SVG-3D: Mining Decision Boundaries with Generative Splatting Priors for Zero-Shot 3D Classification

Ryo Umagami
The University of Tokyo[†]
umagami@g.ecc.u-tokyo.ac.jp

Shohei Ohsawa
Equity Consulting Group[†]
shoheiosawa@ecgjapan.com

Abstract

Zero-shot 3D point cloud classifiers built on large-scale pretrained models (OpenShape, Uni3D, ReCon++, DuoduoCLIP) excel on clean synthetic benchmarks such as ModelNet40 but degrade sharply on real-world scans (ScanObjectNN, ScanNet) and corrupted point clouds (ModelNet-C)—a persistent *real-world generalization gap*. We propose **SVG-3D**, the first 3D extension of Support Vector Generation (SVG), to close this gap without retraining a more capable encoder or accessing target-domain test data. SVG-3D mines a discriminative kernel-machine support set from a frozen encoder paired with DiffSplat, a text-conditioned 3D Gaussian splatting generator. SVG-3D keeps the acceptance rule of SVG and changes two things. (i) A dynamic pair-selection rule scores class pairs on generated samples only, never on test data, and mines only a fixed number of focal pairs, so the number of MCMC chains does not grow with the number of classes. (ii) The Metropolis–Hastings sampler moves from sentences to a text-embedding space, where a Slerp+Gaussian proposal has a closed-form approximate proposal ratio. On ScanObjectNN (OBJ_ONLY/OBJ_BG/PB_T50_RS) and ScanNet, SVG-3D surpasses prior zero-shot 3D classifiers built on large-scale pretrained models. Ablations attribute most of the gain to the pair-selection rule, and the mining gain persists when DiffSplat is replaced with Point-E. In a complementary corruption-aware setting on ModelNet-C, where the corruption family is assumed known, SVG-3D also improves mean accuracy across corruptions.

1 Introduction

Large-scale pretrained 3D-text models—OpenShape [22], Uni3D [50], ULIP-2 [43], ReCon++ [28], and DuoduoCLIP [19]—align 3D features with the joint embedding space of CLIP [29] and achieve strong zero-shot accuracy on clean synthetic benchmarks such as ModelNet40 [40]. On real-world scans (ScanObjectNN [38], ScanNet [7]) and corrupted point clouds (ModelNet-C [30]), however, the same encoders degrade sharply. This persistent *real-world generalization gap* reflects the domain mismatch between the synthetic 3D shapes that dominate pretraining and the partial, noisy observations produced by real sensors and downstream pipelines.

Common approaches to closing this gap fall into data augmentation—which mixes or perturbs synthetic shapes [4, 17, 18, 48] but offers no principled handle on which deformations matter for the decision boundary—and domain adaptation, including test-time adaptation [14, 34], which require target-domain data and are therefore ruled out in truly zero-shot settings. We pursue a complementary axis: we propose **SVG-3D**, which freezes the encoder and replaces nearest-text classification with a kernel machine whose support set is mined from a generative prior. SVG-3D therefore sits as an

[†]Work conducted at I.Y.P Consulting.

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

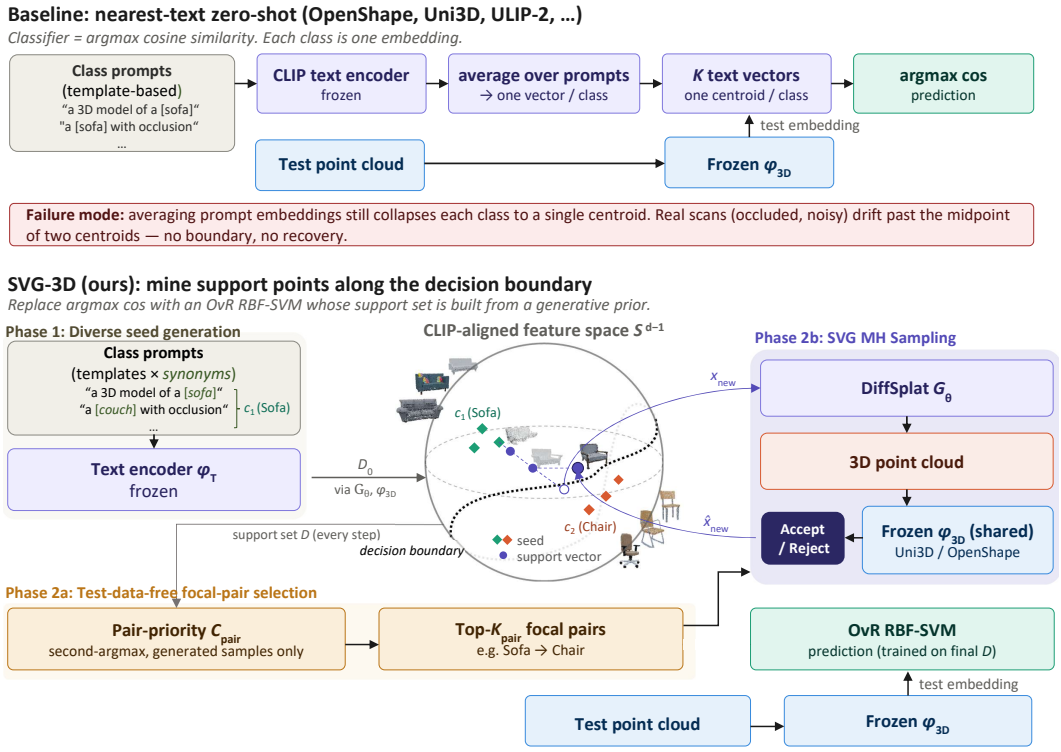

Figure 1: **SVG-3D vs. nearest-text zero-shot.** Standard zero-shot 3D classifiers represent each class by one prompt-averaged centroid in CLIP space (top). SVG-3D keeps the encoder frozen and instead mines a kernel SVM support set along the decision boundary (bottom). A Metropolis–Hastings (MH) sampler walks the text-embedding space, using DiffSplat as a deterministic decoder. A test-data-free pair-priority score selects the focal pairs to mine.

additive layer on a CLIP-aligned 3D encoder and can be applied to newer encoders without retraining them.

Concretely, we extend Support Vector Generation (SVG [24]), a recently introduced kernel-machine framework for zero-shot text classification, to the 3D setting (Fig. 1). SVG converts a frozen language model into a kernel SVM by sampling support sentences with a Metropolis–Hastings (MH) sampler whose acceptance ratio combines an SVM dual-coefficient term with a proposal-density ratio. We replace the language model with DiffSplat [21], a text-conditioned 3D Gaussian Splatting generator, and keep SVG's MH target and acceptance rule unchanged. We also change the MH state from a sentence to a text embedding. For multi-class real-world 3D classification we add prompt-template and class-synonym seed initialisation, one-vs-rest (OvR) SVM training, and a test-data-free rule that decides which class pairs to mine.

Contributions.

- We propose **SVG-3D**, the first 3D extension of Support Vector Generation, for zero-shot 3D point-cloud classification. SVG-3D couples a frozen 3D pretrained encoder, a text-conditioned 3DGS generator, and an SVG-style MH sampler into a single kernel classifier. Its support set is mined from this generative prior without retraining either component.
- We introduce a dynamic, test-data-free rule for choosing which pairwise class boundaries to mine. It computes a priority score for each class pair from generated samples and runs MCMC on the top-K_{pair} pairs. Fixing K_{pair} keeps the number of chains independent of the number of classes.
- Under strict zero-shot evaluation on ScanObjectNN (OBJ_ONLY/OBJ_BG/PB_T50_RS) and ScanNet, SVG-3D outperforms prior zero-shot 3D classifiers built on large-scale pretrained models.

2 Related Work

Zero-shot 3D classification. Two complementary lines dominate. *Multi-view projection* methods (PointCLIP [49], PointCLIP V2 [51], CLIP2Point [13]) render point clouds and reuse a frozen 2D CLIP model. *Native 3D encoders* aligned with CLIP (OpenShape [22], Uni3D [50], ULIP/ULIP-2 [42, 43], ReCon++ [28], CLIP^2 [47], DuoduoCLIP [19], PointLLM [41]) encode point clouds directly and classify by nearest text, with self-supervised 3D pretraining [25] as a common building block. We treat any such encoder as an interchangeable, frozen feature extractor and concentrate our contribution on the classifier built on top.

Robust and real-world 3D recognition. ModelNet-C [30] and ScanObjectNN [38] expose the fragility of point-cloud classifiers under corruption and real-world conditions, respectively. Prior robustness strategies span augmentation-based training [4, 17, 18, 48], set-input and transformer architectures [27, 39, 45], self-supervised pre-training [3, 25], and robustness redesigns [30]. A parallel line performs test-time adaptation by updating parameters (PCoTTA [14]) or denoising inputs (CloudFixer [34]) from unlabelled test data. None of these exploits a generative prior to expand the support set near class boundaries, and—unlike TTA—SVG-3D never consults test inputs, even in unlabelled form.

Generative priors as discriminative samplers. Native 3D generators built on diffusion [12] now span point clouds [46], meshes [10], and Gaussian splats [21, 44] (the last building on 3DGS [16]); 2D-distillation methods [26, 33] instead lift 2D priors into 3D. In 2D, text-to-image diffusion has been used as a synthetic-training source for classification [2, 11, 32, 37], but typically in a "generate-and-train" manner that samples broadly rather than near the decision surface. A complementary line uses gradients of an external discriminator to steer diffusion sampling itself (classifier guidance [8]), and a closely related thread reuses the diffusion model's own likelihood for zero-shot classification [5, 20]. SVG-3D differs from both: it neither modifies the sampler nor evaluates the generator's likelihood, and instead targets the *support* structure of an external discriminator with a closed-form approximate proposal ratio, which keeps the acceptance ratio tractable even for diffusion-based generators (Sec. 3.2). We do not compare against these lines empirically. Classifier guidance outputs samples, not a classifier. Diffusion-likelihood classification needs a generator likelihood for every test shape and class, which is exactly the quantity our closed-form proposal avoids. A controlled comparison with such classifiers is left to future work.

