Dataset Distillation via Saliency-Guided Initialization and Adaptive Gaussian Mixture Frequency Sampling

Anonymous authors
Paper under double-blind review

Abstract

Dataset distillation aims to compress a large training set into a compact synthetic set while preserving its training utility. However, in existing distribution matching methods, representation construction and frequency sampling are typically treated as independent components, which may introduce alignment bias between different representation states. In this paper, we propose a novel dataset distillation framework, termed Saliency-guided Initialization with Gaussian Mixture Sampling (SIGMS), which addresses both weak synthetic-data initialization and the limited adaptability of frequency sampling to varying representation states through two complementary mechanisms. Firstly, we propose SaliPatch, a saliency-guided patch initialization module that selects and stitches informative patches from real images, reducing background interference and improving the quality of synthetic initialization. Secondly, we jointly introduce Stage-Conditioned Feature Interpolation and Adaptive SampleNet, where intermediate representations are constructed between the frozen initial and expert models, while frequency sampling is adaptively adjusted to the corresponding representation states through a shared stage variable, thereby reducing the mismatch between feature representations and frequency probing during distribution alignment. Extensive experiments on multiple benchmark datasets demonstrate that SIGMS achieves the best performance among the compared methods at IPC=1 and remains competitive at higher IPC settings.

1 Introduction

Deep neural networks (DNNs) have achieved remarkable progress across a wide range of tasks, largely driven by the increasing scale of training data and model capacity [17, 22]. However, the growing demand for data also introduces substantial storage and computational costs. Dataset distillation (DD) aims to alleviate this burden by compressing a large real training set into a compact synthetic set while preserving its training utility [40]. Due to its efficiency, dataset distillation has been applied to neural architecture search [12, 34], continual learning [28], medical image analysis [29], and privacy-preserving learning [7, 11].

Among existing DD paradigms, distribution matching has attracted increasing attention because it avoids the expensive nested optimization required by gradient- or trajectory-matching approaches [14, 35, 6], while directly aligning the feature distributions of real and synthetic data. Recent work has further improved distribution matching through increasingly expressive discrepancy measures, including characteristic-function-based modeling [8, 26, 31, 38]. Nevertheless, effective distribution matching depends not only on how discrepancies are measured, but also on how synthetic optimization is initialized and how feature distributions are probed across evolving representation states.

This issue is particularly important because the discrepancy between real and synthetic data can vary across different representation states [31]. Early representations mainly capture coarse or weakly discriminative information, whereas expert representations encode richer task-related semantics [31]. Meanwhile, characteristic-function matching critically depends on the sampled frequency arguments used to probe these distributions [38]. Frequency samples that are informative for one representation state may be less effective for another.

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

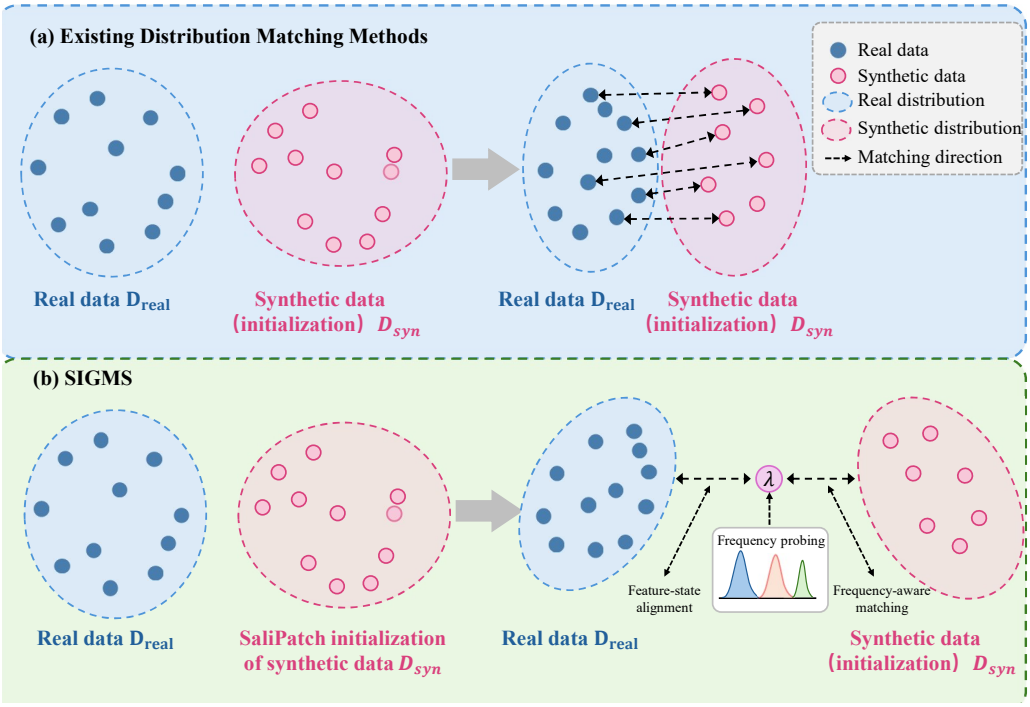

Figure 1: Distribution Matching and SIGMS. Existing methods typically start from weak synthetic initialization and perform distribution matching without explicitly coordinating representation states with frequency probing. In contrast, SIGMS constructs a more informative synthetic prior through SaliPatch and couples feature-state alignment with adaptive frequency sampling through a shared stage variable λ, enabling stage-coordinated distribution matching.

Therefore, representation construction and frequency sampling should be coordinated rather than treated independently.

In highly compressed settings, this challenge is further amplified by weak initialization, since each synthetic sample must preserve as much discriminative information as possible. Figure 1 illustrates this distinction: existing distribution-matching paradigms typically rely on weak synthetic initialization [14, 31] and do not explicitly coordinate representation states with frequency probing, whereas SIGMS improves the synthetic prior and couples the two through a shared stage variable.

Motivated by these observations, we propose Saliency-guided Initialization with Gaussian Mixture Sampling (SIGMS), a unified dataset distillation framework that improves both the semantic starting point and the stage-dependent distribution-matching process. SIGMS first introduces SaliPatch, which extracts high-response patch candidates from real images using class activation maps, performs IPC-aware difficulty-diversity selection, and stitches the selected patches to construct the initial synthetic set. This provides a more informative prior for subsequent optimization than random or weak initialization.

To better capture stage-dependent discrepancies, SIGMS further constructs dual-anchor feature trajectories between a frozen initial model and a frozen expert model. Stage-Conditioned Feature Interpolation generates intermediate feature states, while Adaptive SampleNet uses the same stage variable to control Gaussian-mixture frequency sampling. This shared conditioning explicitly coordinates feature-space alignment with frequency-space probing, enabling the sampled frequency arguments to better adapt to different representa-

tion states. In this way, SIGMS jointly improves the starting point of synthetic optimization and the subsequent characteristic-function-based distribution matching process.

Extensive experiments on CIFAR-10/100, Tiny ImageNet, and ImageNet subsets demonstrate the effectiveness of SIGMS across diverse compression settings. SIGMS achieves particularly strong gains in highly compressed low-IPC regimes while maintaining competitive performance as the IPC increases.

Our main contributions are summarized as follows:

- We propose SIGMS, a stage-coordinated dataset distillation framework that explicitly couples representation-state construction with frequency-space probing during characteristic-function matching.
- We introduce SaliPatch, which combines CAM-guided patch extraction with IPC-aware difficulty-diversity selection and stitching to construct informative synthetic priors.
- We introduce shared stage conditioning, where Stage-Conditioned Feature Interpolation and Adaptive SampleNet are controlled by the same stage variable, enabling frequency probing to adapt to the representation state being aligned.
- Extensive evaluations across CIFAR-10/100, Tiny ImageNet, and ImageNet subsets show that SIGMS achieves particularly strong gains under highly compressed settings while remaining competitive as the IPC increases.

