

A Large-scale Dataset of Gaussian Splats and Their Self-Supervised Pretraining

Appendix

Abstract

This supplementary material first provides the nomenclature of our method for reference, followed by further information for better reproducibility as well as additional evaluations and qualitative results.

A. Nomenclature

Gaussian parameters

C	Center of Gaussian splats
O	Opacity of Gaussian splats
S	Scale of Gaussian splats
R	Rotation as quaternions of Gaussian splats
SH	Spherical harmonics of Gaussian splats
X	Gaussian splats, $X = [C, O, S, R, SH]$
CT	Center splats obtained by Furthest Point Sampling

Method variables

$G(\cdot)$	Grouping feature: selected Gaussian parameters for computing the distance
$E(\cdot)$	Embedding feature: selected Gaussian parameters served as the input for masked autoencoder
f_G	Dimension of the grouping feature
f_E	Dimension of the embedding feature
T	Group tokens obtained from the tokenizer, $T \in \mathbb{R}^{n \times D}$
T_v	Visible tokens after masking, $T_v \in \mathbb{R}^{(1-r)n \times D}$
T_m	Masked tokens, $T_m \in \mathbb{R}^{rn \times D}$
E_{group}	Concatenation of embedding feature of all the k splats in a group
\hat{E}_m	Reconstructed embedding feature for masked regions
E_m	Masked embedding feature
z	Latent variable obtained from the encoder, $z = f_\theta(T_v)$

$f_\theta(\cdot)$	Encoder
$g_\phi(\cdot)$	Decoder
T_l	Learnable token that is concatenated with z
\hat{T}_m	Recovered masked token, $\hat{T}_m = g_\phi(z \oplus T_l)$
Φ	Projector outputting the recovered embedding feature
τ	True distribution function from which E is sampled
$\mathbb{E}[\cdot]$	Expectation operator
$\mathcal{L}_{\text{recon}}$	Reconstruction loss calculated on embedding feature E

Scalars

N	Splats number of the original splatted object
p	Splats number after downsampling
n	Groups numbers splats are splitted into
k	Per-group splats number
r	Mask ratio of the splats groups

B. Implementation Details

Problematic CAD Model: We excluded two models due to error 'No vertex found' encountered when loading into Blender: 4a32519f44dc84aabafe26e2eb69ebf4 and 67ada28ebc79cc75a056f196c127ed77.

Model Rendering. Fig. B.1 illustrates the evenly chosen 72 views across the upper atmosphere when rendering a CAD model in ShapeNet [1] and ModelNet [6].

3DGS Initialization. To balance quality and speed, $5K$ points are sampled for 3DGS initialization. We employ a surface sampling method similar to the point cloud baseline. The normals of the sampled points are determined using the normals of their corresponding faces, and colors are assigned through interpolation from neighboring points, when available. For objects without material data, we initialize the color to gray.

Gaussian-MAE. Gaussian-MAE model is configured with a masked encoder and a decoder, in which the masked encoder consists of the tokenizer and transformer encoder. The transformer encoder has a dimension of 384, a depth of 12 layers, and employs 6 attention heads per layer. The encoder uses a mask ratio of 0.6 and includes positional embeddings. The decoder shares the same dimension of 384

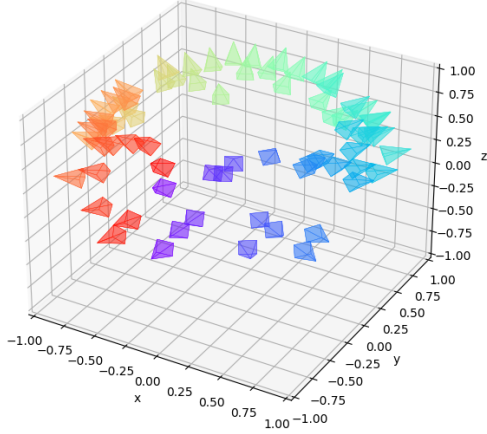


Figure B.1. **Sampled Views for *ShapeSplat* rendering.** We evenly select 72 views on the upper atmosphere for every object.

