SELF-SUPERVISED POINT CLOUD DATASET DISTILLATION FOR EFFICIENT REPRESENTATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Dataset distillation (DD) aims to replace a large training dataset with a compact synthetic set that preserves its training utility, substantially reducing repeated storage and training costs. However, existing point-cloud DD methods are developed under supervised settings and rely on semantic annotations to construct distilled data. While self-supervised learning (SSL) makes large unlabeled point-cloud collections usable without annotation, repeatedly training on the full dataset remains computationally expensive. Moreover, a pretrained checkpoint merely provides model-specific rather than data-level reusability. This motivates a fundamental question: can the representation-learning utility of a massive unlabeled point-cloud dataset be distilled into a small, reusable synthetic set? We study self-supervised dataset distillation for point clouds and propose a distillation framework for constructing compact training sets from unlabeled 3D data. Rather than relying on semantic classes or discrete pseudo-labels, we formulate distillation around self-supervised representation learning and optimize synthetic point clouds according to their ability to train newly initialized encoders. Specifically, our framework combines synthetic point clouds, fixed teacher-derived representation targets, and geometric masking metadata into a compact predictive training resource. Newly initialized encoders consume this resource through the same masked prediction objective, without accessing the teacher or full source collection. We evaluate the distilled synthetic sets on multiple standard point-cloud benchmarks via frozen representation evaluation against equally sized real-subset controls. Extensive experiments demonstrate that the optimized synthetic data yields non-trivial representation-learning utility compared to equally sized real subsets across multiple compression regimes. These results demonstrate the potential of self-supervised point-cloud dataset distillation as a data-level mechanism for compressing and reusing large unlabeled 3D collections.

1 INTRODUCTION

Self-supervised learning (SSL) enables point-cloud encoders to learn from unlabeled geometry through contrastive objectives (Xie et al., 2020; Zhang et al., 2021), reconstruction, or representation prediction (Pang et al., 2022; Knaebel et al., 2023). Repeated full-data pretraining, for example when evaluating different architectures or initializations, still requires substantial computation and data access. A pretrained checkpoint reuses learned weights, whereas a compact training set offers a complementary interface from which a newly initialized encoder can learn. This motivates a question: can the representation-learning utility of a large unlabeled point-cloud collection be distilled into a small, reusable training resource?

Dataset distillation (DD) optimizes synthetic examples for their ability to train models, rather than to reproduce the original inputs (Wang et al., 2018). Point-cloud DD has addressed geometric optimization and synthesis (Zhang et al., 2024; Bo & Wang, 2025; Kim et al., 2026), as well as feature matching and class-aware budgeting (Yim et al., 2025; Zhang et al., 2026; You et al., 2026). These methods use semantic annotations to organize synthesis, matching, or budget allocation. Without such annotations, the problem is therefore not only which synthetic geometries to construct, but what supervision they should carry to support representation learning.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Self-supervised DD provides a foundation by pairing synthetic images with continuous representation targets (Lee et al., 2024; Joshi et al., 2025; Yu et al., 2025). Through representation regression, teacher-guided trajectory matching, or compact parameterization, these methods retain supervision as part of the distilled resource. We build on this principle to study how point-cloud geometry and predictive supervision can be organized together. The aim is to construct examples that teach a representation-learning task under a small object budget, without assigning semantic classes or discrete pseudo-labels.

Masked prediction offers a concrete task for this purpose by learning to predict teacher-derived representations from partial observations (Baevski et al., 2022; Assran et al., 2023; Knaebel et al., 2023; Saito et al., 2025). In our formulation, geometric regions determine what the learner observes, while teacher-derived targets specify what it must predict. Whole-object prediction associates a partial cloud with an object-level description. Hidden-region prediction instead supervises descriptions of the omitted regions. These alternative target scopes define training signals beyond the visible coordinates, allowing us to optimize synthetic geometry for the learner's ability to perform the same task on real objects.

We propose **masked predictive distillation for 3D point clouds (MPD-3D)**, which constructs a compact predictive training resource $\mathcal{S} = \{(\widetilde{X}_j, Z_j, \Omega_j)\}_{j=1}^{M}$. Here, M is the object budget, $\widetilde{X}_j$ contains synthetic coordinates, Z_j contains fixed teacher-derived targets, and Ω_j records fixed geometric region memberships. The targets and memberships are inherited from source objects used for initialization. During production, an encoder and predictor take a short sequence of training updates on masked synthetic clouds. The updated learner's prediction error on other real source objects is differentiated through these updates to optimize only the synthetic coordinates. Geometry is thus optimized for its training effect under the stored supervision. Independently initialized consumers subsequently train on the resource with the same predictive objective, without accessing the teacher or full source collection during pretraining.

We evaluate the resulting frozen representations on ModelNet40, ScanObjectNN, and MVP100FULL at three budgets per dataset. Comparisons use equally sized, class-supervised real-subset pipelines and common linear-probe and nearest-neighbor readouts. With a 10% labeled readout bank, the selected configurations achieve higher mean linear-probe accuracy than these baselines in all nine settings. Additional studies examine target scope, inner-loop updates, initialization, and consumption by other encoder architectures.

Our contributions are threefold:

- We formulate self-supervised point-cloud DD around a compact predictive training resource that combines synthetic geometry, fixed representation targets, and geometric masking information.
- We develop MPD-3D, optimizing synthetic coordinates through short encoder–predictor updates and real-query feedback under alternative whole-object and hidden-region prediction tasks.
- We evaluate independently trained consumers across three benchmarks and multiple budgets, and analyze target strategies and cross-architecture consumption of the same resources.

2 Related Work

Dataset distillation for 3D data. DD constructs compact training sets through learning-outcome optimization (Wang et al., 2018), gradient or trajectory matching (Zhao et al., 2021; Cazenavette et al., 2022), and feature-distribution matching (Zhao & Bilen, 2023). Kernel and neural-feature regression provide tractable inner learning problems (Nguyen et al., 2021; Zhou et al., 2022b; Loo et al., 2022). Point-cloud methods develop gradient-based point optimization and rotation-aware, variable-resolution synthesis (Zhang et al., 2024; Bo & Wang, 2025), improve point-feature matching (Yim et al., 2025; Zhang et al., 2026), and construct compact sets through shape parameterization or class-aware budgeting (Kim et al., 2026; You et al., 2026). These approaches organize training around semantic labels. We study the supervision needed to make a compact point-cloud resource useful for label-free representation learning.

Self-supervised dataset distillation. Self-supervised DD constructs compact datasets for representation pretraining. KRR-ST learns synthetic image–target pairs using a closed-form regression head and real-data representation matching (Lee et al., 2024). MKDT retains fixed teacher-derived targets and matches trajectories of students trained through knowledge distillation (Joshi et al., 2025), while Boost SSDD develops low-dimensional parameterization, predefined augmentation, and approximations of augmented representations (Yu et al., 2025). A complementary line constructs synthetic data for lightweight adaptation of frozen SSL encoders (Cazenavette et al., 2025; Li et al., 2026). Our setting follows the pretraining line: new encoders learn from the distilled resource. We differentiate real-query masked prediction loss through short encoder–predictor updates. KRR-ST instead holds its sampled feature extractor fixed when differentiating through closed-form regression, while MKDT evaluates stored-trajectory discrepancies. Our resource carries whole-object or hidden-region prediction targets together with the synthetic geometry that teaches the task.

Knowledge and representation distillation. Knowledge distillation transfers predictive behavior from a teacher to a student (Hinton et al., 2015). Beyond output probabilities, teacher knowledge can supervise intermediate features, inter-example relations, or contrastive representations (Romero et al., 2015; Park et al., 2019; Tian et al., 2020); self-supervised distillation also transfers instance-level similarity structure (Fang et al., 2021). Data-free KD further synthesizes inputs from trained teachers when the source data are unavailable (Chen et al., 2019; Yin et al., 2020). The central objective in these approaches is teacher-to-student model transfer. Our producer instead uses real source examples to optimize a budgeted training resource through the learner's post-update prediction loss. Subsequent consumers reuse the synthetic inputs and stored targets without online teacher inference.

Masked predictive learning for point clouds. Masked point modeling uses geometric reconstruction (Pang et al., 2022; Zhang et al., 2022), discrete-token prediction (Yu et al., 2022), or masked discrimination (Liu et al., 2022). Teacher-derived prediction targets also underpin representation learning and self-distillation in data2vec, iBOT, and I-JEPA (Baevski et al., 2022; Zhou et al., 2022a; Assran et al., 2023). Point-cloud methods explore masked feature regression and latent prediction (Liu et al., 2023; Knaebel et al., 2023; Saito et al., 2025; Leijenaar & Kasaei, 2025), as well as feature distillation and alignment across different masking conditions (Bahri et al., 2024; Wei et al., 2026). These methods train encoders on the available data. We use masked representation prediction to define the supervision carried by synthetic point clouds, optimizing their coordinates according to how training on them affects prediction on real queries.

3 METHOD

3.1 PROBLEM FORMULATION

Let $\mathcal{D} = \{X_i\}_{i=1}^{N}$ denote an unlabeled point-cloud collection, where $X_i \in \mathbb{R}^{P\times 3}$. Our goal is to distill $\mathcal{D}$ into a compact training resource for point-cloud representation learning:

$$\mathcal{S} = \{(\widetilde{X}_j, Z_j, \Omega_j)\}_{j=1}^{M}, \qquad M \ll N, \tag{1}$$

where $\widetilde{X}_j$ denotes learnable point coordinates, Z_j contains fixed representation targets derived from a frozen self-supervised teacher, and Ω_j specifies the geometric regions used to define partial observations.

A self-supervised teacher provides whole-object or regional targets for masked representation prediction. $\widetilde{X}_j$ is initialized using sampled source objects. We optimize the synthetic coordinates so that training on the resulting point–target pairs improves prediction of the corresponding teacher targets on real source objects. The optimized inputs and fixed supervision together support representation learning through this predictive task.

3.2 SELF-SUPERVISED TEACHER AND FIXED PREDICTIVE TARGETS

Self-supervised teacher. We first learn a representation space from the unlabeled point-cloud collection. This representation space provides the supervision that will subsequently be encoded into the distilled resource. Let E_θ denote a point-cloud encoder and G_ξ a reconstruction decoder. Given

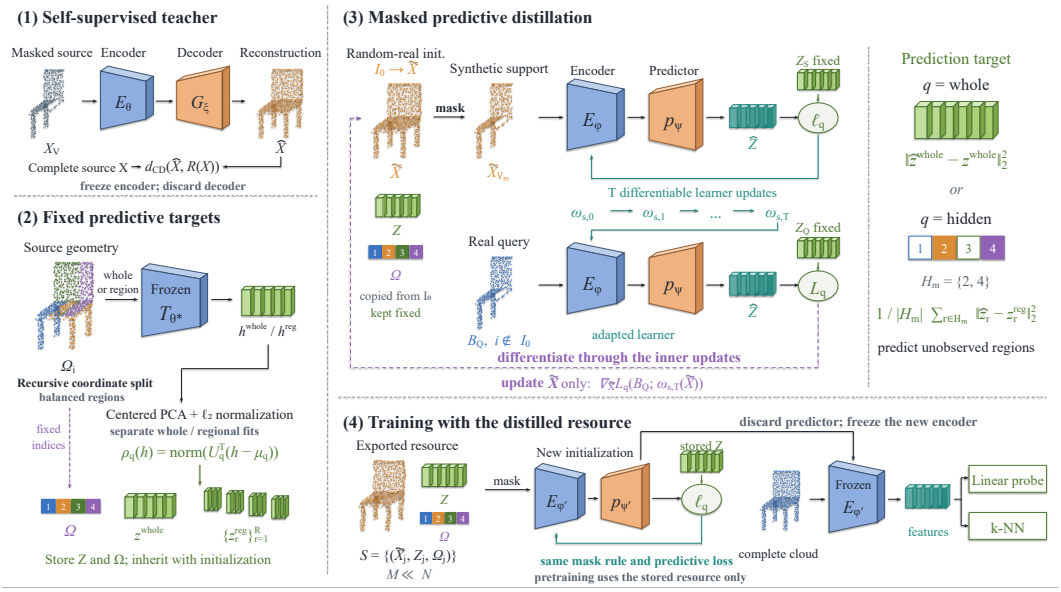


Figure 1: **Overview of masked predictive point-cloud dataset distillation.** (a) A self-supervised encoder–decoder is pretrained on the unlabeled source collection by masked reconstruction; the decoder is then discarded and the encoder is frozen. (b) The frozen encoder produces whole-object and regional features, which are projected by centered PCA and normalized to form fixed predictive targets Z; deterministic geometric memberships Ω define the masking structure. (c) Synthetic point clouds $\widetilde{X}$ are initialized from sampled real objects and inherit their targets and memberships. A learner is adapted on masked synthetic support, while its predictive error on masked real queries is differentiated through the inner updates to optimize only $\widetilde{X}$; Z and Ω remain fixed. (d) The exported resource $\mathcal{S} = \{(\widetilde{X}_j, Z_j, \Omega_j)\}_{j=1}^{M}$ trains a newly initialized encoder and predictor using the same masking rule and predictive objective, without revisiting the source collection or the teacher. The predictor is finally discarded and the frozen encoder is evaluated by linear probing or k-NN.

a masked observation $X_{\mathcal{V}}$ of a source cloud X, the teacher is pretrained by predicting a geometric target $\mathcal{R}(X)$ constructed from the complete cloud:

$$(\theta^\star, \xi^\star) = \arg\min_{\theta,\xi} \mathbb{E}_{X\sim\mathcal{D},\,\mathcal{V}} \left[d_{\mathrm{CD}} \left(G_\xi(E_\theta(X_{\mathcal{V}})), \mathcal{R}(X) \right) \right], \tag{2}$$

where d_{CD} denotes the Chamfer reconstruction loss. After pretraining, the decoder is discarded and the encoder $T_{\theta^\star} \equiv E_{\theta^\star}$ is frozen. The frozen encoder serves only to construct the representation targets used by the distilled predictive task.

