D^3S^2: Diffusion-Guided Dataset Distillation for Semantic Segmentation

Wenjie Zheng , Haoji Hu*, **Jiali Lu , Xingze Zou , Jing Wang**
Zhejiang University
{wenjie_zheng, haoji_hu}@zju.edu.cn

Abstract

Dataset distillation (DD) aims to compress large-scale datasets into compact synthetic sets while preserving training efficacy. However, existing studies mainly focus on image classification, leaving dense prediction tasks such as semantic segmentation largely underexplored. In this work, we identify three key challenges for segmentation DD: (*i*) long-tailed class imbalance, (*ii*) the need for strict pixel-wise alignment between images and dense labels, and (*iii*) the high computational cost of optimizing high-resolution data with complex models. To address these challenges, we propose **D^3S^2**, a **D**iffusion-guided **D**ataset **D**istillation framework for **S**emantic **S**egmentation. Our method adopts a two-stage design. In **Class-Balanced Mask Selection**, we construct a representative mask set via a greedy strategy that prioritizes underrepresented classes. In **Diffusion-Guided Image Synthesis**, we employ a pretrained layout-to-image diffusion model to generate images conditioned on the selected masks, naturally ensuring spatial alignment. To further enhance the training utility of synthesized data, we introduce guided diffusion sampling with two complementary objectives: a segmentation-consistency loss for pixel-level alignment, and a class-wise feature matching loss for aligning per-class feature statistics across layers. Extensive experiments demonstrate the superiority of D^3S^2. Notably, at an extremely compression rate of 1%, our method achieves 24.99% and 35.49% mIoU on ADE20K and COCO-Stuff with Mask2Former (Swin-S), outperforming random selection by 9.34% and 5.70%, respectively. Our code is available at https://github.com/zwj084/D3S2.

1 Introduction

Dataset Distillation (DD) [44, 52, 13] aims to compress large-scale datasets into compact synthetic sets while preserving training efficacy. By reducing storage and computational costs, DD has emerged as a promising paradigm for efficient model training under resource constraints. Recent advances have achieved strong performance in image classification via data matching techniques (see Figure 1 (a)), including Gradient Matching [56, 54, 19], Distribution Matching [43, 55, 57], and Training Trajectory Matching [4, 10]. More recently, decoupled methods [50, 32, 34] (see Figure 1 (b)) disentangle model training from data optimization, avoiding costly bi-level optimization and improving scalability.

Despite these advances, DD remains largely limited to classification, leaving dense prediction tasks such as semantic segmentation underexplored. Extending DD to segmentation is non-trivial due to three key challenges: (*i*) **Long-tailed Class Imbalance.** Unlike relatively balanced classification datasets [11, 20], segmentation datasets [59, 2] exhibit highly skewed pixel distributions (Figure 1 (d)), making it difficult to allocate a limited distillation budget effectively. (*ii*) **Dense Label Coupling with Locality Requirement.** In contrast to classification-based DD methods that operate on global image-level supervision, existing paradigms (e.g., SRe2L [50]) mainly rely on aligning global feature

*Corresponding author.

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

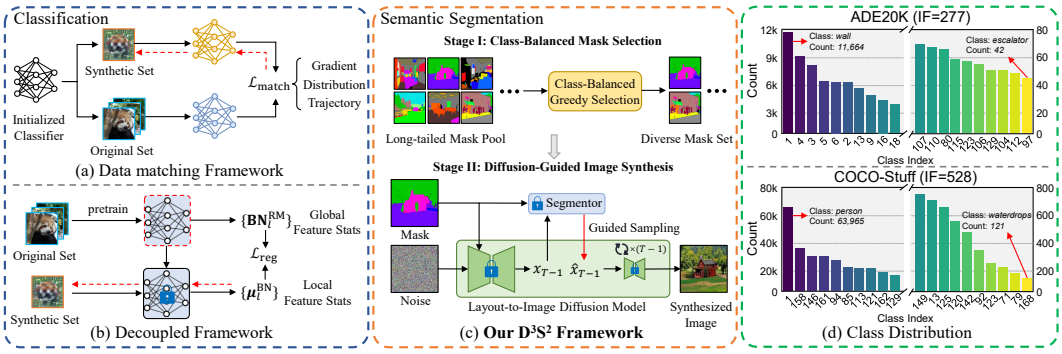


Figure 1: (a) Data-matching methods; (b) Decoupled methods; (c) Our proposed D^{3}S^2, the first DD framework tailored for semantic segmentation, which generates distilled data via a layout-to-image diffusion model; (d) Class distributions of ADE20K [59] and COCO-Stuff [2].

statistics such as BatchNorm statistics, which inherently discard spatial structure. However, semantic segmentation requires pixel-wise supervision, where object semantics are tightly coupled with spatial layouts. This introduces a fundamental gap: DD methods designed for classification optimize images at a global representation level, while segmentation demands fine-grained, region-aware consistency between synthesized images and masks, making purely global feature alignment insufficient. (*iii*) **High Computational Overhead.** The increased complexity of segmentation models and high-resolution supervision makes per-sample optimization (typically 1k–4k iterations) prohibitively expensive, limiting the direct applicability of existing DD paradigms.

To address these challenges, we propose **D^{3}S^2**, a **D**iffusion-guided **D**ataset **D**istillation framework tailored for **S**emantic **S**egmentation. As shown in Figure 1 (c), our method follows a two-stage design. **Stage I: Class-Balanced Mask Selection.** To combat long-tailed class imbalance, we construct a compact yet representative mask set. Specifically, we introduce a class-balanced greedy selection strategy that iteratively selects samples using a scoring function prioritizing underrepresented classes, promoting class diversity and coverage under a limited distillation budget. **Stage II: Diffusion-Guided Image Synthesis.** To enforce strict pixel-wise alignment while avoiding prohibitive per-sample optimization, we build on recent advances in Generative Dataset Distillation [5, 42, 14, 37, 7, 23, 6] and exploit the strong generative priors of diffusion models [36, 15, 35, 29]. We employ a layout-to-image diffusion model [48] pretrained on the target dataset to synthesize images conditioned on selected masks, naturally enforcing pixel-level alignment. To further improve the training utility of the synthesized data, we introduce guided diffusion sampling [1, 51] with two complementary objectives. A segmentation-consistency loss enforces pixel-level alignment with target masks via a pretrained segmentation model, while a class-wise feature matching loss aligns per-class feature statistics across layers. Unlike conventional global BatchNorm statistics matching [50], we compute per-class feature statistics from the full dataset and align them with those extracted from the generated samples. This design captures class-specific characteristics and promotes localized feature refinement. Notably, all guidance objectives are integrated directly into the diffusion sampling process. This enables highly time-efficient generation of high-quality distilled data within a small number of sampling steps, requiring less than one minute to synthesize a single image.

We conduct extensive experiments on ADE20K [59] and COCO-Stuff [2]. Across a wide range of distillation ratios, D^{3}S^2 consistently achieves top-tier performance on multiple segmentation architectures, including Mask2Former [9] and SegFormer [46] with various backbones, demonstrating strong robustness and generalization. For instance, the full COCO-Stuff training set contains 118,287 images, and training Mask2Former (Swin-S) typically requires about two days to reach 48.19% mIoU. In contrast, with only 0.25% data (295 distilled images), D^{3}S^2 finishes training in roughly 3 hours and achieves 26.87% mIoU, outperforming random selection by 7.99%. Although a performance gap remains compared to full-data training, reflecting the challenge of DD for semantic segmentation, these results suggest that diffusion-guided DD for semantic segmentation is a promising direction, with substantial room for improvement. Our main contributions are summarized as follows:

- To the best of our knowledge, we propose **D^{3}S^2, the first dataset distillation framework tailored for semantic segmentation.** It adopts a two-stage design, where *Class-Balanced Mask Selection* constructs a representative mask set, and *Diffusion-Guided Image Synthesis* efficiently generates high-quality distilled data via guided diffusion sampling.

- We introduce two complementary guidance objectives within the diffusion sampling process: a segmentation-consistency loss for pixel-level alignment and a class-wise feature matching loss for class-specific feature fidelity through localized statistics alignment.
- Extensive experiments demonstrate the superiority of D^3S^2. Notably, at a 1% compression ratio, it achieves 24.99% and 35.49% mIoU on ADE20K and COCO-Stuff, respectively, using Mask2Former (Swin-S), surpassing random selection by 9.34% and 5.70%.

2 Related Works

Dataset Distillation. Early DD methods formulate the problem as a meta-learning task [44, 26, 21, 60], optimizing a compact synthetic set such that models trained on it generalize well to the original dataset. Subsequent works move toward a more scalable data matching paradigm, aligning training dynamics between synthetic and real data. Representative approaches include Gradient Matching [56, 54, 19], Distribution Matching [43, 55, 57], and Training Trajectory Matching [4, 10], which respectively match gradients, feature distributions, and long-range optimization trajectories.

Decoupled Methods. Decoupled approaches improve scalability by separating model training from data optimization. SRe2L [50] proposes a Squeeze-Recover-Relabel pipeline, which first squeezes dataset statistics into a model, then recovers synthetic data via BatchNorm statistics matching, and finally refines them with soft labels. Follow-up works [32, 39, 33] extend statistical matching to broader architectures and further enhance sample quality and diversity. Despite their success, these methods are mainly designed for image classification. Inspired by SRe2L, we argue that dense pixel-wise information in semantic segmentation can also be encoded into a segmentor and subsequently recovered. However, extending decoupled methods to segmentation is non-trivial. First, pixel-wise optimization becomes computationally expensive for high-resolution dense prediction. Second, BN statistics rely on globally averaged features and thus fail to capture class-specific semantics. While prior work such as LPLD [45] has explored class-wise statistical matching for classification, applying this to segmentation requires fundamentally different treatment: in classification, class-wise statistics are computed from single-class images with well-defined support; in segmentation, multi-class scenes with variable region sizes demand careful handling to avoid conflating genuine appearance variation with spatial artifacts. To address these issues, we replace explicit pixel optimization with guided diffusion sampling [1, 51], enabling efficient data recovery while preserving pixel-level alignment with masks. In addition, we introduce a class-wise feature matching objective that aligns per-class statistics, providing more discriminative supervision than global BN matching.

Diffusion-based Dataset Distillation. Recent works [14, 37, 7, 30, 41, 61, 23, 38, 6] leverage diffusion models to synthesize realistic and representative datasets, alleviating the limitations of direct pixel-space optimization. Several methods incorporate guided diffusion sampling to improve data quality and training effectiveness. For example, IGD [7] adopts influence-guided sampling, MGD^3 [30] introduces mode guidance to enhance diversity, and LGD [6] employs learnability-aware guidance to generate complementary samples. Despite their promising performance, these methods are primarily designed for image classification and do not account for the structured pixel-level supervision required by semantic segmentation.

Layout-to-Image Diffusion. Diffusion models [36, 15, 35, 29] have achieved remarkable success in image synthesis and can be extended to conditional generation with structural guidance such as semantic layouts. ControlNet [53] introduces additional branches for condition injection, while T2I-Adapter [24] employs lightweight adapters with minimal architectural changes. In this work, we adopt the Freestyle [48], which enables precise layout-conditioned generation without additional modules, making it a lightweight and efficient choice for our framework.

3 Preliminaries

Dataset Distillation for Semantic Segmentation. DD aims to compress a large real dataset $\mathcal{T}$ into a compact synthetic dataset $\mathcal{S}$. We extend the standard DD formulation to semantic segmentation, where each image is associated with dense, pixel-wise annotations. The real dataset $\mathcal{T}$ with N labeled images is defined as $\mathcal{T} = \{(\boldsymbol{x}_i, \boldsymbol{m}_i)\}_{i=1}^{N}$, where $\boldsymbol{x}_i \in \mathbb{R}^{H_i \times W_i \times 3}$ is the input image and $\boldsymbol{m}_i \in \mathbb{R}^{H_i \times W_i}$ is the corresponding segmentation mask, with $\boldsymbol{m}_i(u, v) \in \{0, 1, \ldots, K-1\}$ and K denoting the number the classes. The distilled dataset $\mathcal{S}$ is formulated as $\mathcal{S} = \{(\hat{\boldsymbol{x}}_j, \hat{\boldsymbol{m}}_j)\}_{j=1}^{|\texttt{IPD}|}$,

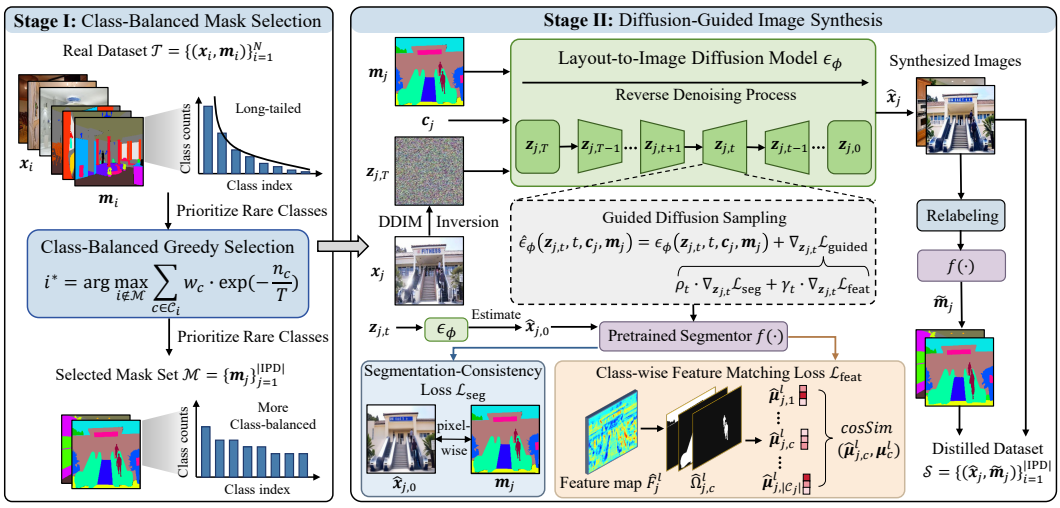


Figure 2: Overview of the D^{3}S^2. In the first stage, we construct a compact mask set that captures long-tailed semantic distributions. In the second stage, we generate layout-aligned images via a layout-to-image diffusion model, further enhanced by guided sampling with segmentation-consistency and class-wise feature matching losses to produce high-quality distilled data.

where $\hat{\boldsymbol{x}}_j$ and $\hat{\boldsymbol{m}}_j$ denote the synthesized images and masks, respectively, and $|\mathtt{IPD}| \ll N$ represents *images per dataset*, reflecting the compression ratio [18]. The goal of DD is to ensure that a model with weights $\theta_{\mathcal{S}}$ trained on $\mathcal{S}$ achieves performance comparable to a model trained on $\mathcal{T}$ with weights $\theta_{\mathcal{T}}$. Formally, the performance gap is bounded by:

$$sup\{|\mathcal{L}_{\theta_{\mathcal{T}}} - \mathcal{L}_{\theta_{\mathcal{S}}}|\}_{(\boldsymbol{x}_v, \boldsymbol{m}_v) \in \mathcal{T}'} \leq \epsilon_{DD}, \tag{1}$$

where $\mathcal{L}$ denotes the loss function, and $(\boldsymbol{x}_v, \boldsymbol{m}_v) \in \mathcal{T}'$ is some test or val set associated with $\mathcal{T}$.