and utilizes 6 attention heads but with a shallower depth of 4 layers. Both the encoder and decoder are regularized using a drop path rate of 0.1. In addition, for the masked auto-encoder (MAE), we use the ℓ_2 Chamfer-Distance [2] following [3]. Let $\hat{E}_m = \Phi(\hat{T}_m)$ and E_m be the reconstructed embedding feature and ground truth embedding feature, respectively. The reconstruction loss $\mathcal{L}_{\text{recon}}$ can be written as:

$$\mathcal{L}_{\text{recon}} = \sum \left[\frac{1}{|\hat{E}(C)|} \sum_{re \in \hat{E}(C)} \min_{gt \in E(C)} \|re - gt\|_2^2 + \sum_{gt \in E(C)} \min_{re \in \hat{E}(C)} \|re - gt\|_2^2 + \sum_{re \in \hat{E}(O, S, R, SH)} \|re - gt\|_1 \right]. \quad (1)$$

Tokenizer. The tokenizer for Gaussian parameters first utilizes a `Conv1d` layer that project the raw Gaussian parameters (embedding feature such as center, opacity, scale, rotation, and sh) into a 128-dimensional tensor. It then employ our splats pooling layer that effectively aggregate information from a larger neighborhood down to a smaller dimension. Finally, a second `Conv1d` followed by max pooling processes the features, resulting in a group token of dimension 384, which is then passed to the transformer encoder.

Parameter Numbers. Tab. B.1 summarizes the total and trainable parameter numbers for our pretraining and finetuning model, as well as the giga floating point operations (GFLOPs) and giga multiply-accumulate operations (GMACs). The pretraining model has the most trainable parameters of 28.79M, while the linear probing in finetuning is the lightest with only 0.008M trainable parameters.

Training Details. Experiments on pretraining were conducted using A6000 GPUs with a batch size of 128. For

finetuning with 2048, 4096, and 8192 sampled splats, we used a batch size of 224 on H100 GPU. Both the pretraining and finetuning stages were trained over 300 epochs, utilizing a cosine learning rate scheduler with a 10-epoch warm-up period. The AdamW optimizer was employed, with learning rates set to 1×10^{-3} for pretraining and 5×10^{-4} for fine-tuning, alongside a weight decay of 0.05. More details are provided in Tab. B.2.

Evaluation. For the classification task, *ShapeSplat* uses the labels from the original dataset. In the part segmentation task, to ensure a fair comparison with point cloud-based methods, our model predicts labels for the given point cloud positions. Features from all Gaussian centers are forwarded to the ground truth point cloud locations through distance-based interpolation, followed by feature integration using a `Conv1d` layer. We report the best results from the separate runs using checkpoints saved at epochs 250, 275, and 300 during pretraining.

As discussed in [5], the ModelNet40 dataset contains notable duplication and label errors that hinder model performance. The evaluation on ModelNet10, which is less affected by these issues, provides a more reliable benchmark and we assign more weight on it when drawing conclusions.

Model	Total (M)	Trainable (M)	GFLOPs	GMACs
Pretrain	28.79	28.79	93.77	46.16
Finetune (full)	21.78	21.78	235.06	116.65
Finetune (mlp3)	21.78	0.267	235.06	116.65
Finetune (linear)	21.52	0.008	235.04	116.64

Table B.1. **Model Parameters and Computation Counts.** The table reports the total and trainable parameters (in millions), as well as GFLOPs and GMACs for the pretraining and finetuning.

C. Per-attribute Reconstruction Error

Tab. C.1 provides the detailed per-attribute reconstruction error corresponding to Figure 6 in the main paper. Note that the target reconstruction is determined by the choice of embedding feature, which results in blank areas in the table. Bundling other Gaussian attributes with the center into either the embedding feature or the grouping feature leads to improved overall reconstruction, and as evident from the classification results, better reconstruction then yields performance increase. We also present the results for *Grouping in Gaussian Feature Space without Center*, where Gaussian centers are entirely excluded during pretraining and finetuning. This experiment shows that reasonable performance can still be achieved by relying solely on other parameters.