Geometric regions. To define prediction from partial observations, each source cloud is partitioned into R balanced geometric regions $\Omega_i = \{\Omega_{i,r}\}_{r=1}^{R}$ using a deterministic recursive coordinate split. The resulting memberships define the visible and hidden portions used during distillation. They are computed once from the source geometry and remain fixed while the synthetic coordinates are optimized.

Fixed representation targets. The frozen teacher is then applied either to the complete object or to each geometric region:

$$h_i^{\mathrm{whole}} = T_{\theta^\star}(X_i), \qquad h_{i,r}^{\mathrm{reg}} = T_{\theta^\star}(X_{i,\Omega_{i,r}}). \tag{3}$$

For each scope $q \in \{\mathrm{whole}, \mathrm{reg}\}$, we fit a centered PCA projection on the corresponding source features and ℓ_2-normalize the projected representation, where $\rho_q(h) = \mathrm{norm}(U_q^\top (h - \mu_q))$. This gives the whole-object and regional targets

$$z_i^{\mathrm{whole}} = \rho_{\mathrm{whole}}(h_i^{\mathrm{whole}}), \qquad z_{i,r}^{\mathrm{reg}} = \rho_{\mathrm{reg}}(h_{i,r}^{\mathrm{reg}}). \tag{4}$$

A synthetic cloud initialized from source object i inherits the corresponding target and region membership. The target and region membership associated with each initialization are fixed throughout distillation, while only the synthetic coordinates are optimized. Consequently, Z and Ω specify the predictive supervision carried by the resource, whereas $\widetilde{X}$ is optimized for effectively teaching that supervision.

3.3 Masked Predictive Distillation

Predictive task. With the fixed targets constructed above, we now optimize a compact set of point clouds for their ability to teach the corresponding predictive task. For mask m, let V_m and H_m denote the visible and hidden regions. A learner consists of an encoder E_ϕ and predictor p_ψ, with parameters $\omega = (\phi, \psi)$, and observes only X_{V_m}. Given the visible points X_{V_m}, it predicts $\widehat{Z} = p_\psi(E_\phi(X_{V_m}))$ in the target representation space. Its prediction is trained with

$$\ell_q = \begin{cases} \|\widehat{z}^{\text{whole}} - z^{\text{whole}}\|_2^2, & q = \text{whole}, \\ \dfrac{1}{|H_m|} \displaystyle\sum_{r \in H_m} \|\widehat{z}_r - z_r^{\text{reg}}\|_2^2, & q = \text{hidden}. \end{cases} \tag{5}$$

Whole-object prediction asks a partial observation to recover an object-level representation, whereas hidden-region prediction focuses the supervision on the unobserved region. Thus, the target scope q specifies the predictive task encoded by the resource, while Z provides its fixed representation supervision.

Bilevel coordinate optimization. At outer iteration s, the learner is adapted on the distilled resource using the predictive objective in Eq. equation 5. We denote the resulting parameters after T differentiable optimization steps by

$$\omega_{s,T} = \mathcal{U}_T\left(\omega_{s,0}; \mathcal{S}, \ell_q\right), \tag{6}$$

where $\mathcal{U}_T$ denotes T differentiable optimization steps on masked samples from $\mathcal{S}$ using the predictive loss ℓ_q in Eq. equation 5.

The distilled coordinates are optimized according to the predictive performance of the adapted learner on real source queries:

$$\widetilde{X}^\star = \arg\min_{\widetilde{X}} \; \mathbb{E}_{\mathcal{B}_Q}\left[\mathcal{L}_q\left(\mathcal{B}_Q; \omega_T(\widetilde{X})\right)\right], \tag{7}$$

where $\mathcal{B}_Q$ contains source objects outside the initialization subset and $\mathcal{L}_q$ is the batch-averaged predictive loss induced by Eq. equation 5.

We minimize the query loss in Eq. equation 7 with respect to $\widetilde{X}$ by differentiating through the inner learner updates. This optimizes the synthetic point clouds, paired with their fixed targets, for reducing teacher-target prediction error on real source objects under the predictive loss in Eq. equation 5. The optimized point–target pairs and masking structure thus serve as a compact resource for teacher-guided representation learning under the prescribed masks and predictive loss.

3.4 Training with the Distilled Resource

To train a new encoder, we initialize an encoder and prediction head and optimize them on masked views of the distilled point clouds. The stored targets provide supervision through the predictive loss in Eq. equation 5, using the same target scope and masking rule as in distillation. Through this task, teacher-derived representation information is transferred through the optimized point–target pairs, without revisiting the source collection or evaluating the teacher during pretraining. We then discard the prediction head and freeze the encoder for downstream representation evaluation on complete point clouds.

4 Experiments

4.1 Experimental setup

Datasets and budgets. We evaluate on ModelNet40 (MN40) (Wu et al., 2015), ScanObjectNN OBJ_BG (SONN) (Uy et al., 2019), and the fixed 100-category MVPNet subset MVP100FULL

Algorithm 1 Masked Predictive Dataset Distillation

Require: Unlabeled source set $\mathcal{D}$, frozen teacher $T_{\theta^\star}$, budget M, target scope q, inner steps T, outer steps S

Construct geometric memberships $\{\Omega_i\}_{i=1}^N$ from the source geometry
Construct fixed representation targets $\{Z_i\}_{i=1}^N$ using $T_{\theta^\star}$
Sample source indices $\mathcal{I}_0 = \{i_1, \ldots, i_M\}$ and initialize $\widetilde{X}_j \leftarrow X_{i_j}$
Copy the corresponding fixed targets and memberships to the synthetic set
for $s = 0, \ldots, S-1$ **do**
    Set the inner learner state $\omega_{s,0}$ according to the configured state policy
    Sample a real query batch $\mathcal{B}_{Q,s}$ from $\{(X_i, Z_i, \Omega_i) : i \notin \mathcal{I}_0\}$
    Form synthetic support batches $\{\mathcal{B}_{S,t}\}_{t=0}^{T-1}$ according to the sampling policy
    for $t = 0, \ldots, T-1$ **do**
        Apply the scheduled mask $m_{s,t}$ to $\mathcal{B}_{S,t}$
        Compute the predictive loss ℓ_q in Eq. equation 5
        Differentiably update the learner to obtain $\omega_{s,t+1}$
    end for
    Evaluate $\omega_{s,T}$ on $\mathcal{B}_{Q,s}$ to obtain the real-query predictive loss
    Update $\widetilde{X}$ through the query loss by differentiating through the inner learner updates
    If persistent, detach and retain the learner parameters and optimizer state
end for
return $\mathcal{S} = \{(\widetilde{X}_j, Z_j, \Omega_j)\}_{j=1}^M$

(MVP) (Yu et al., 2023). The total object budgets are $M \in \{40, 200, 400\}$, $\{15, 75, 150\}$, and $\{100, 500, 1000\}$, respectively. These budgets match one, five, and ten objects per benchmark class for reporting purposes. The producer does not allocate samples by semantic class. Dataset versions and split sizes are in Appendix B.

Consumption and evaluation. PointNet (Qi et al., 2017) is the default producer and consumer backbone. Every consumer starts from random initialization and trains for 200 epochs. Following prior work, We freeze the encoder and evaluate its native features on complete objects using a linear probe (LP) and weighted k-nearest neighbors (kNN) (Lee et al., 2023; Hermosilla et al., 2025; Joshi et al., 2025). LP@10% and kNN@10% share a 10% labeled training bank. Results are means and sample standard deviations over three consumers per resource. (Appendix B.3).

Real-subset baselines. Random, Herding (Welling, 2009), and K-center (Sener & Savarese, 2018) select equally sized real subsets using class-wise budgets. Their consumers train by class-label cross-entropy on complete objects, whereas MPD-3D uses fixed representation targets and 512 visible points. The same epoch schedule and frozen readouts compare the complete pipelines; The details are specified in Appendix B.4.

4.2 Representation learning from compact resources

Table 1 directly tests whether the representation-learning utility of the full unlabeled collection can be compressed into a small training resource. At the same object budget, MPD-3D achieves higher mean LP@10% and kNN@10% than all three real-subset pipelines in all nine settings, using fixed predictive targets during consumer training. The gains are especially clear under linear probing: on ModelNet40 at $M = 200$, LP@10% increases from 45.22% to 63.86%. On MVP100FULL and ScanObjectNN, the improvements appear under both readouts: at their middle budgets, LP@10% improves by 7.26 and 3.56 points, and kNN@10% by 4.86 and 4.65 points, respectively.

The advantage also persists at the smallest budgets, showing that useful representation-learning signal can be retained in highly compact resources, and increasing M further improves both readouts on all three datasets. Together, these results support the central premise of MPD-3D: teacher-derived predictive supervision can be packaged with a small set of point clouds into a reusable data-level resource without revisiting the full source collection. These results should be interpreted as evidence for the utility of the complete predictive resource rather than the optimized geometry alone. The full

Table 1: Frozen PointNet representations with a 10% labeled readout bank. Entries are OA (%), mean ± sample SD over three consumers. Real-subset pipelines use class supervision; MPD-3D uses masked target regression. Bold marks the highest mean within each row.

Dataset	M	Readout	Random	Herding	K-center	MPD-3D
MN40	40	LP	40.01±0.61	41.13±0.16	41.13±0.04	**57.56**±0.35
		kNN	65.82±0.06	65.55±0.21	65.53±0.15	**66.00**±0.36
	200	LP	44.85±0.47	45.22±0.21	43.57±0.53	**63.86**±0.84
		kNN	67.83±0.27	67.67±0.45	69.03±0.26	**69.34**±0.60
	400	LP	45.56±0.10	47.08±0.62	45.73±0.13	**65.71**±0.39
		kNN	69.57±0.35	69.34±0.37	69.38±0.19	**71.04**±0.22
MVP100FULL	100	LP	18.43±0.05	18.59±0.12	18.67±0.12	**25.23**±0.14
		kNN	37.11±0.21	37.36±0.11	37.48±0.12	**42.36**±0.08
	500	LP	21.64±0.11	21.81±0.04	20.49±0.07	**29.07**±0.18
		kNN	41.03±0.25	41.51±0.16	39.84±0.40	**46.37**±0.24
	1000	LP	22.82±0.13	23.10±0.08	21.99±0.15	**29.81**±0.05
		kNN	42.25±0.28	42.45±0.14	40.72±0.15	**46.83**±0.16
SONN	15	LP	35.23±0.53	35.57±0.78	35.57±0.78	**37.23**±0.26
		kNN	35.51±0.70	35.46±0.46	35.46±0.46	**40.33**±1.01
	75	LP	37.12±0.36	37.46±1.22	36.89±0.98	**41.02**±0.26
		kNN	38.15±0.72	38.55±0.96	37.01±0.34	**43.20**±0.60
	150	LP	38.90±0.30	41.02±0.70	38.55±0.00	**43.37**±1.13
		kNN	39.24±0.91	41.25±0.26	38.10±0.60	**43.55**±0.46

labeled bank are shown (Appendix C). The additional effect of coordinate refinement under matched predictive supervision is configuration-dependent (Appendix H).

4.3 Reuse across consumer architectures

Table 2: Same-resource consumption at the middle object budget of each dataset. LP@10% (%), mean ± SD over three consumers. All eight architectures use the same dataset-specific resource and 200-epoch predictive training protocol.

Consumer	MN40 ($M = 200$)	MVP ($M = 500$)	SONN ($M = 75$)
PointNet (producer backbone)	63.86±0.84	29.07±0.18	41.02±0.26
DGCNN	56.43±0.79	25.96±0.19	38.67±1.01
PointMLP	37.47±2.77	18.80±0.19	24.10±1.55
PointMLP-Elite	40.02±0.60	17.80±0.11	21.06±0.95
PointNeXt	38.65±2.04	19.08±0.13	30.58±2.18
Point-MAE encoder	42.64±1.68	20.78±0.05	18.88±6.80

We further evaluate whether a distilled resource can be reused by consumer architectures that differ from the PointNet producer. Each resource is kept entirely fixed—including its synthetic coordinates, predictive targets, region memberships, and masks. It is consumed by newly initialized encoders without any additional coordinate optimization. Table 2 reports representative architectures at the middle budget of each dataset, while Appendix I contains the complete resource–architecture evaluation.