Kernel machines, active learning, and the missing pool. Support Vector Generation (SVG [24]) recasts a frozen language model as a kernel-induced classifier: an MH sampler queries the model to synthesise sentences near the decision surface, an SVM is solved on the resulting support set, and the dual coefficients provide both the classifier and a faithful rationale at inference. SVG proposes new sentences with an LLM completion API and uses the dual coefficient as the MH target. We inherit this target and the resulting acceptance rule, but move the MCMC state to the generator's text-embedding space. Mining samples near the decision boundary is related to active learning [1, 9] and hard-negative mining [35]. Like these approaches, SVG-3D directs computation toward difficult decision regions. SVG-3D instead draws candidate samples from a generative prior rather than selecting them from an existing dataset. Its pair-priority score, computed only from generated samples, determines which pairwise class boundaries receive the MCMC budget. For the multi-class kernel SVM we adopt one-vs-rest [31] rather than the joint Crammer–Singer formulation [6], so per-step retraining decomposes across classes.

What is new in SVG-3D. Tab. 1 separates what SVG-3D inherits from SVG [24] from what is specific to this work. The MH target and acceptance rule are inherited without change, so we claim no novelty there. The main methodological contribution is the test-data-free focal-pair selection rule (Sec. 3.4). SVG has a single binary boundary and needs no such rule; in the multi-class setting it decides where the sampler spends its budget. Sec. 4.3 shows that removing it costs more accuracy than removing the inherited MH filter. The other specific components carry SVG to 3D: a text-embedding state with a closed-form Slerp+Gaussian proposal, a 3DGS text-to-3D generator used as a deterministic decoder (Sec. 3.2), an RBF kernel on CLIP-aligned 3D features, OvR training, and a template$\times$synonym seed grid.

Table 1: Technical delta between SVG [24] and SVG-3D (this work), grouped into inherited and SVG-3D-specific components. The **bold** row is the main methodological contribution.

Component	SVG [24]	SVG-3D (this work)
Inherited		
MH target and acceptance rule	$\pi \propto \alpha$; Eq. 6 of [24]	Same (Eq. (7))
Specific to SVG-3D		
Pair selection	None (single binary boundary)	**Test-data-free focal-pair selection with dynamic refresh**
State space	Natural language sentences ($\mathcal{X}$)	Text-embedding space
Proposal	LLM completion (`text-curie-001`)	Slerp+Gaussian (closed-form approx. ratio)
Generator	The proposing LLM	3DGS text-to-3D decoder (DiffSplat)
Kernel	`text-embedding-ada-002` (RBF)	RBF on CLIP-aligned 3D features
Multi-class scheme	OvR / OvO (proposed, not evaluated)	OvR, retrained after every step
Seed construction	Single task-description prompt	Template $\times$ synonym grid

3 Method

At each step, SVG-3D walks the text-embedding space toward a confusable class, decodes the proposed direction into a 3D shape via DiffSplat [21], and lets the SVM dual coefficient decide whether to keep it. We make this concrete in two phases: *Phase 1* builds an initial support set from prompt-template and class-synonym combinations, and *Phase 2* repeatedly (i) computes a test-data-free pair-priority score, (ii) selects its top-K_{pair} focal pairs, and (iii) runs an SVG-style MH chain per pair, retraining the SVM at every step (Fig. 1).

3.1 Preliminaries: 3D kernel machine on a frozen encoder

Let $\phi_{3D} : \mathcal{P} \to \mathbb{S}^{d-1}$ be a frozen 3D encoder (Uni3D [50] or OpenShape [22]) mapping a point cloud to a unit vector on the CLIP hypersphere $\mathbb{S}^{d-1} \subset \mathbb{R}^d$, and let ϕ_{T} denote the generator's text encoder. The symmetric kernel

$$k(u, u') = \exp\big(-\gamma \, \|u - u'\|^2\big) \tag{1}$$

operates on the CLIP-aligned ϕ_{3D} features. For K-way classification we train one binary OvR SVM per class on a support set $\mathcal{D} = \{(u_i, y_i)\}_{i=1}^n$, $y_i \in \{\pm 1\}$, with decision function $f_\alpha(u) = \sum_i \alpha_i \, k(u, u_i) \, y_i$ and dual

$$\max_{\alpha} \sum_i \alpha_i - \tfrac{1}{2} \sum_{i,j} \alpha_i \alpha_j \, k(u_i, u_j) \, y_i y_j \;\; \text{s.t.} \;\; \sum_i \alpha_i y_i = 0, \;\; 0 \le \alpha_i \le C. \tag{2}$$

The dual coefficient α_i measures how strongly u_i shapes the boundary; SVG-3D, like SVG, uses it as the sampler's target signal.

3.2 Why a 3D Gaussian splat generator?

The role of G_θ is to map proposed text embeddings to point clouds for re-encoding by ϕ_{3D} and insertion into the SVM dual. G_θ appears only as a deterministic decoder: the MH proposal lives in the text-embedding space with a closed-form approximate density (Sec. 3.5), so the intractable likelihood of the underlying diffusion process never enters the acceptance ratio, freeing the choice of G_θ to one of generation quality and speed.

Why splats over point clouds, meshes, or score-distillation pipelines. Three properties make 3DGS-based generators a natural fit for repeated MH proposals across the few tens of rounds the chain requires. *(i) Density-controllable point read-out:* a splat scene exposes Gaussians directly, so a fixed-cardinality point cloud is obtained in a single visibility-filtered, area-weighted pass—no marching cubes, volumetric rendering, or per-shape mesh extraction. *(ii) Single-pass throughput:* DiffSplat [21] decodes a splat scene in one forward pass, whereas score-distillation pipelines (DreamFusion [26],

MVDream [33]) require per-prompt optimization that would make even a few-dozen-step chain across multiple focal pairs prohibitively slow. *(iii) Graceful off-manifold behaviour:* unlike implicit fields, splat scenes degrade smoothly—partial occlusions and asymmetric geometry remain valid Gaussian sets, exactly the regime the MH chain proposes between class centroids. Native point-cloud and implicit-field text-to-3D generators (LION [46], Point-E [23], Shap-E [15]) are valid alternatives for G_θ, and several are open-vocabulary. We default to a 3DGS generator because the text-conditioned splat literature is currently the most actively developing line. The acceptance rule is generator-independent because G_θ does not enter Eq. (7). Attainable accuracy, however, depends strongly on generator quality; Sec. 4.3 tests this by replacing DiffSplat with Point-E.

3.3 Phase 1: diverse seed generation

We use a dataset-independent prompt-template set $\mathcal{T}$ ("a 3D model of a $\langle c \rangle$", "a partial 3D scan of a $\langle c \rangle$", "a $\langle c \rangle$ with occlusion", and a few variants), combined with a per-class synonym set $\mathcal{S}_c$ (e.g. `{table, dining table, wooden table}` for "table"; full lists in Appendix B). $\mathcal{T}$ is fixed across datasets and not tuned. For each $(c, t, s) \in \{c\} \times \mathcal{T} \times \mathcal{S}_c$ we draw $n_{\text{seed}}/(|\mathcal{T}|\,|\mathcal{S}_c|)$ samples from $G_\theta(\phi_{\text{T}}(t(s)))$, extract a 10K-point cloud, and encode it with ϕ_{3D}. The resulting embeddings form the initial support set $\mathcal{D}_0$ (cached for ablations).

3.4 Phase 2a: test-data-free focal-pair selection

A naive multi-class extension of SVG would launch one MH chain per ordered class pair, scaling as $\mathcal{O}(K^2)$. Most pairs are easy, so this spends the budget on boundaries that need no new supports. We therefore restrict MCMC to a dynamically refreshed set of K_{pair} *focal pairs*. They are ranked by a pair-priority matrix C_{pair} computed only from generated and previously mined supports, never from the test set.

After every SVM retraining, we reset C_{pair} to zero, score each $u_i \in \mathcal{D}$ with the per-class decision functions, obtain its top-1/top-2 predictions $(\hat{y}_i^1, \hat{y}_i^2)$, and accumulate directed counts

$$C_{\text{pair}}[s,t] \mathrel{+}= \begin{cases} \mathbf{1}[\hat{y}_i^2 = t] & \text{if } \hat{y}_i^1 = s \text{ matches the SVG-3D label of } u_i, \\ \mathbf{1}[\hat{y}_i^1 = t] & \text{otherwise.} \end{cases} \tag{3}$$

Here the *SVG-3D label* of u_i is the class assigned at synthesis time (seed class for Phase 1, source class for Phase 2); no test-set ground truth is involved. We call this the *second-argmax* rule. When the top-1 prediction matches the SVG-3D label, the runner-up is counted as the closest competitor; otherwise the wrong top-1 class is counted. C_{pair} therefore resembles a confusion matrix, but it counts a different event. The SVM classifies almost all of its own supports correctly, so nearly all of the mass comes from the runner-up branch (Sec. 4.3). C_{pair} thus accumulates on pairs whose margins stay small, which is the regime an SVM-driven sampler should target. We use it as an acquisition score for support mining, not as an estimate of test-domain confusion. We pick the K_{pair} largest entries of C_{pair} as focal pairs. No test data enters this choice, K_{pair} stays independent of K, and the focal set follows the boundary as it moves.