D. Gaussian Splats Reconstruction

Fig. D.1 presents the reconstructed Gaussian splats using the Gaussian-MAE model with input of 4096 Gaussians during pretraining. Our model effectively rebuilds fine details, such as the foot of the table and the frames of the chair.

	Pre-training	Classification		Segmentation
Config	ShapeNet [1]	ScanObjectNN [4]	ModelNet [6]	ShapeNetPart [7]
optimizer	AdamW	AdamW	AdamW	AdamW
learning rate	1e-3	5e-4	5e-4	1e-4
weight decay	5e-2	5e-2	5e-2	5e-2
learning rate scheduler	cosine	cosine	cosine	cosine
training epochs	300	300	300	300
batch size	128	32	128	128
number of splats	1024	2048	1024	2048
number of splats groups	64	128	64	128
splats group size	32	32	32	32
augmentation	Scale&Trans	Scale&Trans	Scale&Trans	Scale&Trans
GPU device	1 A6000 (48G)	1 A6000 (48G)	1 A6000 (48G)	1 A6000 (48G)

Table B.2. **Hyperparameter Recipes for Pretraining and Finetuning.**

Embedding Feature	Grouping Feature	Reconstruction Error					ModelNet10 Accuracy (%)
		centroid	opacity	scale	rotation	sh	
Grouping only by Gaussian Center							
center	center	2.27	-	-	-	-	93.72
center + opacity	center	2.53	0.22	-	-	-	93.83
center + sh	center	2.71	-	-	-	0.361	93.83
center + scale + rotation	center	2.53	-	0.0128	0.126	-	94.27
all	center	2.70	0.20	0.0156	0.132	0.364	95.37
Grouping only by Gaussian Center, use Splats Pooling Layer							
center	center	2.32	-	-	-	-	94.71
center + opacity	center	2.60	0.19	-	-	-	94.82
center + sh	center	3.11	-	-	-	0.050	94.82
center + scale + rotation	center	2.82	-	0.0101	0.125	-	95.37
center+opacity+scale+rotation	center	2.63	0.19	0.0097	0.122	-	95.82
all	center	2.72	0.19	0.0098	0.127	0.047	95.48
Grouping in Gaussian Feature Space, without Center							
opacity	opacity	-	0.007	-	-	-	51.32
scale + rotation	scale + rotation	-	-	0.0089	0.047	-	90.86
sh	sh	-	-	-	-	0.063	78.96
opacity+scale+rotation+sh	opacity+scale+rotation+sh	-	0.058	0.0106	0.069	0.166	92.29
Grouping in Gaussian Feature Space, with Center							
center + opacity	center + opacity	5.12	0.05	-	-	-	94.60
center + scale + rotation	center + scale + rotation	5.76	-	0.010	0.066	-	94.89
center + sh	center + sh	3.50	-	-	-	0.118	95.04
all	all	9.19	0.08	0.011	0.082	0.201	95.26