The same distilled resource remains usable across different point-cloud backbones. In particular, DGCNN obtains 56.43%, 25.96%, and 38.67% LP@10% on MN40, MVP, and SONN, respectively. PointMLP, PointNeXt, and the Point-MAE encoder likewise learn non-trivial frozen representations from the fixed resources. These results indicate that the distilled resource is not restricted to the producer architecture: its stored geometry and predictive supervision can be reused to train different consumers without re-distillation.

4.4 EFFICIENCY OF REPRESENTATION LEARNING

We assess efficiency jointly through frozen LP@10 accuracy and consumer training time, extending the real-subset baselines to PPC values from 1 to 50. PPC denotes the total object budget divided by the benchmark class count; MPD-3D is evaluated at its three produced budgets, PPC$= 1, 5, 10$. All consumers follow the 200-epoch protocol, so these are comparisons across resource sizes at a fixed epoch count. The horizontal axis measures consumer training only, excluding resource construction and frozen readout fitting.

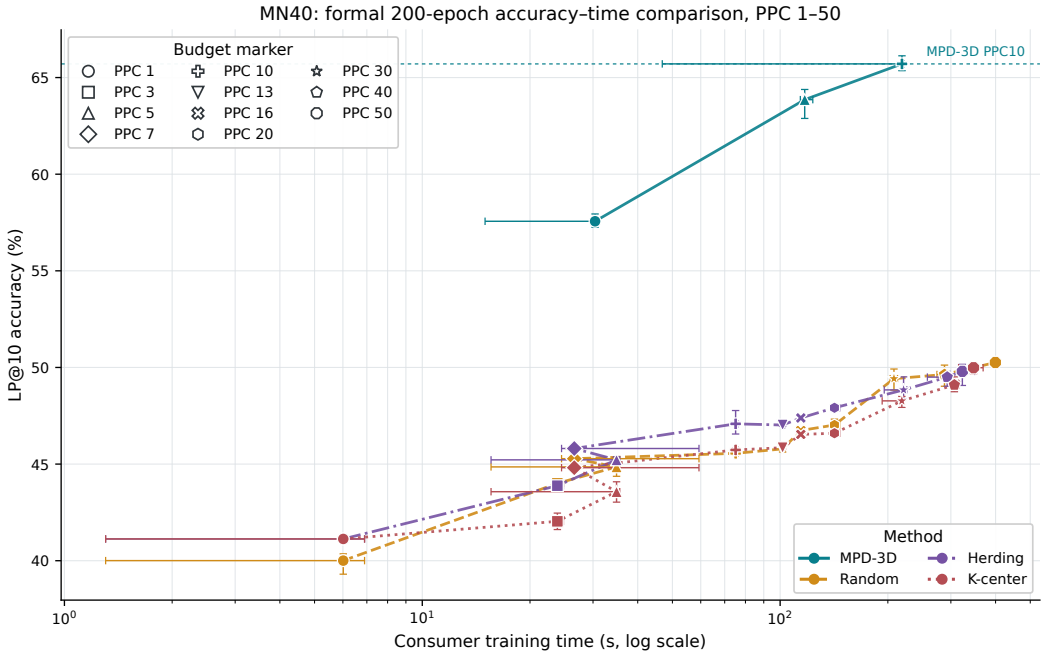


Figure 2: Representation quality versus consumer training time on ModelNet40 under the 200-epoch protocol. The horizontal axis is logarithmic. Vertical bars span the minimum and maximum LP@10 accuracy across consumer seeds, while horizontal bars show the interquartile range of measured training time. MPD-3D is evaluated at PPC$= 1, 5, 10$; real-subset baselines are extended to PPC$= 50$.

Figure 2 shows that increasing the real-subset budget yields only gradual representation gains on ModelNet40. At PPC$= 1$, MPD-3D reaches 57.56% LP@10 with a median consumer time of 30.40 s. In comparison, even at PPC$= 50$, the strongest real-subset LP@10 is 50.26%, with consumer times exceeding 300 s for the reported selection methods. Increasing the predictive-resource budget further raises LP@10 to 63.86% and 65.71% at PPC$= 5$ and 10, respectively.

The same qualitative advantage appears on MVP100FULL: MPD-3D at PPC$= 1$ obtains 25.23% LP@10, whereas the real-subset methods remain below 24% even at PPC$= 50$. For ScanObjectNN, only very larger real subsets can match or slightly exceed the predictive resource while requiring lower median consumer time in several settings. The complete MVP100FULL and ScanObjectNN accuracy–time plots are reported in Appendix D. These results therefore support efficient representation learning through compact resources.

4.5 PRODUCER DESIGN ABLATIONS

Choosing the prediction target

Table 3 compares whole-object and hidden-region prediction with the base producer recipe. Whole-object targets yield higher mean LP@10% in seven settings and higher kNN@10% in eight, with one kNN tie at the displayed precision. The advantage is small at larger MVP100FULL budgets: at $M = 1000$, the two strategies differ by 0.03 points in LP@10% and 0.01 in kNN@10%. Hidden-

Table 3: Target-strategy comparison. Cells show mean LP@10% / kNN@10% over three consumers. Whole-object and hidden-region prediction supervise 128 and 64 target dimensions per update, respectively.

Dataset	M	Whole object	Hidden region
MN40	40	**57.56 / 66.00**	56.98 / 64.74
	200	**63.86 / 69.34**	62.79 / 68.61
	400	**65.71 / 71.04**	65.06 / 71.00
MVP100FULL	100	**25.23 / 42.36**	24.30 / 40.56
	500	**29.07 / 46.37**	28.34 / 45.98
	1000	29.81 / **46.83**	**29.84** / 46.82
SONN	15	35.46 / 39.53	**37.23** / 40.33
	75	**40.22 / 42.05**	39.47 / 41.25
	150	**43.43 / 42.80**	41.71 / 42.17

Table 4: Producer-design ablations. **Top:** geometric masking on ModelNet40 at $M = 400$ with hidden-region prediction. **Bottom:** coordinate initialization on ModelNet40 ($M = 40$) and ScanObjectNN ($M = 150$). All entries are mean LP@10 / kNN@10 OA (%) over three consumers.

Dataset	Configuration	Visible	LP@10	kNN@10
Geometric masking				
MN40	$R = 2$, hide 1	512	**64.91**	71.07
	$R = 8$, hide 4	512	62.55	69.03
	$R = 4$, hide 1	768	64.32	**71.27**
	$R = 4$, hide 3	256	58.06	68.75

region prediction gives higher LP@10% on ScanObjectNN at $M = 15$. Target scope is therefore a useful design choice, rather than a uniform preference for global or local supervision. Because the target dimension also changes, this comparison evaluates the two task configurations, not locality in isolation. Appendix G gives complete readouts and the inner-update, initialization, query, and masking studies.

Geometric masking. As shown in Table 4,the organization of visible geometry has a clear effect on the predictive resource. With the visible-point budget fixed at 512, the two-region setting improves LP@10 from 62.55% to 64.91% compared with the more fragmented eight-region setting. Under four-region partitioning, exposing only 256 points reduces LP@10 to 58.06%, whereas retaining 768 visible points recovers 64.32%. These results suggest that the predictive task benefits from sufficient and structurally coherent context.

Additional producer analyses and ablations are reported in Appendix G, including complete four-readout versions of these studies, head-only inner updates, real-query sampling, inner learning-rate and optimizer-state choices, and unroll length.

5 Conclusion

We introduced MPD-3D, a framework for self-supervised point-cloud dataset distillation. It constructs compact, reusable training resources from unlabeled collections. Synthetic geometry, fixed teacher-derived targets, and geometric masking information jointly specify a masked representation-prediction task. The coordinates are optimized and newly initialized encoders subsequently learn from the resulting resource using the same predictive objective, without accessing the teacher or full source collection during pretraining. Experiments across three benchmarks demonstrate useful frozen representations that can be reused by different consumer architectures. Accuracy–cost comparisons further show favorable consumer-training trade-offs against larger real subsets. Together, these findings support self-supervised dataset distillation as a data-level approach to retaining and reusing the representation-learning utility of unlabeled 3D data.

AI USE STATEMENT

Generative AI tools were used to assist with writing and language polishing, literature retrieval and discovery, and aspects of research ideation and execution, including feedback on methodology and experiments and assistance with implementation and result interpretation. All AI-assisted research suggestions, technical content, and literature-related claims were reviewed and verified by the authors. The authors take full responsibility for the final content of this work, including all text, claims, and artifacts produced with AI assistance.

ETHICS STATEMENT

This work uses publicly available 3D point-cloud benchmark datasets and does not involve human subjects, personally identifiable information, or sensitive personal data. We are not aware of any specific ethical risks beyond those generally associated with machine learning research.

REPRODUCIBILITY STATEMENT

To support reproducibility, we document the complete experimental protocol in the appendix, including dataset versions and budgets, teacher and target construction, producer and consumer configurations, evaluation procedures, and real-subset baselines. We further provide complete numerical results, producer-design ablations, cross-architecture evaluations, and efficiency measurements to facilitate independent verification of the reported findings.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15619–15629, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Assran_Self-Supervised_Learning_From_Images_With_a_Joint-Embedding_Predictive_Architecture_CVPR_2023_paper.html.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. data2vec: A general framework for self-supervised learning in speech, vision and language. In *International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 1298–1312, 2022. URL https://proceedings.mlr.press/v162/baevski22a.html.

Ali Bahri, Moslem Yazdanpanah, Mehrdad Noori, Milad Cheraghalikhani, Gustavo Adolfo Vargas Hakim, David Osowiechi, Farzad Beizaee, Ismail Ben Ayed, and Christian Desrosiers. GeoMask3D: Geometrically informed mask selection for self-supervised point cloud learning in 3D. *arXiv preprint arXiv:2405.12419*, 2024. URL https://arxiv.org/abs/2405.12419.

Deyu Bo and Xinchao Wang. Point cloud dataset distillation. In *International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 4588–4601, 2025. URL https://proceedings.mlr.press/v267/bo25a.html.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9650–9660, 2021.

George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A. Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

George Cazenavette, Antonio Torralba, and Vincent Sitzmann. Dataset distillation for pre-trained self-supervised vision models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/5709163243753c9c9ab7f4b4e5a8766d-Abstract-Conference.html.

Hanting Chen, Yunhe Wang, Chang Xu, Zhaohui Yang, Chuanjian Liu, Boxin Shi, Chunjing Xu, Chao Xu, and Qi Tian. Data-free learning of student networks. In *IEEE/CVF International Conference on Computer Vision*, 2019. URL https://openaccess.thecvf.com/content_ICCV_2019/html/Chen_Data-Free_Learning_of_Student_Networks_ICCV_2019_paper.html.

Zhiyuan Fang, Jianfeng Wang, Lijuan Wang, Lei Zhang, Yezhou Yang, and Zicheng Liu. SEED: Self-supervised distillation for visual representation. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2101.04731.

Pedro Hermosilla, Christian Stippel, and Leon Sick. Masked scene modeling: Narrowing the gap between supervised and self-supervised learning in 3d scene understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14835–14844, 2025.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. URL https://arxiv.org/abs/1503.02531.

Siddharth Joshi, Jiayi Ni, and Baharan Mirzasoleiman. Dataset distillation via knowledge distillation: Towards efficient self-supervised pre-training of deep networks. In *International Conference on Learning Representations*, 2025.

Dongwook Kim, Jae-Young Yim, and Jae-Young Sim. Parameterization-based dataset distillation of 3D point clouds through learnable shape morphing. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Qe7dKZOtWM.

Karim Knaebel, Jonas Schult, Alexander Hermans, and Bastian Leibe. Point2Vec for self-supervised representation learning on point clouds. In *German Conference on Pattern Recognition*, 2023. URL https://arxiv.org/abs/2303.16570.

Dong Bok Lee, Seanie Lee, Joonho Ko, Kenji Kawaguchi, Juho Lee, and Sung Ju Hwang. Self-supervised dataset distillation for transfer learning. In *International Conference on Learning Representations*, 2024.

Jae-Hun Lee, Doyoung Yoon, ByeongMoon Ji, Kyungyul Kim, and Sangheum Hwang. Rethinking evaluation protocols of visual representations learned via self-supervised learning. *arXiv preprint arXiv:2304.03456*, 2023. URL https://arxiv.org/abs/2304.03456.

Remco F. Leijenaar and Hamidreza Kasaei. Asymmetric dual self-distillation for 3D self-supervised representation learning. *arXiv preprint arXiv:2506.21724*, 2025. URL https://arxiv.org/abs/2506.21724.

Mingzhuo Li, Guang Li, Linfeng Ye, Jiafeng Mao, Takahiro Ogawa, Konstantinos N. Plataniotis, and Miki Haseyama. Self-supervised representation-guided generative dataset distillation. *arXiv preprint arXiv:2608.03218*, 2026. URL https://arxiv.org/abs/2608.03218.