Guided Diffusion Sampling. We adopt latent diffusion models [29], where images are encoded into latent representations via an autoencoder $D(E(\cdot))$ and generated through an iterative denoising process [15, 35]. Given a noisy latent $\boldsymbol{z}_t$, DDIM [35] predicts the clean sample as:

$$\hat{\boldsymbol{z}}_{0|t} = \frac{1}{\sqrt{\alpha_t}} \left(\boldsymbol{z}_t - \sqrt{1 - \alpha_t} \boldsymbol{\epsilon}_\phi(\boldsymbol{z}_t, t) \right), \tag{2}$$

where $\boldsymbol{\epsilon}_\phi$ denotes the predicted noise. Guided diffusion sampling further steers the denoising trajectory using task-specific objectives [1, 51]. More details are provided in Appendix A.

4 Method

To address the challenges of dataset distillation for semantic segmentation, we propose **D^{3}S^2**, a diffusion-guided framework with a two-stage pipeline, as illustrated in Figure 2. Specifically, our framework consists of *Class-Balanced Mask Selection* and *Diffusion-Guided Image Synthesis*, which together enable efficient generation of high-quality distilled data.

4.1 Stage I: Class-Balanced Mask Selection

Given the long-tailed class distribution in segmentation datasets (see Figure 1 (d)), a key challenge is to construct a compact yet representative mask set under a limited distillation budget. A naive strategy that enforces strict class-wise balance (e.g., selecting one sample per class) is suboptimal, as segmentation images typically contain multiple co-occurring classes. Such rigid allocation ignores contextual diversity and inter-class relationships. This issue is illustrated in Figure 3, which compares different mask selection strategies on ADE20K under a 1% budget. Random selection severely underrepresents tail classes, with several rare categories never appearing in the selected subset. Rigid uniform selection slightly improves tail coverage by iteratively selecting samples for each class, but still suffers from severe imbalance, yielding an Imbalance Factor (IF) of 129.

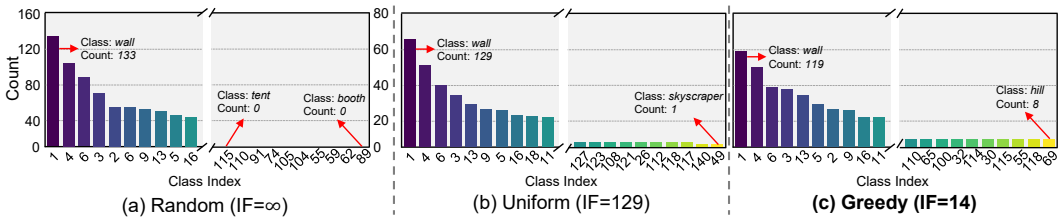


Figure 3: Class distribution comparison under 1% compression ratio on ADE20K.

To address this, we propose a *Class-Balanced Greedy Selection* strategy that progressively builds a mask subset by jointly considering class rarity and coverage. Let $\mathcal{T} = \{(\boldsymbol{x}_i, \boldsymbol{m}_i)\}_{i=1}^{N}$ denote the full dataset, and $\mathcal{C}_i \subset \{0, 1, \ldots, K-1\}$ be the set of classes present in mask $\boldsymbol{m}_i$. Each class c is assigned a weight w_c inversely proportional to its frequency, reflecting its importance under long-tailed distributions. At each step, we select a mask that maximizes the following score:

$$i^* = \arg\max_{i \notin \mathcal{M}} \sum_{c \in \mathcal{C}_i} w_c \cdot \exp\left(-\frac{n_c}{T}\right), \tag{3}$$

where $\mathcal{M}$ is the current selected set, n_c denotes the coverage count of class c, and T is a temperature hyperparameter. The exponential term encourages selecting masks containing underrepresented classes while gradually reducing redundancy as coverage increases.

The selection process is repeated until the distillation budget is reached, yielding the final mask subset $\mathcal{M} = \{\boldsymbol{m}_j\}_{j=1}^{|\texttt{IPD}|}$. Compared to rigid per-class sampling, our strategy better exploits the multi-class nature of segmentation images by favoring masks with diverse and complementary semantic information. As shown in Figure 3 (c), our method substantially improves tail coverage, where even the rarest classes appear at least 8 times and the IF is reduced to 14. However, due to inherent class co-occurrence, achieving a perfectly uniform class distribution is infeasible; instead, our method balances diversity and coverage. The full class distribution statistics are provided in Appendix B. Detailed complexity and theoretical approximation analysis are provided in Appendix D.1.1.

4.2 Stage II: Diffusion-Guided Image Synthesis

Layout-to-Image Diffusion Model. Given the selected mask set from Stage I, we employ a layout-to-image diffusion model to synthesize images aligned with the semantic layouts. Specifically, we adopt Freestyle [48], which enables precise mask-conditioned generation based on a *Stable Diffusion* [29] backbone without introducing additional modules. To reduce domain discrepancy, the model is pretrained on the target dataset $\mathcal{T}$. During sampling, the noise predictor is formulated as $\boldsymbol{\epsilon}_\phi(\boldsymbol{z}_{j,t}, t, \boldsymbol{c}_j, \boldsymbol{m}_j)$, where $\boldsymbol{m}_j$ denotes the input mask and $\boldsymbol{c}_j$ is a textual condition constructed by concatenating the class names present in $\boldsymbol{m}_j$. This design ensures semantically coherent images that are well aligned with the given layouts.

DDIM Inversion. Despite effective layout conditioning, we observe that small objects (e.g., *barrel*, *bench*) are occasionally missing in the generated images (see Figure 4), likely due to being dominated by large-scale structures during the early denoising. To alleviate this issue, we employ DDIM inversion [35] to initialize the sampling from a more informative latent. Given a real image corresponding to mask $\boldsymbol{m}_j$, we invert it into $\boldsymbol{z}_{j,T}$ via the deterministic DDIM trajectory:

$$\boldsymbol{z}_{j,t+1} = \sqrt{\alpha_{t+1}}\hat{\boldsymbol{z}}_{j,0|t} + \sqrt{1-\alpha_{t+1}}\boldsymbol{\epsilon}_\phi(\boldsymbol{z}_{j,t}, t, \boldsymbol{c}_j, \boldsymbol{m}_j). \tag{4}$$

This initialization provides stronger structural priors, improving the preservation of small objects.

4.2.1 Guided Diffusion Sampling

While the diffusion model enforces spatial alignment with masks, it does not explicitly guarantee that the synthesized samples are informative for training. To further enhance the training utility of the distilled data, we introduce guided diffusion sampling with two complementary objectives.

Segmentation-Consistency Loss. We further enforce pixel-wise alignment using a pretrained segmentation model $f(\cdot)$ (readily available from public checkpoints[2]) as guidance. At each step, we

[2] `https://github.com/open-mmlab/mmsegmentation`

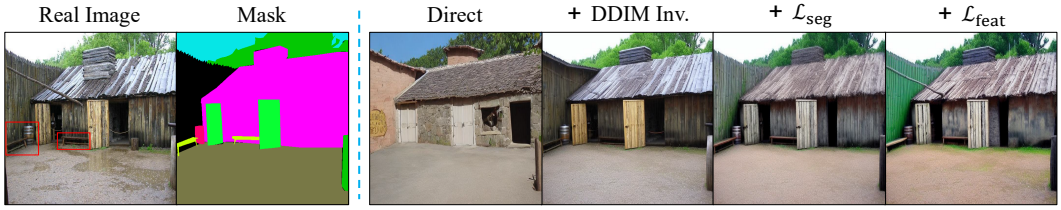


Figure 4: Effect of DDIM Inversion and guided diffusion losses on layout-conditioned image synthesis. We compare generated results under different settings: direct mask-conditioned sampling, with DDIM inversion, and with additional guidance signals using $\mathcal{L}_{\text{seg}}$ and $\mathcal{L}_{\text{feat}}$. Notably, DDIM inversion improves the preservation of small objects (e.g., *barrel* and *bench*), while the proposed guidance further enhances structural completeness and semantic fidelity.

estimate the clean latent $\hat{\boldsymbol{z}}_{j,0|t}$ via Equation (2) and decode it into the image space as $\hat{\boldsymbol{x}}_{j,0} = D(\hat{\boldsymbol{z}}_{j,0|t})$. The $\hat{\boldsymbol{x}}_{j,0}$ and $\boldsymbol{m}_j$ are then fed into f to compute a standard segmentation loss $\mathcal{L}_{\text{seg}}\big(f(\hat{\boldsymbol{x}}_{j,0}), \boldsymbol{m}_j\big)$ [9, 46], which encourages consistency between generated images and target masks throughout the denoising process, enhancing class-specific representations (e.g., see the *bench* in Figure 4).

Class-wise Feature Matching Loss. Global BatchNorm statistics [50] are insufficient for capturing class-specific features due to spatial averaging. To overcome this, we propose a class-wise feature matching loss that enforces alignment at a finer granularity. We denote by $\boldsymbol{F}_i^l \in \mathbb{R}^{H_l \times W_l \times C_l}$ the feature map at the l-th stage of f for image $\boldsymbol{x}_i$. For each class c, we resize the mask $\boldsymbol{m}_i$ to match the spatial resolution of $\boldsymbol{F}_i^l$, and compute the channel-wise mean over its spatial region:

$$\boldsymbol{\mu}_{i,c}^l = \frac{1}{|\Omega_{i,c}^l|} \sum_{(u,v)\in\Omega_{i,c}^l} \boldsymbol{F}_i^l(u,v), \tag{5}$$

where $\Omega_{i,c}^l$ denotes the set of spatial locations belonging to class c and then average over the dataset:

$$\boldsymbol{\mu}_c^l = \frac{1}{N_c} \sum_{i:c\in\mathcal{C}_i} \boldsymbol{\mu}_{i,c}^l. \tag{6}$$

During sampling, we compute $\hat{\boldsymbol{\mu}}_{j,c}^l$ from $\hat{\boldsymbol{x}}_{j,0}$ in the same manner and define $\mathcal{L}_{\text{feat}}$:

$$\mathcal{L}_{\text{feat}} = \sum_{c\in\mathcal{C}_j} \sum_l \left(1 - cosSim(\hat{\boldsymbol{\mu}}_{j,c}^l, \boldsymbol{\mu}_c^l)\right), \tag{7}$$

which encourages class-specific feature consistency across multiple network stages and $cosSim(\cdot,\cdot)$ denotes the cosine similarity.

Design Choice: First-Order Statistics Only. While SRe2L matches both mean and variance for classification, we align only per-class channel means. This reflects a key difference: in classification, BatchNorm statistics aggregate over single-class images, yielding well-defined per-class distributions; in segmentation, statistics are pooled over mask regions within multi-class scenes where spatial support varies drastically. The dataset-level variance thus conflates genuine intra-class variation with nuisance factors (region size, context). Matching variance would inject artificial heterogeneity that conflicts with $\mathcal{L}_{\text{seg}}$ and the canonical appearance goal. We validate this empirically in Table 5.

We incorporate both objectives into the diffusion process by modifying the noise prediction as:

$$\hat{\boldsymbol{\epsilon}}_\phi(\boldsymbol{z}_{j,t}, t, \boldsymbol{c}_j, \boldsymbol{m}_j) = \boldsymbol{\epsilon}_\phi(\boldsymbol{z}_{j,t}, t, \boldsymbol{c}_j, \boldsymbol{m}_j) \; + \rho_t \cdot \nabla_{\boldsymbol{z}_{j,t}} \mathcal{L}_{\text{seg}} + \gamma_t \cdot \nabla_{\boldsymbol{z}_{j,t}} \mathcal{L}_{\text{feat}}, \tag{8}$$

where ρ_t and γ_t control the guidance strength and are normalized to stabilize optimization [7] as:

$$\rho_t = \lambda_{\text{seg}} \cdot \sqrt{1-\alpha_t} \frac{||\boldsymbol{\epsilon}_\phi(\boldsymbol{z}_{j,t}, t, \boldsymbol{c}_j, \boldsymbol{m}_j)||}{||\nabla_{\boldsymbol{z}_{j,t}} \mathcal{L}_{\text{seg}}||}, \quad \gamma_t = \lambda_{\text{feat}} \cdot \sqrt{1-\alpha_t} \frac{||\boldsymbol{\epsilon}_\phi(\boldsymbol{z}_{j,t}, t, \boldsymbol{c}_j, \boldsymbol{m}_j)||}{||\nabla_{\boldsymbol{z}_{j,t}} \mathcal{L}_{\text{feat}}||}. \tag{9}$$

Overall, $\mathcal{L}_{\text{seg}}$ enforces pixel-level consistency, while $\mathcal{L}_{\text{feat}}$ promotes discriminative, class-aware representations. Notably, both objectives are integrated directly into the sampling process, avoiding costly per-sample pixel optimization. In practice, high-quality distilled samples can be generated within 50 sampling steps ($\sim$ 1 minute per image), achieving both efficiency and effectiveness.

Table 1: mIoU (%) on ADE20K [59] under different distillation ratios. Distilled data are generated using a fixed guidance model (Swin-T for Mask2Former and MiT-B2 for SegFormer).

Ratio	Method	Mask2Former [9]				SegFormer [46]	
		Res-50	Swin-T	Swin-S	Swin-B	MiT-B2	MiT-B4
0.5%	Random	9.53±0.5	9.57±0.4	10.98±0.5	13.27±0.3	12.59±0.5	15.04±0.6
	K-Center	7.92±0.4	8.66±0.3	10.52±0.4	11.92±0.5	12.31±0.4	14.83±0.4
	Herding	9.75±0.5	10.50±0.4	12.47±0.4	13.63±0.3	13.30±0.4	15.78±0.3
	Uniform	12.54±0.2	12.51±0.3	15.04±0.2	16.15±0.4	16.32±0.3	19.53±0.2
	Ours	**15.87**±0.6	**16.59**±0.4	**19.45**±0.5	**22.55**±0.6	**20.28**±0.3	**24.45**±0.5
1%	Random	12.99±0.4	13.32±0.3	15.65±0.5	18.09±0.3	16.86±0.2	19.74±0.4
	K-Center	9.88±0.4	11.14±0.3	13.76±0.4	15.09±0.2	14.90±0.3	18.95±0.3
	Herding	13.46±0.4	14.43±0.5	16.32±0.3	17.94±0.3	16.78±0.4	19.88±0.2
	Uniform	17.20±0.3	17.94±0.2	21.42±0.3	22.70±0.4	20.36±0.3	23.48±0.4
	Ours	**20.96**±0.3	**21.69**±0.3	**24.99**±0.5	**27.48**±0.5	**23.82**±0.4	**28.17**±0.5
2%	Random	17.50±0.3	19.16±0.4	21.70±0.4	24.00±0.3	20.72±0.3	24.53±0.2
	K-Center	13.75±0.2	14.67±0.2	17.67±0.3	20.74±0.2	19.14±0.4	22.56±0.3
	Herding	18.34±0.3	18.98±0.2	22.81±0.3	24.61±0.4	22.56±0.4	26.48±0.3
	Uniform	21.63±0.3	23.19±0.2	27.59±0.3	29.10±0.4	25.22±0.3	28.46±0.4
	Ours	**24.67**±0.4	**26.13**±0.6	**30.28**±0.5	**32.63**±0.7	**28.15**±0.3	**31.46**±0.4
Full Dataset		47.03±0.2	48.92±0.2	51.22±0.1	52.82±0.2	45.14±0.3	48.20±0.2

Table 2: mIoU (%) on COCO-Stuff [2] under different distillation ratios. Distilled data are generated using a Swin-T guidance model for sampling and relabeling.