3.5 Phase 2b: SVG MH sampler in text-embedding space

For each focal pair $(s \to t)$ we run an MH chain identical to SVG (Eq. 6 of [24]); only the state space, the proposal, and the deterministic decoder G_θ differ.

State and proposal. The MH state x_t is a text embedding, decoded once as $\hat{x}_t = \phi_{3D}(G_\theta(x_t))$ before insertion into the SVM dual. Given the source state x_t and target embedding e_{target} (cycled through $\mathcal{S}_t$, Sec. 3.3), we propose

$$\mu_{\text{fwd}} = \text{Slerp}(x_t,\, e_{\text{target}},\, \tau), \tag{4}$$

$$x_{\text{new}} = \|\mu_{\text{fwd}}\|\,(\mu_{\text{fwd}} + \epsilon)/\|\mu_{\text{fwd}} + \epsilon\|, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2 I), \tag{5}$$

$$\mu_{\text{bwd}} = \text{Slerp}(x_{\text{new}},\, e_{\text{target}},\, \tau), \tag{6}$$

which we call the *Slerp+Gaussian proposal*: a Slerp drift towards e_{target} followed by isotropic Gaussian noise and rescaling to the norm of μ_{fwd}. Ignoring this rescaling, we approximate the

Algorithm 1 SVG-3D: kernel machine via boundary mining of a 3D generative prior

Require: Frozen encoder ϕ_{3D}, text encoder ϕ_{T}, generator G_θ, classes $\{1,\dots,K\}$ with prompt templates $\mathcal{T}$ and synonyms $\{\mathcal{S}_c\}$, kernel k, C, focal-pair count K_{pair}, MCMC steps N, step size τ, noise σ

▷ *Phase 1: diversified seed generation* (Sec. 3.3)
$\mathcal{D}_0 \leftarrow \emptyset$
for $(c,t,s) \in \{1,\dots,K\} \times \mathcal{T} \times \mathcal{S}_c$**:** append $(\phi_{3D}(G_\theta(\phi_{\mathrm{T}}(t(s)))), c)$ to $\mathcal{D}_0$ $[n_{\mathrm{seed}}/(|\mathcal{T}||\mathcal{S}_c|)$ times]
Train OvR SVMs $\{f_\alpha^{(c)}\}_{c=1}^K$ on $\mathcal{D}_0$; $\mathcal{D} \leftarrow \mathcal{D}_0$
▷ *Phase 2: SVG MH boundary mining* (Sec. 3.4, Sec. 3.5)
for step $n = 1,\dots,N$ **do**
  Compute pair priority C_{pair} on $\mathcal{D}$ via Eq. (3); pick top-K_{pair} focal pairs $\{(s_j \to t_j)\}_{j=1}^{K_{\mathrm{pair}}}$
  for focal pair (s,t) **in parallel do**
    Pick target embedding e_{target} by cycling through $\mathcal{S}_t$
    $\mu_{\mathrm{fwd}} \leftarrow \mathrm{Slerp}(x_t, e_{\mathrm{target}}, \tau)$; $x_{\mathrm{new}} \leftarrow \|\mu_{\mathrm{fwd}}\|\,(\mu_{\mathrm{fwd}} + \epsilon)/\|\mu_{\mathrm{fwd}} + \epsilon\|$
    $P_{\mathrm{new}} \leftarrow G_\theta(x_{\mathrm{new}})$; $\hat{x}_{\mathrm{new}} \leftarrow \phi_{3D}(P_{\mathrm{new}})$
    $\alpha' \leftarrow$ `SVM_train_OvR`$(\mathcal{D}_0 \cup \mathcal{D}_{s\to t} \cup \{(\hat{x}_{\mathrm{new}},\ s)\})$ to obtain α_{new}
    Accept x_{new} with probability A from Eq. (7); if accepted, append to $\mathcal{D}$
  end for
  Retrain $\{f_\alpha^{(c)}\}$ on $\mathcal{D}$
end for
return OvR classifier $\{f_\alpha^{(c)}\}_{c=1}^K$ (predict via highest score)

forward/backward densities by Gaussians, $q(x_t \to x_{\mathrm{new}}) \propto \exp(-\|x_{\mathrm{new}} - \mu_{\mathrm{fwd}}\|^2/2\sigma^2)$ and $q(x_{\mathrm{new}} \to x_t) \propto \exp(-\|x_t - \mu_{\mathrm{bwd}}\|^2/2\sigma^2)$ (Fig. 4 visualises decoded supports along such chains).

The acceptance rule. Setting the target $\pi(x) \propto \alpha(x)$ (the SVM dual coefficient x acquires when added to $\mathcal{D}$, as in SVG [24]) and combining with the proposal-density ratio yields the heart of SVG-3D:

$$\boxed{A(x_t, x_{\mathrm{new}}) = \min\left(1,\ \frac{\alpha_{\mathrm{new}}}{\alpha_t}\ \exp\left(\frac{\|x_{\mathrm{new}} - \mu_{\mathrm{fwd}}\|^2 - \|x_t - \mu_{\mathrm{bwd}}\|^2}{2\sigma^2}\right)\right).} \tag{7}$$

G_θ never enters Eq. (7), so any text-to-3D model can be plugged in without changing the acceptance rule. Within a round, each chain $(s \to t)$ computes α_{new} with the source-class SVM trained on the seeds $\mathcal{D}_0$ and its own accepted supports $\mathcal{D}_{s\to t}$; all SVMs are retrained on the full $\mathcal{D}$ at the end of the round. We retrain the OvR SVM after every step so the pair-priority matrix of Sec. 3.4 reflects the current boundary; the SVM solve is millisecond-scale thanks to small support sets [24].

3.6 Algorithm summary and fixed configuration

Algorithm 1 ties the components together; the SVG-3D–specific additions over the SVG template (Sec. 4 of [24]) are the diversified seed construction of Phase 1 (Sec. 3.3) and the test-data-free focal-pair selection that opens each MCMC iteration (Sec. 3.4), while the inner MH loop keeps SVG's acceptance rule with the Slerp+Gaussian proposal of Sec. 3.5.

4 Experiments

We evaluate SVG-3D in two settings. The main setting is *strict zero-shot* on ScanObjectNN [38] (OBJ_ONLY/OBJ_BG/PB_T50_RS) and ScanNet [7, 50] (Secs. 4.1 and 4.2), where neither the test set nor any target-domain corpus is consulted. Appendix E adds an auxiliary *corruption-aware* study on ModelNet-C [30], where the corruption *family*—not test instances—is assumed known and injected as an augmentation operator. Except for the reference row (e) of Tab. 4, which uses test labels by design, no experiment selects SVM hyperparameters or MCMC focal pairs on the test set.

Setup. We use Uni3D-giant [50] as the point cloud encoder. DiffSplat [21] decodes text embeddings to 3DGS, with 10K points extracted via visibility-filtered, area-weighted upsampling. All main SVG-3D results report mean±std over three independent runs. Ablation rows (a)–(f) of Tab. 4 are single runs. The nearest-text baselines are deterministic given the encoder.

Table 2: Zero-shot classification accuracy (%) on the three ScanObjectNN variants. OBJ_ONLY results are quoted from the cited papers. OBJ_BG and PB_T50_RS are not reported by prior work; we re-evaluate OpenShape and Uni3D on these splits. SVG-3D reports mean ± standard deviation over three runs. Best in column in **bold**.

	OBJ_ONLY			OBJ_BG	PB_T50_RS
Method	Top1	Top3	Top5	Top1	Top1
ULIP [42]	51.6	72.5	82.3	–	–
ReCon++ [28]	65.4	84.1	89.7	–	–
DuoduoCLIP [19]	66.3	85.5	90.2	–	–
OpenShape [22]	52.2	79.7	88.7	54.7	39.5
Uni3D [50]	65.3	85.5	92.7	66.9	49.8
SVG-3D (Ours)	**71.7** $\pm$**0.4**	**87.9** $\pm$**1.5**	**95.3** $\pm$**0.9**	**68.4** $\pm$**1.0**	**54.7** $\pm$**1.4**
without MCMC	65.9	80.7	86.1	65.6	50.1

Table 3: Zero-shot Top-1 classification accuracy (%) on ScanNet. All baseline rows are taken from Uni3D [50]. **Avg.** is the unweighted mean Top-1 across the 17 categories. SVG-3D is averaged over three runs; the standard deviation of Avg. over runs is 0.5. Column abbreviations: Cab=cabinet, Tabl=table, Wind=window, Bksf=bookshelf, Pic=picture, Cntr=counter, Curt=curtain, Fridg=refrigerator, Bath=bathtub, Showr=shower curtain, Toil=toilet. Best in column in **bold**.