Haotian Liu, Mu Cai, and Yong Jae Lee. Masked discrimination for self-supervised learning on point clouds. In *European Conference on Computer Vision*, 2022.

Yang Liu, Chen Chen, Can Wang, Xulin King, and Mengyuan Liu. Regress before construct: Regress autoencoder for point cloud self-supervised learning. In *Proceedings of the 31st ACM International Conference on Multimedia*, 2023. doi: 10.1145/3581783.3612106. URL https://arxiv.org/abs/2310.03670.

Noel Loo, Ramin Hasani, Alexander Amini, and Daniela Rus. Efficient dataset distillation using random feature approximation. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://arxiv.org/abs/2210.12067.

Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual MLP framework. In *International Conference on Learning Representations*, 2022.

Timothy Chieu Nguyen, Zhourong Chen, and Jaehoon Lee. Dataset meta-learning from kernel ridge-regression. In *International Conference on Learning Representations*, 2021. URL https://research.google/pubs/dataset-meta-learning-from-kernel-ridge-regression/.

Yatian Pang, Wenxiao Wang, Francis E. H. Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *European Conference on Computer Vision*, pp. 604–621, 2022. doi: 10.1007/978-3-031-20086-1_35.

Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Park_Relational_Knowledge_Distillation_CVPR_2019_paper.html.

Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Abed Al Kader Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. PointNeXt: Revisiting PointNet++ with improved training and scaling strategies. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. FitNets: Hints for thin deep nets. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6550.

Ayumu Saito, Prachi Kudeshia, and Jiju Poovvancheri. Point-JEPA: A joint embedding predictive architecture for self-supervised learning on point cloud. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 7348–7357, 2025. URL https://openaccess.thecvf.com/content/WACV2025/html/Saito_Point-JEPA_A_Joint_Embedding_Predictive_Architecture_for_Self-Supervised_Learning_on_WACV_2025_paper.html.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkgpBJrtvS.

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 *IEEE/CVF International Conference on Computer Vision*, pp. 1588–1597, 2019.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A. Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph CNN for learning on point clouds. *ACM Transactions on Graphics*, 38(5), 2019.

Lintong Wei, Jian Lu, Haozhe Cheng, Jihua Zhu, and Kaibing Zhang. Point-SRA: Self-representation alignment for 3D representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 10539–10547, 2026. doi: 10.1609/aaai.v40i13.38026. URL https://ojs.aaai.org/index.php/AAAI/article/view/38026.

Max Welling. Herding dynamical weights to learn. In *International Conference on Machine Learning*, pp. 1121–1128, 2009. doi: 10.1145/1553374.1553517.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3D ShapeNets: A deep representation for volumetric shapes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1912–1920, 2015.

Saining Xie, Jiatao Gu, Demi Guo, Charles R. Qi, Leonidas J. Guibas, and Or Litany. PointContrast: Unsupervised pre-training for 3D point cloud understanding. In *European Conference on Computer Vision*, 2020. doi: 10.1007/978-3-030-58580-8_34. URL https://arxiv.org/abs/2007.10985.

Jae-Young Yim, Dongwook Kim, and Jae-Young Sim. Dataset distillation of 3D point clouds via distribution matching. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://arxiv.org/abs/2503.22154.

Hongxu Yin, Pavlo Molchanov, Jose M. Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K. Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via DeepInversion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. URL https://openaccess.thecvf.com/content_CVPR_2020/html/Yin_Dreaming_to_Distill_Data-Free_Knowledge_Transfer_via_DeepInversion_CVPR_2020_paper.html.

Jiahao You, Xu Han, Jinfeng Xu, and Xianzhi Li. Long-tailed 3D point cloud dataset distillation. *arXiv preprint arXiv:2607.26763*, 2026. URL https://arxiv.org/abs/2607.26763.

Sheng-Feng Yu, Jia-Jiun Yao, and Wei-Chen Chiu. Boost self-supervised dataset distillation via parameterization, predefined augmentation, and approximation. In *International Conference on Learning Representations*, 2025.

Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Chenming Zhu, Zhangyang Xiong, Tianyou Liang, Guanying Chen, Shuguang Cui, and Xiaoguang Han. MVImgNet: A large-scale dataset of multi-view images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9150–9161, 2023. URL https://arxiv.org/abs/2303.06042.

Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-BERT: Pre-training 3D point cloud transformers with masked point modeling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Yu_Point-BERT_Pre-Training_3D_Point_Cloud_Transformers_With_Masked_Point_Modeling_CVPR_2022_paper.html.

Hongliang Zhang, Xiaoqi An, Jiawei Lian, Lei Luo, and Jian Yang. CoMPR: Efficient point cloud dataset condensation via bidirectional matching and point recycling. *Pattern Recognition*, 172: 112494, 2026. doi: 10.1016/j.patcog.2025.112494. URL https://www.sciencedirect.com/science/article/abs/pii/S0031320325011574.

Renrui Zhang, Ziyu Guo, Rongyao Fang, Bin Zhao, Dong Wang, Yu Qiao, Hongsheng Li, and Peng Gao. Point-M2AE: Multi-scale masked autoencoders for hierarchical point cloud pre-training. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Wenxiao Zhang, Ziqi Wang, Li Xu, Xun Yang, and Jun Liu. Informative point cloud dataset extraction for classification via gradient-based points moving. In *ACM International Conference on Multimedia*, pp. 6384–6393, 2024. doi: 10.1145/3664647.3680767.

Zaiwei Zhang, Rohit Girdhar, Armand Joulin, and Ishan Misra. Self-supervised pretraining of 3D features on any point-cloud. In *IEEE/CVF International Conference on Computer Vision*, 2021. URL https://openaccess.thecvf.com/content/ICCV2021/html/Zhang_Self-Supervised_Pretraining_of_3D_Features_on_Any_Point-Cloud_ICCV_2021_paper.html.

Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023.

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *International Conference on Learning Representations*, 2021.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. iBOT: Image BERT pre-training with online tokenizer. In *International Conference on Learning Representations*, 2022a. URL https://arxiv.org/abs/2111.07832.

Yongchao Zhou, Ehsan Nezhadarya, and Jimmy Ba. Dataset distillation using neural feature regression. In *Advances in Neural Information Processing Systems*, volume 35, 2022b.

A Appendix

You may include other additional sections here.

B Implementation and evaluation protocol

B.1 Datasets and total budgets

Table 5 gives the actual splits used in this study. ModelNet40 uses the manually aligned mesh version; ScanObjectNN uses OBJ_BG; MVP100FULL denotes the fixed subset used throughout the experiments. Each object contains 1,024 XYZ points. The entire training split is available to the producer without semantic labels. The 10% bank is used only for frozen readout fitting or neighbor lookup.

Table 5: Dataset splits, labeled readout banks, and total object budgets. All objects contain 1,024 XYZ points.

Dataset	Train	Test	Labels@10	Classes	M
ModelNet40	9,843	2,468	984	40	40 / 200 / 400
MVP100FULL	29,288	12,553	2,929	100	100 / 500 / 1,000
ScanObjectNN	2,309	581	230	15	15 / 75 / 150

- **ScanObjectNN:** `https://hkust-vgd.github.io/scanobjectnn/`.
- **ModelNet40:** `https://github.com/lmb-freiburg/orion`.
- **MVPNet (MVImgNet):** `https://github.com/GAP-LAB-CUHK-SZ/MVImgNet`.

B.2 Teacher, targets, and producer settings

The reconstruction-pretrained PointNet teacher uses pointwise layers $3 \to 64 \to 128 \to 1024$ followed by global max pooling. Its pretraining reconstructs 256 sampled points after input-point removal, with keep fractions 0.65 for ModelNet40 and 0.70 for the other two datasets. This reconstruction objective is distinct from the consumer's representation-regression objective.

Normalized teacher features are centered and projected by PCA without whitening, then normalized again. Whole-object targets use a 128-dimensional projection of complete-object features; hidden-region targets use a 64-dimensional projection of isolated-region features. The corresponding PCA fits use N object features and $2N$ region features, respectively. Both target tables are stored as $[M, 2, 64]$ tensors, but whole-object normalization applies to the flattened vector, while regional vectors are normalized separately. Targets, the projector, and region memberships remain fixed during coordinate optimization.

All selected resources use uniform random-real initialization with seed 720913, two equal-sized regions with one hidden, and independently sampled real queries. Region membership is computed from the initial coordinates and then held fixed. The eight-entry cyclic mask bank repeats the two complementary visible subsets. Production performs 200 outer Adam updates with coordinate learning rate 0.01 and support/query batches of at most 32 objects. Query objects exclude those used for coordinate initialization. Table 6 records the dataset-specific producer settings.

Table 6: Selected producer configurations. T is the number of inner updates, not the number of consumer epochs. Whole and hidden targets supervise 128 and 64 target dimensions per update.

Dataset	M	Target	T	Inner optimizer / LR	State
ModelNet40	40 / 200 / 400	Whole	3	SGD, 0.01	Fresh
MVP100FULL	100 / 500 / 1,000	Whole	3	SGD, 0.01	Fresh
ScanObjectNN	15	Hidden	3	SGD, 0.01	Fresh
ScanObjectNN	75	Hidden	5	SGD, 0.001	Fresh
ScanObjectNN	150	Hidden	3	SGD, 0.001	Fresh

Fresh-SGD producers restart the encoder and predictor each outer iteration and reuse the sampled support batch within the unroll. The persistent ScanObjectNN $M = 75$ producer carries detached learner parameters and Adam moments between outer iterations, draws support batches with replacement at each inner step, and differentiates through only the five current updates. Its inner Adam uses $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, backbone weight decay 5×10^{-4}, and no predictor decay. The persistent producer's cosine horizon is 2,000 inner updates. This producer schedule is separate from the consumer schedule below.

B.3 Consumer training and frozen readouts

PointNet (Qi et al., 2017) is the default producer and consumer backbone. Every consumer starts from random initialization and is trained for 200 epochs. After pretraining, we discard the prediction head, freeze the encoder, and extract representations from complete objects. We report both a linear probe (LP) and weighted k-nearest-neighbor (kNN) readout, providing complementary parametric and non-parametric views of representation quality; recent 3D representation-evaluation protocols likewise use linear and nearest-neighbor probing (Hermosilla et al., 2025; Zhang et al., 2022; Saito et al., 2025; Lee et al., 2023). LP@10% and kNN@10% use the same fixed 10% labeled training bank as a low-label diagnostic, while LP@100% and kNN@100% with the full bank are reported in Appendix C. All methods share identical feature processing, labeled banks, and readout settings. Results are mean $\pm$ sample standard deviation over three consumers per resource.

For PointNet consumers, normalized encoder features are passed to a two-layer MLP prediction head with 256 hidden units and 128 outputs. Predictions are regressed to the fixed targets without output normalization; whole-object and hidden-region prediction supervise 128- and 64-dimensional targets, respectively. After pretraining, the prediction head is discarded, the encoder is frozen, and ℓ_2-normalized features are extracted from complete objects.

Following the linear-evaluation protocol of MKDT (Joshi et al., 2025), we fit an ℓ_2-regularized linear classifier using L-BFGS with regularization weight 10^{-3} and an initial 100-iteration budget. We extend the cap to 500 only when needed for convergence. Weighted kNN uses cosine similarity with $k = 20$ and temperature $\tau = 0.07$ (Caron et al., 2021). LP and kNN use the same bank at either 10% or 100% label availability, and all evaluation queries come from the official test split.

B.4 Real-subset selection and supervision

Random uses uniform within-class selection. Herding approximates class feature means, and K-center starts near each class mean and adds points by farthest-point coverage (Welling, 2009; Sener & Savarese, 2018). Herding and K-center use normalized 1,024-dimensional features from an independent, supervised PointNet selector. The selector trains on the full labeled training split for 200 epochs, with batch size 256, Adam learning rate 0.001, weight decay 0.0005, cosine decay to 0.0001. Its final checkpoint is used for selection, not consumer initialization.

Real-subset consumers use the same 200-epoch traversal schedule, optimizer settings, and frozen evaluation as predictive consumers. They train on complete objects with class-label cross-entropy. Table 7 summarizes the information and training conditions of the main comparison.

Table 7: Information access and consumer training. The supervised selector applies to Herding and K-center; Random uses class-wise uniform selection.

Condition	MPD-3D	Real subsets
Production / selection	Unlabeled source; SSL teacher	Class quotas; supervised selector
Consumer input	512 visible points	1,024 points
Consumer objective	Fixed-target regression	Class-label cross-entropy
Consumer schedule	200 epochs; no replacement	200 epochs; no replacement
Checkpoint	Final epoch	Final epoch
Readouts	Frozen LP and weighted kNN	Frozen LP and weighted kNN

C COMPLETE POINTNET READOUTS

Table 8 reports all four readouts for the resources in Table 1. The full-bank LP means favor MPD-3D in all nine settings. Full-bank kNN instead favors real subsets at all three ModelNet40 budgets and at ScanObjectNN $M = 15$. The difference between LP and kNN rankings shows that the measured advantage depends on the readout and labeled reference bank.

Table 8: Complete frozen readouts for the main resources. OA (%), mean ± sample SD over three consumers. These are the same resources and consumers as Table 1.