2 Related Work

2.1 Dataset Distillation and Distribution Matching

Dataset distillation aims to synthesize a compact training set that preserves the training utility of the full dataset [40]. Existing methods can be broadly categorized into optimization-oriented and distribution-matching-based paradigms. Optimization-oriented methods, such as gradient matching and trajectory matching, optimize synthetic data by aligning gradients or long-term training trajectories between real and synthetic datasets [35, 6, 9]. Although effective, these methods typically involve nested optimization or unrolled training, resulting in considerable memory and computational costs.

Distribution matching (DM) provides a more efficient alternative by directly aligning feature distributions of real and synthetic data [14]. Subsequent works improve DM from different perspectives, including improved distribution matching [41], feature regression [36], inter-sample and inter-feature relation modeling [10], maximum-mean-discrepancy-based matching [8], diversified semantic distribution matching [26], dual-view distribution alignment [31], discriminative feature preservation [42], and characteristic-function-based matching [38]. Despite these advances, representation construction and discrepancy probing are generally developed independently, leaving their explicit stage-wise coordination underexplored.

2.2 Initialization Strategies in Dataset Distillation

The initialization of synthetic data can substantially affect the subsequent optimization process, particularly when the synthetic-data budget is highly constrained. Existing dataset distillation methods commonly initialize synthetic samples from random noise or randomly selected real images [14, 31]. While noise initialization provides little semantic guidance, random real-image initialization may retain redundant background content and does not explicitly control the informativeness or diversity of the selected visual regions.

Recent studies have explored more informative initialization and data-construction strategies. For example, RDED selects representative image patches and reconstructs distilled samples through patch composition [16], while SelMatch adopts selection-based initialization to better adapt synthetic data to different IPC regimes [43]. More recently, diffusion-driven patch selection has been explored to identify distinctive regions while preserving intra-class diversity [44]. Other works exploit label augmentation and additional semantic supervision

to improve the informativeness of distilled data [19, 37]. Nevertheless, how to construct an initialization that simultaneously preserves discriminative local semantics and maintains sufficient intra-class diversity remains underexplored, especially under severe compression. Our SaliPatch addresses this issue through CAM-guided candidate extraction, IPC-aware difficulty-diversity selection, and patch stitching.

2.3 Discrepancy Measurement and Frequency Sampling

Accurate discrepancy measurement in high-dimensional feature spaces is central to distribution-matching-based dataset distillation. Early approaches mainly align feature representations or low-order statistics [14, 27, 36], while later methods introduce more expressive distributional objectives, including maximum mean discrepancy (MMD) [5, 8]. However, moment-based alignment may provide limited characterization of complex feature distributions.

More recently, characteristic-function-based distribution modeling has been explored as a more expressive alternative [1, 38]. In particular, NCFM compares real and synthetic feature distributions in the complex plane through both amplitude and phase information and employs a learnable sampling network to optimize the frequency arguments used for characteristic-function matching [38]. This represents an important step beyond fixed frequency sampling. Meanwhile, DANCE improves distribution alignment by constructing intermediate representation states between initialized and expert encoders [31]. However, these two directions—representation-state modeling and frequency-space probing—are largely treated independently. Existing frequency sampling mechanisms are generally not explicitly conditioned on the representation state being aligned. In contrast, SIGMS couples Stage-Conditioned Feature Interpolation with Adaptive SampleNet through a shared stage variable, enabling frequency probing to adapt jointly with the corresponding feature state.

3 Method

3.1 Preliminaries and Overview

Let $\mathcal{T} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{T}|}$ denote the real training set and $\mathcal{S} = \{(\tilde{x}_j, \tilde{y}_j)\}_{j=1}^{|\mathcal{S}|}$ the synthetic set, where $|\mathcal{S}| \ll |\mathcal{T}|$. Dataset distillation aims to optimize $\mathcal{S}$ such that a model trained on $\mathcal{S}$ achieves performance comparable to one trained on $\mathcal{T}$.

Under the distribution-matching paradigm, synthetic data are optimized by aligning the class-conditional feature distributions of the real and synthetic sets. Given a feature extractor $\phi(\cdot)$, the objective can be written as

$$\min_{\mathcal{S}} \sum_{c=1}^{C} \mathcal{D}\left(P_{\phi(x)|y=c},\, P_{\phi(\tilde{x})|\tilde{y}=c}\right), \tag{1}$$

where $\mathcal{D}(\cdot,\cdot)$ denotes a discrepancy measure between the real and synthetic feature distributions.

Recent characteristic-function-based methods provide a more expressive way to measure such discrepancies [38]. For a feature representation $z = \phi(x)$, its empirical characteristic function evaluated at frequency argument t is

$$\Phi_z(t) = \mathbb{E}_z\left[\exp\left(it^\top z\right)\right], \tag{2}$$

where t denotes the frequency argument used to probe the feature distribution. Accordingly, real and synthetic feature distributions can be compared through their characteristic functions evaluated at sampled frequency arguments. The choice of t determines which directions of the feature distribution are emphasized during discrepancy estimation. Although learnable frequency sampling improves the flexibility of characteristic-function matching [38], the sampling behavior is generally not explicitly coordinated with the representation state being aligned.

To address this issue while providing a more informative starting point for synthetic optimization, we propose SIGMS, whose overall framework is illustrated in Figure 2.

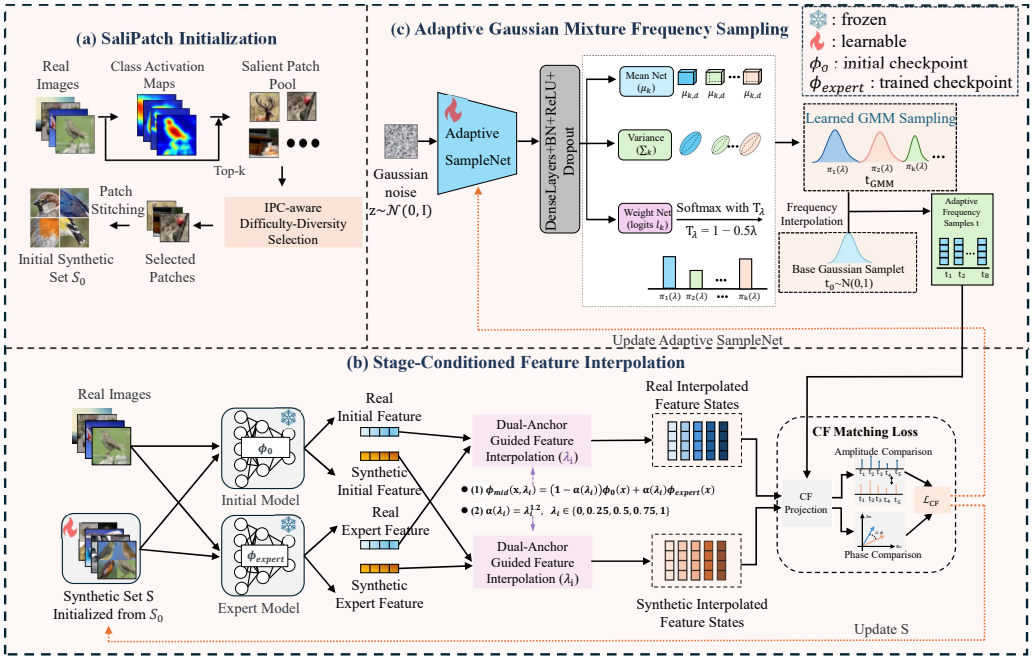

Figure 2: Overview of SIGMS. SaliPatch constructs a semantically informative initialization for the synthetic set, while Stage-Conditioned Feature Interpolation and Adaptive SampleNet are coupled through a shared stage variable λ to coordinate feature-state construction and frequency-space probing for characteristic-function matching.

SIGMS consists of two complementary designs. First, SaliPatch constructs a semantically informative initial synthetic set through CAM-guided candidate extraction, IPC-aware difficulty-diversity selection, and patch stitching. Second, SIGMS couples Stage-Conditioned Feature Interpolation with Adaptive SampleNet through a shared stage variable λ, so that feature-state construction and frequency-space probing evolve in a coordinated manner.

3.2 SaliPatch Initialization

Under severe compression, the quality of synthetic initialization becomes particularly important, as each synthetic sample must preserve sufficient discriminative information. Random noise provides little semantic guidance, while randomly selected real images may contain redundant backgrounds. We therefore introduce SaliPatch, which constructs an informative initial synthetic set from salient and diverse local regions of real images.