Table C.1. **Per-attribute Reconstruction Error of Different Grouping Methods.**

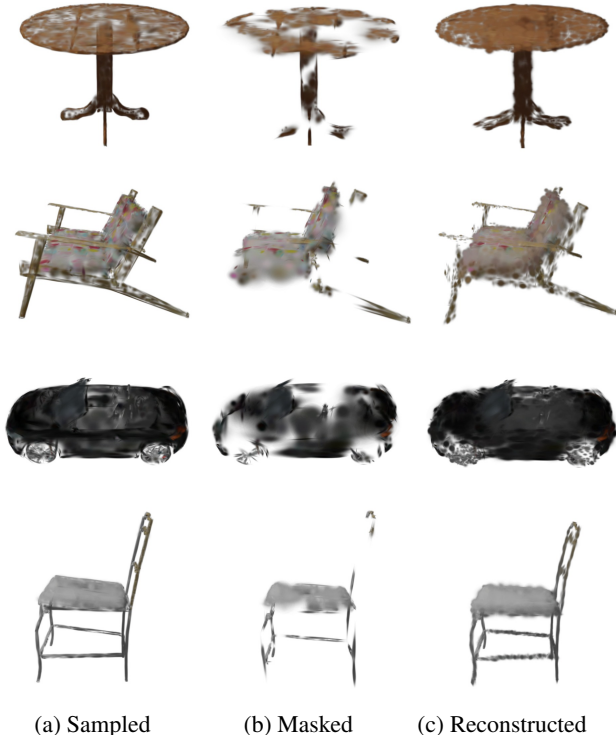


Figure D.1. **Visualizations of Masked Inputs and Reconstructed Gaussians.** We show: (a) Renderings of downsampled 4096 Gaussians, (b) Renderings with 60% Gaussians masked, (c) Renderings of reconstructed Gaussians. The pretrained Gaussian-MAE is able to reconstruct fine details, *e.g.*, frames of the chair.

Additionally, Fig. D.2 compares the reconstructions on the same objects but with 1K, 4K, and 8K Gaussians. The ones with 8K Gaussians clearly capture more color and geometric details. Not only does pretraining on denser inputs result in better reconstructions, but it also boosts performance on downstream tasks (*cf.* Tab. E.1).

E. Ablation on Pretraining Splats Number

Throughout our experiments in the main paper, we use an input number of 1K for pretraining. Tab. E.1 ablates different input number of Gaussians *w.r.t.* downstream task performance. Pretraining using 8K Gaussians leads to the best results on ModelNet10 classification and ShapeNet-Part segmentation. Compare to training from scratch, the pretraining stage significantly boost the downstream task performance.

F. Details on ScanObjectNN Experiments

In the main paper we report the generalization performance of our pretrained model on the real-world point clouds in ScanObjectNN [4] dataset. We didn’t report the performance on the Gaussian splatted objects as the segmented

Input	ModelNet10	ModelNet40	ShapeNet-Part
N\A	93.39	91.44	83.2
1024	95.37	93.35	86.0
2048	95.37	92.95	85.8
4096	95.26	92.58	85.9
8192	95.70	92.70	86.1

Table E.1. **Ablation on Number of Splats in Pretraining** (overall accuracy for ModelNet10 and ModelNet40, instance mIoU for ShapeNet-Part). We increase the total number of Gaussian splats from 1024 to 2048, 4096, and 8192 and the number of splats groups by the respective multiples during pretraining. N\A refers to training from scratch, *i.e.*, without pretraining stage. Evidently, pretraining stage significantly boost the downstream task performance.

mesh is not provided but only point clouds in ScanObjectNN dataset, thus it’s not possible to render views for the objects and train the Gaussian splats. Given Gaussian-MAE already outperforms the baseline [3] in object-only classification, we expect a larger performance gain when finetuned using Gaussian splats.

G. Future Work

While our proposed method successfully employed a self-supervised learning strategy via MAE, achieving competitive results compared to point cloud counterparts, there are several promising avenues for further exploration based on our dataset. One direction is to delve deeper into the established self-supervised learning paradigm by leveraging Gaussian attributes to enhance the encoder’s informativeness for downstream tasks. Also, exploring ways to utilize all Gaussian splats without downsampling is important, as downsampling significantly reduces reconstruction quality, as shown in Fig. D.2. Additionally, transferring knowledge from 2D foundation models into the Gaussian domain presents an intriguing direction. Furthermore, unlike image or point cloud data, which have seen integration with large language models (LLMs), exploring similar trends to integrate LLMs into the 3DGS field could yield valuable insights.