Method	0.25%		0.5%		1%	
	Swin-T	Swin-S	Swin-T	Swin-S	Swin-T	Swin-S
Random	15.62±0.5	18.88±0.6	21.33±0.4	24.75±0.5	26.43±0.5	29.79±0.4
K-Center	15.11±0.4	18.97±0.4	20.58±0.5	24.58±0.4	26.30±0.3	30.19±0.2
Herding	18.75±0.3	22.59±0.3	23.50±0.2	27.14±0.1	28.98±0.2	32.50±0.3
Uniform	17.54±0.4	20.63±0.5	22.83±0.5	27.54±0.3	28.96±0.3	32.46±0.3
Ours	**22.34**±0.5	**26.87**±0.5	**26.16**±0.6	**30.66**±0.4	**31.19**±0.5	**35.49**±0.4
Full Dataset	46.69±0.2	48.19±0.2	46.69±0.2	48.19±0.2	46.69±0.2	48.19±0.2

Relabeling. Following the practice of decoupled methods [50], we further refine the supervision of the synthesized data via relabeling. Given a generated image $\hat{\boldsymbol{x}}_j$, we employ the same segmentation model $f(\cdot)$ used during guided sampling to predict its label:

$$\tilde{\boldsymbol{m}}_j = \arg\max f(\hat{\boldsymbol{x}}_j). \tag{10}$$

The final distilled dataset is thus given by $\mathcal{S} = \{(\hat{\boldsymbol{x}}_j, \tilde{\boldsymbol{m}}_j)\}_{j=1}^{|\texttt{IPD}|}$. This step corrects potential inconsistencies and provides more informative supervision, leading to improved downstream performance.

5 Experiments

5.1 Experiment Setup

Datasets and Evaluation Models. We evaluate D^{3}S^2 on ADE20K [59] and COCO-Stuff [2], with all images resized to 512×512. ADE20K contains 20,210 training and 2,000 validation images with 150 semantic classes, while COCO-Stuff includes 118,287 training and 5,000 validation images with 182 classes. To evaluate the quality of distilled data, we adopt two representative architectures, Mask2Former [9] and SegFormer [46], with multiple backbones. For each setting, we repeat training five times and report the mean Intersection-over-Union (mIoU) as $\bar{x} \pm$ std.

Implementation Details. Following common practice in DD, we evaluate compression ratios around 1%. Specifically, we consider 0.5%, 1%, and 2% on ADE20K, and 0.25%, 0.5%, and 1% on COCO-Stuff. The temperature parameter in mask selection is set to $T = 0.5$. For image synthesis, we use two pretrained checkpoints from FreeStyle [48] and adopt the DDIM sampler [35] with 50 denoising steps. The *classifier-free guidance* [16] scale is set to 2.0, while DDIM inversion also use



(a) ADE20K (b) COCO-Stuff

Figure 5: Qualitative visualization of synthesized samples on (a) ADE20K and (b) COCO-Stuff. The synthesized results exhibit strong alignment with the input layouts under both simple and complex scenes, while highlighting class-discriminative features and suppressing irrelevant visual variations.

50 steps with a scale of 1.0. During guided sampling, we set $\lambda_{\text{seg}} = 0.05$ and $\lambda_{\text{feat}} = 0.2$, and apply guidance at all steps. For segmentation training, we follow the MMSegmentation codebase[3] with minor adjustments to accommodate the limited data size; full details are provided in the Appendix C. All experiments are conducted on a single NVIDIA RTX 6000 Ada GPU (48GB).

Baselines. We adapt representative core-set selection methods from classification [44, 56], including Random [28], K-center [31], and Herding [3, 8]. Random randomly selects samples from the original training. K-center and Herding are adapted by extracting global image features using a pre-trained Swin-S backbone (from Mask2Former) and measuring distances with the L2 norm [27]. In addition, we include a class-balanced baseline, *Uniform*, which iteratively traverses classes from tail to head and selects one image per class in a round-robin manner until the budget is exhausted. A comparison with more advanced coreset selection methods is provided in Appendix E.5.

5.2 Experimental Results

Table 1 and Table 2 report quantitative results on ADE20K and COCO-Stuff. Overall, D^3S^2 consistently outperforms all baselines across architectures and data regimes, demonstrating its effectiveness. On ADE20K, D^3S^2 achieves clear gains at all compression ratios. At 1%, it surpasses Uniform by 3.57% on Mask2Former (Swin-S) and 4.69% on SegFormer (MiT-B4), and remains strong at 0.5%, improving over Random by up to 9.41%. Notably, these gains are consistent across backbones despite using a fixed guidance model, indicating strong cross-architecture generalization. In contrast, coreset-based methods perform worse, as they rely on global image-level features and fail to capture dense pixel-level semantics. Although Uniform benefits from class balance, it still lags behind, highlighting the advantage of combining structured mask selection with diffusion-based synthesis. Similar trends appear on COCO-Stuff. At 0.25%, D^3S^2 outperforms Uniform by 4.80% and 6.24% on Swin-T and Swin-S, respectively, with consistent gains at higher ratios (e.g., +3.33% / +3.12% at 1%), showing the robustness of our framework. Despite these improvements, a gap to full-data training remains. For example, on ADE20K, full-data training achieves 51.22% mIoU on Mask2Former (Swin-S), whereas our best result at 2% reaches 30.28%, indicating room for further improvement. Detailed per-class performance breakdowns for rare and difficult categories are provided in Appendix E.4.

Visualization. Figure 5 shows qualitative results. The generated images exhibit strong spatial alignment with input masks, with well-preserved object boundaries and spatial structures, demonstrating effective pixel-level consistency. Beyond fidelity, our method produces more *informative* training samples: multiple classes are clearly expressed within a single image, emphasizing discriminative features while suppressing irrelevant variations. Compared to real images with cluttered backgrounds or dataset-specific artifacts, the synthesized samples are cleaner and more canonical, which benefits segmentation training by enhancing class-specific patterns and reducing noise. Our approach is particularly beneficial for long-tailed classes. For instance, in ADE20K, the *conveyer belt* category is extremely rare (ratio $< 0.005\%$), whereas under a 2% budget, our method increases its presence

[3] `https://github.com/open-mmlab/mmsegmentation`

Table 3: Ablation study on ADE20K (1% ratio) using Mask2Former (Swin-T).

Mask Sel.	Diff.	DDIM Inv.	$\mathcal{L}_{\text{seg}}$	$\mathcal{L}_{\text{feat}}$	Relabel	mIoU
×	×	×	×	×	×	13.32±0.8
✓	×	×	×	×	×	18.29±0.4
✓	✓	×	×	×	×	16.75±0.5
✓	×	×	×	×	✓	18.34±0.5
✓	✓	✓	×	×	×	17.84±0.3
✓	✓	✓	✓	×	×	18.75±0.6
✓	✓	✓	×	✓	×	19.36±0.5
✓	✓	✓	✓	✓	×	20.95±0.4
✓	✓	✓	✓	✓	✓	**21.69**±0.3

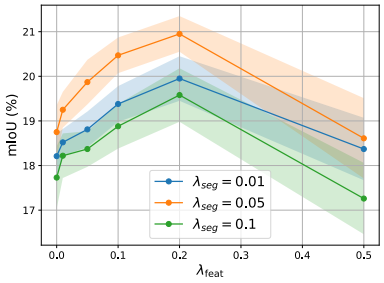


Figure 6: Ablation of λ_{seg} and λ_{feat} on ADE20K (1% ratio) using Mask2Former (Swin-T).

Table 5: Mean-only vs. mean+variance feature matching on ADE20K 1% (mIoU %).

Matching	Res-50	Swin-T	Swin-S	Swin-B	MiT-B2	MiT-B4
Mean only (Ours)	**20.96**±0.3	**21.69**±0.3	**24.99**±0.5	**27.48**±0.5	**23.82**±0.4	**28.17**±0.5
Mean + variance	20.27±0.3	21.11±0.4	24.36±0.4	26.95±0.5	23.19±0.3	27.32±0.4

to $\sim 0.3\%$. The synthesized samples exhibit clear and discriminative *conveyer belt* features with reduced noise, which improves the mIoU of this category from 11.33% (Uniform) to 30.13%. Despite these strengths, some limitations remain. Fine-grained structures (e.g., human hands) may exhibit artifacts, likely due to the inherent limitations of the pretrained layout-to-image diffusion model. Improving the underlying generative quality is an important direction for future work.

5.3 Ablation Study

Effectiveness of Each Component. We perform an ablation study on ADE20K at a 1% ratio, as shown in Table 3, to evaluate the contribution of each component. Starting from a baseline with randomly selected real images (13.32%), introducing our Stage I mask selection significantly improves performance to 18.29%, highlighting the importance of constructing a class-balanced subset. Directly applying diffusion-based synthesis without additional guidance degrades performance to 16.75%, suggesting that naive generation alone is insufficient for producing effective training samples. Incorporating DDIM inversion alleviates this issue (17.84%) by providing better structural initialization. Introducing the proposed guidance brings consistent gains: $\mathcal{L}_{\text{seg}}$ improves pixel-level alignment (+0.91%), while $\mathcal{L}_{\text{feat}}$ yields a larger gain (+1.52%), indicating the importance of class-wise feature consistency. Combining both losses further boosts performance to 20.95%. Finally, relabeling improves the result to 21.69%, suggesting that refining supervision helps correct residual inconsistencies in generated data. Overall, each component contributes positively, and their combination achieves the best performance, validating the effectiveness of our design.

Table 4: Cross-architecture generalization on ADE20K. Distilled datasets generated with different source architectures (Swin-T or MiT-B2) are evaluated on both student architectures to verify whether synthesis overfits the teacher model.

Ratio	Source Archi.	Mask2Former	SegFormer
0.5%	Mask2Former	**16.59**±0.4	20.27±0.4
	SegFormer	16.25±0.5	**20.28**±0.3
1%	Mask2Former	**21.69**±0.3	23.57±0.3
	SegFormer	21.42±0.4	**23.82**±0.4

Mean-Only vs. Mean+Variance Matching. To validate our design choice of using only first-order statistics, we compare mean-only feature matching against a mean+variance variant on ADE20K 1% across six architectures (Table 5). Mean-only matching consistently outperforms mean+variance on all six architectures, with improvements ranging from 0.53 to 0.85 mIoU. The consistent superiority indicates that second-order matching is systematically harmful in segmentation, where variance conflates genuine appearance variation with nuisance factors from variable region sizes and contexts.

Cross-architecture Generalization. We further evaluate whether the distilled data generalizes across different segmentation architectures. As shown in Table 4, datasets distilled with Mask2Former and SegFormer achieve highly consistent performance when evaluated on both models. Under both 0.5% and 1% compression ratios, the performance gap between the two source architectures

Table 6: Efficiency analysis of D^3S^2. We report runtime per stage and peak GPU memory.

Dataset	Ratio (# Images)	Mask Selection	DDIM Inv.		Synthesis		Relabel		Total Time
			Per	Total	Per	Total	Per	Total	
ADE20K	0.5% (101)	∼1.5 min	5 s	∼9 min	∼50 s	∼1h 20m	< 1s	< 1m	∼1h 30m
	1% (202)	∼3 min	5 s	∼18 min	∼50 s	∼2h 40m	< 1s	< 1m	∼3h
	2% (404)	∼6 min	5 s	∼36 min	∼50 s	∼5h 20m	< 1s	< 2m	∼6h
COCO-Stuff	0.25% (295)	∼8 min	5 s	∼26 min	∼50 s	∼4h	< 1s	< 1m	∼4h 30m
	0.5% (591)	∼16 min	5 s	∼52 min	∼50 s	∼8h	< 1s	< 2m	∼9h
	1% (1182)	∼32 min	5 s	∼1h 44m	∼50 s	∼16h	< 1s	< 3m	∼18h
Peak Memory		—	∼8GB		∼28GB		∼3GB		—

remains small, indicating that synthesis captures intrinsic dataset semantics rather than teacher-specific artifacts. Note that core-set baselines are model-agnostic and thus their cross-architecture performance (reported in Table 1) reflects only student capacity differences rather than generalization. Such strong transferability highlights the robustness of our diffusion-guided distillation framework.

Sensitivity of Hyper-Parameters. We analyze the effect of λ_{seg} and λ_{feat} on ADE20K at a 1% ratio (Figure 6). Overall, introducing moderate feature guidance improves performance, highlighting the importance of class-wise feature alignment. As λ_{feat} increases, mIoU steadily improves until 0.2, but drops at $\lambda_{\text{feat}} = 0.5$, suggesting that overly strong feature constraints may harm the generation quality. For λ_{seg}, moderate values perform best. Specifically, $\lambda_{\text{seg}} = 0.05$ consistently yields the highest results across different λ_{feat}. This indicates that appropriate pixel-level guidance is beneficial, but excessive constraints can limit the flexibility of diffusion sampling. The best performance is achieved at $\lambda_{\text{seg}} = 0.05$ and $\lambda_{\text{feat}} = 0.2$, which is the default setting. A comprehensive sensitivity analysis of the temperature parameter T in Stage I mask selection is provided in Appendix E.1.

5.4 Efficiency Analysis

Table 6 summarizes the runtime and memory consumption of each component in D^3S^2. Overall, the proposed framework achieves a favorable efficiency–performance trade-off. The *mask selection* stage is highly efficient, requiring only a few minutes even for larger subsets, as it operates on lightweight statistics without involving model training or GPU-intensive computation. The *DDIM inversion* stage introduces negligible overhead, taking only ∼5 seconds per sample with moderate memory usage (∼8GB). The primary computational cost lies in the *diffusion-based image synthesis*, which requires ∼50 seconds per sample and dominates the total runtime. This is expected, as guided sampling involves gradient backpropagation through the diffusion model, leading to higher memory consumption (∼28GB). Nevertheless, this requirement remains practical on modern GPUs (e.g., 48GB), and can be further reduced via techniques such as gradient checkpointing [5, 23]. The *relabeling* stage introduces negligible overhead in both time and memory (less than 1 second per sample and ∼3GB), owing to the use of lightweight segmentation models. Importantly, the overall distillation time scales linearly with the number of samples and remains manageable in practice (e.g., ∼3 hours for 1% ADE20K on a single GPU). Compared to prior dataset distillation methods that rely on costly per-sample pixel optimization or bi-level training, D^3S^2 avoids iterative optimization loops and instead performs one-pass guided generation. As a result, it provides a significantly more efficient and scalable solution for dense prediction tasks.

6 Conclusion

This paper presents D^3S^2, a diffusion-guided dataset distillation framework for semantic segmentation. It addresses the unique challenges of dense prediction via a two-stage design: (i) a class-balanced greedy selection strategy that constructs a compact and representative mask set for long-tailed distributions, and (ii) layout-to-image diffusion synthesis that generates images aligned with semantic layouts. By integrating segmentation-consistency and class-wise feature matching losses, our method ensures pixel-level alignment and feature fidelity. Extensive experiments on ADE20K and COCO-Stuff demonstrate consistent mIoU improvements over strong baselines across architectures and compression ratios. Overall, this work demonstrates the power of diffusion priors for dataset distillation in semantic segmentation, with substantial room for further improvement.

Acknowledgments and Disclosure of Funding

We thank all reviewers for their invaluable feedback, which significantly strengthened this work. This work received no third-party funding or support, and the authors have no competing interests to disclose.

References

[1] A. Bansal, H.-M. Chu, A. Schwarzschild, S. Sengupta, M. Goldblum, J. Geiping, and T. Goldstein. Universal guidance for diffusion models. In *ICLR*, 2024.

[2] H. Caesar, J. Uijlings, and V. Ferrari. Coco-stuff: Thing and stuff classes in context. In *CVPR*, 2018.