Given a real image x, we compute its class activation map and partition the image into non-overlapping patches

$$\mathcal{P}(x) = \{p_m\}_{m=1}^{M}.$$

Let a_m denote the average activation of patch p_m. We retain the top-K_p patches with the highest activation scores to form a salient candidate pool:

$$\mathcal{I}_{\text{sal}}(x) = \underset{m \in \{1,\dots,M\}}{\arg\operatorname{topK}^{K_p}} a_m, \qquad \mathcal{P}_{\text{sal}}(x) = \{p_m \mid m \in \mathcal{I}_{\text{sal}}(x)\}. \tag{3}$$

To avoid selecting visually redundant patches, we further perform IPC-aware difficulty-diversity selection. For each candidate p_m, its difficulty is measured by

$$d_m = 1 - \max_c q_c(p_m),$$

where $q_c(p_m)$ denotes the predicted probability of class c. Its diversity with respect to the currently selected patch set $\mathcal{Q}$ is defined as

$$v_m = \min_{p_j \in \mathcal{Q}} \left[1 - \cos\left(f(p_m), f(p_j)\right)\right]. \tag{4}$$

We rank the candidate patches using a weighted combination of difficulty and diversity,

$$s_m = (1-\beta)d_m + \beta v_m, \tag{5}$$

where β controls the trade-off between difficulty and diversity according to the IPC setting.

Finally, the selected patches are arranged into an $F \times F$ grid to construct each synthetic image, producing the initial synthetic set $\mathcal{S}_0$. Accordingly, each initialized synthetic image is composed of F^2 selected salient patches. SaliPatch therefore provides a semantically informative and diversity-aware starting point for subsequent distribution matching.

3.3 Stage-Conditioned Feature and Frequency Modeling

To capture distribution discrepancies at different representation states, we construct intermediate features between a frozen initial model ϕ_0 and a frozen expert model ϕ_{expert}. For a stage variable $\lambda_i \in [0,1]$, the representation of an input x is defined as

$$\phi_{\lambda_i}(x) = (1-\alpha(\lambda_i))\,\phi_0(x) + \alpha(\lambda_i)\phi_{\text{expert}}(x), \tag{6}$$

where $\alpha(\lambda_i) = \lambda_i^{1.2}$. The same interpolation is applied to both real and synthetic samples, producing a sequence of feature states from initial representations to expert-level semantics. In our implementation, $\lambda_i \in \{0, 0.25, 0.5, 0.75, 1\}$.

We further adapt the frequency arguments used for characteristic-function matching to the corresponding feature state. Given Gaussian noise $z \sim \mathcal{N}(0, I)$, Adaptive SampleNet predicts the parameters of a Gaussian mixture distribution,

$$p_{\lambda_i}(t) = \sum_{k=1}^{K} \pi_k(\lambda_i)\,\mathcal{N}\left(t; \mu_k, \Sigma_k^2 I\right), \tag{7}$$

where the mixture weights are obtained by temperature-scaled normalization,

$$\pi_k(\lambda_i) = \text{Softmax}\left(\frac{l_k}{T_{\lambda_i}}\right), \qquad T_{\lambda_i} = 1 - 0.5\lambda_i. \tag{8}$$

Let t_{GMM} denote a sample from the learned mixture and $t_0 \sim \mathcal{N}(0, I)$ a base Gaussian sample. The final frequency argument is constructed as

$$t_{\lambda_i} = (1-\rho(\lambda_i))\,t_0 + \rho(\lambda_i)t_{\text{GMM}}, \qquad \rho(\lambda_i) = \lambda_i^{1.5}. \tag{9}$$

By conditioning both feature interpolation and frequency sampling on the same stage variable λ_i, SIGMS explicitly coordinates representation-state construction with frequency-space probing. Early stages retain broader base-Gaussian exploration, while later stages place greater emphasis on the learned multimodal frequency distribution, enabling characteristic-function matching to better adapt to stage-dependent feature discrepancies.

3.4 Optimization Objective

Given the stage-conditioned representation $\phi_{\lambda_i}(\cdot)$ and the corresponding frequency argument t_{λ_i}, we perform characteristic-function matching between the real and synthetic feature distributions across all stages. The overall SIGMS objective is formulated as

$$\min_{\mathcal{S}} \max_{\psi} \mathcal{L}_{\text{SIGMS}} = \min_{\mathcal{S}} \max_{\psi} \sum_{i=1}^{L} w_i \left\| \mathbb{E}_{x\sim\mathcal{T}}\left[e^{\,\mathrm{i}t_{\lambda_i}^{\top}\phi_{\lambda_i}(x)}\right] - \mathbb{E}_{\tilde{x}\sim\mathcal{S}}\left[e^{\,\mathrm{i}t_{\lambda_i}^{\top}\phi_{\lambda_i}(\tilde{x})}\right] \right\|_2^2, \tag{10}$$

where w_i denotes the weight assigned to the i-th representation stage and ψ denotes the parameters of Adaptive SampleNet. The synthetic set $\mathcal{S}$ is optimized to minimize the discrepancy between real and synthetic feature distributions, whereas Adaptive SampleNet is updated in the opposite direction to identify informative frequency arguments. Through this alternating optimization, SIGMS jointly aligns stage-dependent feature representations while continually refining the frequency-space regions used for characteristic-function matching.

Table 1: Test accuracy (%) on CIFAR-10, CIFAR-100, and Tiny ImageNet under different IPC settings.

Method	CIFAR-10			CIFAR-100			Tiny ImageNet		
	IPC=1	IPC=10	IPC=50	IPC=1	IPC=10	IPC=50	IPC=1	IPC=10	IPC=50
Random	14.4±2.0	26.0±1.2	43.4±1.0	4.2±0.3	14.6±0.5	30.0±0.4	1.4±0.1	5.0±0.2	15.0±0.4
Herding	21.5±1.2	31.6±0.7	40.4±0.6	8.4±0.3	17.3±0.3	33.7±0.5	2.8±0.2	6.3±0.2	16.7±0.3
Forgetting	13.5±1.2	23.3±1.0	23.3±1.1	4.5±0.2	15.1±0.3	30.5±0.3	1.6±0.1	5.1±0.2	15.0±0.3
DC	28.3±0.5	44.9±0.5	53.9±0.5	12.8±0.3	25.2±0.3	–	–	–	–
DSA	28.8±0.7	52.1±0.5	60.6±0.5	13.9±0.3	32.3±0.3	42.8±0.4	–	–	–
DCC	32.9±0.8	49.4±0.5	61.6±0.4	13.3±0.3	30.6±0.4	40.0±0.3	–	–	–
FrePo	46.8±0.7	65.5±0.4	71.7±0.2	28.7±0.1	42.5±0.2	44.3±0.2	15.4±0.3	25.4±0.2	–
MTT	46.3±0.8	65.3±0.7	71.6±0.2	24.3±0.3	40.1±0.4	47.7±0.2	8.8±0.3	23.2±0.2	28.0±0.3
ATT	48.3±1.0	67.7±0.6	74.5±0.4	26.1±0.3	44.2±0.5	51.2±0.3	11.0±0.5	25.8±0.4	–
FTD	46.8±0.3	66.6±0.3	73.8±0.2	25.2±0.2	43.4±0.3	48.5±0.3	10.4±0.3	24.5±0.2	–
TESLA	48.5±0.8	66.4±0.8	72.6±0.7	24.8±0.4	41.7±0.3	47.9±0.3	–	–	–
DATM	46.9±0.5	66.8±0.2	76.1±0.3	27.9±0.2	47.2±0.4	55.0±0.2	17.1±0.3	31.1±0.3	39.7±0.3
CAFE	30.3±1.1	46.3±0.6	55.5±0.6	12.9±0.3	27.8±0.3	37.9±0.3	–	–	–
DM	26.0±0.8	48.9±0.6	63.0±0.4	11.4±0.3	29.7±0.3	43.6±0.4	3.9±0.2	12.9±0.4	24.1±0.3
IDM	45.6±0.7	58.6±0.1	67.5±0.1	20.1±0.3	45.1±0.1	50.0±0.2	10.1±0.2	21.9±0.2	27.7±0.3
DANCE	47.1±0.2	70.8±0.2	76.1±0.1	27.9±0.2	49.8±0.1	52.8±0.1	11.6±0.2	26.4±0.3	28.9±0.4
M3D	45.3±0.3	63.5±0.2	69.9±0.5	26.2±0.3	42.4±0.2	50.9±0.7	–	–	–
IID	47.1±0.1	59.9±0.1	69.0±0.3	24.6±0.1	45.7±0.4	51.3±0.4	–	–	–
DSDM	45.0±0.4	66.5±0.3	75.8±0.3	19.5±0.2	46.2±0.3	54.0±0.2	–	–	–
NCFM	49.5±0.3	70.6±0.3	77.7±0.2	32.6±0.3	48.9±0.3	53.5±0.1	18.5±0.2	26.3±0.2	29.8±0.1
SIGMS	52.9±0.3	72.5±0.3	78.9±0.3	34.5±0.1	51.2±0.4	53.8±0.2	20.7±0.3	27.0±0.1	30.4±0.3
Whole Dataset		84.8±0.1			56.2±0.3			37.6±0.4	