Dataset / M	Method	LP@10%	kNN@10%	LP@100%	kNN@100%
MN40 / 40	MPD-3D	57.56±0.35	66.00±0.36	57.56±0.37	77.66±0.24
MN40 / 40	Random	40.01±0.61	65.82±0.06	39.95±0.61	79.88±0.44
MN40 / 40	Herding	41.13±0.16	65.55±0.21	40.63±0.40	80.20±0.53
MN40 / 40	K-center	41.13±0.04	65.53±0.15	40.69±0.37	80.23±0.60
MN40 / 200	MPD-3D	63.86±0.84	69.34±0.60	64.75±0.93	80.63±0.39
MN40 / 200	Random	44.85±0.47	67.83±0.27	43.94±0.26	81.69±0.12
MN40 / 200	Herding	45.22±0.21	67.67±0.45	44.33±0.12	81.44±0.08
MN40 / 200	K-center	43.57±0.53	69.03±0.26	42.92±0.52	82.35±0.41
MN40 / 400	MPD-3D	65.71±0.39	71.04±0.22	66.95±0.76	81.32±0.08
MN40 / 400	Random	45.56±0.10	69.57±0.35	44.62±0.15	82.70±0.19
MN40 / 400	Herding	47.08±0.62	69.34±0.37	45.76±0.08	82.37±0.45
MN40 / 400	K-center	45.73±0.13	69.38±0.19	44.65±0.07	82.78±0.35
MVP100FULL / 100	MPD-3D	25.23±0.14	42.36±0.08	25.13±0.05	60.26±0.14
MVP100FULL / 100	Random	18.43±0.05	37.11±0.21	18.23±0.13	57.96±0.37
MVP100FULL / 100	Herding	18.59±0.12	37.36±0.11	18.36±0.11	57.40±0.37
MVP100FULL / 100	K-center	18.67±0.12	37.48±0.12	18.44±0.13	57.45±0.44
MVP100FULL / 500	MPD-3D	29.07±0.18	46.37±0.24	29.41±0.08	65.40±0.30
MVP100FULL / 500	Random	21.64±0.11	41.03±0.25	21.97±0.22	61.12±0.07
MVP100FULL / 500	Herding	21.81±0.04	41.51±0.16	22.16±0.14	61.91±0.15
MVP100FULL / 500	K-center	20.49±0.07	39.84±0.40	21.15±0.24	60.81±0.38
MVP100FULL / 1000	MPD-3D	29.81±0.05	46.83±0.16	30.32±0.07	65.99±0.17
MVP100FULL / 1000	Random	22.82±0.13	42.25±0.28	22.71±0.07	62.88±0.30
MVP100FULL / 1000	Herding	23.10±0.08	42.45±0.14	23.07±0.08	62.82±0.10
MVP100FULL / 1000	K-center	21.99±0.15	40.72±0.15	22.19±0.13	62.19±0.18
SONN / 15	MPD-3D	37.23±0.26	40.33±1.01	46.13±1.20	50.49±1.25
SONN / 15	Random	35.23±0.53	35.51±0.70	41.14±0.30	52.38±0.50
SONN / 15	Herding	35.57±0.78	35.46±0.46	41.08±1.40	51.92±1.54
SONN / 15	K-center	35.57±0.78	35.46±0.46	41.08±1.40	51.92±1.54
SONN / 75	MPD-3D	41.02±0.26	43.20±0.60	49.05±0.30	56.86±0.70
SONN / 75	Random	37.12±0.36	38.15±0.72	44.87±0.43	52.90±0.36
SONN / 75	Herding	37.46±1.22	38.55±0.96	43.78±0.40	52.84±0.96
SONN / 75	K-center	36.89±0.98	37.01±0.34	43.66±0.95	52.95±0.36
SONN / 150	MPD-3D	43.37±1.13	43.55±0.46	51.75±0.43	56.28±0.52
SONN / 150	Random	38.90±0.30	39.24±0.91	46.76±0.26	53.24±0.50

Continued on next page.

Table 8 continued.

Dataset / M	Method	LP@10%	kNN@10%	LP@100%	kNN@100%
SONN / 150	Herding	41.02±0.70	41.25±0.26	46.24±0.85	55.13±0.53
SONN / 150	K-center	38.55±0.00	38.10±0.60	45.09±0.91	53.59±0.95

D EXTENDED ACCURACY–COST ANALYSIS

Figures 3–5 show the complete budget-labeled accuracy–time comparisons. The real subsets cover PPC$\in \{1, 3, 5, 7, 10, 13, 16, 20, 30, 40, 50\}$, while MPD-3D uses its three produced budgets, PPC$\in \{1, 5, 10\}$. PPC denotes the total object budget divided by the benchmark class count, rather than class-wise allocation by our producer. Every consumer trains for 200 epochs with batch size 32 and the final incomplete batch retained, giving $200\lceil M/32 \rceil$ updates. The curves connect successive budgets, not checkpoints from one run.

Points show mean LP@10 accuracy and median consumer training time. Vertical bars span the observed minimum–maximum accuracy over the plotted consumer seeds; horizontal bars span the first and third quartiles of timing measurements. These are empirical ranges and interquartile ranges, not confidence intervals or standard deviations. The real-subset methods share recorded timing entries at PPC≤ 20 and have method-specific entries at PPC$= 30, 40, 50$; all accuracies are method-specific. Consumer timing excludes teacher pretraining, target construction, coordinate optimization, and frozen readout fitting.

The plots compare complete training pipelines: real-subset consumers use class-label cross-entropy, whereas MPD-3D uses masked representation regression. Several timing distributions are broad and median runtime is not monotonic in budget, so small runtime differences are not interpreted as reliable speed advantages.

Figure 3: ModelNet40 accuracy–time comparison over PPC$=$ 1–50. Points show mean LP@10 and median consumer time on a logarithmic axis. Bars show accuracy minimum–maximum ranges and timing interquartile ranges. The dashed reference marks MPD-3D at PPC$=$ 10.

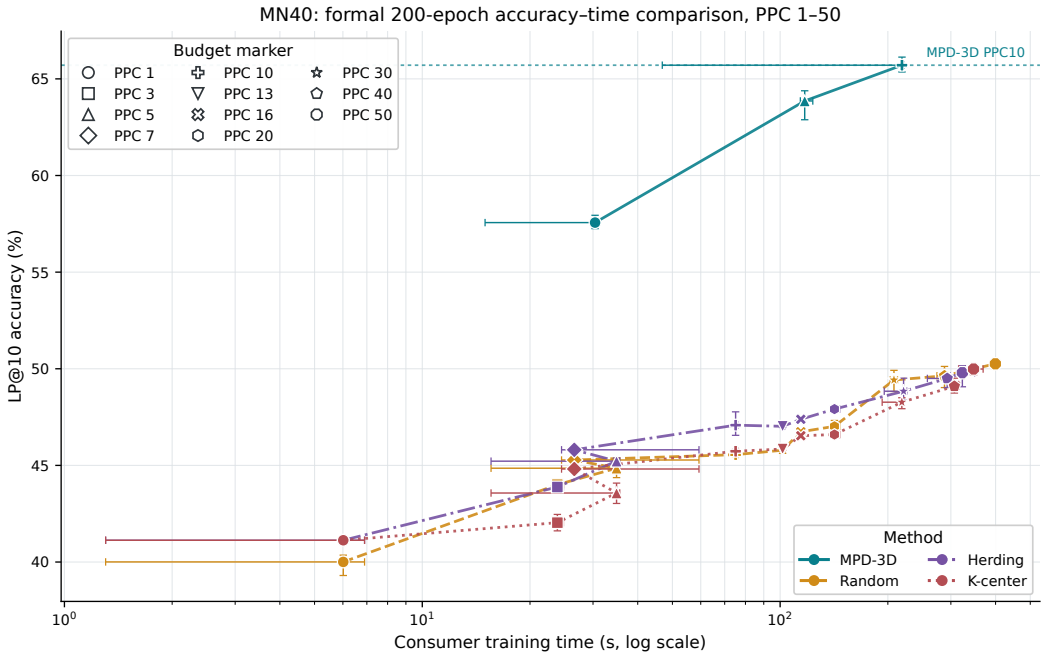


Figure 4: MVP100FULL accuracy–time comparison, using the same summaries as Figure 3. MPD-3D uses PPC= $1, 5, 10$; real-subset budgets extend to 50.

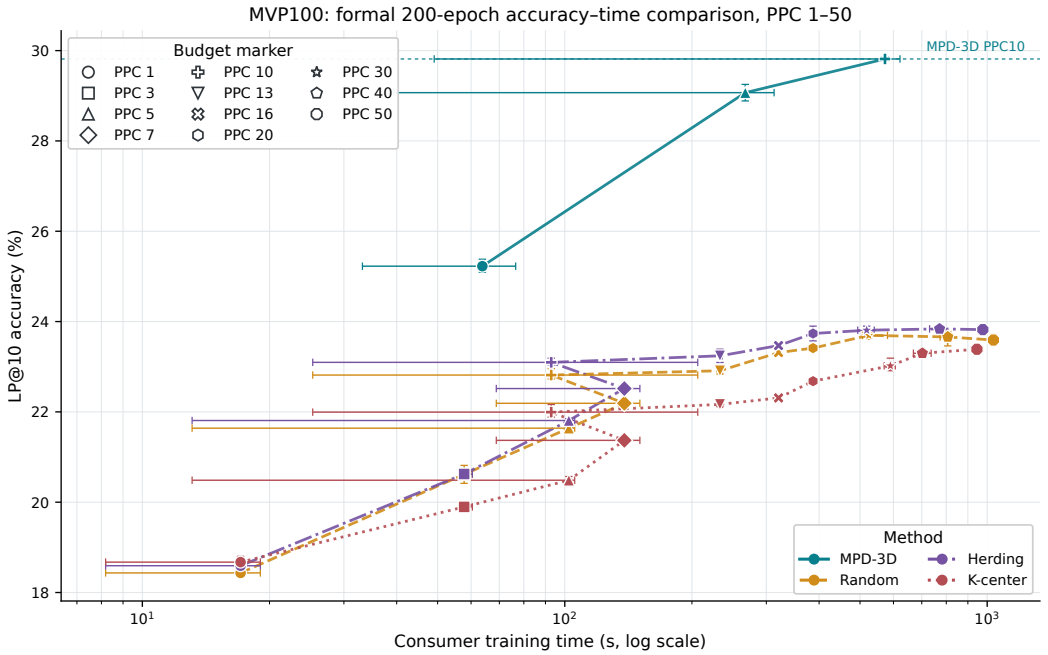
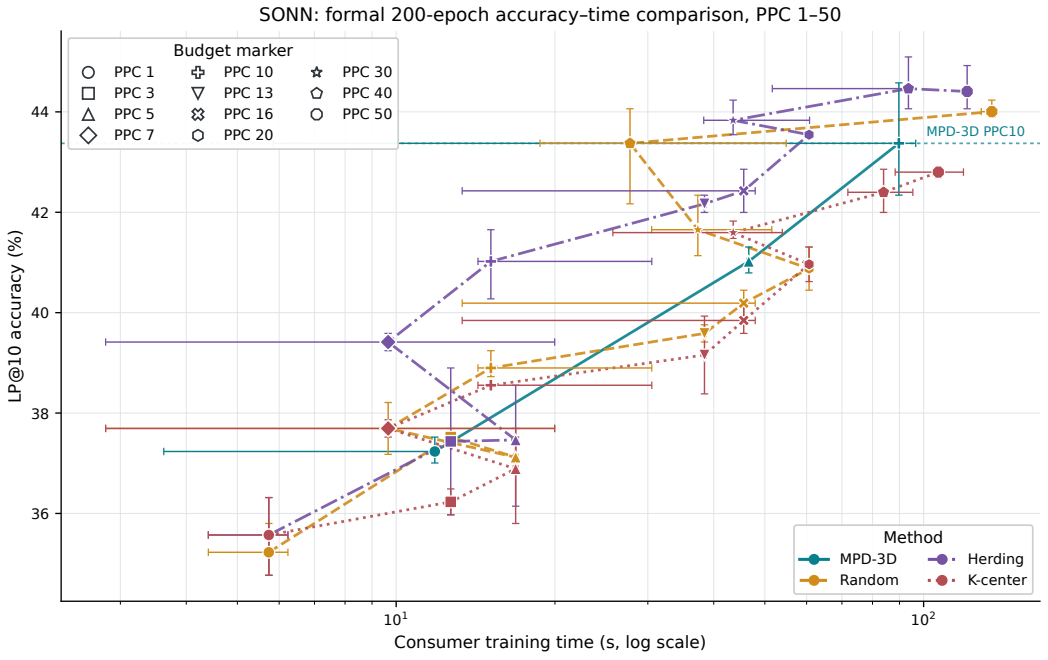


Figure 5: ScanObjectNN accuracy–time comparison. Larger real subsets can match or exceed the synthetic resources in LP@10 with lower median consumer time in some settings, illustrating the dataset dependence of the trade-off.



