Efficient Dataset Distillation for Pre-Trained Self-Supervised Models via Statistical Flow Matching

Qianxin Xia[1,2*] **Jiawei Du**[2] ✉ **Yuhan Zhang**[3] **Xin Zhang**[2]
Xuewan He[1] **Wenbo Jiang**[1] **Jielei Wang**[1] **Tao Luo**[2] **Guoming Lu**[1] ✉
[1]University of Electronic Science and Technology of China, China
[2]Institute of Advanced Intelligence and Computing, Agency for Science, Technology and Research, Singapore
[3]Sichuan University, China
{xqx023, hershel}@std.uestc.edu.cn {wenbo_jiang, jlwang, lugm}@uestc.edu.cn
{dujw, zhangx7, luo_tao}@a-star.edu.sg zyhyuhan@stu.scu.edu.cn

Abstract

Dataset distillation seeks to synthesize a compact surrogate dataset that enables performance comparable to training on the original dataset for downstream tasks. For the scenario where pre-trained self-supervised models serve as priors, traditional *Linear Gradient Matching* optimizes synthetic images by encouraging them to mimic the gradient updates induced by real images on the linear probe. However, this batch-level formulation requires loading thousands of real images and applying multiple differentiable augmentations to synthetic images at each distillation step, leading to substantial computational and memory overheads. In this paper, we revisit the linear gradient and theoretically derive that it is essentially a *local* relative distribution directed from target class centers toward non-target class centers, which we term "flow". This property causes suboptimality and instability, often necessitating expensive multiple augmentations to compensate. To address this, we introduce *Statistical Flow Matching*, an optimal, stable, and efficient supervised learning framework that optimizes synthetic images by aligning *global* statistical flows in the original data. Our approach loads raw statistics only once and performs a single augmentation pass on the synthetic data, achieving performance comparable to or better than the state-of-the-art method with 10× less GPU memory usage and 4× faster distillation time. Moreover, increasing the number of augmentations for our method yields further performance gains while incurring lower additional cost. Our code is publicly available at https://github.com/einsteinxia/SFM.

1 Introduction

In machine learning, data serves as the "fuel" that drives improvements in model capabilities. As datasets grow larger and more diverse, models can learn richer patterns and achieve better performance. However, this reliance on data also introduces significant practical overheads, since large-scale dataset acquisition, management, transmission, and processing require substantial storage, communication, and computational capacity, as well as considerable time and operational resources. [19]. To reconcile high performance with prohibitive resource costs, *Dataset Distillation* (DD) [36] has been proposed to compress a cumbersome original dataset into a compact surrogate dataset that yields training efficacy comparable to using the full dataset. This enables more efficient model training and benefits downstream tasks such as continual learning [41, 25] and neural architecture search [24, 11]. On the other hand, the application of DD has been extended by numerous works to a range of research fields, e.g., privacy protection [42, 48], multimodal learning [38, 45], and federated learning [40, 17].

✉ Corresponding author.
* Work completed during internship at IAIC, A*STAR.

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

Several recent methods [46, 4, 43, 32] and their extensions [35, 12, 44, 39] have advanced the ambitious goal of learning high-quality models from synthetic images. However, they typically require distilling a relatively large Image-Per-Class (IPC) and training models from scratch to achieve merely mediocre or modest performance, both of which are inefficient and limit practical training and deployment in specific scenarios, such as resource-constrained edge environments. Consequently, there is an urgent need to investigate a more efficient, practice-ready scheme to break this impasse.

Fortunately, computer vision has increasingly shifted toward exploiting representations learned by large-scale pre-trained self-supervised vision models (e.g., DINO [26], CLIP [28], EVA [13], and MoCo [8]) for downstream tasks. In practice, such models are either lightly fine-tuned on the target task or used as frozen backbone encoders to provide transferable features for lightweight task-specific heads. With these strong priors, competitive models can often be obtained rapidly using only a small amount of data [31]. Therefore, we argue that developing efficient data synthesis and training strategies tailored to such pre-trained self-supervised vision models is a key prerequisite for advancing DD beyond a "toy" setting toward real-world applicability.

The pioneering work, Linear Gradient Matching (LGM) [3], first forwards both real images and noise-initialized synthetic images through the frozen backbone of the pre-trained self-supervised model, and then optimizes the synthetic images so that they induce gradient updates on the backbone-attached linear classification head that are similar to those induced by real images. On the one hand, this batch-level formulation requires loading numerous real images at each distillation step. On the other hand, each synthetic image undergoes multiple rounds of differentiable augmentations, so the *autograd* computation graphs for all views and their intermediate activations must be retained to enable backpropagation, which incurs significant computational overhead and GPU memory usage.

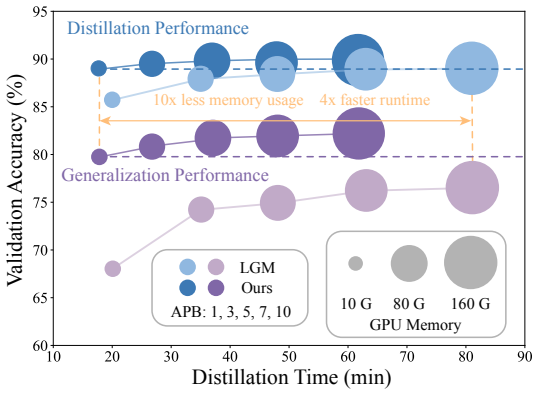


Figure 1: Comparison between LGM and our SFM in terms of distillation time, GPU memory usage, and validation accuracy on ImageNet-100 with IPC=1. The size of the circle denotes different amounts of Augmentations Per Batch (APB). The distillation model is EVA-02 and the generalized models are CLIP, DINO-v2 and MoCo-v3. With only one augmentation, our SFM beats LGM using multiple augmentations, which substantially reduces resource consumption while achieving state-of-the-art validation performance.