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate SIGMS on CIFAR-10, CIFAR-100, Tiny ImageNet, and representative ImageNet subsets. These benchmarks cover different dataset scales and category complexities, allowing us to assess the effectiveness of SIGMS under both standard and highly compressed settings.

Baselines. We compare SIGMS with representative dataset distillation methods from different paradigms, including gradient- and trajectory-matching approaches as well as recent distribution-matching methods. In particular, our comparisons include DM [14], DSA [33], MTT [6], IDC [20], DANCE [31], RDED [16], and NCFM [38], together with other competitive methods when corresponding results are available.

Evaluation Protocol. Following common dataset distillation practice, we evaluate all methods under different images-per-class (IPC) settings. For each synthetic dataset, a fresh evaluation network is trained from scratch and tested on the corresponding real test set. Unless otherwise specified, we report the average test accuracy over multiple independent runs.

Implementation Details. SIGMS initializes the synthetic set using SaliPatch and then optimizes it with stage-conditioned characteristic-function matching. The initial and expert feature extractors are kept frozen during distillation, while the synthetic images and Adaptive SampleNet are updated according to the min-max objective in Eq. (10). We use the stage set $\{0, 0.25, 0.5, 0.75, 1\}$ throughout the experiments. Detailed hyperparameters, network configurations, and additional implementation settings are provided in the Appendix.

4.2 Main Results

We compare SIGMS with representative dataset distillation methods on CIFAR-10, CIFAR-100, Tiny ImageNet, and six ImageNet subsets under different IPC settings. Among these methods, NCFM [38] serves as the most directly comparable baseline due to its characteristic-function-based distribution matching formulation. The results are summarized in Tables 1 and 2.

CIFAR-10/100 and Tiny ImageNet. As shown in Table 1, SIGMS achieves clear improvements over NCFM, especially under highly compressed low-IPC settings. At IPC=1, SIGMS

Table 2: Test accuracy (%) on ImageNet subsets at 128×128 resolution under different IPC settings.

Method	ImageNette		ImageWoof		ImageFruit		ImageMeow		ImageSquawk		ImageYellow	
	IPC=1	IPC=10	IPC=1	IPC=10	IPC=1	IPC=10	IPC=1	IPC=10	IPC=1	IPC=10	IPC=1	IPC=10
Random	23.5±4.8	47.7±2.4	14.2±0.9	27.0±1.9	–	–	–	–	–	–	–	–
MTT	47.7±0.9	63.0±1.3	28.6±0.8	35.8±1.8	–	–	–	–	–	–	–	–
DM	32.8±0.5	58.1±0.3	21.1±1.2	31.4±0.5	–	–	–	–	–	–	–	–
RDED	33.8±0.8	63.2±0.7	18.5±0.9	40.6±2.0	–	–	–	–	–	–	–	–
NCFM	54.0±1.6	80.2±1.0	31.4±1.1	50.0±1.3	33.2±1.2	44.8±1.0	35.8±1.3	60.5±1.1	44.0±1.0	75.2±1.5	50.2±1.6	74.2±1.5
SIGMS	56.2±1.0	80.5±0.8	33.1±0.7	50.5±1.3	35.4±0.4	45.2±0.5	36.7±0.4	60.8±0.8	45.6±0.8	75.7±1.0	52.4±0.4	74.5±1.2

Table 3: Ablation study of the main components in SIGMS. Test accuracy (%) is reported under representative IPC settings.

Method	CIFAR-10		CIFAR-100		Tiny ImageNet	
	IPC=1	IPC=50	IPC=1	IPC=50	IPC=1	IPC=50
NCFM [38]	49.5±0.3	77.7±0.2	32.6±0.3	53.5±0.1	18.5±0.2	29.8±0.1
+ SaliPatch	50.1±0.3	78.5±0.5	33.4±1.0	53.6±0.7	19.7±0.1	29.9±0.6
+ Adaptive SampleNet	51.5±0.2	78.2±0.5	34.2±0.9	53.4±1.0	20.4±1.0	30.1±1.2
SIGMS	52.9±0.3	78.9±0.3	34.5±0.1	53.8±0.2	20.7±0.3	30.4±0.3

improves accuracy from 49.5% to 52.9% on CIFAR-10, from 32.6% to 34.5% on CIFAR-100, and from 18.5% to 20.7% on Tiny ImageNet. These results demonstrate that SIGMS is particularly effective when the synthetic-data budget is severely constrained, while maintaining competitive performance at larger IPC values.

ImageNet Subsets. Table 2 further shows that SIGMS consistently improves upon NCFM across all six ImageNet subsets. At IPC=1, SIGMS improves ImageNette from 54.0% to 56.2%, ImageFruit from 33.2% to 35.4%, and ImageYellow from 50.2% to 52.4%. Similar improvements are also observed on ImageWoof, ImageMeow, and ImageSquawk, indicating that SIGMS remains effective on more complex and higher-resolution visual distributions.

Overall, SIGMS exhibits its strongest advantage under severe compression while maintaining strong performance across larger IPC settings.

4.3 Ablation Study

We conduct ablation studies to analyze the effect of each component in SIGMS. Starting from the NCFM baseline, we separately add SaliPatch and Adaptive SampleNet and compare them with the full SIGMS framework. We also compare different ways of combining feature interpolation and frequency sampling under the same SaliPatch initialization.

As shown in Table 3, adding either SaliPatch or Adaptive SampleNet improves performance in most settings, with larger gains generally observed in low-IPC regimes. SaliPatch improves synthetic-data initialization, while Adaptive SampleNet provides more effective frequency sampling for characteristic-function matching. Using both components together gives the best overall performance.

We further study the shared stage variable λ under the same SaliPatch initialization, with detailed results reported in Section 4.4.

4.4 Further Analysis

We further analyze SIGMS in terms of stage-wise coordination, hyperparameter sensitivity, computational overhead, and optimization stability.

Effect of the Shared Stage Variable. We investigate whether feature interpolation and frequency sampling should be coordinated through the same stage variable. All variants use the same SaliPatch initialization to isolate the effect of stage-wise coordination. As shown in Table 4, Stage-Conditioned Feature Interpolation alone provides a strong baseline. Simply introducing Adaptive SampleNet without conditioning it on the feature stage slightly degrades performance across the three datasets. In contrast, sharing the same stage variable λ consistently improves the results, indicating that adaptive frequency sampling benefits from being aligned with the corresponding representation state.

Table 4: Effect of the shared stage variable λ. All variants use the same SaliPatch initialization.

Variant	CIFAR-10 IPC=1	CIFAR-100 IPC=1	Tiny ImageNet IPC=1
Stage-Conditioned Feature Interpolation	51.6±0.5	33.3±0.4	19.6±0.5
+ Adaptive SampleNet (independent)	51.2±0.6	33.0±0.4	19.4±0.5
+ Adaptive SampleNet (shared λ)	52.9±0.3	34.5±0.1	20.7±0.3

Table 5: Sensitivity to the main hyperparameters at IPC=1.