D.1 COMPLETE NUMERICAL RESULTS

Tables 9–11 give the numerical summaries for the expanded plots. Panel (a) reports mean LP@10 with $[\min, \max]$; panel (b) reports median consumer time with $[Q_1, Q_3]$. A dash denotes an unproduced synthetic budget. Bold marks the highest displayed mean accuracy at each budget, including ties at the reported precision.

Table 9: ModelNet40 accuracy and consumer cost over PPC$=$ 1–50. Accuracy is mean $[\min, \max]$ in %; time is median $[Q_1, Q_3]$ in seconds. All consumers train for 200 epochs.

PPC	MPD-3D	Random	Herding	K-center
(a) LP@10 accuracy (%)				
1	**57.56** [57.25, 57.94]	40.01 [39.30, 40.36]	41.13 [40.96, 41.29]	41.13 [41.09, 41.17]
3	–	**44.00** [43.76, 44.25]	43.88 [43.60, 44.17]	42.04 [41.61, 42.46]
5	**63.86** [62.88, 64.38]	44.85 [44.37, 45.30]	45.22 [44.98, 45.38]	43.57 [43.03, 44.08]
7	–	45.28 [45.18, 45.38]	**45.81** [45.79, 45.83]	44.81 [44.77, 44.85]
10	**65.71** [65.36, 66.13]	45.56 [45.46, 45.66]	47.08 [46.56, 47.77]	45.73 [45.58, 45.83]
13	–	45.79 [45.62, 45.95]	**47.02** [46.96, 47.08]	45.85 [45.79, 45.91]
16	–	46.74 [46.68, 46.80]	**47.39** [47.29, 47.49]	46.54 [46.39, 46.68]
20	–	47.02 [46.72, 47.33]	**47.91** [47.85, 47.97]	46.60 [46.56, 46.64]
30	–	**49.42** [48.82, 49.92]	48.84 [48.38, 49.51]	48.27 [47.93, 48.50]
40	–	**49.64** [49.03, 50.12]	49.50 [49.03, 49.80]	49.10 [48.74, 49.51]
50	–	**50.26** [50.00, 50.49]	49.80 [49.07, 50.16]	49.99 [49.68, 50.20]
(b) Consumer training time (s)				
1	30.40 [14.98, 30.82]	6.01 [1.30, 6.90]	6.01 [1.30, 6.90]	6.01 [1.30, 6.90]
3	–	23.85 [23.85, 24.81]	23.85 [23.85, 24.81]	23.85 [23.85, 24.81]
5	117.13 [113.88, 123.40]	34.90 [15.57, 35.63]	34.90 [15.57, 35.63]	34.90 [15.57, 35.63]
7	–	26.59 [24.50, 59.33]	26.59 [24.50, 59.33]	26.59 [24.50, 59.33]
10	219.03 [46.85, 222.30]	75.09 [74.18, 76.20]	75.09 [74.18, 76.20]	75.09 [74.18, 76.20]
13	–	101.61 [101.26, 102.92]	101.61 [101.26, 102.92]	101.61 [101.26, 102.92]
16	–	114.24 [113.05, 116.54]	114.24 [113.05, 116.54]	114.24 [113.05, 116.54]
20	–	141.85 [141.29, 146.76]	141.85 [141.29, 146.76]	141.85 [141.29, 146.76]
30	–	208.02 [202.96, 212.84]	221.30 [195.41, 225.99]	218.84 [192.81, 219.69]
40	–	287.93 [274.30, 290.48]	293.06 [257.93, 301.78]	306.86 [301.99, 316.53]
50	–	399.46 [399.14, 399.61]	323.00 [315.91, 349.53]	347.15 [315.28, 369.21]

Table 10: MVP100FULL accuracy and consumer cost over PPC$=$ 1–50. Accuracy is mean $[\min, \max]$ in %; time is median $[Q_1, Q_3]$ in seconds. All consumers train for 200 epochs.

PPC	MPD-3D	Random	Herding	K-center
(a) LP@10 accuracy (%)				
1	**25.23** [25.09, 25.38]	18.43 [18.39, 18.49]	18.59 [18.48, 18.72]	18.67 [18.55, 18.80]
3	–	**20.62** [20.42, 20.82]	**20.62** [20.62, 20.63]	19.90 [19.84, 19.95]
5	**29.07** [28.89, 29.25]	21.64 [21.52, 21.75]	21.81 [21.76, 21.85]	20.49 [20.43, 20.57]
7	–	22.19 [22.10, 22.28]	**22.52** [22.44, 22.59]	21.37 [21.32, 21.42]
10	**29.81** [29.78, 29.87]	22.82 [22.73, 22.96]	23.10 [23.04, 23.19]	21.99 [21.88, 22.16]
13	–	22.91 [22.84, 22.98]	**23.24** [23.09, 23.39]	22.17 [22.11, 22.22]
16	–	23.31 [23.30, 23.33]	**23.47** [23.46, 23.48]	22.31 [22.30, 22.32]
20	–	23.41 [23.39, 23.44]	**23.74** [23.57, 23.90]	22.68 [22.65, 22.71]
30	–	23.69 [23.61, 23.75]	**23.81** [23.70, 23.90]	23.02 [22.91, 23.19]
40	–	23.66 [23.46, 23.78]	**23.84** [23.72, 23.91]	23.30 [23.21, 23.36]
50	–	23.59 [23.48, 23.72]	**23.82** [23.73, 23.87]	23.39 [23.33, 23.42]
(b) Consumer training time (s)				
1	63.81 [33.19, 76.52]	17.08 [8.19, 19.03]	17.08 [8.19, 19.03]	17.08 [8.19, 19.03]
3	–	57.78 [56.14, 60.43]	57.78 [56.14, 60.43]	57.78 [56.14, 60.43]
5	267.48 [39.40, 313.00]	102.24 [13.11, 105.56]	102.24 [13.11, 105.56]	102.24 [13.11, 105.56]
7	–	138.35 [68.85, 150.60]	138.35 [68.85, 150.60]	138.35 [68.85, 150.60]
10	572.88 [49.07, 621.64]	92.88 [25.31, 206.44]	92.88 [25.31, 206.44]	92.88 [25.31, 206.44]
13	–	233.02 [232.62, 235.91]	233.02 [232.62, 235.91]	233.02 [232.62, 235.91]
16	–	320.72 [318.52, 324.53]	320.72 [318.52, 324.53]	320.72 [318.52, 324.53]
20	–	387.05 [382.07, 389.40]	387.05 [382.07, 389.40]	387.05 [382.07, 389.40]
30	–	524.58 [506.80, 580.77]	518.40 [493.83, 540.26]	589.95 [570.24, 605.98]
40	–	806.47 [773.67, 806.77]	771.29 [730.48, 778.77]	702.31 [668.84, 736.82]
50	–	1034.52 [1008.96, 1045.92]	976.48 [973.67, 977.87]	945.58 [942.84, 970.71]

Table 11: ScanObjectNN accuracy and consumer cost over PPC$=$ 1–50. Accuracy is mean $[\min, \max]$ in %; time is median $[Q_1, Q_3]$ in seconds. All consumers train for 200 epochs.

PPC	MPD-3D	Random	Herding	K-center
(a) LP@10 accuracy (%)				
1	**37.23** [37.01, 37.52]	35.23 [34.77, 35.80]	35.57 [34.77, 36.32]	35.57 [34.77, 36.32]
3	–	**37.52** [37.35, 37.69]	37.44 [35.97, 38.90]	36.23 [35.97, 36.49]
5	**41.02** [40.79, 41.31]	37.12 [36.83, 37.52]	37.46 [36.14, 38.55]	36.89 [35.80, 37.69]
7	–	37.69 [37.18, 38.21]	**39.41** [39.24, 39.59]	37.69 [37.52, 37.87]
10	**43.37** [42.34, 44.58]	38.90 [38.73, 39.24]	41.02 [40.28, 41.65]	38.55 [38.55, 38.55]
13	–	39.59 [39.41, 39.76]	**42.17** [42.00, 42.34]	39.16 [38.38, 39.93]
16	–	40.19 [39.93, 40.45]	**42.43** [42.00, 42.86]	39.85 [39.59, 40.10]
20	–	40.88 [40.45, 41.31]	**43.55** [43.55, 43.55]	40.96 [40.62, 41.31]
30	–	41.65 [41.14, 42.34]	**43.83** [43.55, 44.23]	41.59 [41.48, 41.82]
40	–	43.37 [42.17, 44.06]	**44.46** [44.06, 45.09]	42.40 [42.00, 42.86]
50	–	44.00 [43.89, 44.23]	**44.41** [44.06, 44.92]	42.80 [42.69, 42.86]
(b) Consumer training time (s)				
1	11.84 [3.63, 12.00]	5.73 [4.40, 6.23]	5.73 [4.40, 6.23]	5.73 [4.40, 6.23]
3	–	12.69 [12.58, 12.87]	12.69 [12.58, 12.87]	12.69 [12.58, 12.87]
5	46.61 [46.25, 47.01]	16.85 [16.68, 17.01]	16.85 [16.68, 17.01]	16.85 [16.68, 17.01]
7	–	9.66 [2.81, 19.98]	9.66 [2.81, 19.98]	9.66 [2.81, 19.98]
10	89.75 [5.85, 96.60]	15.12 [14.29, 30.53]	15.12 [14.29, 30.53]	15.12 [14.29, 30.53]
13	–	38.42 [38.27, 38.71]	38.42 [38.27, 38.71]	38.42 [38.27, 38.71]
16	–	45.60 [13.34, 47.94]	45.60 [13.34, 47.94]	45.60 [13.34, 47.94]
20	–	60.65 [60.13, 61.66]	60.65 [60.13, 61.66]	60.65 [60.13, 61.66]
30	–	37.32 [30.51, 51.54]	43.54 [38.29, 60.78]	43.55 [25.75, 53.97]
40	–	27.75 [18.74, 54.88]	93.58 [51.65, 94.71]	83.96 [71.84, 95.41]
50	–	134.66 [128.59, 136.81]	120.92 [120.55, 122.31]	106.64 [88.32, 119.00]

E CONSTRUCTION COSTS AND DELIVERED STORAGE

Figure 8 separates the measured target-cache allocation, coordinate optimization, consumer training, and LP@10 readout costs. The cache is shared across PPC$= 1, 5, 10$, with one third charged to each resource; this allocation applies to the three-budget use case. Teacher pretraining is excluded, so these totals are partial construction-and-use costs rather than end-to-end costs. The consumer-only measurements in the preceding section and these stage totals have different cost boundaries.

Figure 6: Measured stages for constructing and using the selected resources. Cache cost is allocated equally across PPC$= 1, 5, 10$. Teacher pretraining is excluded; consumer training and LP@10 readout are shown separately.

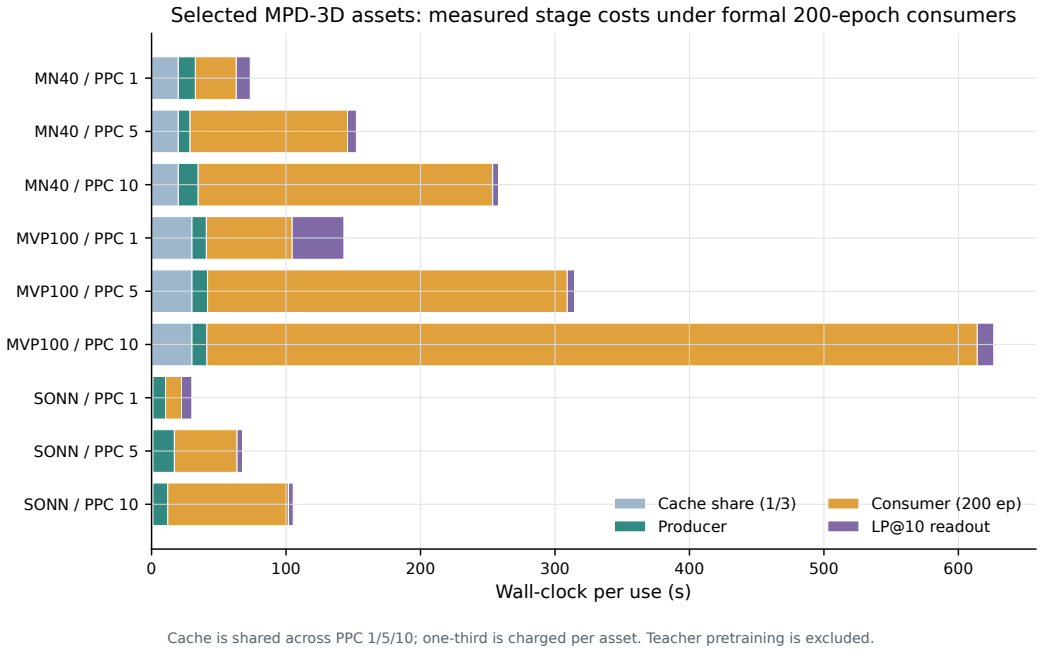


Figure 9 relates LP@10 accuracy to the reported partial one-use cost and complete serialized resource size. The resource includes coordinates, fixed representation targets, region indices, and metadata, but no teacher weights. Table 12 summarizes the sizes at the precision shown in the figure. These measurements characterize construction and reuse costs, but without matched teacher-pretraining timings they do not establish a full-cost break-even point for repeated use.