H. Impact Statement

The introduction of the ShapeSplat dataset and the Gaussian-MAE model mark new advancements in 3D representation and understanding. ShapeSplat, a large-scale dataset with around 65K objects across 87 categories, was created using 2 GPU years of computation on a TITAN XP GPU, providing a robust resource for research in 3D Gaussian Splatting (3DGS). Our work facilitates unsupervised pretraining and supervised finetuning for classification and segmentation tasks, revealing critical insights into the distribution of optimized Gaussian parameters and their differing impacts on these tasks. To fully exploit the contribution of

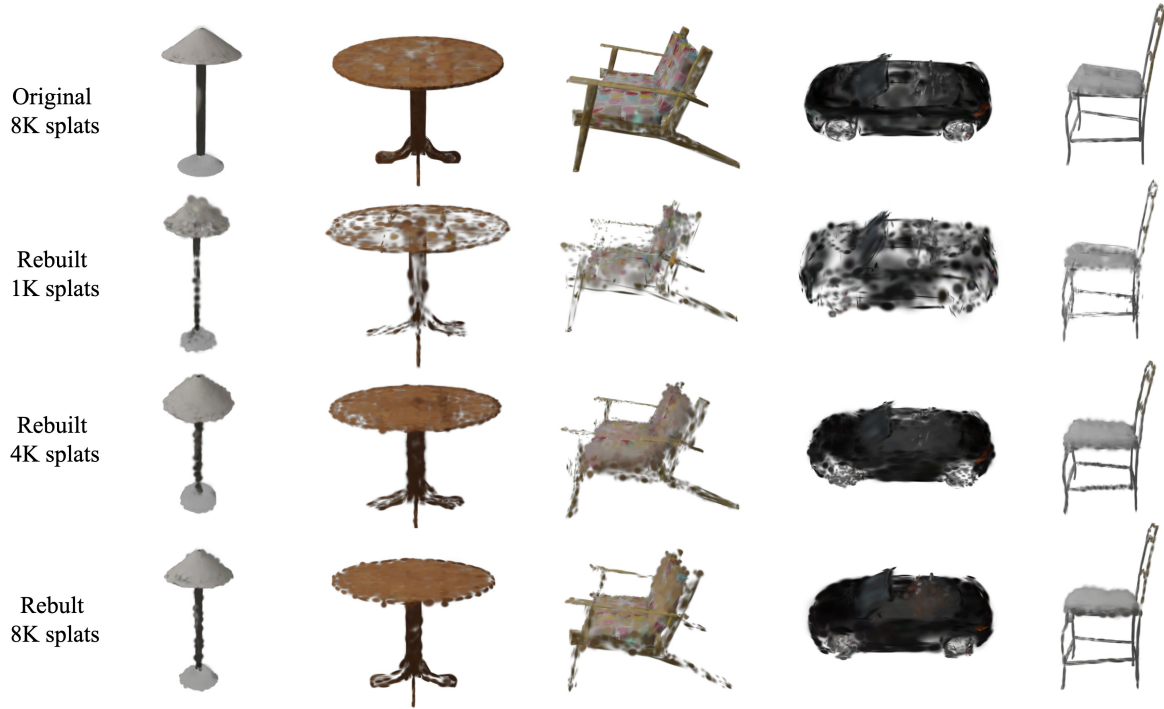


Figure D.2. **Qualitative Comparison of Reconstructed Gaussian Splats with Different Total Numbers.** We compare the reconstruction results with 1024, 4096, and 8192 Gaussian splats. As the number of splats increases, the reconstructions by Gaussian-MAE capture notably more color and geometric details, such as the colors of the bench and the frames of the chair, highlighting the importance of denser splat inputs as the finer color and geometry boost the downstream task performance.

gaussian parameters space, we introduce Gaussian feature grouping and splats pooling layers, which effectively embed similar Gaussians. By making our dataset and model publicly available, we aim to drive further research in 3D representation learning, enabling the community to explore and expand upon our work.

References

- [1] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. [1](#), [3](#)
- [2] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. [2](#)
- [3] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *European conference on computer vision*, pages 604–621. Springer, 2022. [2](#), [4](#)
- [4] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1588–1597, 2019. [3](#), [4](#)
- [5] Jarne Van den Herrewegen, T Tourwé, et al. Point cloud classification with modelnet40: What is left? In *DMLR, Data-centric Machine Learning Research Workshop at the 40 th International Conference on Machine Learning*, 2023. [2](#)
- [6] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. [1](#), [3](#)
- [7] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016. [3](#)