[3] F. M. Castro, M. J. Marín-Jiménez, N. Guil, C. Schmid, and K. Alahari. End-to-end incremental learning. In *ECCV*, 2018.

[4] G. Cazenavette, T. Wang, A. Torralba, A. A. Efros, and J.-Y. Zhu. Dataset distillation by matching training trajectories. In *CVPR*, 2022.

[5] G. Cazenavette, T. Wang, A. Torralba, A. A. Efros, and J.-Y. Zhu. Generalizing dataset distillation via deep generative prior. In *CVPR*, 2023.

[6] J. A. Chan-Santiago and M. Shah. Learnability-guided diffusion for dataset distillation. In *CVPR*, 2026.

[7] M. Chen, J. Du, B. Huang, Y. Wang, X. Zhang, and W. Wang. Influence-guided diffusion for dataset distillation. In *ICLR*, 2025.

[8] Y. Chen, M. Welling, and A. Smola. Super-samples from kernel herding. In *UAI*, 2010.

[9] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar. Masked-attention mask transformer for universal image segmentation. In *CVPR*, 2022.

[10] J. Cui, R. Wang, S. Si, and C.-J. Hsieh. Scaling up dataset distillation to imagenet-1k with constant memory. In *ICML*, 2023.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[12] U. Feige. A threshold of $\ln n$ for approximating set cover. *JACM*, 45:634–652, 1998.

[13] J. Geng, Z. Chen, Y. Wang, H. Woisetschläger, S. Schimmler, R. Mayer, Z. Zhao, and C. Rong. A survey on dataset distillation: approaches, applications and future directions. In *IJCAI*, 2023.

[14] J. Gu, S. Vahidian, V. Kungurtsev, H. Wang, W. Jiang, Y. You, and Y. Chen. Efficient dataset distillation via minimax diffusion. In *CVPR*, 2024.

[15] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[16] J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[17] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, 1972.

[18] S. K. A. Khatib, A. ElHagry, S. Shao, and Z. Shen. OD^3: Optimization-free dataset distillation for object detection. In *ICLR*, 2026.

[19] J.-H. Kim, J. Kim, S. J. Oh, S. Yun, H. Song, J. Jeong, J.-W. Ha, and H. O. Song. Dataset condensation via efficient synthetic-data parameterization. In *ICML*, 2022.

[20] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[21] N. Loo, R. Hasani, A. Amini, and D. Rus. Efficient dataset distillation using random feature approximation. In *NeurIPS*, 2022.

[22] A. Maharana, P. Yadav, and M. Bansal. D2 pruning: Message passing for balancing diversity and difficulty in data pruning. In *ICLR*, 2024.

[23] B. B. Moser, F. Raue, S. Palacio, S. Frolov, and A. Dengel. Unlocking dataset distillation with diffusion models. In *NeurIPS*, 2025.

[24] C. Mou, X. Wang, L. Xie, Y. Wu, J. Zhang, Z. Qi, and Y. Shan. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. In *AAAI*, 2024.

[25] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions—I. *Mathematical Programming*, 14:265–294, 1978.

[26] T. Nguyen, Z. Chen, and J. Lee. Dataset meta-learning from kernel ridge-regression. In *ICLR*, 2021.

[27] D. Qi, J. Li, J. Peng, B. Zhao, S. Dou, J. Li, J. Zhang, Y. Wang, C. Wang, and C. Zhao. Fetch and forge: Efficient dataset condensation for object detection. In *NeurIPS*, 2024.

[28] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017.

[29] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[30] J. A. C. Santiago, P. Tirupattur, G. K. Nayak, G. Liu, and M. Shah. MGD^3: Mode-guided dataset distillation using diffusion models. In *ICML*, 2025.

[31] O. Sener and S. Savarese. Active learning for convolutional neural networks: A core-set approach. In *ICLR*, 2018.

[32] S. Shao, Z. Yin, M. Zhou, X. Zhang, and Z. Shen. Generalized large-scale data condensation via various backbone and statistical matching. In *CVPR*, 2024.

[33] S. Shao, Z. Zhou, H. Chen, and Z. Shen. Elucidating the design space of dataset condensation. In *NeurIPS*, 2024.

[34] Z. Shen, A. Sherif, Z. Yin, and S. Shao. Delt: A simple diversity-driven earlylate training for dataset distillation. In *CVPR*, 2025.

[35] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

[36] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

[37] D. Su, J. Hou, W. Gao, Y. Tian, and B. Tang. D^4M: Dataset distillation via disentangled diffusion model. In *CVPR*, 2024.

[38] D. Su, H. Wu, H. Chen, Y. Shi, Y. Wang, X. Ye, and J. Zhu. Diffusion models as dataset distillation priors. In *ICLR*, 2026.

[39] P. Sun, B. Shi, D. Yu, and T. Lin. On the diversity and realism of distilled dataset: An efficient dataset distillation paradigm. In *CVPR*, 2024.

[40] H. Wang, L. Zhang, W. Liu, D. Jiang, W. Wei, and C. Ding. Jodiffusion: Jointly diffusing image with pixel-level annotations for semantic segmentation promotion. In *AAAI*, 2026.

[41] H. Wang, Z. Zhao, J. Wu, Y. Shang, G. Liu, and Y. Yan. CaO_2: Rectifying inconsistencies in diffusion-based dataset distillation. In *ICCV*, 2025.

[42] K. Wang, J. Gu, H. Zhang, D. Zhou, Z. Zhu, W. Jiang, and Y. You. Dim: Distilling dataset into generative model. In *ECCV Workshops*, 2024.

[43] K. Wang, B. Zhao, X. Peng, Z. Zhu, S. Yang, S. Wang, G. Huang, H. Bilen, X. Wang, and Y. You. Cafe: Learning to condense dataset by aligning features. In *CVPR*, 2022.

[44] T. Wang, J.-Y. Zhu, A. Torralba, and A. A. Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

[45] L. Xiao and Y. He. Are large-scale soft labels necessary for large-scale dataset distillation? In *NeurIPS*, 2024.

[46] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. In *NeurIPS*, 2021.

[47] T. Xie, J. Zhu, G. Ma, M. Lin, W. Chen, W. Yang, and S. Liu. Structural-entropy-based sample selection for efficient and effective learning. In *ICLR*, 2025.

[48] H. Xue, Z. Huang, Q. Sun, L. Song, and W. Zhang. Freestyle layout-to-image synthesis. In *CVPR*, 2023.

[49] L. Yang, X. Xu, B. Kang, Y. Shi, and H. Zhao. Freemask: Synthetic images with dense annotations make stronger segmentation models. In *NeurIPS*, 2023.

[50] Z. Yin, E. Xing, and Z. Shen. Squeeze, recover and relabel: Dataset condensation at imagenet scale from a new perspective. In *NeurIPS*, 2023.

[51] J. Yu, Y. Wang, C. Zhao, B. Ghanem, and J. Zhang. Freedom: Training-free energy-guided conditional diffusion model. In *ICCV*, 2023.

[52] R. Yu, S. Liu, and X. Wang. Dataset distillation: A comprehensive review. *IEEE TPAMI*, 46:150–170, 2024.

[53] L. Zhang, A. Rao, and M. Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, 2023.

[54] B. Zhao and H. Bilen. Dataset condensation with differentiable siamese augmentation. In *ICML*, 2021.

[55] B. Zhao and H. Bilen. Dataset condensation with distribution matching. In *WACV*, 2023.

[56] B. Zhao, K. R. Mopuri, and H. Bilen. Dataset condensation with gradient matching. In *ICLR*, 2021.

[57] G. Zhao, G. Li, Y. Qin, and Y. Yu. Improved distribution matching for dataset condensation. In *CVPR*, 2023.

[58] H. Zheng, R. Liu, F. Lai, and A. Prakash. Coverage-centric coreset selection for high pruning rates. In *ICLR*, 2023.

[59] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017.

[60] Y. Zhou, E. Nezhadarya, and J. Ba. Dataset distillation using neural feature regression. In *NeurIPS*, 2022.

[61] Y. Zou, G. Li, D. Su, Z. Wang, J. Yu, and C. Zhang. Dataset distillation via vision-language category prototype. In *ICCV*, 2025.

This appendix provides additional technical details, experimental results, and analyses that complement the main paper. Appendix A derives the gradient-based guidance mechanism for diffusion-based synthesis and provides the complete algorithm. Appendix B visualizes class-distribution statistics for the ADE20K and COCO-Stuff datasets. Appendix C provides extended implementation details. Appendix D presents the complete algorithmic formulation of our two-stage framework. Appendix E presents comprehensive experimental analyses including temperature sensitivity, scaling behavior across compression ratios, comparison with mask-conditioned synthesis methods, fine-grained class-wise performance breakdown for rare and difficult categories, and comparison with advanced coreset selection methods. Appendix F discusses limitations and promising future directions. Appendix G addresses broader societal impacts. Appendix H provides additional qualitative visualizations on both datasets. Appendix I documents large language model usage in manuscript preparation.

A More Details on Guided Diffusion Sampling

Diffusion models [15, 36] learn a parameterized data distribution through a forward noising and reverse denoising process. Given a clean sample $\boldsymbol{z}_0$, the forward process gradually perturbs it with Gaussian noise:

$$\boldsymbol{z}_t = \sqrt{\alpha_t}\boldsymbol{z}_0 + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, I), \tag{11}$$

where α_t controls the noise scale at timestep t.

The diffusion model is trained to predict the injected noise $\boldsymbol{\epsilon}_\phi(\boldsymbol{z}_t, t, \boldsymbol{c})$ under condition $\boldsymbol{c}$ using a mean squared error objective. During sampling, DDIM [35] estimates the clean latent:

$$\hat{\boldsymbol{z}}_{0|t} = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{z}_t - \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_\phi(\boldsymbol{z}_t, t, \boldsymbol{c})\right), \tag{12}$$

and progressively reconstructs the sample through iterative denoising steps.

In this work, we adopt latent diffusion models [29], where images are encoded into latent representations $\boldsymbol{z} = E(\boldsymbol{x})$ and reconstructed using a decoder $D(\cdot)$, significantly improving computational efficiency.

To further steer generation toward desired properties, guided diffusion sampling [1, 51] modifies the denoising trajectory using gradients from external objectives. Given a guidance function $f(\cdot)$ and loss function $\ell(\cdot,\cdot)$, the predicted noise is adjusted as:

$$\hat{\boldsymbol{\epsilon}}_\phi(\boldsymbol{z}_t, t, \boldsymbol{c}) = \boldsymbol{\epsilon}_\phi(\boldsymbol{z}_t, t, \boldsymbol{c}) + s(t) \cdot \nabla_{\boldsymbol{z}_t}\ell\big(f(\boldsymbol{z}_t), C\big), \tag{13}$$

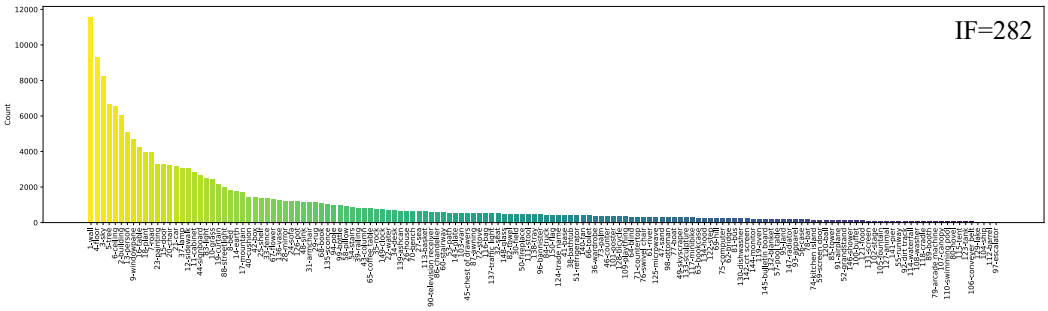
where $s(t)$ controls the guidance strength and C denotes additional conditions. This formulation enables flexible integration of task-specific objectives into the sampling process.

B Dataset Distributions

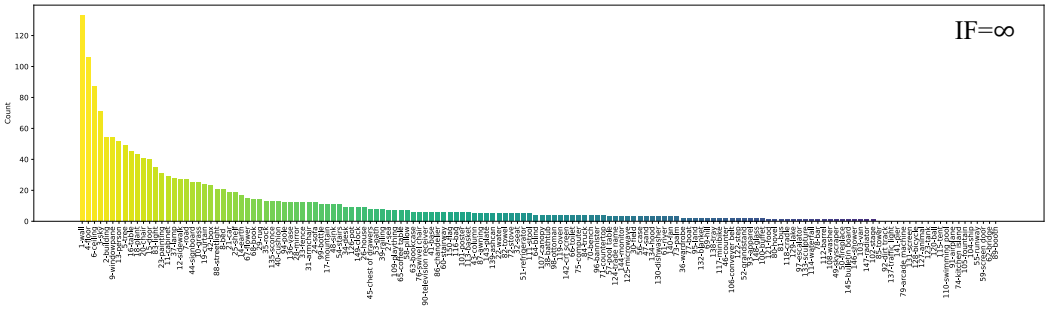
In this section, we analyze the class distributions of different selection strategies on ADE20K [59] and COCO-Stuff [2], as shown in Figure 7 and Figure 8, respectively. For each dataset, we compare four settings: (a) the original dataset, (b) Random selection, (c) Uniform selection, and (d) our Class-Balanced Greedy Selection. The imbalance factor (IF) is reported in each subfigure.

ADE20K. ADE20K contains 20,210 training images with 150 classes. We fix the compression ratio to 1%, resulting in 202 selected samples. As shown in Figure 7 (a), the original dataset exhibits a highly long-tailed distribution with IF=282. Random selection (Figure 7 (b)) fails to cover all classes, leading to an infinite IF. Uniform selection (Figure 7 (c)) partially alleviates the imbalance (IF=129), but tail classes are still severely underrepresented, with only 1–2 samples per class. In contrast, our greedy selection (Figure 7 (d)) significantly improves the balance, reducing IF to 14 and ensuring that even the rarest classes are sampled at least 8 times.