To address these issues, we delve into the mechanism of LGM. We first theoretically reformulate the linear gradient into an analytical form of local relative distribution, which we call **flow**. The flow is directed from the **target class**'s distribution center toward that of the **non-target class**. Subsequently, we reveal some key limitations (e.g., suboptimality, instability, and inefficiency) of this local matching paradigm and propose a novel supervised learning framework called **S**tatistical **F**low **M**atching **(SFM)**. SFM first statistically summarizes the global distribution center of the original dataset just once to form a constant flow, which remains fixed throughout the optimization. It then induces the synthetic flow of the distilled images to align with the statistical flow, achieving optimal, stable, and efficient synthesis.

As shown in Figure 1, SFM requires training only a linear classifier with IPC=1, taking just seconds, yet achieves 80%∼90% validation accuracy across multiple self-supervised models. Compared to LGM with ten augmentations, our SFM with just a single augmentation achieves comparable distillation performance and superior generalization performance, while reducing GPU memory usage by 10× (16 GB vs. 165 GB) and shortening the distillation time by 4× (18 min vs. 80 min). Moreover, applying more augmentations to SFM yields further gains, confirming the favorable scalability. In practice, we can also adjust the trade-off between resources and performance according to requirements. Notably, since SFM does not require frequently loading large batches of real images from the dataloader or performing additional gradient computations at each distillation step, our distillation time is significantly lower than that of LGM under the same augmentation settings.

Overall, our contributions are summarized as follows:

- We theoretically identify linear gradient as a local relative distribution, thereby establishing for the first time the concept of "flow" for dataset distillation.
- We propose statistical flow matching, a novel framework for global supervised learning that ensures optimality, stability, and efficiency in the process of data synthesis.
- Extensive experiments demonstrate that our method consistently outperforms the state-of-the-art linear gradient matching in both performance and efficiency.

2 Related Work

Dataset Distillation. As the scale of datasets in machine learning continues to grow, efficient storage and learning paradigms have attracted the attention of researchers from various fields. As a result, they have shifted from the initial coreset approach [9, 37], which directly selects representative samples from the raw dataset, to dataset distillation, which generates abstract or non-raw images containing insightful information from the original dataset. This has led to a range of modes, including the introduction of consistency loss that matches distribution [20], gradient [47], trajectory [49], statistics [30], and other parameter optimization [1], generation-based approaches [5, 6]. However, under low-IPC settings, these methods do not perform particularly well when training models from scratch. Even under high IPC configurations, they exhibit only modest to mediocre performance while resource requirements grow proportionally with IPC. LGM [3] is the first to incorporate pre-trained model priors into DD to bridge the above performance gap, but at a prohibitive cost.

Self-Supervised Learning. Due to the expensive and time-consuming nature of collecting labeled data on a large scale in reality, self-supervised learning has become a representative method for pre-training networks, allowing models to learn transferable representations from abundant unlabeled data for subsequent downstream tasks. Widely adopted self-supervised vision models mainly include CLIP [28], DINO [26], EVA [13], and MoCo [8].

In this work, we consider an extreme setting where we distill only **one image per class** from pre-trained self-supervised vision models, and for downstream tasks we train only **a linear classifier**. In addition, we mainly address the limitations imposed by LGM, including suboptimality, instability, and inefficiency. The satisfactory results have indeed unlocked the potential of this paradigm.

3 Method

In this section, we revisit linear gradient matching in pre-trained self-supervised vision models and deeply explore its mechanism. We reveal its essence and the limitations it brings. Based on this, we propose statistical flow matching, thereby improving distillation efficiency and performance. An overview of our SFM and the comparison to LGM are presented in Figure 2.

3.1 Rethinking Linear Gradient Matching

Formally, we consider a frozen pre-trained self-supervised model or feature extractor $\boldsymbol{\phi}$ that maps images $\boldsymbol{x}$ into a latent representation space $\boldsymbol{\phi}(\boldsymbol{x})^{F\times 1}$, where F is the output dimension of $\boldsymbol{\phi}$. Given an original large-scale dataset $\mathcal{O}=\{(\boldsymbol{x}_i^o,\boldsymbol{y}_i^o)\}_{i=1}^{|\mathcal{O}|}$, our objective is to construct a compact synthetic dataset $\mathcal{S}=\{(\boldsymbol{x}_i^s,\boldsymbol{y}_i^s)\}_{i=1}^{|\mathcal{S}|}$, which enables training a linear classification head on top of $\boldsymbol{\phi}$ with performance on par with that trained on the full real dataset. where $|\mathcal{S}|\ll|\mathcal{O}|$ and $\boldsymbol{x}_i^o,\boldsymbol{x}_i^s\sim P(\boldsymbol{x}^o),P(\boldsymbol{x}^s)$ are the i-th original and synthetic images with the corresponding one-hot labels $\boldsymbol{y}_i^o,\boldsymbol{y}_i^s\in\mathcal{Y}^{C\times 1}$. C is the number of classes.

