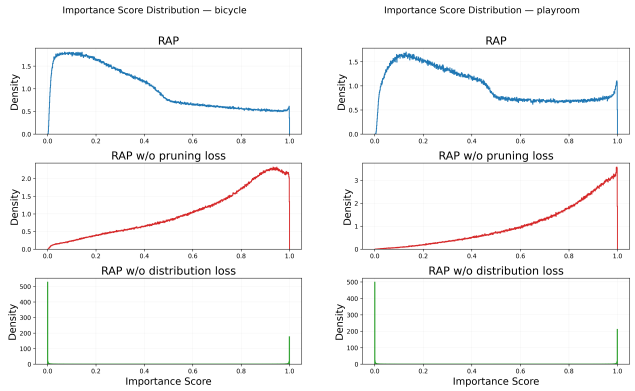


Figure 8. Predicted score distributions for RAP, RAP without pruning-aware loss, and RAP without distribution loss.

work tends to assign overly large scores to most primitives, making it difficult to identify truly important points. In contrast, removing the distribution loss causes the scores to collapse toward near-binary values around 0 and 1, which prevents setting flexible pruning thresholds and makes fine-grained importance discrimination unreliable. The full RAP model produces a smooth and well-spread distribution—dominated by low scores but with clear separation among mid- and high-importance primitives—enabling stable and accurate pruning across a wide range of ratios.

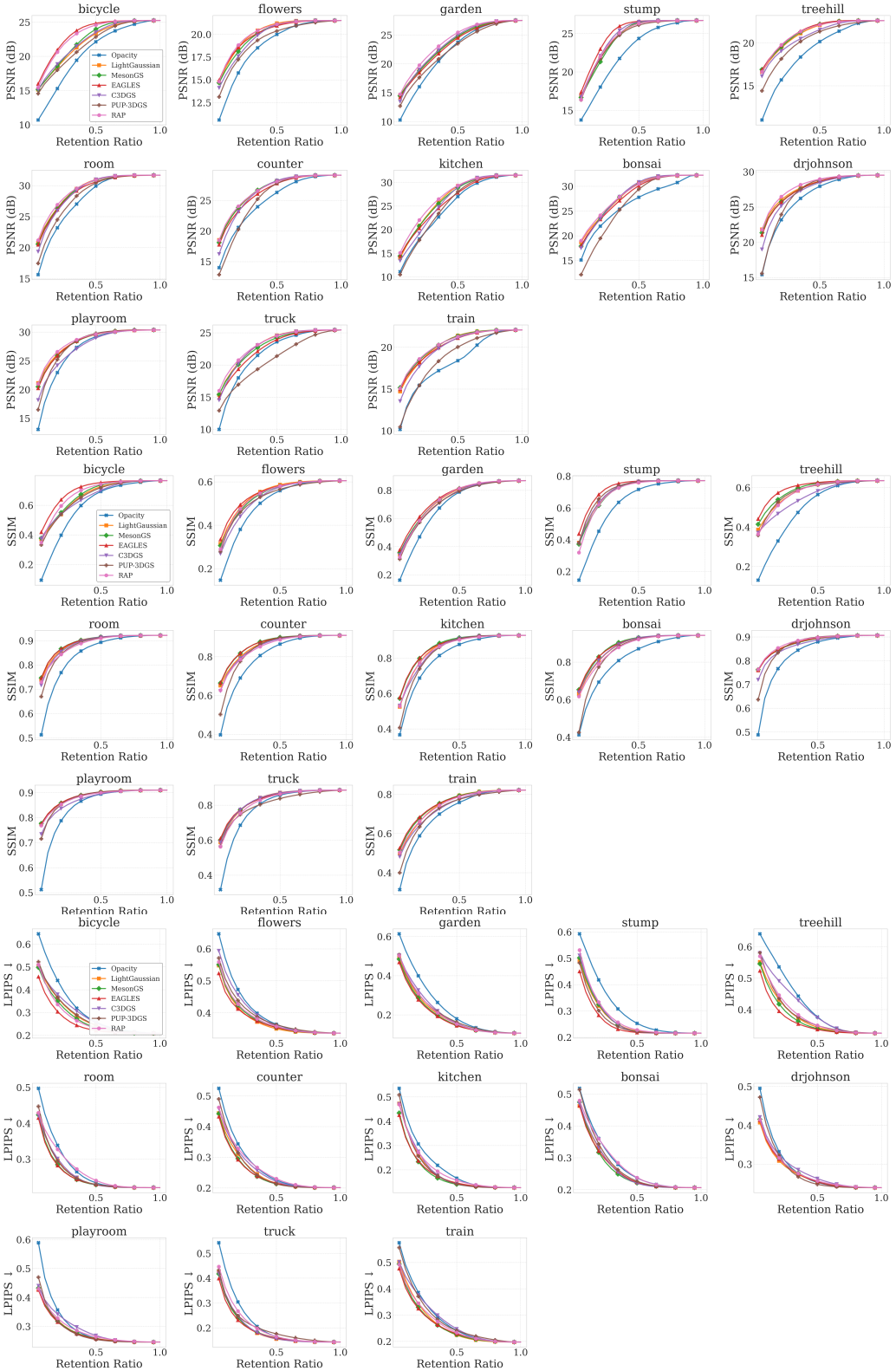


Figure 9. Per-scene post-hoc pruning results across three metrics: PSNR (top), SSIM (middle), and LPIPS (bottom).