Method	**Avg.**	Bed	Cab	Chair	Sofa	Tabl	Door	Wind	Bksf	Pic	Cntr	Desk	Curt	Fridg	Bath	Showr	Toil	Sink
PointCLIP [49]	6.3	0.0	0.0	0.0	0.0	0.7	0.0	0.0	91.8	0.0	0.0	0.0	15.0	0.0	0.0	0.0	0.0	0.0
PointCLIP V2 [51]	11.0	0.0	0.0	23.8	0.0	0.0	0.0	0.0	7.8	0.0	**90.7**	0.0	0.0	0.0	0.0	64.4	0.0	0.0
CLIP2Point [13]	24.9	20.8	0.0	**85.1**	43.3	26.5	**69.9**	0.0	20.9	1.7	31.7	27.0	0.0	1.6	46.5	0.0	22.4	25.6
PointCLIP w/ TP. [49]	26.1	0.0	55.7	72.8	5.0	5.1	1.7	0.0	77.2	0.0	0.0	51.7	0.3	0.0	0.0	40.3	85.3	49.2
CLIP2Point w/ TP. [13]	35.2	11.8	3.0	45.1	27.6	10.5	61.5	2.6	71.9	0.3	33.6	29.9	4.7	11.5	**72.2**	**92.4**	86.1	34.0
CLIP^2 [47]	38.5	32.6	**67.2**	69.3	42.3	18.3	19.1	4.0	62.6	1.4	12.7	52.8	40.1	9.1	59.7	41.0	71.0	45.5
Uni3D [50]	45.8	**58.5**	3.7	78.8	83.7	54.9	31.3	**39.4**	70.1	35.1	1.9	27.3	**94.2**	**13.8**	38.7	10.7	88.1	47.6
SVG-3D (Ours)	**48.4**	53.8	7.2	70.8	**85.0**	**58.6**	32.4	35.8	89.5	51.9	43.9	**60.3**	40.1	0.8	0.0	43.3	**89.5**	**59.0**
without MCMC	45.1	47.4	3.7	73.0	75.6	49.8	30.2	22.4	**93.0**	**80.5**	9.8	55.7	40.7	2.3	0.0	40.0	86.8	55.4

Hyperparameter selection. SVM and MCMC hyperparameters are selected per dataset on a synthetic DiffSplat-generated validation set disjoint from the seeds, scored by OvR Top-1; no test point or label is consulted at any stage. Per-dataset values are listed in Appendix C.

Baselines. For ScanObjectNN OBJ_ONLY, where prior work reports zero-shot numbers, we quote published results of ULIP [42], ReCon++ [28], DuoduoCLIP [19], OpenShape [22], and Uni3D [50]. For OBJ_BG and PB_T50_RS, which no prior work reports, we re-evaluate OpenShape and Uni3D ourselves under the same protocol. Throughout, nearest-text classification serves as the primary baseline.

4.1 Real-world recognition on ScanObjectNN

ScanObjectNN provides the cleanest test of real-world generalization: object scans derived from real RGB-D capture exhibit partial observation, sensor noise, and pose perturbation absent from synthetic pretraining shapes. Tab. 2 reports zero-shot accuracy on the three variants of increasing difficulty (OBJ_ONLY $\rightarrow$ OBJ_BG $\rightarrow$ PB_T50_RS). SVG-3D sets a new state of the art on OBJ_ONLY across all three Top-k columns, reaching 71.7 ± 0.4 Top-1. This is $+6.4$ pt over the Uni3D baseline with the same encoder (65.3) and $+5.4$ pt over the strongest published result we are aware of (DuoduoCLIP, 66.3); run-to-run variance is an order of magnitude smaller than this gain. The same configuration also improves OBJ_BG (68.4 ± 1.0, $+1.5$ pt) and PB_T50_RS (54.7 ± 1.4, $+4.9$ pt) over the Uni3D baseline. On both splits the gain is positive in every run and exceeds the run-to-run standard deviation. The seed-only ablation ("without MCMC") stays close to the Uni3D baseline on all three splits. The gain therefore comes from the MCMC-mined supports rather than from the generated seeds alone.

Table 4: Exploration ablations on ScanObjectNN OBJ_ONLY. All rows use the Uni3D backbone unless otherwise noted (row (f) uses OpenShape). GT-confusion pair selection (row (e)) refreshes its focal pairs after every SVM retraining from the ground-truth confusion matrix evaluated on the test set, so it explicitly violates the zero-shot protocol. It is a reference, not an upper bound. Best in column in **bold**.

#	Configuration	Top-1 (%)	Top-3 (%)	Top-5 (%)
(a)	Seed-only: Phase 1 alone, no MCMC mining	65.9	80.7	86.1
(b)	Always accept: skip the MH acceptance test, keep every proposal	69.5	84.7	91.4
(c)	Random focal pairs: replace second-argmax with uniform pair sampling	66.1	80.9	87.6
(d)	Argmax-only pairs: count only top-1 misclassifications, drop runner-ups	68.2	88.1	94.7
(e)	GT-confusion pairs (*ref.*; test labels; not an upper bound)	71.1	**89.3**	95.0
(f)	Encoder swap: full pipeline on the OpenShape backbone	62.0	77.6	83.3
(g)	SVG-3D full (second-argmax pair selection)	**71.7**	87.9	**95.3**

4.2 Generalization to scene-level zero-shot: ScanNet

ScanNet evaluates zero-shot recognition over object instances extracted from real indoor scans. SVG-3D lifts the unweighted mean Top-1 across the 17 categories from 45.8 (Uni3D) to 48.4 ± 0.5 (Tab. 3). Instance-weighted accuracy also rises, from 50.7 to 51.5. The seed-only ablation reaches 45.1, below the nearest-text baseline, so the whole gain comes from MCMC ($+3.3$ over seed-only). The largest gains occur for Counter ($+42.0$), Desk ($+33.0$), and Shower curtain ($+32.6$), all classes on which nearest-text is weak. Curtain (-54.1), Bathtub (-38.7), and Refrigerator (-13.0) regress.

Per-class regressions. Every regressing class has already dropped at the seed-only stage. MCMC leaves Bathtub ($0.0 \rightarrow 0.0$) and Curtain ($40.7 \rightarrow 40.1$) flat and partly recovers Bed ($47.4 \rightarrow 53.8$) and Window ($22.4 \rightarrow 35.8$). Bathtub fails before any mining. Its errors go to Sofa and Counter rather than to its focal-pair partner, and we cannot yet tell whether the generated bathtubs are poor or the encoder places real bathtub scans far from them. Mining can thus raise the mean while individual classes regress.

4.3 Ablations

Inherited vs. new components. Tab. 4 can be read as a leave-one-out over the full pipeline (g, 71.7). Removing all of Phase 2 (a) costs 5.8 points. Removing the inherited MH filter (b) costs 2.2 points. Removing the new pair-selection rule (c) costs 5.6 points; with random focal pairs, the acceptance rule alone stays only 0.2 above seed-only. Both components are needed, but transplanting SVG's acceptance rule to 3D does not by itself produce the gain. Row (d) further shows that the runner-up branch of Eq. (3) matters: counting only top-1 errors reaches 68.2.

The pair-priority matrix is not a test-confusion estimate. Over three runs, the Spearman correlation between C_{pair} and the true test confusion matrix is weak over all 210 off-diagonal entries ($\rho = 0.314 \pm 0.075$) and vanishes among the 15 pairs actually mined ($\rho = 0.043 \pm 0.273$). The two matrices count different events. The true confusion matrix counts only test errors, whereas $99.8 \pm 0.1\%$ of the mass of C_{pair} comes from the runner-up branch, i.e. from correctly classified supports. The rule should therefore be judged by its output, the set of mined pairs. Random pairs reach 66.1, while pairs chosen from the true test confusion (e, using test labels) reach 71.1, slightly below (g). Privileged access to test labels thus gives no measurable advantage, and row (e) is a reference rather than an upper bound.

Encoder and generator replacement. Row (f) lifts OpenShape OBJ_ONLY Top-1 from 52.2 (Tab. 2) to 62.0 ($+9.8$ pt), so the gain transfers across encoders. To test the generator, we replace DiffSplat with Point-E [23] (`base40M-textvec` plus upsampler; 4,096 points), keep everything else fixed, and run OBJ_ONLY three times. Mining raises Point-E from 26.2 (seed-only) to 30.6 ± 0.9 ($+4.5$), compared with $65.9 \rightarrow 71.7 \pm 0.4$ ($+5.8$) for DiffSplat. Point-E is far below the nearest-text baseline, so this is a mechanism check rather than a competitive configuration. The mining mechanism transfers to a different generator architecture, but attainable accuracy tracks generator quality. We have tested only these two generators.

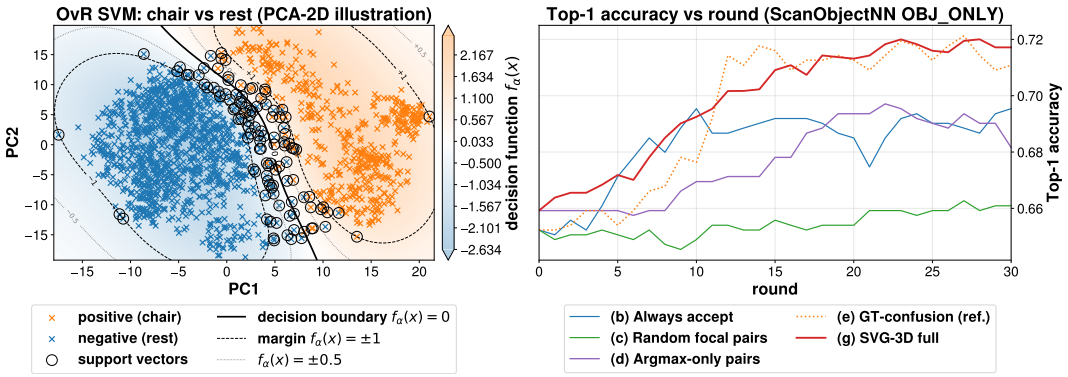

Figure 2: **Left:** 2D PCA illustration of the OvR SVM decision boundary for the chair-vs-rest classifier, fit on PCA-projected features (positives: chair seeds and mined supports; negatives: 50 samples per other class). Solid line: $f_\alpha(x)$=0; dashed: $f_\alpha(x)$=$\pm$1; dotted: $f_\alpha(x)$=$\pm$0.5. The 2D SVM is re-fit on the PCA projection so that the contour visualizes rather than reproduces the 1024-d boundary. **Right:** Top-1 accuracy on ScanObjectNN OBJ_ONLY across rounds for the rows of Tab. 4: (b) always accept, (c) random focal pairs, (d) argmax-only pairs, (e) GT-confusion pairs (*ref.*), and (g) SVG-3D full (mean over 3 runs).