To achieve this, LGM samples a **random** linear classifier matrix $\boldsymbol{W}\sim\mathcal{N}(0,0.01^2)\in\mathbb{R}^{C\times F}$ at each distillation step to enforce that training on the synthetic data mimics the gradient updates from training on real data under diverse initialization. After the original and synthetic images pass forward through $\boldsymbol{\phi}$ and $\boldsymbol{W}$, the gradient of the cross-entropy loss $\boldsymbol{\ell}$ with respect to $\boldsymbol{W}$ is:

$$\frac{\partial\boldsymbol{\ell}}{\partial\boldsymbol{W}}=\frac{1}{B}\sum_{i=1}^{B}(\boldsymbol{p}_i-\boldsymbol{y}_i)\boldsymbol{\phi}(\boldsymbol{x}_i)^\top, \tag{1}$$

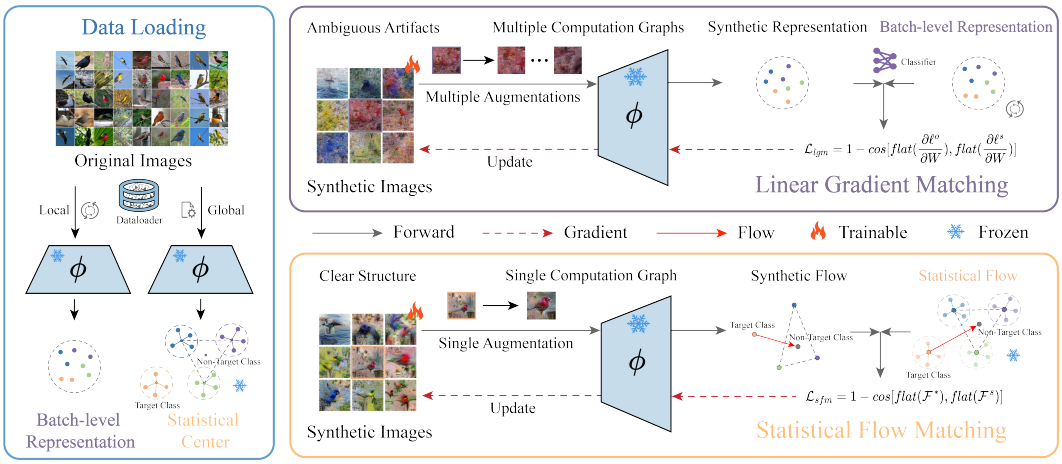


Figure 2: Comparison of the pipelines: Linear Gradient Matching vs. Statistical Flow Matching. During the original data loading phase, LGM dynamically loads a large batch of samples at every distillation step per epoch, whereas SFM computes the statistics only once and then fixes them during subsequent optimization. During the synthetic data distillation phase, LGM employs multiple augmentations to effectively align gradients or capture local relative distributions, while retaining expensive model computation graphs for backpropagation. In contrast, SFM requires only a single augmentation to stably and efficiently match the optimal statistical flow of the original data.

where $B = aC$ is the batch size under the setting of IPC=1, a is the number of augmentation for each class. $\boldsymbol{p}_i = [p_{i1}, p_{i2}, \cdots, p_{ic}, \cdots, p_{iC}]^\top$ is the probability distribution of the i-th sample:

$$p_{ic} = \text{softmax}(z_{ic}) = \frac{\exp(z_{ic})}{\sum_{j=1}^{C} \exp(z_{ij})}, \tag{2}$$

where $\boldsymbol{z}_i = [z_{i1}, z_{i2}, \cdots, z_{ic}, \cdots, z_{iC}]^\top = \boldsymbol{W}\boldsymbol{\phi}(\boldsymbol{x}_i)$ is the logits of the i-th sample.

Then, the LGM loss can be defined as:

$$\mathcal{L}_{lgm} = 1 - cos[flat(\frac{\partial \boldsymbol{\ell}^o}{\partial \boldsymbol{W}}), flat(\frac{\partial \boldsymbol{\ell}^s}{\partial \boldsymbol{W}})], \tag{3}$$

where $flat$ represents the $flatten$ operation. $\boldsymbol{\ell}^o$ and $\boldsymbol{\ell}^s$ represent the losses on the original image "o" and the synthetic image "s", respectively. $\mathcal{L}_{lgm}$ is subsequently backpropagated through the computational graph, which includes the inner gradient step, feature extractor, and the linear classifier, to obtain $\partial \mathcal{L}_{lgm} / \partial \boldsymbol{x}^s$ and update the synthetic images.

For Equation 1, given the determinism of the labels $\boldsymbol{y}$ and the feature extractor $\boldsymbol{\phi}$, the update of the synthetic data depends entirely on the probability distribution $\boldsymbol{p}$, which is in turn governed exclusively by the dynamic classifier's linear matrix $\boldsymbol{W}$.

Theorem 1 (Exchangeability)**.** *In a multiclass classification using the softmax function. Let the number of classes be C, and the weight vectors $\boldsymbol{W}_c \in \mathbb{R}^{F\times 1}$ for each class c are independently initialized from a Gaussian distribution with mean zero $\boldsymbol{W}_c \sim \mathcal{N}(0, \sigma_c^2)$. For sample i, the predicted class probabilities p_{ic} for each class c are exchangeable and uniformly distributed across the classes:*

$$\mathbb{E}[p_{i1}] = \cdots = \mathbb{E}[p_{ic}] = \cdots = \mathbb{E}[p_{iC}] = 1/C.$$

Theorem 2 (LogNormal Distribution)**.** *Let $Y \sim \mathcal{N}(\mu, \sigma^2)$ be a normally distributed random variable, and define $X = e^Y$. Then X follows a lognormal distribution, denoted $X \sim \mathcal{LN}(\mu, \sigma^2)$, with:*

$$\mathbb{E}[X] = \exp\big(\mu + \sigma^2/2\big), \quad \text{Var}[X] = \exp\big(2\mu + \sigma^2\big)\big[\exp(\sigma^2) - 1\big].$$

According to **Theorem 1**, the expectation of Equation 1 is:

$$\mathbb{E}[\frac{\partial \boldsymbol{\ell}}{\partial \boldsymbol{W}}] = \frac{1}{B}\sum_{i=1}^{B}(\mathbb{E}[\boldsymbol{p}_i] - \boldsymbol{y}_i)\boldsymbol{\phi}(\boldsymbol{x}_i)^\top = \frac{1}{B}\sum_{i=1}^{B}(\frac{1}{C}\mathbf{1} - \boldsymbol{y}_i)\boldsymbol{\phi}(\boldsymbol{x}_i)^\top. \tag{4}$$

We are aware that the c-th row of linear classifier matrix $\boldsymbol{W}_c \sim \mathcal{N}(0, 0.01^2)$, $z_{ic} = \boldsymbol{W}_c\boldsymbol{\phi}(\boldsymbol{x}_i) \sim \mathcal{N}(0, \sigma^2)$ ($\sigma = 0.01 \cdot ||\boldsymbol{\phi}(\boldsymbol{x}_i||$). A LayerNorm operation is typically applied at the end of self-supervised models, which ensures that $||\boldsymbol{\phi}(\boldsymbol{x}_i|| = O(1)$ and σ has a bounded and small value.