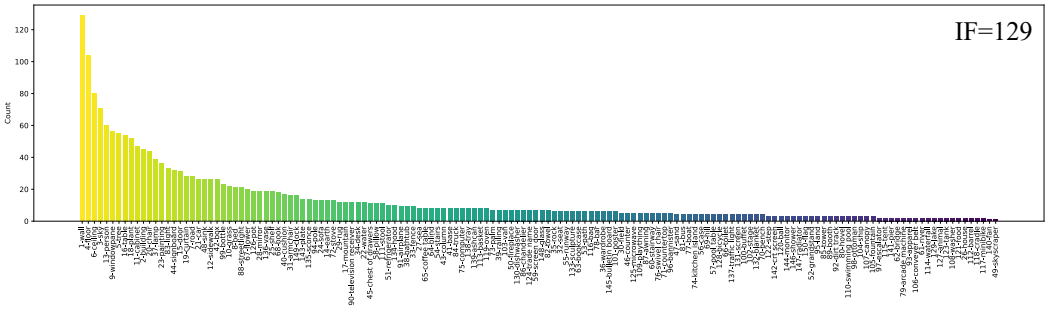
COCO-Stuff. COCO-Stuff contains 118,287 training images with 182 classes. We use a 0.5% compression ratio, corresponding to 591 selected samples. The original dataset (Figure 8 (a)) has an even more skewed distribution with IF=528. Similar to ADE20K, Random selection (Figure 8 (b)) leads to missing classes, while Uniform selection (Figure 8 (c)) improves the distribution (IF=78) but still provides limited coverage for tail classes (4–5 samples). Our greedy strategy (Figure 8 (d))



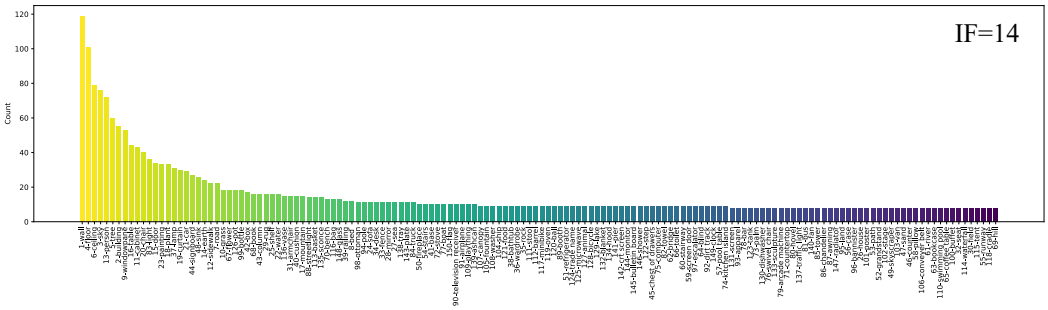
(a) Full Dataset



(b) Random Selection

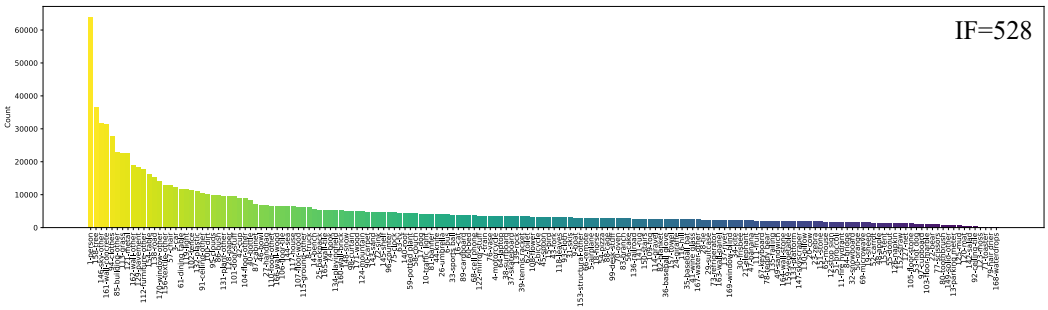


(c) Uniform Selection

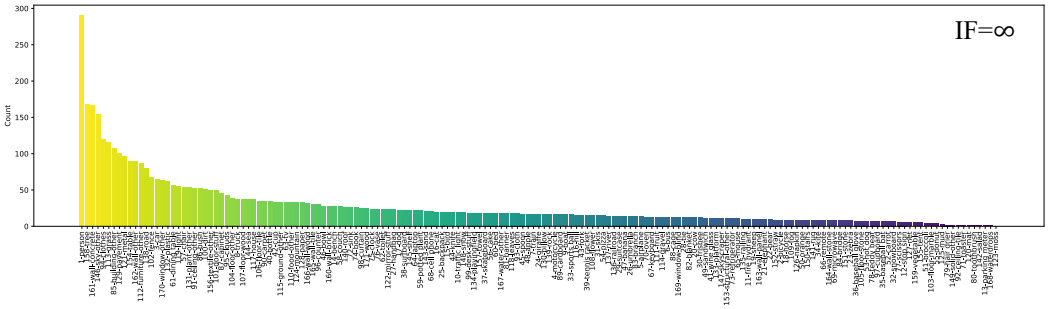


(d) Greedy Selection

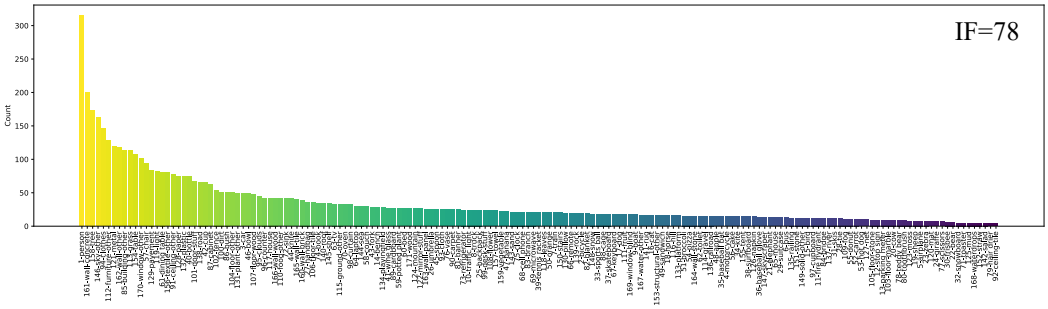
Figure 7: Class distribution comparison on ADE20K under a 1% selection ratio (202 images). We compare (a) the original dataset, (b) Random selection, (c) Uniform selection, and (d) the proposed Class-Balanced Greedy Selection. The imbalance factor (IF) is reported in each subfigure. Our method significantly reduces class imbalance and improves coverage of tail classes.



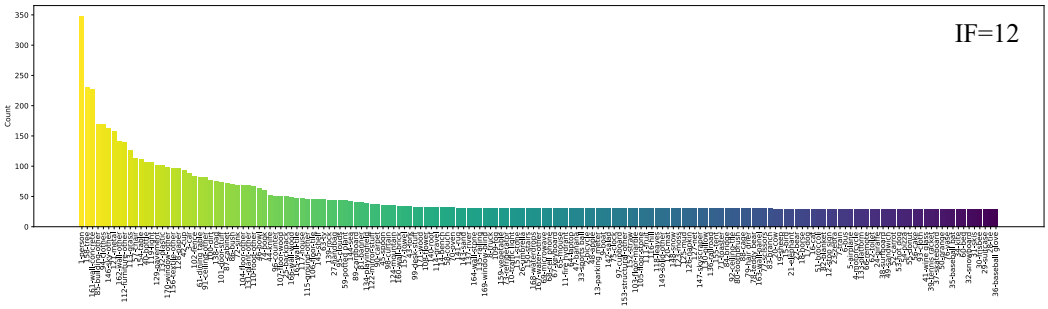
(a) Full Dataset



(b) Random Selection



(c) Uniform Selection



(d) Greedy Selection

Figure 8: Class distribution comparison on COCO-Stuff under a 0.5% selection ratio (591 images). We compare (a) the original dataset, (b) Random selection, (c) Uniform selection, and (d) the proposed Class-Balanced Greedy Selection. The imbalance factor (IF) is shown in each subfigure. Our method achieves a much more balanced distribution, especially for rare classes.

Algorithm 1: Class-Balanced Greedy Selection

Input: Dataset $\mathcal{T} = \{(\boldsymbol{x}_i, \boldsymbol{m}_i)\}_{i=1}^{N}$, class sets $\{\mathcal{C}_i\}$, budget $B = |\texttt{IPD}|$, temperature T
Output: Selected mask set $\mathcal{M}$
Initialize $\mathcal{M} \leftarrow \emptyset$; $\quad n_c \leftarrow 0$ for all classes c ;
Compute class weights $w_c \propto 1/\text{freq}(c)$;
while $|\mathcal{M}| < B$ **do**
  foreach $i \notin \mathcal{M}$ **do**
    Compute score:

$$s_i = \sum_{c \in \mathcal{C}_i} w_c \cdot \exp\left(-\frac{n_c}{T}\right)$$

  end
  Select $i^* = \arg\max_i s_i$;
  $\mathcal{M} \leftarrow \mathcal{M} \cup \{\boldsymbol{m}_{i^*}\}$;
  foreach $c \in \mathcal{C}_{i^*}$ **do**
    $n_c \leftarrow n_c + 1$
  end
end
return $\mathcal{M}$

further reduces the imbalance to IF=12, with tail classes appearing at least 29 times, demonstrating substantially improved coverage.

Discussion. Despite the improved balance, achieving a perfectly uniform class distribution is inherently infeasible in semantic segmentation due to frequent class co-occurrence within images. A single image often contains multiple classes with varying spatial extents, making independent per-class allocation impractical. From a dataset distillation perspective, our strategy is also more principled under extreme compression. Dominant classes (e.g., *person*, *wall*, *sky*) already appear abundantly in the original dataset and contribute less marginal information. Allocating excessive budget to such classes is inefficient. Instead, our method prioritizes underrepresented categories, leading to a more informative and balanced distilled set. This design better utilizes the limited budget and provides more comprehensive supervision, which is crucial for dense prediction tasks like semantic segmentation.

C More Implementation Details

For downstream evaluation, we train segmentation models using the official MMSegmentation codebase.[4] Specifically, we adopt Mask2Former [9] and SegFormer [46] as the evaluation architectures. For training on the full dataset, we follow the official configurations provided by MMSegmentation, such as `mask2former_r50_8xb2-160k_ade20k-512x512.py` and `segformer_mit-b2_8xb2-160k_ade20k-512x512.py`. All models are retrained on a server with 8 NVIDIA GeForce RTX 3090 GPUs to ensure a fair comparison. For training on the distilled datasets, we adjust the training schedule to account for the significantly reduced data size. Specifically, we reduce the number of training iterations from 160k to 20k while keeping all other hyperparameters unchanged. This modification ensures efficient training while maintaining consistency with the original setup. Code is available at `https://github.com/zwj084/D3S2`.

D Method Algorithm

D.1 Stage I: Class-Balanced Greedy Selection

Algorithm 1 presents our proposed Class-Balanced Greedy Selection. It progressively constructs a representative mask subset by balancing class rarity and coverage. The class weights prioritize underrepresented categories, while the exponential decay term discourages repeatedly selecting already well-covered classes. This dynamic trade-off enables the selection of diverse and comple-

[4] `https://github.com/open-mmlab/mmsegmentation`

Algorithm 2: D^{3}S^2: Diffusion-Guided Dataset Distillation for Semantic Segmentation

Input: Dataset $\mathcal{T} = \{(\boldsymbol{x}_i, \boldsymbol{m}_i)\}_{i=1}^{N}$, budget B, pretrained layout-to-image diffusion model ϵ_ϕ, segmentation model $f(\cdot)$
Output: Distilled dataset $\mathcal{S}$
Stage I: Mask Selection ;
Obtain mask set $\mathcal{M} \leftarrow \texttt{MaskSelection}(\mathcal{T}, B)$ using Algorithm 1 ;
Stage II: Diffusion-Guided Image Synthesis ;
Initialize $\mathcal{S} \leftarrow \emptyset$;
foreach $\boldsymbol{m}_j \in \mathcal{M}$ **do**
  Construct condition $\boldsymbol{c}_j$ from class names in $\boldsymbol{m}_j$;
  // DDIM Inversion ;
  Obtain latent $\boldsymbol{z}_{j,T}$ by inverting corresponding real image $\boldsymbol{x}_j$;
  // Guided Sampling ;
  for $t = T$ *to* 1 **do**
    Predict noise $\boldsymbol{\epsilon}_\phi(\boldsymbol{z}_{j,t}, t, \boldsymbol{c}_j, \boldsymbol{m}_j)$;
    Estimate $\hat{\boldsymbol{z}}_{j,0|t}$ and decode $\hat{\boldsymbol{x}}_{j,0}$;
    Compute $\mathcal{L}_{\text{seg}}$ and $\mathcal{L}_{\text{feat}}$;
    Update noise with guidance:

$$\hat{\boldsymbol{\epsilon}}_\phi = \boldsymbol{\epsilon}_\phi + \rho_t \nabla \mathcal{L}_{\text{seg}} + \gamma_t \nabla \mathcal{L}_{\text{feat}}$$

    Sample $\boldsymbol{z}_{j,t-1}$ using DDIM update ;
  end
  Decode final image $\hat{\boldsymbol{x}}_j = D(\boldsymbol{z}_{j,0})$;
  // Relabeling ;
  Obtain $\tilde{\boldsymbol{m}}_j = \arg\max f(\hat{\boldsymbol{x}}_j)$;
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{(\hat{\boldsymbol{x}}_j, \tilde{\boldsymbol{m}}_j)\}$;
end
return $\mathcal{S}$

mentary samples under a limited budget. Unlike rigid class-wise sampling, the proposed strategy naturally leverages multi-class co-occurrence in segmentation data, resulting in a more balanced yet contextually rich mask set.

D.1.1 Computational Complexity and Approximation Analysis

While the greedy selection in Algorithm 1 is conceptually straightforward, its theoretical justification is rigorous. We show that the algorithm greedily maximizes a monotone submodular objective function, yielding provable approximation guarantees.

Submodular Formulation. Define $\gamma := e^{-1/T}$ and let $n_c(\mathcal{M})$ denote the coverage count of class c in set $\mathcal{M}$. We introduce the *soft class-weighted coverage* objective:

$$F(\mathcal{M}) = \sum_c w_c \cdot \phi(n_c(\mathcal{M})), \quad \text{where} \quad \phi(n) = \frac{1-\gamma^n}{1-\gamma}, \tag{14}$$

This formulation assigns diminishing returns to successive appearances of class c: the first occurrence contributes w_c, the second $w_c\gamma$, and so on. The marginal gain of adding a mask i to $\mathcal{M}$ is precisely:

$$\Delta_i F(\mathcal{M}) = F(\mathcal{M} \cup \{i\}) - F(\mathcal{M}) = \sum_{c \in \mathcal{C}_i} w_c \gamma^{n_c(\mathcal{M})} = s_i, \tag{15}$$

which matches Equation (3) in the main paper. Thus, Algorithm 1 is standard greedy maximization of $\max_{|\mathcal{M}| \le B} F(\mathcal{M})$.

Submodularity. The objective F is monotone and submodular for every $T > 0$. Monotonicity follows immediately since $\phi(n)$ is non-decreasing in n. Submodularity holds because γ^n is a non-increasing sequence in n, which ensures that marginal gains $\Delta_i F(\mathcal{M})$ can only decrease as $\mathcal{M}$ grows. Formally, for any $\mathcal{M} \subseteq \mathcal{M}'$ and item $i \notin \mathcal{M}'$:

$$\Delta_i F(\mathcal{M}') = \sum_{c \in \mathcal{C}_i} w_c \gamma^{n_c(\mathcal{M}')} \le \sum_{c \in \mathcal{C}_i} w_c \gamma^{n_c(\mathcal{M})} = \Delta_i F(\mathcal{M}). \tag{16}$$

Approximation Guarantee. Two important consequences follow: **(1) NP-hardness.** For $w_c \equiv 1$ (uniform class weights) and small T (negligible decay), the problem $\max_{|\mathcal{M}| \le B} F(\mathcal{M})$ reduces to the Maximum Coverage problem, which is known to be NP-hard [17, 12]. **(2) Optimal Polynomial-Time Approximation.** By the classical result of Nemhauser–Wolsey–Fisher [25], the greedy algorithm achieves $(1 - 1/e) \approx 0.632$ of the optimal value for maximizing any monotone submodular function subject to a cardinality constraint. Moreover, by Feige [12], this $(1 - 1/e)$ ratio is the best guarantee any polynomial-time algorithm can achieve unless P=NP. Therefore, our Algorithm 1 is not only simple but also theoretically optimal in the context of polynomial-time approximations.

Computational Complexity. The cost of Algorithm 1 is $O(B \cdot N\bar{K})$, where B is the distillation budget, N is the dataset size, and $\bar{K}$ is the average number of classes per image. Since mask selection operates only on lightweight statistics (class frequencies and coverage counts) without model training, it completes in a few CPU-minutes even for large datasets.