Figure 7: Quality, partial one-use cost, and delivered storage of the selected resources. Annotations give the PPC budget and serialized size. The cost axis includes the measured stages in Figure 8, excluding teacher pretraining.

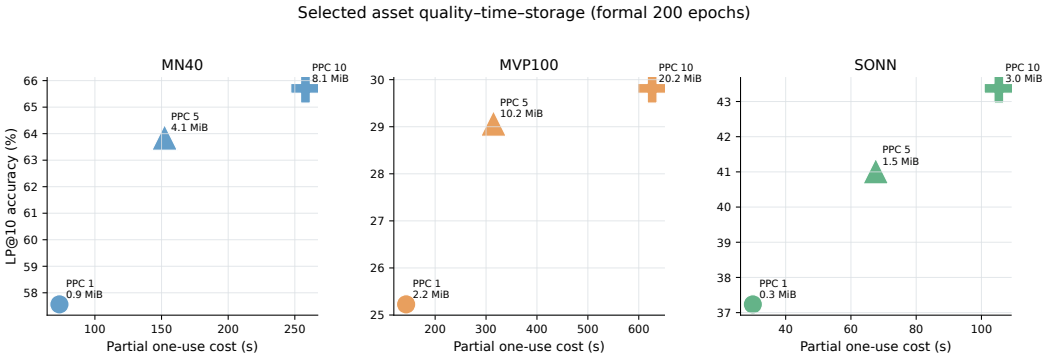


Table 12: Complete resource sizes (MiB), reported at the one-decimal precision used in Figure 9. MiB denotes 2^{20} bytes.

Dataset	PPC= 1	PPC= 5	PPC= 10
ModelNet40	0.9	4.1	8.1
MVP100FULL	2.2	10.2	20.2
ScanObjectNN	0.3	1.5	3.0

F Construction costs and delivered storage

Figure 8 separates the measured target-cache allocation, coordinate optimization, consumer training, and LP@10 readout costs. The cache is shared across PPC= $1, 5, 10$, with one third charged to each resource; this allocation applies to the three-budget use case. Teacher pretraining is excluded, so the totals are partial construction-and-use costs rather than end-to-end costs. Figure 9 complements this breakdown with LP accuracy, the reported partial one-use cost, and serialized storage. The complete resources occupy 0.324–20.250 MiB, including coordinates, fixed targets, region indices, and metadata, but no teacher weights.

G Prediction-task and producer-configuration studies

All resources in this section are consumed under the same 200-epoch protocol, with three consumer seeds. The studies compare archived producer configurations. A selected-resource row provides context; when its target scope, learning rate, or inner state differs from a study variant, that comparison changes more than one factor.

G.1 Whole-object and hidden-region targets

Table 13 expands the target comparison in Table 3. The base recipe uses two regions with one hidden, fresh SGD with learning rate 0.01 for three inner updates, and independent real queries. Changing target scope also changes the teacher input, PCA fit, normalization scope, and active supervision dimension. The ScanObjectNN selected resources are listed separately from their base-task counterparts. In particular, the selected $M = 15$ resource and the base hidden-region resource have distinct results despite sharing the nominal target scope.

Figure 8: Measured stages for constructing and using the selected resources. The cache cost is allocated equally across PPC$= 1, 5, 10$. Teacher pretraining is excluded, while consumer training and LP@10 readout are shown separately.

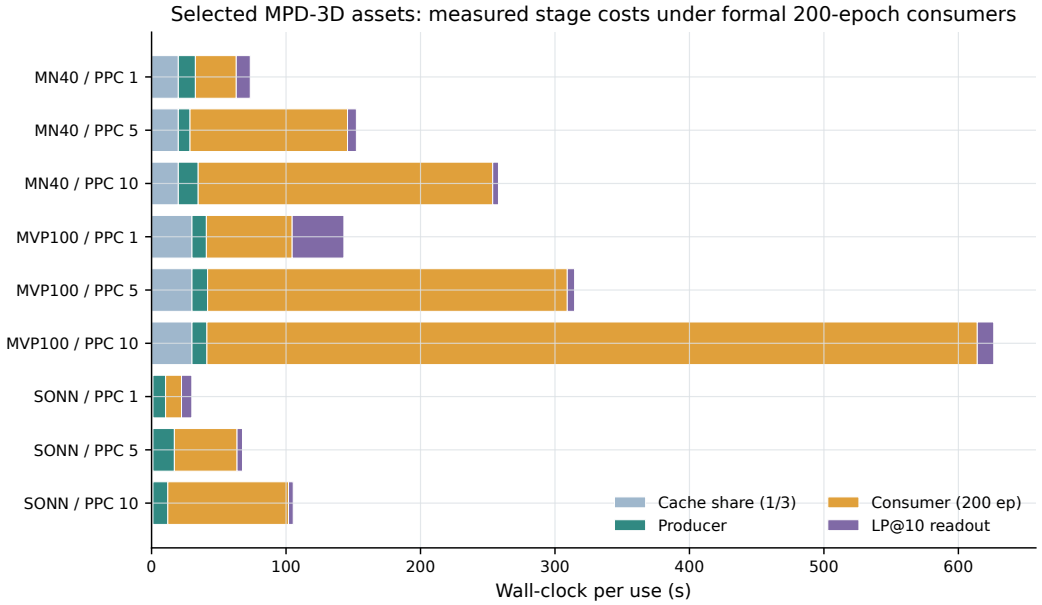


Figure 9: Quality, partial one-use cost, and delivered storage of the selected resources. Annotations give the PPC budget and serialized size.

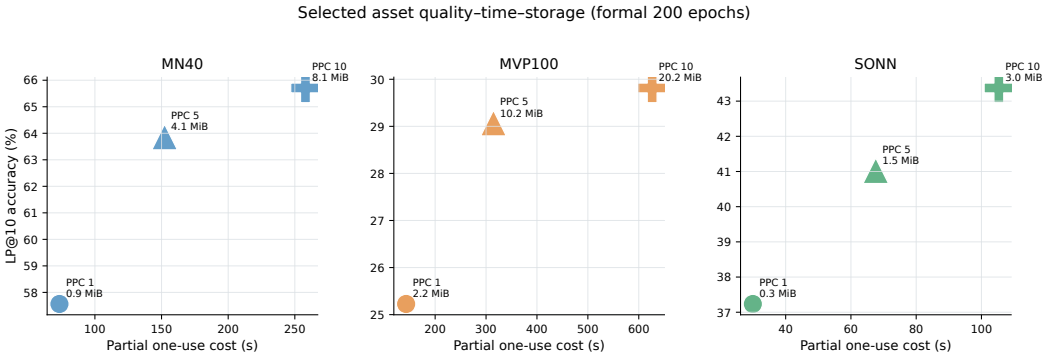


Table 13: Target configurations and selected-resource references. Four frozen readouts (%), mean ± SD over three consumers.

Dataset / M	Resource	LP@10%	kNN@10%	LP@100%	kNN@100%
MN40 / 40	Whole (selected)	57.56±0.35	66.00±0.36	57.56±0.37	77.66±0.24
	Hidden, base	56.98±0.33	64.74±0.48	57.16±0.76	76.08±0.28
MN40 / 200	Whole (selected)	63.86±0.84	69.34±0.60	64.75±0.93	80.63±0.39
	Hidden, base	62.79±0.71	68.61±0.25	64.11±0.89	80.05±0.41
MN40 / 400	Whole (selected)	65.71±0.39	71.04±0.22	66.95±0.76	81.32±0.08
	Hidden, base	65.06±0.75	71.00±0.66	66.10±0.44	81.87±0.51
MVP / 100	Whole (selected)	25.23±0.14	42.36±0.08	25.13±0.05	60.26±0.14
	Hidden, base	24.30±0.10	40.56±0.18	24.32±0.25	57.79±0.09

Continued on next page.

Table 13 continued.

Dataset / M	Resource	LP@10%	kNN@10%	LP@100%	kNN@100%
MVP / 500	Whole (selected)	29.07±0.18	46.37±0.24	29.41±0.08	65.40±0.30
	Hidden, base	28.34±0.33	45.98±0.24	28.70±0.39	64.71±0.18
MVP / 1000	Whole (selected)	29.81±0.05	46.83±0.16	30.32±0.07	65.99±0.17
	Hidden, base	29.84±0.06	46.82±0.22	30.14±0.08	65.84±0.16
SONN / 15	Whole, base	35.46±1.30	39.53±0.55	45.50±0.60	50.43±1.53
	Hidden (selected)	37.23±0.26	40.33±1.01	46.13±1.20	50.49±1.25
SONN / 75	Whole, base	40.22±0.87	42.05±0.36	49.17±0.43	54.62±0.72
	Hidden (selected)	41.02±0.26	43.20±0.60	49.05±0.30	56.86±0.70
SONN / 150	Whole, base	43.43±0.26	42.80±0.53	52.90±1.17	55.59±0.17
	Hidden (selected)	43.37±1.13	43.55±0.46	51.75±0.43	56.28±0.52

G.2 Inner update scope

Table 14 reports the head-only producer variants. Only the predictor is updated inside their differentiable producer unroll, while gradients to synthetic coordinates remain enabled. Downstream consumers still train both encoder and predictor. These producer-study variants use hidden-region targets; the selected ModelNet40 and MVP100FULL resources instead use whole-object targets. Thus, selected-minus-head-only differences also include target changes. The base hidden-region resources in Table 13 provide the relevant task context.

Table 14: Head-only inner-update variants, evaluated with full encoder–predictor consumer training. OA (%), mean ± SD over three consumers.

Dataset / M	LP@10%	kNN@10%	LP@100%	kNN@100%
MN40 / 40	57.21±0.55	64.07±0.12	57.77±0.69	76.49±0.38
MN40 / 200	63.25±0.61	68.73±0.16	64.52±0.55	80.31±0.59
MN40 / 400	64.99±0.98	70.91±0.49	66.00±0.55	81.36±0.24
MVP / 100	24.56±0.28	40.24±0.81	24.49±0.28	57.39±0.86
MVP / 500	28.09±0.28	45.86±0.23	28.28±0.30	64.43±0.17
MVP / 1000	29.59±0.25	46.76±0.19	30.11±0.24	65.64±0.09
SONN / 15	36.89±0.70	39.99±0.36	45.04±0.95	50.66±0.50
SONN / 75	39.93±0.62	41.54±0.36	49.11±1.79	53.99±0.81
SONN / 150	41.48±0.96	41.48±1.05	52.84±0.75	55.71±0.43

G.3 Coordinate initialization

Table 15 compares real-data and Gaussian-noise initializations in the archived producer studies. Herding and K-center name initialization procedures here, not the supervised real-subset baselines in the main table: the coordinates are subsequently optimized. The ModelNet40 study uses hidden-region variants, whereas its selected resource uses whole-object prediction. The ScanObjectNN selected resource also differs in inner learning rate from the initialization-study variants. The Gaussian-noise variants have much lower LP accuracy than the Herding- and K-center-initialized variants on both datasets. The selected-reference differences are configuration comparisons rather than isolated initialization effects.

Table 15: Initialization/configuration study. All entries are optimized resources, not unmodified real subsets. OA (%), mean ± SD over three consumers.

Dataset / M	Initialization	LP@10%	kNN@10%	LP@100%	kNN@100%
MN40 / 40	Selected resource	57.56±0.35	66.00±0.36	57.56±0.37	77.66±0.24
	Herding initialization	55.98±0.71	63.78±0.58	55.32±1.07	76.46±0.27
	K-center initialization	56.01±1.13	63.84±0.27	55.67±1.12	76.42±0.76
	Gaussian noise	19.42±1.74	57.29±1.45	20.34±2.24	73.16±1.07
SONN / 150	Selected resource	43.37±1.13	43.55±0.46	51.75±0.43	56.28±0.52

Continued on next page.

Table 15 continued.

Dataset / M	Initialization	LP@10%	kNN@10%	LP@100%	kNN@100%
	Herding initialization	41.08±1.65	42.74±0.65	50.55±0.88	55.54±0.26
	K-center initialization	43.72±0.69	43.95±1.12	52.78±1.34	57.54±0.60
	Gaussian noise	24.50±1.84	31.84±1.82	23.92±1.80	42.17±1.34

G.4 Real-query organization and geometric masking

The query variants in Table 16 use hidden-region prediction; the selected ModelNet40 reference uses whole-object prediction. The stratified query policies group fixed descriptors and alter how real queries are drawn. The reported query variants are close in LP@10%, ranging from 62.70% to 63.07%.

The masking study changes the number of regions and hidden slots. With 512 visible points, the two-region and eight-region variants reach 64.91% and 62.55% LP@10%, respectively. Four regions with three hidden leave only 256 visible points and give 58.06%. Both the input information and target structure can change, so these rows evaluate masking configurations rather than a fixed-information comparison of region granularity.

Table 16: Query and masking configurations. OA (%), mean ± SD over three consumers. The selected reference additionally differs in target scope.