According to **Theorem 2**, the expectation and variance of $\exp(z_{ic})$ can be expresses as:

$$\mathbb{E}[\exp(z_{ic})] = \exp(\sigma^2/2), \quad \text{Var}[\exp(z_{ic})] = \exp(\sigma^2) \cdot [\exp(\sigma^2) - 1], \tag{5}$$

when C is relatively large (e.g., 100), according to the law of large numbers $\mathbb{E}[\exp(z_{ic})] \approx \sum_{j=1}^{C} \exp(z_{ij})/C$, the variance of p_{ic} in Equation 2 can be expressed as:

$$\begin{aligned} \text{Var}[p_{ic}] &= \text{Var}[\frac{\exp(z_{ic})}{C \cdot \exp(\sigma^2/2)}] = \frac{\text{Var}[\exp(z_{ic})]}{C^2 \cdot \exp(\sigma^2)} \\ &= \frac{\exp(\sigma^2) \cdot [\exp(\sigma^2) - 1]}{C^2 \cdot \exp(\sigma^2)} = \frac{\exp(\sigma^2) - 1}{C^2}, \end{aligned} \tag{6}$$

since σ is very small, we can see that $\text{Var}[p_{ic}] \to 0$. According to **Theorem 1** : $\mathbb{E}[p_{ic}] = 1/C$, $p_{ic} \to 1/C$ and its slight fluctuation is determined solely by $\boldsymbol{W}$.

To investigate the impact of $\boldsymbol{W}$, we establish three baselines: (1) the raw LGM with **random** $\boldsymbol{W}$ sampled at every distillation step, (2) **fixed** $\boldsymbol{W}$ over the whole distillation process, and (3) an **analytical** form in Equation 4 that is independent of $\boldsymbol{W}$. As shown in Table 1, all modes under various models achieve similar performance, which indicates that $\boldsymbol{W}$ has no significant impact on performance. This primarily stems from the fact that $p_{ic} \to 1/C$ is weakly correlated with $\boldsymbol{W}$. Although randomly sampling $\boldsymbol{W}$ intends to enhance generalization by aligning gradients under different initial conditions, both our theoretical analysis and experimental results indicate that this constitutes a redundant constraint.

Table 1: A comparison of average generalization performance under different $\boldsymbol{W}$ settings. The number of augmentation per batch is set to 1. It proves to be insensitive to changes in $\boldsymbol{W}$.

Mode	CLIP	DINO-v2	EVA-02	MoCo-v3
Random	64.9±0.2	78.9±0.1	72.2±0.1	72.5±0.1
Fixed	64.4±0.4	78.4±0.2	72.6±0.3	72.4±0.2
Analytic	64.3±0.4	78.4±0.2	73.0±0.1	71.9±0.2

Therefore, the gradient of the loss $\boldsymbol{\ell}$ with respect to $\boldsymbol{W}_c$ for each class c degenerates entirely into the analytical form shown in Equation 4:

$$\begin{aligned} \frac{\partial \boldsymbol{\ell}}{\partial \boldsymbol{W}_c} &= \frac{1}{B}\sum_{i=1}^{B} \frac{1}{C}\boldsymbol{\phi}(\boldsymbol{x}_i|_{\boldsymbol{y}_{ic}=0})^\top - \frac{C-1}{C}\boldsymbol{\phi}(\boldsymbol{x}_i|_{\boldsymbol{y}_{ic}=1})^\top \\ &= \frac{1}{aC} \cdot \frac{a(C-1)}{C} \cdot [\frac{\sum_{i=1}^{a(C-1)} \phi(x_i|_{y_{ic}=0})^\top}{a(C-1)} - \frac{\sum_{i=1}^{a} \phi(x_i|_{y_{ic}=1})^\top}{a}] \\ &= \frac{C-1}{C^2}[\overline{\phi}(\boldsymbol{x}|_{\boldsymbol{y}_c=0})^\top - \overline{\phi}(\boldsymbol{x}|_{\boldsymbol{y}_c=1})^\top], \end{aligned} \tag{7}$$

where the overline "$-$" denotes the mean. This effectively models a pattern of relative distribution. We refer to it as **flow** and the corresponding matching strategy is called flow matching. The flow originates from the distribution center of **target class** $\overline{\phi}(x|_{y_c=1})$ and terminates at the distribution center of **non-target class** $\overline{\phi}(x|_{y_c=0})$. Notably, our terminology is somewhat similar to that of flow-based generative models [22, 23], but fundamentally different. In generative models, a flow typically refers to a mapping from a Gaussian distribution to the data distribution, whereas our flow denotes a vector representation pointing from the target class center to the non-target class center.

3.2 Statistical Flow Matching

In Equation 7, we derive the analytical form of the linear gradient and show that, given a fixed number of classes C, it depends solely on the distribution centers of the data. For the original dataset, the matching strategy is invariably performed at the batch level, which results in some limitations:

Suboptimality. The local target and non-target class centers inadequately represent the full data distribution within a batch, leading to biased optimization that deviates from the global optimum. This limitation becomes more pronounced when the batch size is small.