Seeds and round-wise behaviour. Prompt diversification and synonym expansion both matter for Phase 1 (Tab. 5). Across rounds (Fig. 2, right), random pairs (c) barely leave the seed-only level. Always accept (b) rises early but plateaus once non-margin samples dilute the SVM dual, and argmax-only pairs (d) plateau in between. The full run (g) tracks the GT-confusion reference (e) throughout.

5 Analysis

Why MH-mined boundaries help: a kernel-machine perspective. The gain over nearest-text is structural. A nearest-text classifier collapses each class to one prompt-averaged direction in $\mathbb{S}^{d-1}$ and flips under domain shift (Fig. 1, top), whereas targeting the SVM dual coefficient $\alpha(x)$ routes MCMC budget to non-trivial support vectors: α is non-zero only for points within the margin or on the wrong side of the boundary, so an MH chain with target $\pi(x) \propto \alpha(x)$ samples the margin band by construction (Fig. 2, left, visualises this for chair-vs-rest). Boundary mining is thus a property of the *sampler-classifier coupling*, not specific to DiffSplat, consistent with the Point-E result of Sec. 4.3.

Limitations and future work. *Scope and cost.* SVG-3D requires a CLIP-aligned encoder, ruling out e.g. Point-MAE [25]. Phase 1 and the pair-priority matrix scale linearly in K, so SVG-3D suits dozens-of-class taxonomies. A run costs $\sim$10 A100-hours, 94% of it in Phase 1, which seed caching and sharing amortise (Appendix F). *Generator coverage.* On OBJ_ONLY, 3 of 15 classes have fewer than half of their seeds recognised as the intended class, although this does not predict per-class accuracy (Appendix D). Coverage will drop for categories outside the generator's training data, and we tested only two generators. *Harmful supports.* Supports for out-of-distribution classes can be harmful (Bathtub on ScanNet), and a naive nearest-text filter lowers accuracy by 3.9 points (Appendix D). *Per-class trade-offs.* Mining can raise the mean while individual classes regress (Sec. 4.2). *Future work* includes comparisons with generative classifiers and newer text-to-3D generators, and extensions to open-vocabulary and dense prediction (e.g. OpenMask3D [36]).

6 Conclusion

We presented SVG-3D, the first 3D extension of Support Vector Generation: a frozen 3D pretrained encoder, a text-to-3D Gaussian splat generator, and an SVG-style Metropolis–Hastings sampler are coupled into a single zero-shot kernel classifier whose support set is mined from a generative prior. The acceptance rule is inherited from SVG; the new element is a test-data-free focal-pair selection rule that decides where the sampler spends its budget and keeps the number of chains fixed as the number of classes grows. Ablations attribute most of the gain to this rule. SVG-3D outperforms prior

zero-shot 3D classifiers on ScanObjectNN and ScanNet under strict zero-shot, and its mining gain persists under a different generator.

Acknowledgments and Disclosure of Funding

This research was funded by I.Y.P Consulting. Travel support was provided by Equity Consulting Group.

References

[1] Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*, 2019.

[2] Shekoofeh Azizi, Simon Kornblith, Chitwan Saharia, Mohammad Norouzi, and David J Fleet. Synthetic data from diffusion models improves imagenet classification. *arXiv preprint arXiv:2304.08466*, 2023.

[3] Guangyan Chen, Meiling Wang, Yi Yang, Kai Yu, Li Yuan, and Yufeng Yue. Pointgpt: Auto-regressively generative pre-training from point clouds. *Advances in Neural Information Processing Systems*, 36: 29667–29679, 2023.

[4] Yunlu Chen, Vincent Tao Hu, Efstratios Gavves, Thomas Mensink, Pascal Mettes, Pengwan Yang, and Cees GM Snoek. Pointmixup: Augmentation for point clouds. In *European conference on computer vision*, pages 330–345. Springer, 2020.

[5] Kevin Clark and Priyank Jaini. Text-to-image diffusion models are zero shot classifiers. *Advances in Neural Information Processing Systems*, 36:58921–58937, 2023.

[6] Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of machine learning research*, 2(Dec):265–292, 2001.

[7] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pages 2432–2443. IEEE, 2017.

[8] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[9] Melanie Ducoffe and Frederic Precioso. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841*, 2018.

[10] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *Advances in neural information processing systems*, 35:31841–31854, 2022.

[11] Ruifei He, Shuyang Sun, Xin Yu, Chuhui Xue, Wenqing Zhang, Philip Torr, Song Bai, and Xiaojuan Qi. Is synthetic data from generative models ready for image recognition? *arXiv preprint arXiv:2210.07574*, 2022.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[13] Tianyu Huang, Bowen Dong, Yunhan Yang, Xiaoshui Huang, Rynson WH Lau, Wanli Ouyang, and Wangmeng Zuo. Clip2point: Transfer clip to point cloud classification with image-depth pre-training. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22100–22110. IEEE, 2023.

[14] Jincen Jiang, Qianyu Zhou, Yuhang Li, Xinkui Zhao, Meili Wang, Lizhuang Ma, Jian Chang, Jian J Zhang, and Xuequan Lu. Pcotta: Continual test-time adaptation for multi-task point cloud understanding. *Advances in Neural Information Processing Systems*, 37:96229–96253, 2024.

[15] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023.

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

[17] Sihyeon Kim, Sanghyeok Lee, Dasol Hwang, Jaewon Lee, Seong Jae Hwang, and Hyunwoo J Kim. Point cloud augmentation with weighted local transformations. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 528–537. IEEE, 2021.

[18] Dogyoon Lee, Jaeha Lee, Junhyeop Lee, Hyeongmin Lee, Minhyeok Lee, Sungmin Woo, and Sangyoun Lee. Regularization strategy for point cloud via rigidly mixed sample. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15895–15904. IEEE, 2021.

[19] Han-Hung Lee, Yiming Zhang, and Angel Chang. Duoduo clip: Efficient 3d understanding with multi-view images. In *International Conference on Learning Representations*, volume 2025, pages 48070–48091, 2025.

[20] Alexander C Li, Mihir Prabhudesai, Shivam Duggal, Ellis Brown, and Deepak Pathak. Your diffusion model is secretly a zero-shot classifier. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2206–2217. IEEE, 2023.

[21] Chenguo Lin, Panwang Pan, Bangbang Yang, Zeming Li, and Yadong Mu. Diffsplat: Repurposing image diffusion models for scalable gaussian splat generation. In *International Conference on Learning Representations*, volume 2025, pages 40745–40767, 2025.

[22] Minghua Liu, Ruoxi Shi, Kaiming Kuang, Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su. Openshape: Scaling up 3d shape representation towards open-world understanding. *Advances in neural information processing systems*, 36:44860–44879, 2023.

[23] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022.

[24] Shohei Ohsawa. Support vector generation: Kernelizing large language models for efficient zero-shot nlp. *Advances in Neural Information Processing Systems*, 38:97993–98006, 2025.

[25] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *ECCV*, pages 604–621. Springer, 2022.

[26] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

[27] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[28] Zekun Qi, Runpei Dong, Shaochen Zhang, Haoran Geng, Chunrui Han, Zheng Ge, Li Yi, and Kaisheng Ma. Shapellm: Universal 3d object understanding for embodied interaction. In *European Conference on Computer Vision*, pages 214–238. Springer, 2024.

[29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 2021.

[30] Jiawei Ren, Liang Pan, and Ziwei Liu. Benchmarking and analyzing point cloud classification under corruptions. In *International conference on machine learning*, pages 18559–18575. PMLR, 2022.

[31] Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of machine learning research*, 5(Jan):101–141, 2004.

[32] Mert Bülent Sarıyıldız, Karteek Alahari, Diane Larlus, and Yannis Kalantidis. Fake it till you make it: Learning transferable representations from synthetic imagenet clones. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8011–8021, 2023.

[33] Yichun Shi, Peng Wang, Jianglong Ye, Long Mai, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. In *International conference on learning representations*, volume 2024, pages 39838–39859, 2024.

[34] Hajin Shim, Changhun Kim, and Eunho Yang. Cloudfixer: Test-time adaptation for 3d point clouds via diffusion-guided geometric transformation. In *European Conference on Computer Vision*, pages 454–471. Springer, 2024.

[35] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 761–769, 2016.

[36] Ayça Takmaz, Elisabetta Fedele, Robert W Sumner, Marc Pollefeys, Federico Tombari, and Francis Engelmann. Openmask3d: Open-vocabulary 3d instance segmentation. *arXiv preprint arXiv:2306.13631*, 2023.

[37] Yonglong Tian, Lijie Fan, Phillip Isola, Huiwen Chang, and Dilip Krishnan. Stablerep: Synthetic images from text-to-image models make strong visual representation learners. *Advances in Neural Information Processing Systems*, 36:48382–48402, 2023.

[38] 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 *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1588–1597. IEEE, 2019.

[39] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. *Advances in neural information processing systems*, 35: 33330–33342, 2022.

[40] 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.

[41] Runsen Xu, Xiaolong Wang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. Pointllm: Empowering large language models to understand point clouds. In *European Conference on Computer Vision*, pages 131–147. Springer, 2024.