D.2 Stage II: Diffusion-Guided Image Synthesis

Algorithm 2 summarizes the overall pipeline of D^3S^2. The framework first constructs a representative mask set via class-balanced selection (Stage I), and then synthesizes high-quality training samples through diffusion-based generation with guidance (Stage II). Compared to existing dataset distillation approaches, D^3S^2 offers several advantages. First, it explicitly accounts for long-tailed distributions through adaptive mask selection with theoretical guarantees. Second, it avoids costly pixel-level optimization by integrating task-specific objectives directly into the diffusion sampling process. Finally, the combination of segmentation-consistency and class-wise feature matching enables the generation of structurally aligned and semantically discriminative samples, leading to strong performance and generalization across architectures.

Synthesis vs. Refinement. While we term Stage II *synthesis*, it is more precisely understood as *refinement*. Each selected mask $\boldsymbol{m}_j$ comes paired with a corresponding real image $\boldsymbol{x}_j$ from the original dataset. Via DDIM inversion, we project this real image into latent space as the starting point for denoising, rather than sampling from random noise. Without our guidance losses, the deterministic DDIM trajectory would return close to the original image. Our two objectives—$\mathcal{L}_{\text{seg}}$ for pixel-level alignment and $\mathcal{L}_{\text{feat}}$ for class-wise feature consistency—then guide refinement away from this natural point. The endpoint is thus neither the original image nor an unrestricted generation, but a conditioned refinement that balances fidelity to the mask with dataset-level feature statistics. This perspective clarifies how Stage II operates: we preserve the real image as an informative prior and refine it toward improved training utility.

E Additional Experimental Results

E.1 Sensitivity Analysis of Temperature Parameter T

Motivation. The class-balanced greedy selection (Algorithm 1) relies on a temperature parameter T that controls the trade-off between prioritizing underrepresented classes and maintaining sample diversity. To justify our choice of $T = 0.5$ and demonstrate robustness, we provide a comprehensive sensitivity analysis across a $50\times$ range of values. Crucially, $T = 0.5$ was selected by minimizing the Imbalance Factor (IF) of the selected mask subset—a criterion that uses only Stage-I mask statistics, requiring no synthesis, student training, or test labels. This ensures T was never tuned on downstream performance and can be reproduced on new datasets at negligible cost.

Experimental Setup. We evaluate six temperature values $T \in \{0.1, 0.2, 0.5, 1.0, 2.0, 5.0\}$ on ADE20K at 1% compression (202 images) using Mask2Former (Swin-T). For each T, we report: (1) Imbalance Factor (IF) and minimum per-class count from Stage I; (2) Stage-I-only mIoU—training on the selected *real* image-mask pairs, isolating selection quality from synthesis; and (3) full D^3S^2 mIoU with diffusion-guided synthesis. All results are averaged over 5 runs.

Results and Analysis. Four key patterns emerge from Table 7: **(1) Robustness across wide range.** Across the $50\times$ range, full-pipeline mIoU spans only 0.55 points (21.14–21.69) with consistently low variance ($\sigma \le 0.4$), while Stage-I-only mIoU spans merely 0.25 points (18.04–18.29) with minimal variance ($\sigma \le 0.2$). Within the reasonable range $T \in [0.2, 2.0]$, the spread narrows to 0.31 mIoU, demonstrating that performance is insensitive to the specific choice of T. **(2) Consistent superiority over baselines.** Every tested T outperforms the strongest baseline (Uniform, 17.94

Table 7: Sensitivity analysis of temperature parameter T on ADE20K at 1% compression (202 images) using Mask2Former (Swin-T). IF: Imbalance Factor (lower indicates better balance). Min: minimum per-class count (higher indicates better coverage). Uniform baseline achieves 17.94 mIoU.

T	IF↓	Min↑	Stage-I-only	Full D^{3}S^2
0.1	22	6	18.04±0.1	21.14±0.4
0.2	17	7	18.15±0.1	21.38±0.3
0.5	**14**	**8**	**18.29±0.1**	**21.69±0.3**
1.0	18	7	18.19±0.1	21.45±0.2
2.0	18	7	18.17±0.1	21.40±0.3
5.0	25	5	18.05±0.2	21.15±0.4

mIoU) by substantial margins. Even the worst configurations ($T = 0.1$ and $T = 5.0$) exceed Uniform by +3.20 and +3.21 mIoU, respectively. Stage-I-only results surpass Uniform for *all* T values ($\geq$18.04), confirming that our method's advantage is robust and does not depend on careful tuning. **(3) Optimality of $T = 0.5$.** The default choice $T = 0.5$ simultaneously achieves the lowest IF (14), highest minimum per-class count (8), and best mIoU for both Stage-I-only (18.29±0.1) and full pipeline (21.69±0.3). This choice also exhibits the smallest or near-smallest variance, providing both point-estimate optimality and stability benefits. **(4) Behavior at extremes.** Performance degrades predictably at both extremes due to the exponential decay term $\exp(-n_c/T)$: as $T \to 0$, it approaches a binary indicator (never-covered vs. covered); as $T \to \infty$, it approaches 1, reducing the score to a static rarity sum. Both extremes diminish the dynamic balance between class rarity and coverage, yielding suboptimal and less stable selection.

Discussion. The robustness across T stems from our selection mechanism: class weights $w_c \propto 1/\text{freq}(c)$ span over two orders of magnitude (original IF=282), dominating the selection score $s_i = \sum_{c \in \mathcal{C}_i} w_c \exp(-n_c/T)$. T modulates only how quickly covered classes are discounted, providing fine-grained control without drastically altering the core prioritization of rare classes. Our IF-based selection criterion naturally identifies the optimal T without requiring downstream task evaluation, making it reproducible and transferable to new datasets at negligible cost.

E.2 Scaling Experiment: Budget vs. Full-Data Performance

Motivation. To quantify the regime where D^{3}S^2 provides maximum value, we evaluate scaling behavior across compression ratios from 0.5% to 20% of ADE20K (101 to 4,042 training images out of 20,210 total). This experiment characterizes how our advantage over baselines changes as the data budget increases, validating our design focus on extreme compression ratios.

Experimental Setup. We compare D^{3}S^2 against Random selection and Uniform across seven budget ratios: 0.5%, 1%, 2%, 5%, 10%, 15%, and 20%. All experiments use a consistent configuration: $T = 0.5$, Swin-T for guidance and relabeling, and identical downstream training protocols with Mask2Former (Swin-T). Results are reported as means over 5 runs, with the 0.5%, 1%, and 2% points corresponding to the experiments from Table 1 in the main paper.

Results and Analysis. Three key findings emerge from Figure 9: **(1) Maximum advantage in low-budget regime.** The margin over Uniform selection decreases monotonically: +4.08% (0.5%) → +3.75% (1%) → +2.94% (2%) → +2.12% (5%) → +1.43% (10%) → +0.81% (20%). This demonstrates that D^{3}S^2 delivers the largest gains precisely where data constraints are most severe—the regime that motivates dataset distillation. **(2) Bounded gap to full data.** At 20% compression, our method achieves 43.30% mIoU compared to full-data performance of 48.92%, leaving a gap of 5.62 percentage points. This plateau is theoretically expected: dataset distillation compresses a large-scale distribution into a compact surrogate, and once real data coverage becomes sufficient (at higher ratios), any distillation method must asymptotically approach and eventually fall behind real images. This phenomenon has been observed in other dense-prediction condensation work [27]. **(3) Design positioning validation.** Our focus on 0.25%–2% compression ratios (where 101 images at 0.5% roughly match Random's performance with 404 images) aligns with the distillation objective of achieving 4× data savings in the regime where our advantage is greatest. The scaling curves confirm this is not an arbitrary choice but reflects the natural operating point where distilled data provides maximum value compared to alternatives.

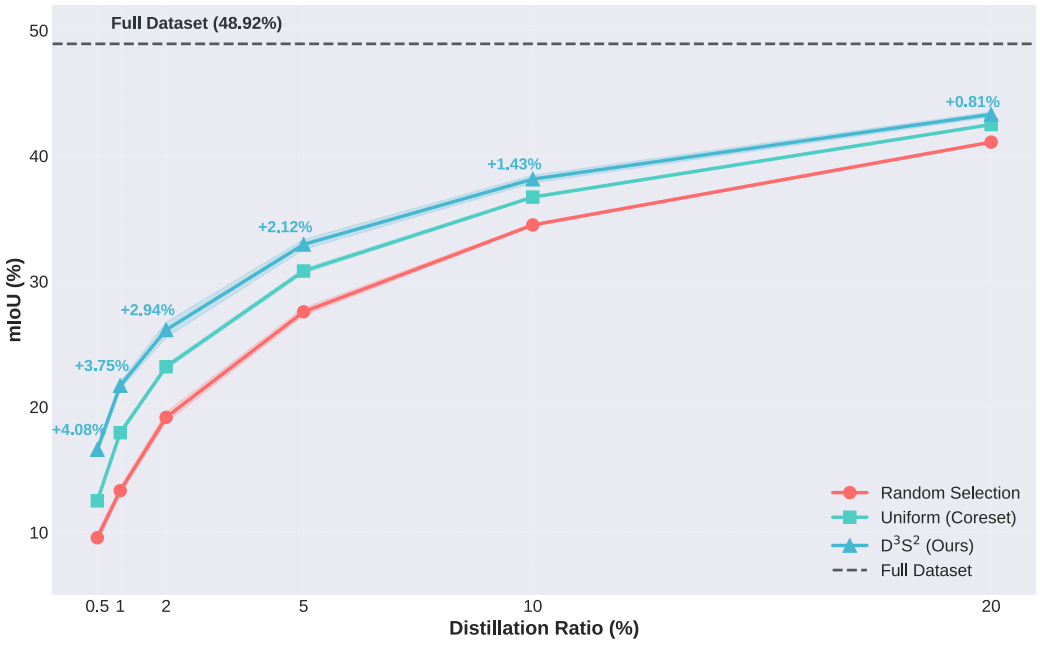


Figure 9: Scaling behavior on ADE20K (Mask2Former Swin-T) across compression ratios from 0.5% to 20%. D^3S^2 consistently outperforms Random and Uniform across all budgets, with advantages decreasing monotonically from +4.08% at 0.5% to +0.81% at 20%.

E.3 Comparison with Mask-Conditioned Synthesis Methods

Motivation. While D^3S^2 shares its synthesis mechanism with mask-conditioned image generation methods, the objectives differ fundamentally. Such methods typically perform **Generative Data Augmentation (GDA)**, enriching datasets at modest expansion ratios (1:5 to 1:20) to train *together with* the full real dataset. In contrast, our setting is **Generative Dataset Distillation (GDD)**, operating at extreme compression ratios (100:1 to 1000:1) where synthesized samples *replace* rather than augment real data. This inversion fundamentally changes design requirements: expansion affords redundancy where weak samples are harmless since real data remains available; distillation cannot, and must make every sample carry dataset-level information.

Compared Method. We compare against FreeMask [49], a representative mask-conditioned synthesis method for generating segmentation training data. FreeMask employs a three-stage pipeline: (1) selecting diverse layouts from real data and generating synthetic images conditioned on these layouts using pretrained diffusion models; (2) filtering noisy synthetic regions by comparing generated images with their conditioning masks; and (3) re-sampling synthetic images based on mask-level hardness to improve data quality. For fair comparison, we use identical generator checkpoints (FreeStyleNet [48] fine-tuned on ADE20K) and provide FreeMask with *exactly our Stage-I selected masks*, so the pipelines differ only in Stage II synthesis objectives. We implement stages (1) and (2) of FreeMask—layout-conditioned generation and noise filtering—but exclude the hardness-based re-sampling (stage 3) as it requires additional forward passes and iterative refinement orthogonal to the core synthesis mechanism being compared.

Experimental Setup. We evaluate on ADE20K at a 1% compression ratio (202 images) across six backbone architectures spanning both Mask2Former (Res-50, Swin-T, Swin-S, Swin-B) and SegFormer (MiT-B2, MiT-B4). All methods use identical Stage-I masks selected by our method, Swin-T for guidance during synthesis, and the same downstream training protocols. We report FreeMask results both without and with relabeling to isolate the contribution of mask-image alignment. Uniform selection serves as the real-image baseline. All results are reported as means±std over 5 runs, with ours corresponding to the experiments from Table 1.

Table 8: Comparison with FreeMask on ADE20K at 1% compression (202 images, mIoU %). All methods use identical Stage-I selected masks, differing only in Stage II synthesis objectives.

Method	Relabel	Res-50	Swin-T	Swin-S	Swin-B	MiT-B2	MiT-B4
Uniform	–	17.20±0.3	17.94±0.2	21.42±0.3	22.70±0.4	20.36±0.3	23.48±0.4
FreeMask	×	15.09±0.1	15.42±0.4	18.29±0.3	21.95±0.2	19.60±0.2	21.77±0.2
FreeMask	✓	17.27±0.2	18.10±0.3	21.62±0.2	24.76±0.2	21.29±0.2	24.90±0.3
Ours	✓	**20.96**±0.3	**21.69**±0.3	**24.99**±0.5	**27.48**±0.5	**23.82**±0.4	**28.17**±0.5

Results and Analysis. Three patterns emerge from Table 8: **(1) GDA fails at extreme compression.** Without relabeling, FreeMask falls *below* the Uniform baseline across all six backbones by –0.75to –3.13 mIoU, meaning its synthetic images train worse than the real images whose masks produced them. With relabeling, FreeMask barely edges past Uniform by only +0.16 to +2.06 mIoU (mean +0.81), demonstrating that a GDA pipeline at 100:1 compression barely justifies generation at all. In contrast, our method surpasses Uniform by +3.46 to +4.78 mIoU (mean +4.00). **(2) Alignment as the primary bottleneck.** Relabeling lifts FreeMask by up to +3.33 mIoU but our pipeline by only +0.74 mIoU (from Table 3), indicating FreeMask's generated images drift significantly from the masks used as ground truth. Our $\mathcal{L}_{\text{seg}}$ enforces alignment *during* sampling rather than via post-hoc correction, explaining the smaller relabeling gain. **(3) Dataset-level statistics matter.** The remaining 2.5–4.2 mIoU advantage after accounting for alignment is attributable to $\mathcal{L}_{\text{feat}}$, which matches dataset-level feature statistics aggregated over real training images. A GDA objective has no reason to match such statistics, as expansion never requires one sample to be representative—redundancy across the augmented set and retention of real data make individual sample quality less critical. In distillation, every sample must encode dataset-level information to compensate for 100:1 compression, necessitating guidance objectives beyond per-image mask conditioning.



Figure 10: Qualitative comparison between FreeMask and D^{3}S^2 on ADE20K. For each example, we show: real image, segmentation mask, FreeMask synthesis, and our synthesis. Our method demonstrates superior mask-image alignment, particularly for small objects, and generates textures that better capture class-intrinsic attributes.