Instability. The centers or flows vary dynamically across distillation steps or batches, which causes optimization instability that induces artifacts in Figure 5 (left) and thereby degrades performance. These also explain why LGM implicitly adopts a large batch size on original data and multiple augmentations on synthetic data, which proves to be generally effective in capturing global distribution.

Inefficiency. Employing a large batch size and extensive augmentations can alleviate the negative effects of the above two limitations to some extent, but they incur substantial demands on memory and computational resources. This stems from the necessity to load large volumes of original images and maintain intricate computational graphs for gradient computation across numerous augmented synthetic images, rendering the process highly inefficient.

Consequently, we propose a simple but comprehensive method to simultaneously address the aforementioned three issues. We precompute class-wise statistical centers $\phi^*(\boldsymbol{x})$ from the original dataset "o", thereby constructing a constant statistical flow $\boldsymbol{\mathcal{F}}^* = [\mathcal{F}_1^*, \cdots, \mathcal{F}_c^*, \cdots, \mathcal{F}_C^*]$, where:

$$\mathcal{F}_c^* = \phi^*(\boldsymbol{x}^o|_{\boldsymbol{y}_c^o=0})^\top - \phi^*(\boldsymbol{x}^o|_{\boldsymbol{y}_c^o=1})^\top. \tag{8}$$

The statistical flow remains fixed during the optimization process, enabling synthetic data to learn a global objective in an optimal, stable, and efficient manner. Our meta loss is designed as the cosine distance between the synthetic flow $\boldsymbol{\mathcal{F}}^s$ and the statistical flow $\boldsymbol{\mathcal{F}}^*$:

$$\mathcal{L}_{sfm} = 1 - cos[flat(\boldsymbol{\mathcal{F}}^*), flat(\boldsymbol{\mathcal{F}}^s)]. \tag{9}$$

Operating within a **supervised learning** paradigm, our method can stably capture the latent structure in the original data. This achieves efficient optimization without mandating the loading of numerous images from the original data and the execution of multiple augmentations on synthetic data.

3.3 Classifier Inheritance

We find that the misalignment between the distillation model ϕ_d and the evaluation model ϕ_e may lead to some degradation in generalization. Motivated by hypothesis transfer learning [7, 29], which closely matches our scene by leveraging a source-domain classifier for target-domain training with limited labeled data and no access to source data, we inherit the pre-trained classifier f from the distillation model trained on the full images, and focus on learning a projector $\mathcal{P}$ composed solely of a lightweight linear layer on the synthetic images $\boldsymbol{x}^s$:

$$\mathcal{L}_{eval} = ||\phi_d(\boldsymbol{x}^s) - \mathcal{P}(\phi_e(\boldsymbol{x}^s))||_2^2. \tag{10}$$

Notably, label information is no longer required for computing the loss, and the feature alignment loss in Equation 10 becomes the only source of optimization gradients. We don't need complex fine-tuning of the hyperparameters for different loss weights like soft labels. Once the projector has been trained, we directly reuse the pre-trained classifier for inference on the validation images $\boldsymbol{x}^v$:

$$\mathcal{I} = f(\mathcal{P}(\phi_e(\boldsymbol{x}^v))). \tag{11}$$

We emphasize that Classifier Inheritance (CI) serves only as an optional plug-in to further improve generalization during evaluation. Unless otherwise specified, all experimental results in this paper are obtained using SFM alone to ensure a fair comparison. More details are provided in the Appendix D.

4 Experiments

4.1 Experimental Setup

Datasets and Backbones. To evaluate the effectiveness of our method, we conduct all experiments on various datasets with 224 × 224 resolution, including universal ImageNet-1k [10], its subsets (ImageFruits, ImageNette, ImageWoof [15]), and fine-grained Stanford Dogs [18] and CUB-200-2011 [34]. We mainly use the "ViT-B" version of four pre-trained self-supervised models or feature extractors as backbones: CLIP [28], DINO-v2 [26], EVA-02 [13], and MoCo-v3 [8].

Baselines and Evaluation. We compare our method with the state-of-the-art method LGM [3], diffusion-based method MGD^3 [6], and three real-image baselines, including **Random** (simply select

Table 2: **The distillation performance on specific models.** We compare our SFM with LGM and several real-image baselines on various datasets. Our SFM, along with its derivatives SFM* and Neighbors (SFM), consistently outperforms other relevant baselines. The superscript "*" indicates multiple augmentations, otherwise a single augmentation. "–" indicates out-of-memory due to resource limitations. The best and second-best results are highlighted in **bold** and underlined.