[42] Le Xue, Mingfei Gao, Chen Xing, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. Ulip: Learning a unified representation of language, images, and point clouds for 3d understanding. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1179–1189. IEEE, 2023.

[43] Le Xue, Ning Yu, Shu Zhang, Artemis Panagopoulou, Junnan Li, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. Ulip-2: Towards scalable multimodal pre-training for 3d understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27091–27101, 2024.

[44] Taoran Yi, Jiemin Fang, Junjie Wang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. Gaussiandreamer: Fast generation from text to 3d gaussians by bridging 2d and 3d diffusion models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6796–6807. IEEE, 2024.

[45] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

[46] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. *arXiv preprint arXiv:2210.06978*, 2022.

[47] Yihan Zeng, Chenhan Jiang, Jiageng Mao, Jianhua Han, Chaoqiang Ye, Qingqiu Huang, Dit-Yan Yeung, Zhen Yang, Xiaodan Liang, and Hang Xu. CLIP^2: Contrastive language-image-point pretraining from real-world point cloud data. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15244–15253. IEEE, 2023.

[48] Jinlai Zhang, Lyujie Chen, Bo Ouyang, Binbin Liu, Jihong Zhu, Yujin Chen, Yanmei Meng, and Danfeng Wu. Pointcutmix: Regularization strategy for point cloud classification. *Neurocomputing*, 505:58–67, 2022.

[49] Renrui Zhang, Ziyu Guo, Wei Zhang, Kunchang Li, Xupeng Miao, Bin Cui, Yu Qiao, Peng Gao, and Hongsheng Li. Pointclip: Point cloud understanding by clip. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8542–8552. IEEE, 2022.

[50] Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Uni3d: Exploring unified 3d representation at scale. In *International Conference on Learning Representations*, volume 2024, pages 46766–46782, 2024.

[51] Xiangyang Zhu, Renrui Zhang, Bowei He, Ziyu Guo, Ziyao Zeng, Zipeng Qin, Shanghang Zhang, and Peng Gao. Pointclip v2: Prompting clip and gpt for powerful 3d open-world learning. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2639–2650. IEEE, 2023.

A Derivation of the Acceptance Probability

Starting from the standard Metropolis–Hastings acceptance ratio:

$$A(x_t, x_{\text{new}}) = \min\left(1,\ \frac{\pi(x_{\text{new}})\, q(x_{\text{new}} \to x_t)}{\pi(x_t)\, q(x_t \to x_{\text{new}})}\right). \tag{8}$$

Substituting $\pi(x) \propto \alpha(x)$ (the SVM dual coefficient acquired by x when it is added to the support set) and the Gaussian approximation of the proposal densities (ignoring the norm rescaling):

$$\frac{\pi(x_{\text{new}})}{\pi(x_t)} = \frac{\alpha_{\text{new}}}{\alpha_t}, \tag{9}$$

$$\frac{q(x_{\text{new}} \to x_t)}{q(x_t \to x_{\text{new}})} = \frac{\exp\!\big(-\|x_t - \mu_{\text{bwd}}\|^2/2\sigma^2\big)}{\exp\!\big(-\|x_{\text{new}} - \mu_{\text{fwd}}\|^2/2\sigma^2\big)} = \exp\!\left(\frac{\|x_{\text{new}} - \mu_{\text{fwd}}\|^2 - \|x_t - \mu_{\text{bwd}}\|^2}{2\sigma^2}\right). \tag{10}$$

Combining yields Eq. (7). The proposal correction term is non-trivial because the Slerp drift makes the proposal asymmetric ($\mu_{\text{fwd}} \neq \mu_{\text{bwd}}$ in general); the same algebra appears in SVG (Eq. 6 of [24]) and, under this approximation, ensures detailed balance with respect to the target $\pi(x) \propto \alpha(x)$.

B Prompt templates and class synonyms

This appendix lists every prompt template and per-class synonym used by SVG-3D (Sec. 3.3). Both lists were generated once with Claude Sonnet 4.6 and held fixed across runs. We supplied neither dataset names nor test data or labels, and did not select, reorder, or iterate on the outputs in response to evaluation results. The only manually specified prior was the three generic thematic groups below, shared by every dataset.

Prompt templates $\mathcal{T}$. The template set $\mathcal{T}$ is identical for every dataset and comprises nine entries, organised into three thematic groups (`{c}` denotes the class name). The complete instruction was: "for each of clean object, scan-style observation, and object condition, list three prompt templates for point-cloud generation."

- *Clean object:* "a typical `{c}`", "a 3D model of a `{c}`", "a clean 3D model of a `{c}`";
- *Scan-style observation:* "a partial 3D scan of a `{c}`", "a noisy 3D scan of a `{c}`", "a `{c}` with occlusion";
- *Object condition:* "a broken `{c}`", "a damaged `{c}`", "a `{c}` with missing parts".

Table 5: Prompt/synonym ablations on ScanObjectNN OBJ_ONLY (referenced from Sec. 4.3). Best in column in **bold**.

#	Configuration	Top-1 (%)	Top-3 (%)	Top-5 (%)
(a)	Single template + single synonym	53.9	67.8	75.7
(b)	Single template + multiple synonyms	55.9	77.6	84.7
(c)	Multiple templates + single synonym	62.1	81.9	89.5
(d)	Multiple templates + multiple synonyms (full)	**71.7**	**87.9**	**95.3**

Class synonyms $\mathcal{S}_c$. Each class is paired with three alternatives generated from the class name alone using the instruction "list three classes similar to this class." The class name was the only input: the benchmark name and all test information were withheld. This procedure injects the general vocabulary evoked by a class name, but no target-domain or evaluation-specific information. Tabs. 6 to 8 list the exact resulting sets for ScanObjectNN (15 classes), ScanNet (17 classes), and ModelNet40 (40 classes; reused as-is for ModelNet-C).

Table 6: Class synonym list $\mathcal{S}_c$ for ScanObjectNN (15 classes).

Class	Synonyms
bag	bag, handbag, backpack
bin	trash bin, wastebasket, recycling bin
box	cardboard box, storage box, small box
cabinet	cabinet, storage cabinet, filing cabinet
chair	chair, office chair, armchair
desk	desk, office desk, computer desk
display	computer monitor, flat screen monitor, TV screen
door	door, wooden door, room door
shelf	bookshelf, wall shelf, storage shelf
table	table, dining table, wooden table
bed	bed, single bed, double bed
pillow	pillow, cushion, bed pillow
sink	sink, bathroom sink, kitchen sink
sofa	sofa, couch, loveseat
toilet	toilet, toilet bowl, lavatory toilet

Table 7: Class synonym list $\mathcal{S}_c$ for ScanNet (17 classes).

Class	Synonyms
bed	bed, double bed, single bed
cabinet	cabinet, kitchen cabinet, filing cabinet
chair	chair, office chair, armchair
sofa	sofa, couch, loveseat
table	table, dining table, wooden table
door	door, wooden door, room door
window	window, glass window, room window
bookshelf	bookshelf, wall shelf, storage shelf
picture	picture, wall picture, painting
counter	counter, kitchen counter, bathroom counter
desk	desk, office desk, computer desk
curtain	curtain, window curtain, drape
refrigerator	refrigerator, fridge, kitchen refrigerator
bathtub	bathtub, freestanding bathtub, claw-foot bathtub
shower curtain	shower curtain, bathroom shower curtain, plastic shower curtain
toilet	toilet, toilet bowl, lavatory toilet
sink	sink, kitchen sink, bathroom sink

C Hyperparameters

This appendix lists the full SVG-3D configuration used in the experiments of Sec. 4. Tab. 9 collects components held constant across every dataset and backbone, while Tab. 10 reports the few values selected per dataset on the synthetic validation set described in Sec. 4.

D Generator Coverage and Harmful Supports

Measuring coverage. We measure how well the generator covers each class with a test-free *fidelity* score. Fidelity is the fraction of a class's 500 Phase-1 seeds that the nearest-text classifier assigns to the intended class. The classifier uses the official Uni3D prompt set (64 templates), with each class embedding averaged over templates and re-normalised. No test data or labels are involved, so fidelity can be computed before deployment. Tab. 11 reports it on the 15 classes of ScanObjectNN OBJ_ONLY, averaged over three runs.

Coverage does not limit accuracy here. Fidelity does not predict per-class SVG-3D accuracy on the real test scans (Spearman $\rho = +0.20$, $p = 0.48$). Cabinet is the clearest case. Only 5.4% of its generated seeds are recognised as cabinets, yet its test accuracy rises from 10.7% (nearest-text) to

Table 8: Class synonym list $\mathcal{S}_c$ for ModelNet40 (40 classes; reused as-is for ModelNet-C).