Qualitative Analysis. Figure 10 visually corroborates the quantitative findings: **(1) Stronger spatial alignment.** Our synthesized images adhere more faithfully to input masks compared to

FreeMask, especially for small and thin objects where precise boundaries are critical. FreeMask's generations exhibit noticeable spatial drift—objects appear displaced, truncated, or with incorrect boundaries—explaining why relabeling provides large gains (+3.33 mIoU) for FreeMask but minimal gains (+0.74 mIoU) for ours. Our $\mathcal{L}_{\text{seg}}$ enforces pixel-wise consistency during sampling, ensuring geometry matches supervision from the outset. **(2) Class-intrinsic texture quality.** Beyond alignment, our samples exhibit textures and appearances that better capture essential attributes of each semantic category. This is attributable to $\mathcal{L}_{\text{feat}}$, which guides synthesis toward feature distributions measured from real training images, ensuring each sample encodes dataset-level representativeness rather than arbitrary visual diversity. **(3) Training utility.** The combination of precise alignment and class-representative appearance makes our synthesized samples significantly more *informative*—each image carries clearer supervision signals and more canonical exemplars of its constituent classes. In the distillation regime where every sample must compensate for 100:1 compression, this informativeness directly translates to improved downstream performance, as evidenced by consistent 2.5–4.8 mIoU gains across all architectures.

Discussion. This comparison clarifies the distinction between mask-conditioned synthesis as a *mechanism* versus its *objective*. FreeMask and similar GDA methods optimize for diversity and coverage under the assumption that weak samples are harmless when training includes the full real dataset. Our GDD setting inverts this: the synthesized set must *replace* real data, requiring each sample to be informationally dense and aligned with dataset-level statistics. The results validate that simply applying a GDA pipeline to extreme compression ratios is insufficient—distillation requires objectives that explicitly target representativeness and alignment, which our two-stage design addresses through balanced selection and feature-matching guidance.

E.4 Class-Wise Performance Analysis

Motivation. Aggregate mIoU can obscure performance on categories that matter most for dataset distillation: rare classes nearly absent from random subsets, and difficult classes that are intrinsically hard to synthesize and segment. To provide fine-grained insights, we conduct a class-wise analysis on two groups of five classes per dataset: **rare**, the lowest-frequency classes in the original training set, and **difficult**, classes that are *not* rare but intrinsically challenging due to small size, thin structure, or visual ambiguity. This separation disentangles the failure modes of scarcity from those of learnability.

Experimental Setup. We select the 5 lowest-frequency classes as the rare group for each dataset. For the difficult group, we manually identify 5 classes that appear with moderate frequency but pose inherent challenges: thin structures (*poles*, *streetlights*), small objects (*hair drier*), and ambiguous categories (*flags*). The reference baseline is Uniform, the strongest coreset method that also enforces class balance, isolating what Stage II synthesis adds beyond selection. Experiments follow the main paper's configuration: Mask2Former (Swin-T), 2% distillation on ADE20K and 1% on COCO-Stuff, averaged over 5 runs. We report per-class IoU, original and distilled class frequencies, and group means. Per-class IoU variance is omitted for brevity (all $\leq$0.7).

Results and Analysis. Four key patterns emerge from Tables 9 and 10 and Figures 11 and 12: **(1) Gains concentrate where they matter most.** On ADE20K, rare classes gain +13.35 IoU (6.48 $\rightarrow$ 19.83) and difficult classes +7.15 IoU (8.38 $\rightarrow$ 15.53), compared to an all-class gain of only +2.94—representing $4.5\times$ and $2.4\times$ amplification, respectively. On COCO-Stuff, the pattern is even stronger: +7.84 IoU for rare classes and +10.31 IoU for difficult classes, versus +2.23 overall ($3.5\times$ and $4.6\times$ amplification). This demonstrates that aggregate mIoU significantly understates what our method accomplishes for the categories that matter most in dataset distillation—those that are scarce or inherently challenging. **(2) Two mechanisms map onto two stages.** For rare classes, Stage I is the primary driver: representation increases by $120\times$ on ADE20K (0.0031% $\rightarrow$ 0.37%) and $146\times$ on COCO-Stuff (0.0024% $\rightarrow$ 0.35%), both converging to the 0.35–0.40% coverage floor that balanced selection guarantees at these compression ratios. This direct exposure lift explains the IoU jumps. For difficult classes, however, representation is *not* the explanation. On ADE20K, their frequency actually *decreases* (0.60% $\rightarrow$ 0.48%), with *streetlight* dropping from 1.20% to 0.59%, yet IoU still gains +7.15. On COCO-Stuff, the modest increase (0.25% $\rightarrow$ 0.40%) cannot account for the +10.31 gain. What changes is the *content* of each sample—Stage II synthesizes cleaner, more canonical instances than the cluttered real images they replace, as visualized in Figures 11 and 12. Thus, Stage I fixes scarcity through balanced selection, while Stage II fixes learnability through informative synthesis. **(3) Balance over real images is insufficient.** Uniform also enforces

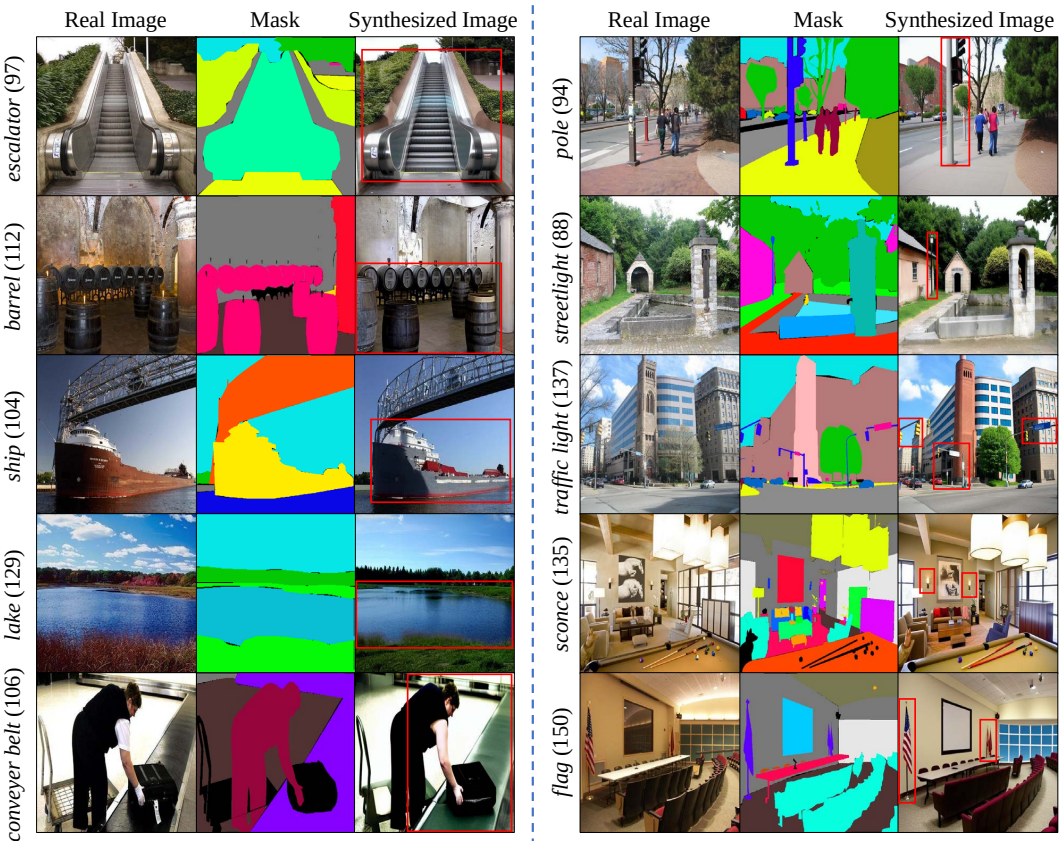
Table 9: Class-wise performance on ADE20K at 2% compression (mIoU %). Freq. indicates the fraction of images containing each class in the original vs. distilled set.

Group	Class (Index)	Freq. (orig.)	Freq. (ours)	Uniform	**Ours**
Rare	escalator (97)	0.0025%	0.38%	6.30	**18.59**
	barrel (112)	0.0030%	0.38%	1.29	**18.71**
	ship (104)	0.0032%	0.38%	5.98	**16.64**
	lake (129)	0.0032%	0.38%	7.48	**15.10**
	conveyer belt (106)	0.0035%	0.34%	11.33	**30.13**
	Rare mean	0.0031%	0.37%	6.48	**19.83**
Difficult	pole (94)	0.60%	0.47%	9.75	**11.18**
	streetlight (88)	1.20%	0.59%	11.64	**19.65**
	traffic light (137)	0.30%	0.34%	9.73	**12.36**
	sconce (135)	0.62%	0.59%	10.79	**17.48**
	flag (150)	0.26%	0.42%	0.00	**16.98**
	Difficult mean	0.60%	0.48%	8.38	**15.53**
All-class mIoU		–	–	23.19	**26.13**

Table 10: Class-wise performance on COCO-Stuff at 1% compression (mIoU %). Freq. indicates the fraction of images containing each class in the original vs. distilled set.

Group	Class (Index)	Freq. (orig.)	Freq. (ours)	Uniform	**Ours**
Rare	waterdrops (168)	0.0012%	0.35%	2.82	**9.70**
	hair drier (79)	0.0020%	0.35%	1.40	**12.32**
	toaster (71)	0.0023%	0.35%	22.89	**30.50**
	moss (123)	0.0027%	0.35%	7.80	**14.10**
	ceiling-tile (92)	0.0036%	0.35%	11.96	**19.27**
	Rare mean	0.0024%	0.35%	9.34	**17.18**
Difficult	toothbrush (80)	0.09%	0.35%	10.71	**15.18**
	scissors (77)	0.10%	0.35%	21.29	**44.06**
	spoon (45)	0.33%	0.38%	2.76	**9.54**
	knife (44)	0.41%	0.54%	4.85	**10.28**
	fork (43)	0.33%	0.37%	7.88	**19.73**
	Difficult mean	0.25%	0.40%	9.45	**19.76**
All-class mIoU		–	–	28.96	**31.19**

class balance, yet leaves several categories nearly unlearnable: *flag* at 0.00 IoU, *barrel* at 1.29, *hair drier* at 1.40, *spoon* at 2.76, and *waterdrops* at 2.82. Our distilled set lifts these to 16.98, 18.71, 12.32, 9.54, and 9.70, respectively—gains of +17.42 to +6.88 IoU from near-zero baselines. This reveals a critical insight: at extreme compression, a few real images where a rare class appears small and occluded provide little learnable signal. Simply balancing class coverage is not enough—what matters is the *informativeness* of each sample. Our Stage II guidance ensures that every synthesized image presents clear, class-discriminative features aligned with dataset-level statistics, making each sample carry more training utility than its real counterpart. **(4) Honest limits and concrete remedies.** Absolute IoU remains modest for the thinnest structures: *pole* gains only +1.43 (9.75 $\rightarrow$ 11.18), and categories like *spoon* (9.54), *waterdrops* (9.70), and *toothbrush* (15.18) remain relatively low despite significant improvements. This limitation traces to the layout-to-image generator's fidelity on near-pixel-wide geometry rather than a flaw in our objective design. Since Stage I supplies *real* masks with exact boundaries, what limits these classes is the generator's rendering competence—consistent with our findings in the FreeMask comparison, where a weaker prior costs appearance realism while geometry and supervision survive. Stronger diffusion backbones (e.g., higher-resolution generators) or boundary-aware guidance terms are concrete remedies that future work can explore. Importantly, even with this bottleneck, every single class improves over Uniform, and substantial gains (+8.01



(a) Rare Classes (b) Difficult Classes

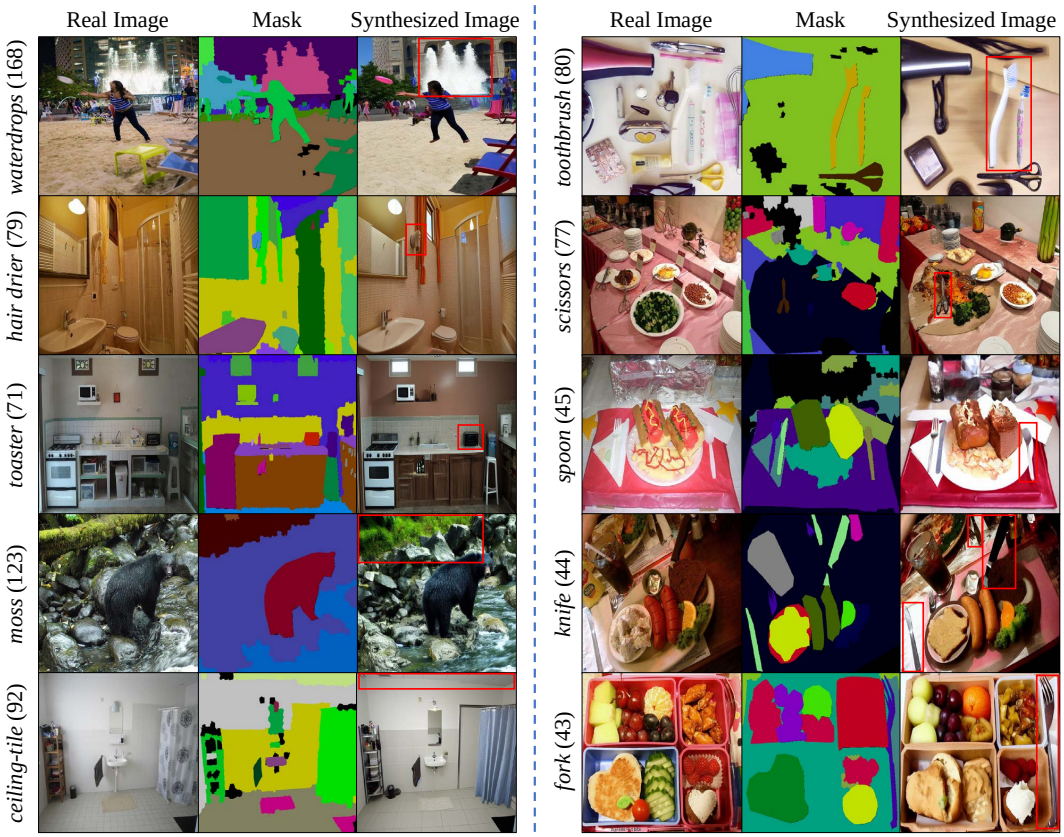
Figure 11: Qualitative examples for rare and difficult classes on ADE20K. For each class, we show the real image, segmentation mask, and our synthesized image. Stage I lifts rare class representation to 0.35–0.40%, while Stage II generates clean, canonical instances that enhance learnability even for thin structures.

for *streetlight*, +22.77 for *scissors*) demonstrate that the framework is not fundamentally limited to common or easy categories.

E.5 Comparison with Advanced Coreset Selection Methods

Motivation. The baselines in the main paper (Random, K-Center, Herding) represent foundational coreset principles but do not reflect recent advances. To rigorously validate our class-balanced selection strategy, we compare against three state-of-the-art coreset methods that embody modern design principles: **Coverage-centric Coreset Selection (CCS)** [58] optimizes coverage over feature space with high pruning rates; **D2Pruning** [22] balances diversity and sample difficulty via message passing; and **Structural-Entropy-based Sample Selection (SES)** [47] scores samples by their structural entropy contribution. These methods represent the frontier of coreset research in classification, and comparing against them tests whether class-level coverage is genuinely necessary or whether advanced diversity-based selection suffices.

Experimental Setup. We evaluate all three methods on ADE20K using Mask2Former (Swin-S) across compression ratios of 0.5%, 1%, and 2%. For fair comparison, we use each method's official implementation with default hyperparameters. We adapt CCS, D2Pruning, and SES to semantic segmentation using image-level difficulty estimates derived from a Mask2Former model trained on the full ADE20K training set. SES and D2Pruning additionally construct image-similarity graphs from global Swin-S features, extracted following the same protocol as all operate on global image-level features extracted from a pretrained Swin-S backbone (same as used for K-Center and Herding). All



(a) Rare Classes (b) Difficult Classes

Figure 12: Qualitative examples for rare and difficult classes on COCO-Stuff. For each class, we show the real image, segmentation mask, and our synthesized image. Our method generates visually distinct and class-discriminative instances even for small, ambiguous objects and uncommon stuff categories.