Train Set (1 Img/Cls)	ImageNet-100					ImageNet-1k				
	CLIP	DINO-v2	EVA-02	MoCo-v3	Average	CLIP	DINO-v2	EVA-02	MoCo-v3	Average
Random	56.6±1.6	74.8±2.8	64.5±2.7	61.4±2.6	64.3±2.4	31.7±0.5	50.3±0.5	37.7±0.4	38.8±0.6	39.6±0.5
Centroids	77.1±0.1	86.9±0.3	80.9±0.2	77.7±0.1	80.6±0.2	53.9±0.0	69.5±0.1	58.1±0.1	57.4±0.0	59.7±0.1
Neighbors (LGM*)	67.8±0.3	86.0±0.2	78.8±0.2	77.1±0.1	77.4±0.2	38.8±0.1	67.7±0.1	49.9±0.1	56.4±0.0	53.2±0.1
Neighbors (SFM)	72.2±0.2	88.0±0.1	80.4±0.0	77.8±0.1	79.6±0.1	48.1±0.1	69.6±0.0	57.3±0.0	57.0±0.0	58.0±0.0
LGM	77.3±0.2	89.8±0.0	85.7±0.1	81.1±0.1	83.5±0.1	55.5±0.0	73.5±0.2	67.6±0.0	61.6±0.0	64.6±0.1
SFM (Ours)	86.7±0.1	90.6±0.0	88.9±0.2	84.7±0.0	87.7±0.1	**67.1**±0.0	**75.1**±0.1	**71.6**±0.1	**63.4**±0.0	**69.3**±0.1
LGM*	84.9±0.1	91.5±0.1	89.0±0.0	83.4±0.1	87.2±0.1	63.0±0.0	75.0±0.1	70.3±0.1	63.2±0.0	67.9±0.0
SFM* (Ours)	**86.9**±0.1	**91.6**±0.1	**89.7**±0.2	**85.6**±0.0	**88.5**±0.1	–	–	–	–	–
Full Dataset	92.5±0.0	95.2±0.1	94.1±0.1	89.4±0.3	92.8±0.1	78.7±0.0	83.0±0.0	81.7±0.1	76.5±0.0	80.0±0.0
Train Set (1 Img/Cls)	Stanford Dogs					CUB-200				
	CLIP	DINO-v2	EVA-02	MoCo-v3	Average	CLIP	DINO-v2	EVA-02	MoCo-v3	Average
Random	23.3±1.5	51.9±1.8	38.3±1.8	36.6±1.4	37.5±1.6	37.5±1.6	64.4±1.5	44.3±1.5	19.1±0.5	41.3±1.3
Centroids	43.3±0.1	73.0±0.2	60.9±0.2	55.2±0.2	58.1±0.2	54.3±0.2	78.5±0.2	59.9±0.3	30.2±0.1	55.7±0.2
Neighbors (LGM*)	33.4±0.1	71.3±0.2	58.5±0.2	56.3±0.1	54.9±0.2	39.4±0.1	76.9±0.0	52.6±0.3	28.1±0.0	49.2±0.1
Neighbors (SFM)	35.7±0.1	72.3±0.2	60.3±0.2	55.5±0.3	56.0±0.2	47.3±0.3	78.7±0.1	56.0±0.2	28.7±0.1	52.7±0.2
LGM	15.0±0.0	79.8±0.2	65.0±0.2	63.5±0.2	55.8±0.2	16.4±0.0	81.1±0.1	51.5±0.2	25.5±0.0	43.6±0.1
SFM (Ours)	55.0±0.3	81.8±0.2	72.9±0.2	67.2±0.3	69.2±0.3	67.6±0.2	85.7±0.1	72.5±0.2	41.4±0.1	66.8±0.2
LGM*	52.1±0.2	83.0±0.1	74.8±0.1	69.6±0.2	69.9±0.2	62.2±0.2	86.0±0.1	74.1±0.2	42.5±0.2	66.2±0.2
SFM* (Ours)	**58.9**±0.0	**83.2**±0.1	**75.0**±0.1	**70.4**±0.2	**71.9**±0.1	**70.5**±0.1	**86.5**±0.0	**75.0**±0.0	**45.5**±0.1	**69.4**±0.1
Full Dataset	76.9±0.1	88.6±0.1	82.6±0.1	72.3±0.5	80.1±0.2	77.5±0.7	90.2±0.2	84.0±0.3	43.7±0.8	73.8±0.5

a random image for each class), **Centroids** (take the real image closest to the mean embedding for each class), and **Neighbors** (choose the real image with the minimum embedding distance to the synthetic image). We use only the final layer of the backbone network to provide transferable features for the task-specific classification head. The reported results represent the averages of 3 trials.

Implementation details. To ensure a fair comparison, we always follow the relevant implementation standards of LGM [3]. Unless otherwise specified, a single differentiable augmentation is applied to our synthetic images. Under the multi-augmentation setting, SFM uses 5 augmentations for CUB-200-2011, 8 for Stanford Dogs, and 10 for ImageNet-100, while LGM uses 3 augmentations for ImageNet-1k and 10 for other datasets. The corresponding LGM* results are cited from the original paper [3]. More implementation details are presented in the Appendix B.

4.2 Distillation Performance on Specific Models

As shown in Table 2, we distill synthetic images using diverse backbones and evaluate them under consistent architectures. With only one augmentation, our SFM consistently outperforms LGM. For example, on ImageNet-1k with CLIP, SFM achieves 67.1%, which is 11.6% higher than LGM (55.5%) and even 4.1% higher than LGM* (63.0%) that uses multiple augmentations. LGM performs extremely poorly or even crashes on fine-grained datasets. For example, on CUB-200 with EVA-02, SFM reaches 72.5%, outperforming LGM (51.5%) by 21%. Moreover, LGM achieves only 15.0% and 16.4% on fine-grained datasets with CLIP, performing even worse than real-image baselines, which further confirms its suboptimality and instability.

Under multiple augmentations, LGM's inability to capture the global optimal distribution is alleviated to some extent, and the gap with SFM* is significantly reduced, but LGM* still underperforms SFM*. Notably, our single-augmentation Neighbors (SFM) generally outperforms the multi-augmentation Neighbors (LGM*). This indicates that our synthetic images reside in a more representative feature space of the entire dataset, benefiting from our optimal global statistical flow objective.

Table 3: **The generalization performance across models.** We synthesize images using a given model and then evaluate them across other models on ImageNet-100. Compared to other baselines, our distilled datasets generalize well and achieve optimal performance across different model pairs, aside from outlier pairs of CLIP and MoCo-v3, which is caused by discrepancies or inconsistencies inherent in the models themselves. The superscript "*" indicates multiple augmentations, otherwise a single augmentation. The best and second-best results are highlighted in **bold** and underlined.