Configuration	LP@10%	kNN@10%	LP@100%	kNN@100%
MN40, $M = 200$: query policy				
Selected resource	63.86±0.84	69.34±0.60	64.75±0.93	80.63±0.39
Global queue	62.99±0.64	69.08±0.56	64.38±0.67	80.06±0.51
IID	62.98±0.46	68.85±0.27	64.30±0.62	80.12±0.47
Stratified, empirical	62.70±0.59	68.81±0.47	63.94±0.56	80.01±0.80
Stratified, square-root mixture	63.07±0.50	68.95±0.63	64.74±0.56	80.46±0.96
MN40, $M = 400$: masking				
Selected resource	65.71±0.39	71.04±0.22	66.95±0.76	81.32±0.08
$R = 2$, hide 1 (512 visible)	64.91±0.57	71.07±0.57	66.15±0.52	81.74±0.28
$R = 8$, hide 4 (512 visible)	62.55±0.65	69.03±0.51	63.26±0.19	80.40±0.12
$R = 4$, hide 3 (256 visible)	58.06±0.36	68.75±0.10	57.96±0.46	80.59±0.18
$R = 4$, hide 1 (768 visible)	64.32±0.28	71.27±0.60	65.69±0.31	82.13±0.18

G.5 Inner learning rate, state, and unroll length

Table 17 groups the inner-optimizer studies. On MVP100FULL $M = 100$, the two hidden-region learning rates have equal LP@10% means, but the smaller rate gives higher kNN@10%. For ScanObjectNN $M = 75$, three-step persistent Adam has higher LP@10% and kNN@10% than three-step reset Adam, while LP@100% favors the reset variant. The selected five-step resource also changes within-unroll support sampling, so its difference is not a pure persistence effect. For the ModelNet40 $M = 40$ hidden-region study, increasing the unroll from one to five updates lowers all four reported readouts.

Table 17: Inner-optimizer configuration studies. OA (%), mean ± SD over three consumers. Selected resources are contextual references, not necessarily single-factor controls.

Configuration	LP@10%	kNN@10%	LP@100%	kNN@100%
MVP100FULL, $M = 100$: inner learning rate				
Selected resource	25.23±0.14	42.36±0.08	25.13±0.05	60.26±0.14
Hidden, SGD 0.01	24.41±0.03	40.20±0.34	24.48±0.11	57.58±0.42
Hidden, SGD 0.001	24.41±0.38	41.64±0.60	24.73±0.34	58.88±0.65
SONN, $M = 75$: learner state				
Selected resource	41.02±0.26	43.20±0.60	49.05±0.30	56.86±0.70

Continued on next page.

Table 17 continued.

Configuration	LP@10%	kNN@10%	LP@100%	kNN@100%
Persistent Adam, $T = 3$	40.62±0.52	43.14±0.85	47.27±1.21	56.11±0.52
Reset Adam, $T = 3$	39.13±0.95	41.82±0.60	49.86±1.29	54.04±0.60
MN40, $M = 40$: unroll length				
Selected resource	57.56±0.35	66.00±0.36	57.56±0.37	77.66±0.24
Hidden, $T = 1$	55.66±1.60	64.18±0.90	54.97±1.62	75.84±0.83
Hidden, $T = 5$	54.24±0.71	62.13±0.70	53.05±0.65	74.72±0.63

H GEOMETRY REFINEMENT UNDER FIXED PREDICTIVE SUPERVISION

A predictive resource contains both its input geometry and the supervision used to train a consumer. To condition on that supervision, we compare the optimized resource with its original real support X_0, keeping source-object identities, teacher targets, region memberships, masks, visible-point counts, BatchNorm policy, consumer initialization, and the 200-epoch schedule fixed. Only the coordinates are replaced. Table 18 reports the mean paired difference between optimized and initial geometry.

Coordinate refinement gives positive LP@10% differences in four of nine settings, LP@100% in six, kNN@10% in three, and kNN@100% in two. MVP100FULL at $M = 1000$ improves all four readouts, whereas $M = 100$ decreases all four. The conditioned comparison therefore identifies budget- and readout-dependent refinement effects, alongside the useful training signal already supplied by real support with fixed predictive targets. It does not identify the contribution of the targets alone, which would require changing the supervision while controlling the remaining resource and training conditions.

Table 18: Incremental coordinate effect with fixed predictive supervision. Entries are paired mean differences (percentage points), optimized coordinates minus X_0, over the same three consumer seeds. Absolute optimized scores are in Table 8.

Dataset / M	ΔLP@10%	ΔkNN@10%	ΔLP@100%	ΔkNN@100%
MN40 / 40	−1.22	−0.41	−1.26	−0.42
MN40 / 200	+0.34	−0.77	+0.42	−0.61
MN40 / 400	+0.80	+0.15	+1.40	−0.30
MVP / 100	−1.11	−2.26	−1.23	−2.80
MVP / 500	+0.69	−0.17	+0.79	−0.08
MVP / 1000	+1.66	+0.14	+1.76	+0.45
SONN / 15	−0.69	+0.92	+0.06	−2.58
SONN / 75	−1.09	−0.29	−0.06	+1.32
SONN / 150	−1.78	−0.23	+0.98	−1.49

I COMPLETE CROSS-ARCHITECTURE CONSUMPTION

Each of the nine selected resources is used unchanged by seven non-PointNet consumers, yielding 63 combinations with three seeds each. The backbones are DGCNN (Wang et al., 2019), PointMLP and its Elite variant (Ma et al., 2022), PointNeXt (Qian et al., 2022), and the Point-MAE encoder (Pang et al., 2022). Only the consumer backbone changes; coordinates, targets, region memberships, and masks remain fixed. Every consumer uses the 200-epoch predictive objective, followed by the same frozen readouts. Point-MAE refers to its encoder with the target-regression head, rather than its original reconstruction objective. Tables 19–21 retain every measured architecture, including low means and large standard deviations.

DGCNN improves with resource size on ModelNet40 and MVP100FULL. Other responses are less uniform: the Point-MAE encoder reaches 42.64% LP@10% on ModelNet40 at $M = 200$, but 18.88% on ScanObjectNN at $M = 75$.

Architecture-matched real-subset results are not part of this evaluation, so these tables describe consumption outcomes rather than architecture-specific distillation gains.

Table 19: ModelNet40 cross-architecture consumption. OA (%), mean ± SD over three consumers. Each budget uses its selected PointNet-produced resource.

M	Consumer	LP@10%	kNN@10%	LP@100%	kNN@100%
40	DGCNN	47.34±0.65	59.09±0.21	47.30±0.60	66.86±0.40
40	PointMLP	26.99±1.66	33.86±2.33	26.35±0.94	42.17±1.20
40	PointMLP-Elite	30.81±0.46	34.43±1.08	31.43±1.01	43.76±0.70
40	PointNeXt	26.38±2.22	39.01±3.65	26.82±3.60	48.99±4.56
40	Point-MAE encoder	23.43±4.44	33.55±8.63	23.57±3.84	43.67±10.29
200	DGCNN	56.43±0.79	65.64±0.71	57.04±0.80	74.65±0.35
200	PointMLP	37.47±2.77	45.95±1.59	37.51±3.18	57.35±0.87
200	PointMLP-Elite	40.02±0.60	45.57±4.09	41.09±0.77	56.28±3.35
200	PointNeXt	38.65±2.04	55.02±0.67	38.18±1.90	66.29±0.63
200	Point-MAE encoder	42.64±1.68	60.49±0.20	42.69±1.77	71.93±0.89
400	DGCNN	61.71±0.53	68.35±0.84	62.02±0.17	77.15±0.42
400	PointMLP	45.54±0.25	55.36±0.89	45.56±1.17	67.76±0.45
400	PointMLP-Elite	44.69±0.56	52.23±0.47	44.98±0.51	64.17±0.82
400	PointNeXt	42.04±2.60	58.41±1.53	41.40±2.51	70.52±1.40
400	Point-MAE encoder	48.80±0.70	62.83±1.50	48.76±1.12	75.34±0.74

Table 20: MVP100FULL cross-architecture consumption. OA (%), mean ± SD over three consumers. Each budget uses its selected PointNet-produced resource.

M	Consumer	LP@10%	kNN@10%	LP@100%	kNN@100%
100	DGCNN	22.33±0.49	32.16±0.37	22.87±0.45	44.13±0.65
100	PointMLP	10.96±0.93	12.60±1.36	10.99±0.92	16.76±2.78
100	PointMLP-Elite	13.53±0.49	13.24±0.16	13.87±0.68	19.19±0.23
100	PointNeXt	9.60±0.70	19.64±1.28	9.61±0.62	27.61±1.23
100	Point-MAE encoder	9.27±2.05	14.22±3.71	9.33±2.04	21.23±4.44
500	DGCNN	25.96±0.19	40.11±0.09	25.82±0.31	55.78±0.20
500	PointMLP	18.80±0.19	23.10±1.00	18.48±0.40	33.03±1.52
500	PointMLP-Elite	17.80±0.11	21.29±1.54	17.76±0.38	30.06±2.58
500	PointNeXt	19.08±0.13	35.01±0.41	18.96±0.15	52.07±2.53
500	Point-MAE encoder	20.78±0.05	35.62±0.25	21.04±0.35	52.41±0.73
1000	DGCNN	26.18±0.11	41.77±0.17	25.93±0.28	57.87±0.39
1000	PointMLP	21.67±1.32	30.65±1.07	21.47±1.10	43.61±1.37
1000	PointMLP-Elite	20.66±0.87	27.97±1.37	20.44±0.77	40.40±1.98
1000	PointNeXt	19.23±0.58	35.72±1.08	19.34±0.65	55.45±1.80
1000	Point-MAE encoder	21.72±0.26	37.12±0.37	21.88±0.27	54.52±0.27

Table 21: ScanObjectNN cross-architecture consumption. OA (%), mean ± SD over three consumers. Each budget uses its selected PointNet-produced resource.

M	Consumer	LP@10%	kNN@10%	LP@100%	kNN@100%
15	DGCNN	30.75±1.70	32.59±0.87	37.52±0.46	41.88±1.11
15	PointMLP	24.56±2.59	24.90±0.87	33.68±1.38	29.32±1.22
15	PointMLP-Elite	19.33±1.56	18.88±1.34	28.92±0.62	25.76±1.39
15	PointNeXt	24.44±0.62	21.80±0.40	24.33±0.60	33.45±1.38
15	Point-MAE encoder	14.80±2.38	33.05±4.50	14.40±1.69	46.18±3.48
75	DGCNN	38.67±1.01	37.18±1.47	43.66±0.81	49.00±2.18
75	PointMLP	24.10±1.55	21.40±1.01	32.53±3.32	27.42±2.12
75	PointMLP-Elite	21.06±0.95	19.22±1.89	28.46±2.08	25.13±0.46
75	PointNeXt	30.58±2.18	31.55±4.72	35.86±1.70	40.85±2.76
75	Point-MAE encoder	18.88±6.80	21.00±5.57	23.01±7.84	34.77±5.27
150	DGCNN	36.66±1.13	38.55±0.46	43.09±1.01	49.57±1.30

Continued on next page.

Table 21 continued.

M	Consumer	LP@10%	kNN@10%	LP@100%	kNN@100%
150	PointMLP	25.01 ± 0.70	20.14 ± 0.79	33.28 ± 1.56	29.26 ± 1.95
150	PointMLP-Elite	21.17 ± 1.70	19.74 ± 1.50	28.80 ± 1.48	26.97 ± 0.36
150	PointNeXt	30.46 ± 0.96	28.63 ± 0.70	33.73 ± 0.89	37.92 ± 0.99
150	Point-MAE encoder	13.65 ± 2.76	18.93 ± 6.41	16.47 ± 3.87	29.20 ± 10.05

I.1 Accounting for repeated use

Consumer timing excludes teacher pretraining, target construction, coordinate optimization, and frozen readout fitting. For K independently trained consumers, total pretraining cost would include

$$T_{\text{total}}(K) = T_{\text{teacher}} + T_{\text{targets}} + T_{\text{production}} + KT_{\text{consumer}}. \tag{8}$$

Teacher pretraining lacks a timing record matched to the current checkpoint, so the available measurements do not establish a full-cost break-even point. The target-cache construction can be shared across the three resource budgets; charging one third to each resource is an allocation for that three-budget use case, not the cost of constructing one standalone resource. The tables vary object budget at a fixed 200-epoch endpoint and are not learning curves over training duration.

J Teacher reference

Table 22 reports direct linear-probe evaluation of the frozen SSL teacher, pretrained on the full unlabeled source collection. LP@10 and LP@100 use 10% and 100% of the labeled training bank, respectively. This is a reference for checkpoint reuse, not a consumer trained from scratch on M objects. The teacher scores are not treated as an upper bound on separately trained consumers.

Table 22: Direct frozen-teacher LP accuracy (%). The teacher is evaluated without compact-resource consumer training.

Dataset	LP@10	LP@100
ModelNet40	37.44	37.03
MVP100FULL	17.63	18.08
ScanObjectNN	37.69	45.78