Class	Synonyms
airplane	airplane, jet plane, propeller plane
bathtub	bathtub, freestanding bathtub, claw-foot bathtub
bed	bed, single bed, double bed
bench	bench, park bench, wooden bench
bookshelf	bookshelf, tall bookshelf, wooden bookshelf
bottle	bottle, glass bottle, water bottle
bowl	bowl, ceramic bowl, mixing bowl
car	car, sedan, sports car
chair	chair, office chair, dining chair
cone	cone, traffic cone, paper cone
cup	cup, coffee mug, tea cup
curtain	curtain, window curtain, drape
desk	desk, office desk, writing desk
door	door, wooden door, panel door
dresser	dresser, chest of drawers, wooden dresser
flower pot	flower pot, planter, ceramic flower pot
glass box	glass box, glass display case, glass container
guitar	guitar, acoustic guitar, electric guitar
keyboard	keyboard, computer keyboard, piano keyboard
lamp	lamp, table lamp, floor lamp
laptop	laptop, notebook computer, laptop computer
mantel	mantel, fireplace mantel, wooden mantel
monitor	monitor, computer monitor, flat screen monitor
night stand	night stand, bedside table, nightstand
person	person, human figure, standing person
piano	piano, grand piano, upright piano
plant	plant, potted plant, indoor plant
radio	radio, vintage radio, portable radio
range hood	range hood, kitchen hood, exhaust hood
sink	sink, kitchen sink, bathroom sink
sofa	sofa, couch, loveseat
stairs	stairs, staircase, steps
stool	stool, wooden stool, bar stool
table	table, dining table, wooden table
tent	tent, camping tent, dome tent
toilet	toilet, toilet bowl, lavatory toilet
tv stand	TV stand, entertainment center, TV cabinet
vase	vase, flower vase, ceramic vase
wardrobe	wardrobe, armoire, closet wardrobe
xbox	xbox, game console, video game console

36.4% (SVG-3D). SVG-3D uses generated samples as supports near the boundary, not as typical class exemplars. A shape that separates two classes need not be a typical member of either.

Caveats. These 15 classes are everyday objects, and DiffSplat is trained on GObjaverse-derived data. Fidelity is expected to drop for specialised or rare categories outside that distribution. Fidelity also measures the (generator, encoder) pair jointly, so it cannot tell a generation failure from a recognition failure.

Filtering harmful supports. Supports for classes the generator cannot synthesise may be harmful; on ScanNet, Bathtub is already at 0.0 before any mining. We tested the simplest detection criterion: drop every support that nearest-text does not assign to its intended class. This needs only one encoder forward pass and no test data. Tab. 12 shows that it hurts accuracy. Supports near the boundary lie exactly where nearest-text errs, so the filter discards informative hard samples together with artifacts. Separating the two requires a different criterion, which we leave to future work.

Table 9: Hyperparameters shared across all datasets and encoders. Components with per-dataset variation are listed separately in Tab. 10; all values below are held constant across every benchmark and every backbone reported in the paper.

Component	Symbol	Value
SVM kernel	κ	RBF
RBF bandwidth	γ	"scale" (sklearn default)
Probability calibration	–	none
Multi-class scheme	–	one-vs-rest
Focal-pair count	K_{pair}	15
Retraining interval	–	every step
Slerp step size	τ	0.05
Pair-priority rule	–	second-argmax (Eq. (3))
Target-synonym schedule	–	cycle through $\mathcal{S}_t$
Point cloud size	–	10,000 points
Voxel downsample	–	0.005

Table 10: Per-dataset hyperparameters selected on a synthetic DiffSplat-generated validation set following the procedure of Sec. 4. The remaining hyperparameters are held constant across all rows (see Tab. 9).

Dataset	C	σ	N	n_{seed} per class
ScanObjectNN (OBJ_ONLY/OBJ_BG/PB_T50_RS)	0.5	0.3	30	500
ScanNet	0.3	0.3	20	500
ModelNet-C	0.5	0.2	20	100

E Auxiliary study: corruption-aware classification on ModelNet-C

This study is *not* strict zero-shot and does not support the claims of Secs. 4.1 and 4.2. We assume that the corruption *family*—not the test instances—is known in advance, as is common in practice (e.g. LiDAR jitter or depth-camera dropouts). SVG-3D uses this prior as an augmentation operator on generated point clouds. We apply the seven operators (severity 4, parameters from [30]) during both seed construction and MCMC, and train a single classifier for all columns of Tab. 13 without per-corruption tuning.

ModelNet-C [30] corrupts ModelNet40 along seven axes grouped into Density, Noise, and Transformation categories at five severity levels. We report severity 4 in Tab. 13. SVG-3D reaches a mean of 62.2 ± 0.2, compared with 60.5 for OpenShape and 55.4 for Uni3D. Relative to the without-MCMC initialization, boundary mining improves all seven corruption types, most on Add-L ($+2.1$), Scale ($+1.7$), and Rotate ($+1.6$). SVG-3D still trails OpenShape on Scale (64.6 ± 0.2 vs. 74.0). Starting from nearest-text in the same Uni3D feature space (55.4), an SVM on clean generated seeds reaches 57.4 ($+2.0$), corruption-augmented seeds raise it to 61.0 ($+3.6$), and MCMC adds a further $+1.2$ (62.2 ± 0.2). The strict zero-shot settings show the opposite pattern: MCMC supplies 5.8 of the 6.4-point gain on OBJ_ONLY and all of the gain on ScanNet.

F Complexity Analysis

Wall-clock cost. On a single A100 GPU, the dominant cost is generator decoding. A complete SVG-3D run on ScanObjectNN OBJ_BG (15 classes, $K_{\text{pair}} = 15$, $N = 30$, $n_{\text{seed}} = 500$) takes roughly 10 GPU-hours. No encoder or generator weight is updated, so memory stays at the inference level. Clamping K_{pair} keeps Phase 2 independent of the class count K, while Phase 1 scales as $\mathcal{O}(K \cdot |\mathcal{S}_c| \cdot |\mathcal{T}|)$ (Sec. 3.3).

Breakdown. In this run the generator is called 7,500 times for Phase-1 seeds and 450 times for Phase-2 proposals. Phase 1 therefore accounts for about 94% of the cost, and boundary mining for about 6%. Mitigation should target Phase 1.

Table 11: Generator coverage on ScanObjectNN OBJ_ONLY, measured by the fidelity of Phase-1 seeds (mean over three runs).

Fidelity	Classes	Members
$< 25\%$	1/15	cabinet
$< 50\%$	3/15	cabinet, sink, shelf
$\geq 50\%$	12/15	all others

Table 12: Filtering supports by nearest-text agreement on ScanObjectNN OBJ_ONLY (three runs).

Configuration	Top-1 (%)	Supports removed
No filter (SVG-3D full)	$\mathbf{71.7 \pm 0.4}$	–
Keep only supports matching the intended class	67.8 ± 0.6	29%
Keep only non-matching supports (control)	41.4 ± 0.4	71%

Table 13: Per-corruption classification accuracy (%) on ModelNet-C under corruption-aware augmentation (Appendix E) at severity 4. SVG-3D reports mean ± standard deviation over three runs; its without-MCMC row is the common seed-only initialization of those runs. "Mean" is the unweighted average over the seven columns. All numbers are evaluated by us, since the cited papers do not report ModelNet-C numbers under this protocol. Best in column in **bold**.

Method	**Mean**	Scale	Jitter	Drop-G	Drop-L	Add-G	Add-L	Rotate
OpenShape [22]	60.5	**74.0**	32.6	**61.8**	**58.4**	65.8	60.0	71.0
Uni3D [50]	55.4	70.4	38.9	43.3	43.9	63.1	55.2	**73.2**
SVG-3D (Ours)	$\mathbf{62.2 \pm 0.2}$	64.6 ± 0.2	$\mathbf{60.0 \pm 0.4}$	57.3 ± 0.4	52.4 ± 0.1	$\mathbf{66.6 \pm 0.5}$	$\mathbf{65.3 \pm 0.3}$	69.2 ± 0.3
without MCMC	61.0	62.9	58.6	56.8	51.9	65.7	63.2	67.6

Mitigations. *(i) Caching the seed pool.* Additional runs reuse the Phase-1 output and pay only for Phase 2. *(ii) Sharing the classifier across splits.* Because no test data is consulted, one classifier serves all three ScanObjectNN splits. *(iii) Sharing seeds across datasets.* Templates are identical across datasets, and synonyms depend only on the class name. Seeds for a class name can therefore be reused wherever that name appears.

Inference-time cost. At test time the OvR SVM evaluates $K \cdot n_{\mathrm{SV}}$ kernel inner products in CLIP space. On our datasets n_{SV} remains in the low hundreds per class thanks to the support-vector sparsity characteristic of kernel machines, so the additional latency over nearest-text classification ($K \cdot d$ inner products) is a constant factor that, on an A100, is dwarfed by the ϕ_{3D} forward pass on the test point cloud—empirically below a few milliseconds per shape and never the bottleneck in deployment. This makes SVG-3D a drop-in replacement for nearest-text inference wherever the encoder forward pass is already on the critical path.

Table 14: Computational complexity of SVG-3D components. n: total support count, d: CLIP embedding dimension, K: number of classes, K_{pair}: number of focal pairs (constant in K), N: MCMC steps.

Component	Cost
Phase 1 seed generation	$O(K \cdot \lvert\mathcal{S}_c\rvert \cdot \lvert\mathcal{T}\rvert)$ generator calls
Phase 2 MCMC (total proposals)	$O(K_{\mathrm{pair}} \cdot N)$
DiffSplat decoding (per proposal)	$O(1)$ GPU forward pass
ϕ_{3D} encoding (per proposal)	$O(1)$ GPU forward pass
Per-step OvR-SVM retraining	$O(K \cdot n^2 \cdot d)$
Inference (per test point cloud)	$O(K \cdot n_{\mathrm{SV}} \cdot d)$

G Visualisations

We include qualitative visualisations for representative focal pairs on ScanObjectNN: Fig. 3 shows the Phase 1 seed pool, and Fig. 4 shows mined supports along MCMC chains, with each row corresponding to a single focal pair and the columns laid out as source seed, MCMC steps $1, \ldots, 10$, and target seed. The seeds in Fig. 3 are near-canonical exemplars that vary mainly in viewpoint, colour, and minor stylistic details, so the seed pool is homogeneous within each row. In Fig. 4, the decoded supports stay close to the source class in the early steps. In later steps, the decoded geometry often departs from both classes: many rows show flattened, fragmented, or text-like shapes, or unrelated objects, rather than a gradual transition towards the target class. The mined supports are thus more diverse than the seeds. These later-step shapes are hard to interpret visually, but they were accepted because they receive large SVM dual coefficients, i.e. they lie near the decision boundary in the frozen encoder's feature space; this may be why they help despite their appearance, and the contribution of Phase 2 is quantified by the ablations in Sec. 4.3.