Distill. Model	CLIP					DINO-v2				
Eval. Model	CLIP	DINO-v2	EVA-02	MoCo-v3	Average	CLIP	DINO-v2	EVA-02	MoCo-v3	Average
Centroids	77.1±0.1	85.1±0.2	82.6±0.2	**74.7**±0.1	79.9±0.2	72.4±0.0	86.9±0.3	80.1±0.4	75.8±0.1	78.8±0.2
Neighbors (LGM*)	67.8±0.3	81.1±0.1	75.0±0.2	70.4±0.0	73.5±0.1	71.2±0.1	86.0±0.2	78.0±0.4	74.7±0.0	77.6±0.2
Neighbors (SFM)	72.2±0.2	83.6±0.2	75.8±0.0	72.3±0.1	76.0±0.1	72.7±0.0	88.0±0.1	80.0±0.2	75.4±0.0	79.0±0.1
LGM	77.3±0.2	70.2±0.4	70.4±0.1	41.8±0.2	64.9±0.2	70.8±0.1	89.8±0.0	82.4±0.2	72.5±0.1	78.9±0.1
SFM (Ours)	86.7±0.1	84.9±0.3	85.5±0.2	68.7±0.0	81.5±0.2	78.6±0.0	90.6±0.0	86.2±0.1	80.5±0.0	84.0±0.0
LGM*	84.9±0.1	80.8±0.4	83.8±0.2	61.6±0.2	77.8±0.2	77.0±0.1	91.5±0.1	86.8±0.1	78.8±0.1	83.5±0.1
SFM* (Ours)	**86.9±0.1**	**86.5±0.2**	**87.5±0.1**	72.0±0.1	**83.2±0.1**	**79.8±0.0**	**91.6±0.1**	**87.1±0.1**	**81.4±0.0**	**85.0±0.1**
Distill. Model	EVA-02					MoCo-v3				
Eval. Model	CLIP	DINO-v2	EVA-02	MoCo-v3	Average	CLIP	DINO-v2	EVA-02	MoCo-v3	Average
Centroids	70.7±0.1	85.2±0.1	80.9±0.2	72.6±0.1	77.4±0.1	75.5±0.2	85.8±0.1	81.7±0.1	77.7±0.1	80.2±0.1
Neighbors (LGM*)	69.5±0.1	85.4±0.3	78.8±0.2	72.3±0.1	76.5±0.2	72.6±0.3	86.2±0.2	81.1±0.5	77.1±0.1	79.7±0.3
Neighbors (SFM)	69.4±0.3	85.2±0.1	80.4±0.0	74.7±0.1	77.4±0.1	**75.8**±0.0	85.7±0.1	81.2±0.1	77.8±0.1	80.1±0.1
LGM	69.7±0.0	81.4±0.3	85.7±0.1	52.0±0.0	72.2±0.1	54.3±0.1	81.7±0.1	72.9±0.1	81.1±0.1	72.5±0.1
SFM (Ours)	81.0±0.2	88.0±0.2	88.9±0.2	70.5±0.0	82.1±0.2	66.3±0.1	86.7±0.2	79.3±0.2	84.7±0.0	79.3±0.1
LGM*	75.5±0.2	86.4±0.1	89.0±0.0	67.7±0.1	79.7±0.1	65.6±0.1	86.6±0.1	82.3±0.2	83.4±0.1	79.5±0.1
SFM* (Ours)	**81.5±0.1**	**88.9±0.2**	**89.7±0.2**	**75.5±0.0**	**83.9±0.1**	72.5±0.0	**88.9±0.2**	**84.5±0.2**	**85.6±0.0**	**82.9±0.1**
Full Dataset	92.5±0.0	95.2±0.1	94.1±0.1	89.4±0.3	92.8±0.1	92.5±0.0	95.2±0.1	94.1±0.1	89.4±0.3	92.8±0.1

Table 4: The Independent and combined effects of Target Class (TCDD) and Non-target Class (NCDD) Dataset Distillation on datasets of varying scales. The results represent the average cross-architecture generalization performance distilled from CLIP.

TCDD	NCDD	IN-Fruits	IN-100	IN-1k
×	✓	15.8±1.2	0.77±0.0	0.10±0.1
✓	×	**77.6**±0.7	79.4±0.1	57.9±0.0
✓	✓	76.5±1.1	**81.5**±0.2	**60.7**±0.1

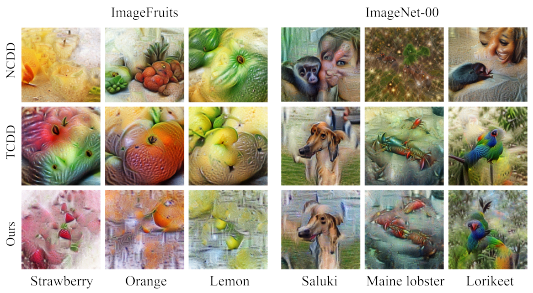


Figure 3: A comparison between images synthesized on ImageNet-100 by TCDD and NCDD.

4.3 Generalization Performance Across Models

As depicted in Table 3, we distill images using a given model and then evaluate them across other models that are not seen during distillation. The generalization of our SFM/SFM*/Neighbors(SFM) generally surpasses that of LGM/LGM*/Neighbors (LGM*). We observe that (1) whether used as a distillation model or an evaluation model, the generalization performance of DINO-v2 remains at the highest level, and (2) even when evaluating images distilled using the same model, the performance gap remains significant across different models. For example, when MoCo-v3 is used as the distillation model, DINO-v2 can reach 86.6%, while CLIP is only 66.3%, which is far inferior to other real-image baselines. Both cases indicate that the type and quality of self-supervised models can affect the generalization ability of the synthetic dataset. Additionally, we find that CLIP and MoCo-v3 form an anomalous pair, making them unsuitable for distillation or evaluation, as their performance falls short of the real-image baselines. This is due to the gap and inconsistency in the feature representations learned by the two models [3, 16].