Table 11: Comparison with advanced coreset selection methods on ADE20K using Mask2Former (Swin-S), mIoU (%).

Method	0.5%	1%	2%
Random	10.98±0.5	15.65±0.5	21.70±0.4
K-Center	10.52±0.4	13.76±0.4	17.67±0.3
Herding	12.47±0.4	16.32±0.3	22.81±0.4
Uniform	15.04±0.2	21.42±0.3	27.59±0.3
CCS	8.25±0.4	10.67±0.4	17.61±0.3
SES	13.64±0.3	17.81±0.3	24.33±0.2
D2Pruning	14.51±0.5	19.28±0.4	24.50±0.3
Ours	**19.45**±0.5	**24.99**±0.5	**30.28**±0.5
Full Dataset	*51.22*±0.1	*51.22*±0.1	*51.22*±0.1

methods select real images, which are then used for segmentation training following the protocol in Table 1. Results are averaged over 5 runs.

Results and Analysis. Three key findings emerge from Table 11: **(1) Advanced methods are stronger, and our margin holds.** D2Pruning and SES substantially outperform the foundational baselines, confirming they represent the state of the art. D2Pruning gains +1.04 to +2.04 over Herding and +1.53 to +3.63 over Random, while SES gains +1.17 to +1.49 over Herding. Despite this,

D^3S^2 maintains significant advantages: +4.94 mIoU over D2Pruning at 0.5%, +5.71 at 1%, and +5.78 at 2%. Crucially, the margin *grows* as the budget tightens—the regime dataset distillation targets—demonstrating that our approach is not merely competitive but addresses a fundamentally different bottleneck. **(2) Class balance is the right principle for dense prediction under extreme compression.** Remarkably, Uniform—the simplest class-balanced baseline with no learned scoring—outperforms all three advanced methods at every compression ratio. The gap widens from +0.53 mIoU over D2Pruning at 0.5% to +2.14 at 1% and +3.09 at 2%. This is not a defect of CCS, SES, or D2Pruning but a consequence of the task shift from classification to dense prediction. All three methods assign a scalar score to each whole image, which is well-defined when one image carries one label. In semantic segmentation, a multi-class scene contains 10–30 overlapping categories, and a scalar score necessarily averages away the tail signal—an image containing one rare class among twenty common ones becomes indistinguishable from an ordinary scene. With no mechanism enforcing per-class coverage, ADE20K's severe imbalance (Imbalance Factor = 282) survives selection, leaving tail categories nearly absent from the distilled subset. **(3) Synthesis amplifies the advantage.** While Uniform beats D2Pruning by +2.14 to +3.09 mIoU, our full pipeline extends this to +5.71 to +5.78, demonstrating that Stage II synthesis compounds the benefit of balanced selection. The selected masks provide diverse, class-representative layouts, and guided diffusion refines them into canonical, informative training samples. This two-stage design—class-aware selection followed by feature-matching synthesis—proves essential for segmentation distillation, where both coverage and sample quality matter more than in classification.

Discussion. This comparison clarifies a broader principle: coreset methods designed for classification excel at capturing global diversity and sample difficulty, but these criteria are *class-agnostic*. For tasks with severe long-tail distributions—where performance hinges on tail coverage—scalar image scoring cannot distinguish a rare-class-bearing image from a common one. Our class-balanced selection explicitly tracks per-class coverage, ensuring no category is neglected. Combined with diffusion-guided synthesis, this yields a distilled set that is both balanced and informative, outperforming even the most advanced diversity-based methods by a substantial margin.

F Limitations and Future Works

Despite the strong empirical performance, D^3S^2 has several limitations that suggest promising directions for future research. **First**, our framework relies on a layout-to-image diffusion model pretrained on the target dataset. When such a model is unavailable, a domain gap may arise, leading to degraded generation quality and reduced effectiveness of the distilled data. In future work, this limitation could be alleviated by leveraging more general-purpose diffusion models with stronger cross-domain generalization or by introducing lightweight adaptation techniques (e.g., parameter-efficient fine-tuning) to quickly align pretrained models with new target domains. **Second**, the proposed guided sampling strategy requires a pretrained segmentation model as the guidance network. For datasets lacking publicly available checkpoints, training such a model from scratch introduces additional computational overhead. A potential direction is to reduce this dependency by exploring self-supervised or weakly supervised alternatives for guidance, or by distilling transferable segmentation priors from large-scale foundation models to new domains with minimal supervision. **Finally**, our method directly selects masks from the original dataset, which may not be optimal under extreme compression ratios. The selected layouts are constrained by existing data distributions and cannot fully explore the space of informative semantic configurations. A promising direction is to move beyond selection-based strategies and instead jointly synthesize images and layouts [40]. Such a layout–image co-generation could enable more flexible and diverse semantic compositions, potentially leading to more compact and informative distilled datasets.

G Broader Impacts

This work introduces a diffusion-based framework for dataset distillation in semantic segmentation, enabling efficient training with significantly reduced data. By lowering the data and computational requirements, our method can facilitate research and deployment in resource-constrained settings, such as edge devices or domains where large-scale annotation is costly (e.g., medical or environmental applications). However, as the method relies on generative models, it may inherit biases present in the original dataset, potentially amplifying class imbalance or under-representation in downstream

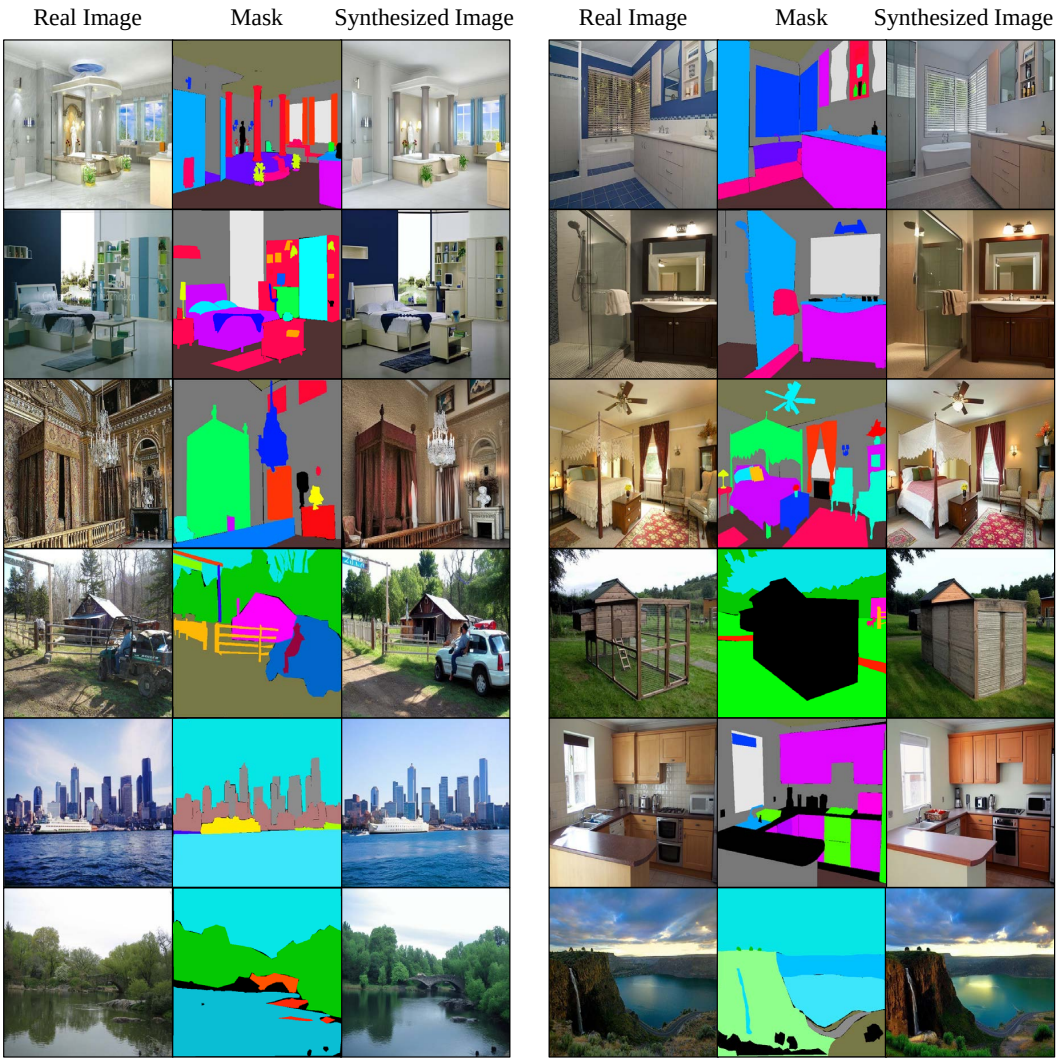


Figure 13: Additional qualitative results on ADE20K. Each example shows the real image, segmentation mask, and synthesized image generated by D^3S^2, demonstrating strong layout alignment and clear semantic structures.

tasks. In addition, synthesized data may introduce artifacts or unrealistic patterns that could affect model robustness if not properly controlled. There is also a risk that synthetic datasets could be misused to obscure data provenance or bypass data governance policies. Future work should explore bias mitigation, improved controllability of generation, and mechanisms for ensuring transparency and traceability of synthetic data. Overall, we believe this line of research has the potential to make data-efficient learning more accessible while highlighting the importance of responsible use of generative models.

H More Visualizations

Figure 13 and Figure 14 present additional qualitative results on ADE20K and COCO-Stuff, respectively. Each example includes the real image, the corresponding segmentation mask, and the synthesized image generated by D^3S^2. Consistent with the observations in the main paper, the synthesized images exhibit strong spatial alignment with the input masks across diverse scenes. Object boundaries and spatial structures are well preserved, indicating effective pixel-level consistency during generation. Moreover, the generated samples highlight class-discriminative features while suppressing irrelevant variations, resulting in cleaner and more canonical representations compared

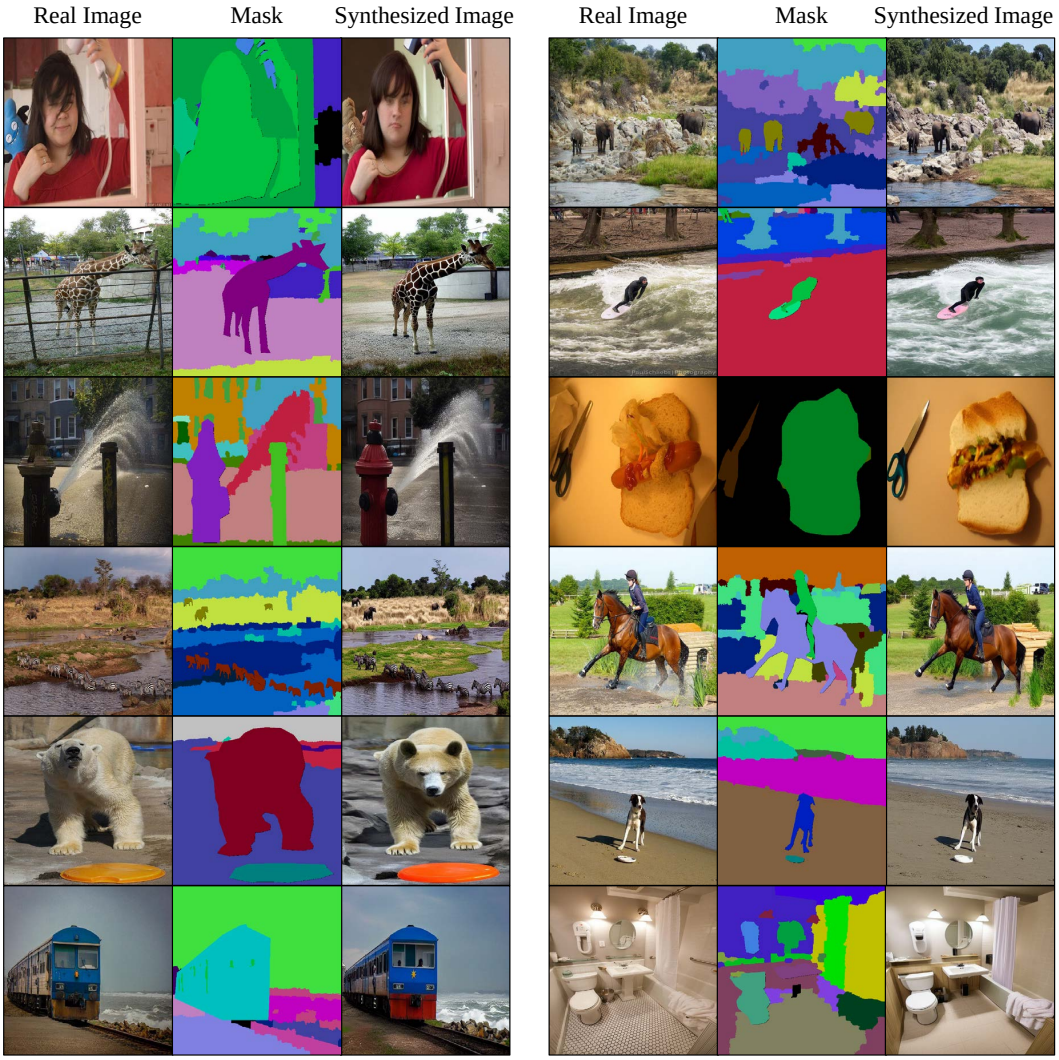


Figure 14: Additional qualitative results on COCO-Stuff. The synthesized images exhibit consistent alignment with input masks and emphasize class-discriminative features across diverse and complex scenes.

to real images. Even in complex layouts with multiple co-occurring objects, the synthesized images maintain clear foreground structures and semantic coherence. These results further demonstrate that the proposed diffusion-guided framework produces structurally faithful and semantically informative samples, which are well-suited for training segmentation models.

I Large Language Model Usage

We use ChatGPT (GPT-5, by OpenAI) as a writing assistant to polish the language of this paper. Specifically, it was employed to improve clarity, grammar, and style in certain sections, but it did not contribute to research ideation, technical content, experiments, or analysis. All scientific contributions, methods, and conclusions are solely the work of the authors. The authors take full responsibility for the accuracy and integrity of the paper's contents.

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: Our abstract and introduction accurately reflect the paper's contributions to diffusion-guided dataset distillation for semantic segmentation.

   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: Yes, we discuss this in Appendix F.

   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: This paper does not include theorems or lemmas.

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: Yes, we provide sufficient implementation details in Section 5.1 and Appendix C.

   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: [Yes]

Justification: We provide code in the supplementary materials and provide a GitHub link in the Abstract and Appendix C.

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: Experimental setting is described in Section 5.1 and Appendix C.

   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: We report error bars in Tables 1 to 4.

   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: We provide the computer resources in Section 5.1 and Appendix C.

   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: Our research conform with the NeurIPS Code of Ethics.

   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: [Yes]

    Justification: We discuss the broader impact in Appendix G.

    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 poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All assets used in the paper, including datasets, are publicly available. Proper credits are given to the creators or original owners of these datasets where applicable. The licenses and terms of use for these datasets are explicitly mentioned and respected in accordance with their respective guidelines.

    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: [Yes]

    Justification: We have attached our code and user instructions in the supplementary materials.

    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 does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

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