	F			K		
	1	2	4	128	256	512
CIFAR-10	51.0±0.4	52.9±0.3	50.7±0.5	52.1±0.5	52.9±0.3	52.2±0.6
CIFAR-100	33.2±0.5	34.5±0.1	33.9±0.6	34.0±0.3	34.5±0.1	34.2±0.1

Hyperparameter Sensitivity. We next examine the sensitivity of SIGMS to the patch splicing factor F and the number of Gaussian components K, which respectively affect SaliPatch construction and frequency sampling. Table 5 reports the results on CIFAR-10 and CIFAR-100 at IPC=1. The default settings $F = 2$ and $K = 256$ achieve the best performance on both datasets, indicating that moderate patch granularity and mixture complexity provide a better balance between semantic preservation and adaptive frequency modeling.

Computational Cost. We further compare the computational cost of SIGMS with NCFM under the same experimental setting. SIGMS requires a notably longer runtime due to stage-conditioned feature construction and adaptive frequency modeling, while the increase in peak GPU memory remains relatively small. Detailed runtime, memory usage, and sampler parameter counts are reported in the Appendix.

Convergence and Stability. Figure 3 shows the matching-loss curves of SIGMS under different IPC settings. The loss decreases rapidly during early optimization and subsequently converges to a stable level across different compression ratios. Together with the repeated-run results reported in Tables 1 and 2, this suggests that the optimization remains stable under different IPC settings.

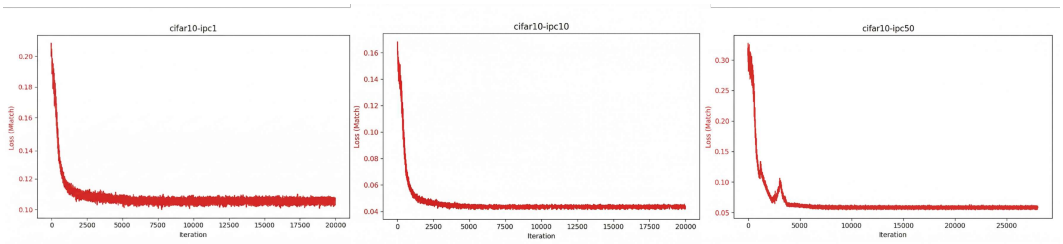

Figure 3: Convergence of SIGMS under different IPC settings on CIFAR-10.

5 Conclusion

In this work, we propose SIGMS, a dataset distillation framework that improves synthetic-data initialization and characteristic-function-based distribution matching. SaliPatch constructs informative synthetic priors from salient local regions, while Stage-Conditioned Feature Interpolation and Adaptive SampleNet use a shared stage variable to adapt feature representations and frequency sampling across different representation states. Experiments on CIFAR-10/100, Tiny ImageNet, and ImageNet subsets show that SIGMS consistently improves over its characteristic-function baseline, with particularly clear gains under highly compressed low-IPC settings. Ablation and further analyses verify the effectiveness of the proposed components and demonstrate stable optimization behavior.

AI Use Statement

Large language models were used to assist with several parts of the research and writing process. Specifically, they were used for language polishing, improving the clarity and organization of the manuscript, drafting and revising portions of the paper, and providing limited support for research ideation and implementation.

For research and implementation support, large language models were used to help discuss experimental design choices, organize ablation and sensitivity studies, suggest implementation and debugging steps, and assist with writing or modifying small portions of experimental and visualization code. They were also used to help interpret experimental outputs and to improve the presentation of figures, tables, and appendix material.

All methodological decisions, experimental configurations, source code changes, numerical results, statistical summaries, figures, tables, and conclusions were reviewed and verified by the authors. The authors independently ran all experiments and evaluations and checked the reported results against the actual experimental outputs. Large language models were not used to generate or fabricate experimental results, and no unverified AI-generated claims were included in the manuscript.

The authors remain fully responsible for the technical content, correctness, and conclusions of this work.

References

[1] Ansari, A. F., Scarlett, J. & Soh, H. A Characteristic Function Approach to Deep Implicit Generative Modeling. ArXiv Preprint ArXiv:1909.07425. (2020).

[2] Arjovsky, M. & Bottou, L. Towards Principled Methods for Training Generative Adversarial Networks. ArXiv Preprint ArXiv:1701.04862. (2017).

[3] Shao, S., Yin, Z., Zhou, M., Zhang, X. & Shen, Z. Generalized Large-Scale Data Condensation via Various Backbone and Statistical Matching. 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16709–16718. (2024).

[4] Bińkowski, M., Sutherland, D. J., Arbel, M. & Gretton, A. Demystifying MMD GANs. ArXiv Preprint ArXiv:1801.01401. (2021).

[5] Sriperumbudur, B. K., Gretton, A., Fukumizu, K., Schölkopf, B. & Lanckriet, G. R. G. Hilbert space embeddings and metrics on probability measures. ArXiv Preprint ArXiv:0907.5309. (2010).

[6] Cazenavette, G., Wang, T., Torralba, A., Efros, A. A. & Zhu, J.-Y. Dataset Distillation by Matching Training Trajectories. ArXiv Preprint ArXiv:2203.11932. (2022).

[7] Chen, D., Kerkouche, R. & Fritz, M. Private Set Generation with Discriminative Information. ArXiv Preprint ArXiv:2211.04446. (2022).

[8] Zhang, H., Li, S., Wang, P., Zeng, D. & Ge, S. M3D: Dataset Condensation by Minimizing Maximum Mean Discrepancy. ArXiv Preprint ArXiv:2312.15927. (2024).

[9] Cui, J., Wang, R., Si, S. & Hsieh, C.-J. Scaling Up Dataset Distillation to ImageNet-1K with Constant Memory. Proceedings of the 40th International Conference on Machine Learning (ICML). pp. 6565–6590. (2023).

[10] Deng, W., Li, W., Ding, T., Wang, L., Zhang, H., Huang, K., Huo, J. & Gao, Y. Exploiting Inter-sample and Inter-feature Relations in Dataset Distillation. 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 17057–17066. (2024).

[11] Dong, T., Zhao, B. & Lyu, L. Privacy for Free: How does Dataset Condensation Help Privacy? ArXiv Preprint ArXiv:2206.00240. (2022).

[12] Such, F. P., Rawal, A., Lehman, J., Stanley, K. O. & Clune, J. Generative Teaching Networks: Accelerating Neural Architecture Search by Learning to Generate Synthetic Training Data. ArXiv Preprint ArXiv:1912.07768. (2019).

[13] Reich, K. Chapter 13 - Leonhard Euler, 'Introduction' to analysis (1748). In I. Grattan-Guinness, R. Cooke, L. Corry, P. Crépel, N. Guicciardini (eds.), Landmark Writings in Western Mathematics 1640-1940. pp. 181–190. Elsevier Science, Amsterdam. (2005).

[14] Zhao, B. & Bilen, H. Dataset Condensation with Distribution Matching. ArXiv Preprint ArXiv:2110.04181. (2022).

[15] Gao, S., Liu, X., Zeng, B., Xu, S., Li, Y., Luo, X., Liu, J., Zhen, X. & Zhang, B. Implicit Diffusion Models for Continuous Super-Resolution. ArXiv Preprint ArXiv:2303.16491. (2023).

[16] Sun, P., Shi, B., Yu, D. & Lin, T. On the Diversity and Realism of Distilled Dataset: An Efficient Dataset Distillation Paradigm. ArXiv Preprint ArXiv:2312.03526. (2024).

[17] He, K., Zhang, X., Ren, S. & Sun, J. Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2016).

[18] Howard, J. & Gugger, S. fastai: A Layered API for Deep Learning. Information. Vol. 11, p. 108. (2020).

[19] Kang, S., Lim, Y. & Shim, H. Label-Augmented Dataset Distillation. ArXiv Preprint ArXiv:2409.16239. (2024).

[20] Kim, J.-H., Kim, J., Oh, S. J., Yun, S., Song, H., Jeong, J., Ha, J.-W. & Song, H. O. Dataset Condensation via Efficient Synthetic-Data Parameterization. Proceedings of the 39th International Conference on Machine Learning (ICML). pp. 11102–11118. (2022).