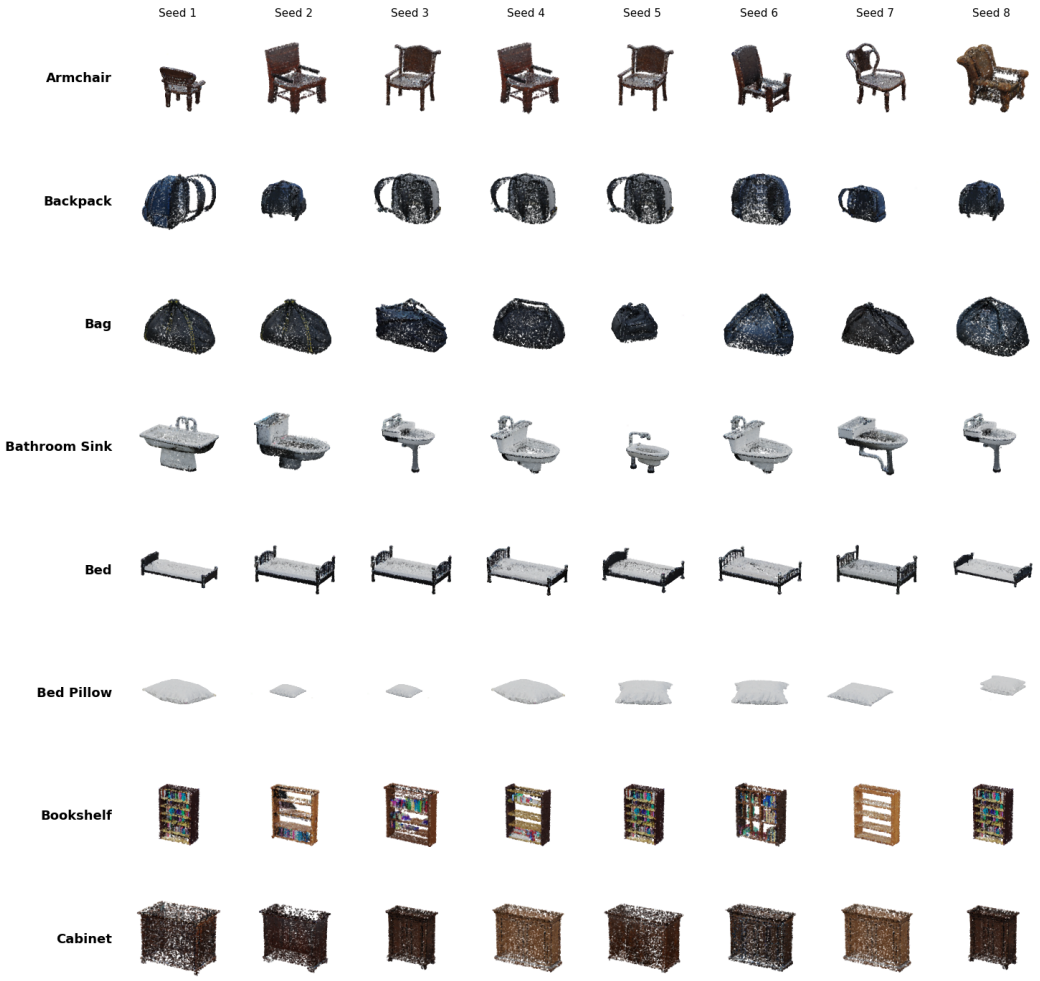

Figure 3: Phase 1 seed pool (no MCMC). Each row shows eight seeds produced by the splat generator for one class synonym (`armchair`, `backpack`, `bag`, `bathroom sink`, `bed`, `bed pillow`, `bookshelf`, `cabinet`; alphabetical). Within each row the seeds are near-canonical exemplars and differ mainly in viewpoint, colour, and minor stylistic variation.



Figure 4: MCMC trajectories of decoded supports for representative focal pairs on ScanObjectNN. Each row corresponds to a single focal pair; the leftmost column shows a source-class seed, the rightmost column a target-class seed, and the ten intermediate columns show decoded supports of accepted samples at MCMC steps $1, \ldots, 10$. Pairs include `bag–pillow`, `handbag–pillow`, `bed/double bed/single bed–door/pillow`, `recycling bin/wastebasket/cardboard box–bag/cabinet`, `small box/storage box/cabinet–cabinet/door`, `storage cabinet–door`, and `computer desk/desk/office desk–table`. Early-step supports remain close to the source class; later steps often produce flattened, fragmented, or unrelated shapes rather than a gradual transition to the target class.

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: The abstract states two design choices (test-data-free pair selection and the SVG-style MH sampler with a closed-form approximate Slerp+Gaussian proposal ratio) and Sec. 1 lists three contributions; each maps directly to a section of the paper (Secs. 3.4 to 3.6) and to the experimental claims in Sec. 4. The scope is restricted to dozens-of-class zero-shot classification on CLIP-aligned encoders, as discussed in Sec. 5.

   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: Sec. 5 contains a "Limitations and future work" paragraph covering the CLIP-alignment requirement, K-scaling and wall-clock cost, generator coverage, harmful supports, and per-class trade-offs within focal pairs.

   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: [N/A]

   Justification: The paper does not introduce new theorems. The only formal derivation, the MH acceptance ratio of Eq. (7), follows the form of SVG's acceptance rule [24] and is re-derived for completeness in Appendix A; the detailed-balance argument and its assumptions (a Gaussian approximation of the proposal density that ignores the norm rescaling, target $\pi(x) \propto \alpha(x)$) are stated explicitly in Sec. 3.5 and Appendix A.

   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: Algorithm 1 specifies the full procedure; Sec. 4 fixes encoders, generator, and evaluation protocol; Tables 9 and 10 list every hyperparameter (kernel, C, K_{pair}, N, τ, σ, n_{seed}, point-cloud size, voxel downsample); Appendix B gives the full prompt-template and synonym lists. All encoders, the generator, and all benchmarks are publicly available with the citations given in the paper.

   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 code is not publicly released. To support reproduction, the paper specifies the full pipeline (Sec. 3), all hyperparameters (Table 9), and the complete prompt-template and synonym lists with their generation procedure (Appendix B). All datasets, encoders, and the generator (DiffSplat) are publicly available.

   Guidelines:

   - 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: Sec. 4 and Table 9 (full hyperparameter table in Appendix C) specify the encoders, generator, point-cloud preprocessing, benchmark splits (ScanObjectNN OBJ_ONLY/OBJ_BG/PB_T50_RS, ModelNet-C severity 4, ScanNet via the Uni3D pipeline), kernel choice, SVM regulariser, MCMC parameters $(\tau, \sigma, N, K_{\text{pair}})$, and the synthetic validation set on which the per-dataset values were selected.

   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: [Yes]

   Justification: All main SVG-3D results (ScanObjectNN OBJ_ONLY/OBJ_BG/PB_T50_RS, ScanNet, ModelNet-C, and the Point-E replacement) report mean$\pm 1\sigma$ (standard deviation) over three independent runs of the full pipeline (Sec. 4); ablation rows (a)–(f) of Tab. 4 are single runs, and the nearest-text baselines are deterministic given the encoder. On every benchmark the gain over the strongest nearest-text baseline exceeds the run-to-run standard deviation (Secs. 4.1 and 4.2 and Appendix E).

   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: Appendix F reports a complete SVG-3D run on ScanObjectNN OBJ_BG (K=15, K_{pair}=15, N=30, n_{seed}=500) at roughly 10 A100 GPU-hours, broken down into 7,500 Phase-1 and 450 Phase-2 generator calls, with mitigations by seed caching and sharing; the SVM solve is millisecond-scale, and inference latency on a held-out point cloud is below a few milliseconds. Table 14 gives the asymptotic cost of every component.

   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 uses only publicly released encoders, generators, and synthetic/real-scan benchmarks under their original licences, involves no human-subject data, and introduces no models or datasets that could plausibly be misused.

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: The work is a methodological contribution to zero-shot 3D classification on existing benchmarks; it neither releases new data, models, or surveillance-capable systems, nor opens a direct path to deployment beyond the academic 3D-recognition pipelines it builds on. We therefore do not see a specific positive or negative societal impact distinct from those of the underlying CLIP-aligned 3D encoders and text-to-3D generators it cites.

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 releases neither pretrained generative models nor scraped data.

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: All used assets—OpenShape [22], Uni3D [50], ULIP/ULIP-2 [42, 43], ReCon++ [28], DuoduoCLIP [19], DiffSplat [21], ScanObjectNN [38], ModelNet/ModelNet-C [30, 40], ScanNet [7], CLIP [29]—are cited at first use; we use each at the public release version specified in its repository and respect the original licences (academic / non-commercial research use, as applicable per asset).

    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 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 involves no crowdsourcing and no human-subject experiments.

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: No human-subject research is involved.

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: [Yes]

Justification: The prompt templates and class-synonym lists used to build the Phase-1 seeds (Sec. 3.3) were generated once with an LLM (Claude Sonnet 4.6). Appendix B gives the model, the exact instructions, what was withheld from the LLM (dataset names, test data, labels), and the complete resulting lists. No LLM is used elsewhere in the method.

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.