[21] Kogon, S. M. & Williams, D. B. Characteristic function based estimation of stable distribution parameters. A Practical Guide to Heavy Tails: Statistical Techniques and Applications. pp. 311–335. Birkhauser Boston Inc., USA. (1998).

[22] Krizhevsky, A. Learning Multiple Layers of Features from Tiny Images. Technical Report. (2009).

[23] Le, Y. & Yang, X. Tiny ImageNet Visual Recognition Challenge. Stanford University Technical Report. (2015).

[24] Lee, S., Chun, S., Jung, S., Yun, S. & Yoon, S. Dataset Condensation with Contrastive Signals. ArXiv Preprint ArXiv:2202.02916. (2022).

[25] Li, C.-L., Chang, W.-C., Cheng, Y., Yang, Y. & Póczos, B. MMD GAN: towards deeper understanding of moment matching network. Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS). pp. 2200–2210. (2017).

[26] Li, H., Zhou, Y., Gu, X., Li, B. & Wang, W. Diversified Semantic Distribution Matching for Dataset Distillation. Proceedings of the 32nd ACM International Conference on Multimedia (MM '24). pp. 7542–7550. (2024).

[27] Wang, K., Zhao, B., Peng, X., Zhu, Z., Yang, S., Wang, S., Huang, G., Bilen, H., Wang, X. & You, Y. CAFE: Learning to Condense Dataset by Aligning Features. ArXiv Preprint ArXiv:2203.01531. (2022).

[28] Yang, E., Shen, L., Wang, Z., Liu, T. & Guo, G. An efficient dataset condensation plugin and its application to continual learning. Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS '23). (2023).

[29] Li, Z. & Kainz, B. Image Distillation for Safe Data Sharing in Histopathology. ArXiv Preprint ArXiv:2406.13536. (2024).

[30] Li, Z. et al. DD-Ranking: Rethinking the Evaluation of Dataset Distillation. ArXiv Preprint ArXiv:2505.13300. (2025).

[31] Zhang, H., Li, S., Lin, F., Wang, W., Qian, Z. & Ge, S. DANCE: Dual-View Distribution Alignment for Dataset Condensation. ArXiv Preprint ArXiv:2406.01063. (2024).

[32] Levy, P. S. Théorie de l'addition des variables aléatoires. The Mathematical Gazette. Vol. 39, p. 344. (1955).

[33] Zhao, B. & Bilen, H. Dataset Condensation with Differentiable Siamese Augmentation. Proceedings of the 38th International Conference on Machine Learning (ICML). pp. 12674–12685. (2021).

[34] Medvedev, D. & D'yakonov, A. Learning to Generate Synthetic Training Data using Gradient Matching and Implicit Differentiation. ArXiv Preprint ArXiv:2203.08559. (2022).

[35] Zhao, B., Mopuri, K. R. & Bilen, H. Dataset Condensation with Gradient Matching. International Conference on Learning Representations (ICLR 2021). pp. 1–20. (2021).

[36] Zhou, Y., Nezhadarya, E. & Ba, J. Dataset Distillation using Neural Feature Regression. ArXiv Preprint ArXiv:2206.00719. (2022).

[37] Qin, T., Deng, Z. & Alvarez-Melis, D. A Label is Worth a Thousand Images in Dataset Distillation. ArXiv Preprint ArXiv:2406.10485. (2025).

[38] Wang, S., Yang, Y., Liu, Z., Sun, C., Hu, X., He, C. & Zhang, L. Dataset Distillation with Neural Characteristic Function: A Minmax Perspective. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 25570–25580. (2025).

[39] Tran, M.-T., Le, T., Le, X.-M., Do, T.-T., & Phung, D. Enhancing Dataset Distillation via Non-Critical Region Refinement. ArXiv Preprint ArXiv:2503.18267. (2025).

[40] Wang, T., Zhu, J.-Y., Torralba, A., & Efros, A. A. Dataset Distillation. ArXiv Preprint ArXiv:1811.10959. (2020).

[41] Zhao, G., Li, G., Qin, Y. & Yu, Y. Improved Distribution Matching for Dataset Condensation. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7856–7865. (2023).

[42] Wang, K., Li, Z., Cheng, Z.-Q., Khaki, S., Sajedi, A., Vedantam, R., Plataniotis, K. N., Hauptmann, A. & You, Y. Emphasizing Discriminative Features for Dataset Distillation in Complex Scenarios. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2025).

[43] Lee, Y. & Chung, H. W. SelMatch: Effectively Scaling Up Dataset Distillation via Selection-Based Initialization and Partial Updates by Trajectory Matching. ArXiv Preprint ArXiv:2406.18561. (2024).

[44] Zhong, X., Sun, S., Gu, X., Xu, Z., Wang, Y., Wu, J. & Chen, B. Efficient Dataset Distillation via Diffusion-Driven Patch Selection for Improved Generalization. ArXiv Preprint ArXiv:2412.09959. (2024).

A Appendix

A.1 Implementation Details

This section provides additional implementation details of SIGMS, including the backbone configuration, optimization settings, SaliPatch initialization, stage-conditioned feature modeling, Adaptive SampleNet, and the evaluation protocol.

A.1.1 Datasets and Network Architectures

For CIFAR-10, the input resolution is 32×32. We use a 3-layer ConvNet with instance normalization as the feature extractor. The real-data batch size is set to 1024.

During dataset distillation, the initial and expert feature extractors are kept frozen. The synthetic images and Adaptive SampleNet are optimized while the two representation anchors remain fixed. This prevents the anchor representations from drifting during distillation and allows the intermediate feature states to be constructed consistently between the initial and expert models.

For evaluation, a fresh network is initialized from scratch, trained only on the distilled dataset, and evaluated on the corresponding real test set.

A.1.2 Optimization Settings

We use AdamW to optimize the synthetic images. For CIFAR-10, the synthetic-image learning rate is set to 0.01, and the momentum parameter for synthetic-image optimization is set to 0.5. The learning rate of Adaptive SampleNet is set to 1×10^{-3}.

The distillation process is performed for 20,000 iterations. The calibration weight and PLTDA weight are both set to 1.0. Unless otherwise specified, all ablation and sensitivity experiments use the same optimization budget and backbone configuration as the full SIGMS model.

The default implementation settings are summarized in Table 6.

Table 6: Default implementation settings of SIGMS on CIFAR-10.

Setting	Value
Input resolution	32×32
Backbone	3-layer ConvNet
Normalization	Instance normalization
Real-data batch size	1024
Distillation iterations	20,000
Synthetic-image learning rate	0.01
Synthetic-image momentum	0.5
Adaptive SampleNet learning rate	1×10^{-3}
Frequency arguments	4096
Gaussian components K	256
SaliPatch stitching factor F	2
Representation stages	5
Calibration weight	1.0
PLTDA weight	1.0

A.1.3 SaliPatch Initialization

SaliPatch constructs the initial synthetic set from salient local regions of real images. For each real image, we first compute a class activation map (CAM) and divide the image into local patches. The average CAM response within each patch is used to identify salient candidate regions.

For the default CIFAR-10 configuration, the patch size is set to 8, the top-8 salient candidates are retained, and the patch-selection ratio is set to 0.2. The default patch-stitching factor is

$$F = 2.$$

After salient candidate extraction, SaliPatch performs IPC-aware difficulty-diversity selection. Under highly compressed settings, relatively informative and difficult regions are favored while redundant patches are suppressed. The selected patches are then resized when necessary and arranged into an $F \times F$ grid to construct the initial synthetic image.

For a stitching factor F, each initialized synthetic image contains

$$F^2$$

selected local regions. Therefore, the default $F = 2$ setting combines four salient regions within each synthetic image.

The SaliPatch initialization procedure can be summarized as follows:

1. Compute CAMs for real training images.
2. Divide each image into local patches.
3. Rank patches according to their CAM responses.
4. Construct a salient candidate pool.
5. Perform IPC-aware difficulty-diversity selection.
6. Select F^2 local regions for each synthetic image.
7. Stitch the selected patches to construct the initial synthetic set.

A.1.4 Stage-Conditioned Feature Interpolation

We construct intermediate representation states between a frozen initial feature extractor ϕ_0 and a frozen expert feature extractor ϕ_{expert}.

Five representation stages are used throughout distillation:

$$\lambda \in \{0,\, 0.25,\, 0.5,\, 0.75,\, 1\}.$$

For an input image x, the representation at stage λ is defined as

$$\phi_\lambda(x) = (1 - \alpha(\lambda))\phi_0(x) + \alpha(\lambda)\phi_{\text{expert}}(x),$$

where

$$\alpha(\lambda) = \lambda^{1.2}.$$

The same interpolation rule is applied to both real and synthetic samples. Small values of λ emphasize features closer to the initial network, whereas larger values progressively move the representation toward the expert state.

A.1.5 Adaptive Frequency Sampling

Adaptive SampleNet models the frequency distribution used in characteristic-function matching with a learnable Gaussian mixture.

For the default configuration, the number of Gaussian components is

$$K = 256,$$

and 4096 frequency arguments are sampled during each matching step.

The stage-dependent temperature is defined as

$$T_\lambda = 1 - 0.5\lambda.$$

Let t_{GMM} denote a sample drawn from the learned Gaussian mixture and let

$$t_0 \sim \mathcal{N}(0, I)$$

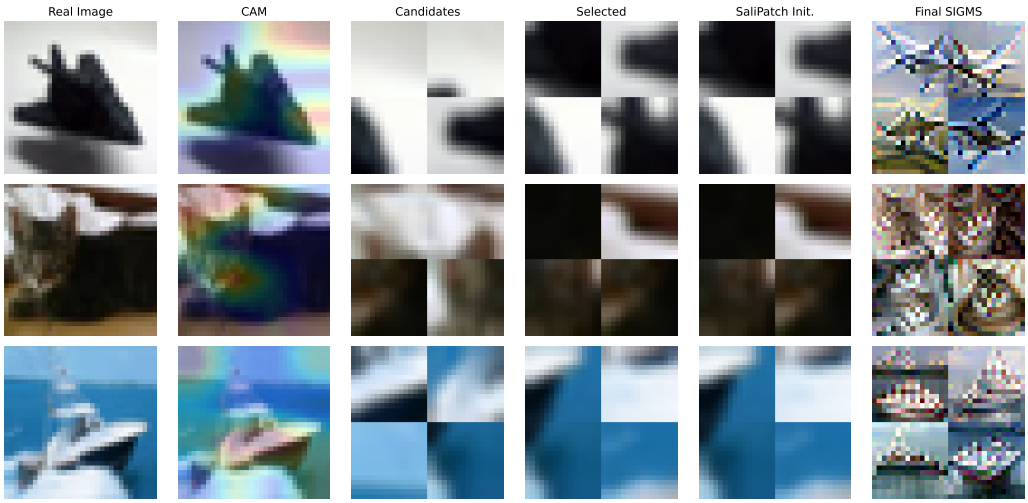

Figure 4: Visualization of the SaliPatch initialization process on three representative CIFAR-10 classes. Each row corresponds to one class. From left to right: real image, class activation map, salient candidate patches, selected patches after difficulty-diversity selection, stitched SaliPatch initialization, and the final optimized SIGMS sample.

denote a base Gaussian sample. The final stage-conditioned frequency argument is

$$t_\lambda = (1 - \rho(\lambda))t_0 + \rho(\lambda)t_{\mathrm{GMM}},$$

where

$$\rho(\lambda) = \lambda^{1.5}.$$

Therefore, early representation stages retain more base-Gaussian exploration, while later stages place progressively greater emphasis on the learned Gaussian-mixture distribution.

A.1.6 Evaluation Protocol

For evaluation, a fresh network is trained from scratch on the distilled dataset and tested on the corresponding real test set.

For CIFAR-10, the evaluation network is trained for 2000 epochs with a batch size of 128 using AdamW. The initial learning rate is set to 0.01, the momentum is 0.9, and the weight decay is

$$5 \times 10^{-4}.$$

The random seed is set to 0 in the default configuration. For experiments reported with uncertainty estimates, independent evaluation networks are initialized and trained from scratch, and the mean test accuracy and standard deviation are reported.

A.2 Visualization of SaliPatch Initialization

To provide a qualitative view of the SaliPatch initialization process, we visualize three representative CIFAR-10 classes in Figure 4. For each class, we show a real image, its class activation map, salient candidate patches, the patches retained after difficulty-diversity selection, the stitched SaliPatch initialization, and the final optimized SIGMS sample.

The visualization shows that SaliPatch concentrates the initialization on salient class-related local regions while preserving complementary patch information. Subsequent distribution-matching optimization further updates the synthetic pixels, such that the final distilled samples are optimized primarily for training utility rather than visual realism.

A.3 Adaptive SampleNet Architecture

Adaptive SampleNet consists of three prediction branches: a mean branch, a scale branch, and a mixture-weight branch.

The mean branch uses fully connected layers with feature dimension 2048. Batch normalization, LeakyReLU activation, and dropout are inserted between the intermediate layers.

The scale branch follows a similar architecture and predicts one positive scalar scale parameter for each Gaussian component through a Softplus activation.

The mixture-weight branch maps the 2048-dimensional component representation to a hidden dimension of 1024 and subsequently predicts one mixture logit.

Table 7 summarizes the architecture.

Table 7: Architecture of Adaptive SampleNet.

Module	Architecture
Mean branch	$2048 \to 2048 \to 2048 \to 2048$
Scale branch	$2048 \to 2048 \to 2048 \to 1$
Weight branch	$2048 \to 1024 \to 1$
Default Gaussian components	$K = 256$
Frequency arguments	4096

The prediction networks are shared across Gaussian components. Therefore, changing K changes the number of Gaussian component codes processed by the shared prediction networks rather than directly changing the dimensions of the MLP parameters.

A.4 Additional Experimental Analysis

A.4.1 Hyperparameter Sensitivity

We investigate the sensitivity of SIGMS to the SaliPatch stitching factor F and the number of Gaussian components K.

When varying F, the Gaussian-mixture configuration is fixed to its default value

$$K = 256.$$

The underlying data-decoding factor is kept fixed, and only the SaliPatch-specific stitching factor is varied. This ensures that the sensitivity experiment isolates the influence of SaliPatch composition.

For CIFAR-10 at IPC=1, the results for

$$F \in \{1, 2, 4\}$$

are

$$51.0 \pm 0.4, \qquad 52.9 \pm 0.3, \qquad 50.7 \pm 0.5,$$

respectively.

The default $F = 2$ achieves the best performance among the tested settings. When $F = 1$, each synthetic image is initialized from only one salient region, limiting the amount of complementary local information. When $F = 4$, each synthetic image is composed of sixteen smaller regions, which increases local diversity but may fragment object-level semantic structures.

When analyzing the number of Gaussian components, the SaliPatch factor is fixed to

$$F = 2,$$

and only

$$K \in \{128, 256, 512\}$$

is varied. The default setting uses $K = 256$. The complete results are reported in Table 5 of the main paper.

A.4.2 Effect of the Shared Stage Variable

We further examine whether feature interpolation and adaptive frequency sampling should be coordinated through the same stage variable.

All compared variants use the same SaliPatch initialization.

Stage-Conditioned Feature Interpolation alone achieves 51.6 ± 0.5, 33.3 ± 0.4, and 19.6 ± 0.5 on CIFAR-10, CIFAR-100, and Tiny ImageNet at IPC=1, respectively.

Introducing Adaptive SampleNet without conditioning it on the representation stage gives 51.2 ± 0.6, 33.0 ± 0.4, and 19.4 ± 0.5.

In contrast, sharing the same stage variable λ between feature interpolation and Adaptive SampleNet improves the results to 52.9 ± 0.3, 34.5 ± 0.1, and 20.7 ± 0.3, respectively.

These results indicate that the benefit of Adaptive SampleNet arises not only from using a flexible Gaussian-mixture distribution, but also from adapting frequency probing to the corresponding representation state.

A.4.3 Computational Cost

We compare the computational cost of SIGMS with NCFM under the same CIFAR-10 IPC=1 experimental setting.

Both methods use the same backbone architecture, optimization budget, and hardware configuration. We report the total distillation time, peak GPU memory usage, and the number of trainable parameters in the sampling network.

Table 8: Computational cost comparison with NCFM on CIFAR-10 at IPC=1.

Method	Time (h)	Peak Memory (GB)	Sampler Params.
NCFM [38]	3.02	2.37	12.60M
SIGMS	10.65	2.50	23.10M

As shown in Table 8, SIGMS requires a longer distillation time than NCFM. The additional runtime mainly comes from stage-conditioned feature construction and adaptive frequency modeling.

Specifically, the total distillation time increases from 3.02 hours for NCFM to 10.65 hours for SIGMS. In contrast, the increase in peak GPU memory is relatively small, from 2.37 GB to 2.50 GB.

The number of trainable parameters in the sampling network increases from 12.60M to 23.10M. Overall, the additional computational cost of SIGMS is mainly reflected in runtime rather than GPU memory usage.This trade-off is most favorable in highly compressed low-IPC regimes, where SIGMS yields its largest accuracy improvements; at higher IPC settings, the relative gains become smaller.

A.4.4 Optimization Stability

During distillation, the synthetic images and Adaptive SampleNet are optimized while the initial and expert feature extractors remain frozen.

The use of stage-conditioned interpolation provides a gradual transition between different representation states. Similarly, the frequency distribution gradually evolves from the base Gaussian distribution toward the learned Gaussian mixture.

This progressive construction avoids abruptly changing either the feature representation or the frequency distribution and contributes to stable optimization across representation stages.

A.4.5 Qualitative Visualization

SaliPatch provides a semantically informed initialization before synthetic-data optimization begins.

A qualitative comparison can be constructed using:

1. representative real images,
2. SaliPatch initialized synthetic images,
3. NCFM distilled images, and
4. final SIGMS distilled images.

The initialized images preserve salient local class-related structures, while the final synthetic images are further optimized according to the distribution-matching objective. Therefore, the final distilled samples are optimized primarily for training utility rather than photorealistic appearance.

A.5 Reproducibility Details

For reproducibility, we summarize the main implementation choices used by SIGMS:

- The initial and expert feature extractors remain frozen throughout distillation.
- The synthetic images are directly optimized through gradients from the stage-conditioned characteristic-function matching objective.
- Adaptive SampleNet is optimized jointly with the synthetic images under the min–max objective.
- Five representation stages are used:
$$\{0, 0.25, 0.5, 0.75, 1\}.$$
- The feature interpolation schedule is
$$\alpha(\lambda) = \lambda^{1.2}.$$
- The adaptive frequency temperature is
$$T_\lambda = 1 - 0.5\lambda.$$
- The frequency interpolation schedule is
$$\rho(\lambda) = \lambda^{1.5}.$$
- The number of sampled frequency arguments is 4096.
- The default Gaussian-mixture component number is
$$K = 256.$$
- The default SaliPatch stitching factor is
$$F = 2.$$
- In the full SIGMS model, Stage-Conditioned Feature Interpolation and Adaptive SampleNet share the same stage variable λ.
- In the F-sensitivity experiments, the underlying data-decoding factor is fixed while only the SaliPatch-specific stitching factor is varied.
- In the K-sensitivity experiments, the SaliPatch configuration and all remaining optimization settings are fixed.
- Evaluation networks are freshly initialized and trained from scratch on the distilled datasets.

A.6 Detailed Characteristic-Function Matching Objective

We provide additional details of the characteristic-function matching objective used in SIGMS.

For a representation $z = \phi_\lambda(x)$ at stage λ, the characteristic function evaluated at a frequency argument t_λ is defined as

$$\Phi_\lambda(t_\lambda) = \mathbb{E}_z \left[\exp\left(i t_\lambda^\top z \right) \right].$$

Given a mini-batch of N_r real samples, the empirical characteristic function is

$$\widehat{\Phi}_\lambda^{\,r}(t_\lambda) = \frac{1}{N_r} \sum_{n=1}^{N_r} \exp\left(i t_\lambda^\top \phi_\lambda(x_n) \right).$$

Similarly, for N_s synthetic samples,

$$\widehat{\Phi}_\lambda^{\,s}(t_\lambda) = \frac{1}{N_s} \sum_{m=1}^{N_s} \exp\left(i t_\lambda^\top \phi_\lambda(\tilde{x}_m) \right).$$

The stage-wise discrepancy can therefore be written as

$$\mathcal{L}_{\mathrm{CF}}^{(\lambda)} = \left| \widehat{\Phi}_\lambda^{\,r}(t_\lambda) - \widehat{\Phi}_\lambda^{\,s}(t_\lambda) \right|^2.$$

Considering all representation stages, the overall matching objective becomes

$$\mathcal{L}_{\mathrm{SIGMS}} = \sum_{\lambda \in \Lambda} w_\lambda \mathcal{L}_{\mathrm{CF}}^{(\lambda)},$$

where

$$\Lambda = \{0, 0.25, 0.5, 0.75, 1\}.$$

SIGMS follows a min–max optimization scheme:

$$\min_{\mathcal{S}} \max_{\psi} \mathcal{L}_{\mathrm{SIGMS}},$$

where $\mathcal{S}$ denotes the learnable synthetic set and ψ represents the parameters of Adaptive SampleNet.

The synthetic images minimize the discrepancy between real and synthetic feature distributions, whereas Adaptive SampleNet maximizes the discrepancy by identifying informative frequency arguments. This adversarial interaction encourages synthetic samples to match the real distribution along frequency directions that expose the largest remaining discrepancies.

A.7 Stage-Conditioning Schedule

To provide a more explicit view of the stage-conditioned transition, Table 9 reports the feature interpolation coefficient $\alpha(\lambda)$, the Gaussian-mixture interpolation coefficient $\rho(\lambda)$, and the sampling temperature T_λ at each representation stage.

Table 9: Stage-dependent coefficients used in SIGMS.

λ	$\alpha(\lambda) = \lambda^{1.2}$	$\rho(\lambda) = \lambda^{1.5}$	$T_\lambda = 1 - 0.5\lambda$
0.00	0.000	0.000	1.000
0.25	0.189	0.125	0.875
0.50	0.435	0.354	0.750
0.75	0.708	0.650	0.625
1.00	1.000	1.000	0.500

At $\lambda = 0$, both feature construction and frequency sampling remain close to their initial states: the representation is dominated by the initial model and the frequency arguments are sampled entirely from the base Gaussian distribution.

As λ increases, the representation gradually moves toward the expert feature space, while Adaptive SampleNet increasingly relies on the learned Gaussian-mixture distribution. At the same time, the decreasing temperature sharpens the mixture weights, allowing later stages to place greater emphasis on selected frequency components.

At $\lambda = 1$, the expert representation and the learned Gaussian-mixture frequency distribution are fully activated.

These schedules are simple monotonic parameterizations chosen to provide smooth transitions between the two endpoints. We do not claim that the specific exponents are theoretically optimal; the central design choice of SIGMS is that representation construction and frequency sampling are conditioned on the same stage variable.

A.8 Limitations

SIGMS introduces additional computation compared with NCFM because each distillation step involves stage-conditioned representation construction and adaptive frequency modeling. The computational analysis shows that this additional overhead is primarily reflected in runtime, while the increase in peak GPU memory remains relatively small.

SaliPatch relies on class activation maps obtained from a pretrained feature extractor. Consequently, the quality of the initialization may depend on the representation quality of the model used for saliency estimation.

The SaliPatch stitching factor F controls the granularity of local semantic composition. The sensitivity analysis indicates that excessively coarse or excessively fine patch decomposition can reduce performance, while the default $F = 2$ provides a more balanced initialization.

Adaptive SampleNet additionally introduces the number of Gaussian components K as a hyperparameter. Different values of K affect the flexibility and computational workload of frequency sampling.

Future work may investigate more efficient stage-conditioned frequency parameterizations and adaptive mechanisms for automatically determining appropriate patch and mixture granularity.