4.4 Ablation and Extension

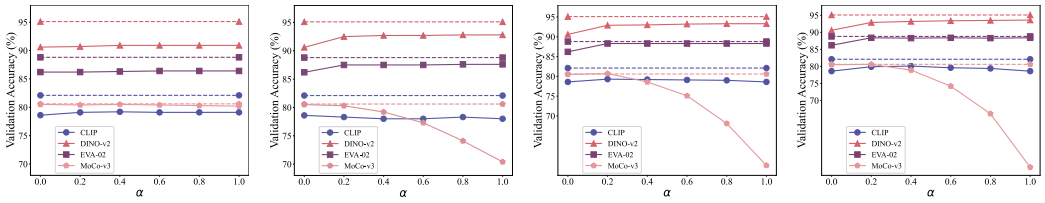
Target Class and Non-target Class. We distill synthetic images using the two parts in Equation 8 separately, termed Target Class Dataset Distillation (TCDD) and Non-target Class Dataset Distillation

Table 5: Comparison of LGM and SFM under different IPC settings. The distillation model is DINO-v2 and the eval model is EVA-02.

IPC	1	2	5	10
Random	64.5±2.7	72.9±0.2	80.9±0.1	85.0±0.2
MGD^3	77.0±0.2	80.9±0.0	83.5±0.1	85.8±0.1
LGM	82.4±0.2	84.9±0.2	85.5±0.0	85.5±0.1
SFM	**86.2**±0.1	**86.5**±0.1	**87.5**±0.1	**88.4**±0.0

Table 6: Performance of our CI combined with existing distillation methods.

Method	CLIP	DINO-v2	EVA-02	MoCo-v3
MGD^3	64.4±0.4	84.0±0.3	77.0±0.2	71.9±0.1
MGD^3+**CI**	71.4±0.1	94.8±0.0	82.6±0.1	74.6±0.1
LGM	70.8±0.1	89.8±0.0	82.4±0.2	72.5±0.1
LGM+**CI**	76.7±0.1	95.0±0.0	84.6±0.1	75.1±0.0
SFM	78.6±0.0	90.6±0.0	86.2±0.1	80.5±0.0
SFM+**CI**	**82.1**±0.0	**95.1**±0.1	**88.8**±0.1	**80.6**±0.1



(a) T=1 (b) T=4 (c) T=8 (d) T=16

Figure 4: A comparison between soft label (solid line) with different temperatures and our CI (dashed line). Our CI outperforms soft labels across all settings.

(NCDD). As shown in Table 4, NCDD individually collapses across all dataset scales. In contrast, TCDD alone outperforms our SFM (TCDD+NCDD) on the small-scale ImageFruits with only 10 classes, indicating that preserving basic representative features (absolute distribution centers) is sufficient. However, when handling large-scale datasets, our SFM demonstrates greater insight and delivers superior performance gains. As shown in Figure 3, NCDD deviates entirely from the target-class representation and produces unrelated non-target classes that harm performance, while TCDD effectively captures discriminative features of the target classes and synthesizes images relevant to the target class. Notably, our synthetic images appear fragmented on the small-scale ImageFruits and are less informative than TCDD, which may explain the performance drop. On the large-scale ImageNet-100, our synthetic images share similar semantics or structures with TCDD, yet our misty quality brings more expressiveness and insight, making them stand out in evaluation.

Few-shot Learning. To demonstrate the scalability of our method and its stability under different settings, we extend the extreme 1 IPC to few-shot learning (IPC=2, 5, 10). As shown in Table 5, our SFM can be successfully extended to different IPC and always maintains optimal performance. In contrast, LGM quickly reaches saturation as IPC increases (85.5% at both IPC=5 and IPC=10), and the gap with Random and MGD^3 narrows (85.5% vs. 85.0% and 85.8%).

Classifier Inheritance Helps Model Alignment. In our previous Table 3, we find that misalignment between the distillation and evaluation models hinders generalization. To address this, we propose an alternative classifier inheritance (CI) strategy that trains the evaluation model directly using the distillation model's extracted features as "labels", followed by inference with a classifier pre-trained on the original dataset, ensuring model alignment. As illustrated in Table 6, we use DINO-v2 (while MGD^3 uses the original VAE and diffusion) on ImageNet-100 for distillation and evaluate on other models. Results demonstrate that CI consistently boosts generalization as a plug-in module.

Classifier Inheritance Beats Soft Label. The soft label [27, 33] has been widely established as an effective evaluation method for DD. In Figure 4, we train our classifier using a weighted loss $\alpha\ell_{kl}+(1-\alpha)\ell_{ce}$, where soft labels are provided by the distillation model along with its corresponding classifier. We observe that soft labels are highly architecture-sensitive. For example, when T=4, they improve performance for DINO-v2 and EVA-02, but they provide no benefit for CLIP and even degrade results for MoCo-v3, as performance drops sharply with increased Kullback–Leibler (KL) divergence loss weight α. Overall, CI consistently outperforms soft labels, confirming that our inherited classifier is crucial for generalization. Additionally, we find that regardless of how the temperature is adjusted, soft labels consistently fail to surpass our CI. But as T increases, the gap between them becomes smaller. When T=1, soft labels hardly affect performance. In addition, it is worth noting that MoCo-v3 is highly sensitive to both the temperature and the weight α of the soft labels. As T and α increase, the more detrimental the impact of soft labels on performance becomes.

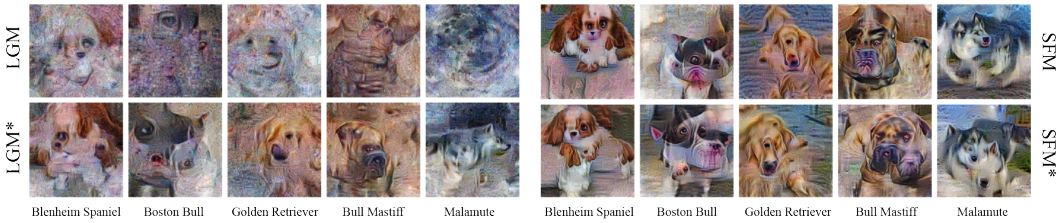
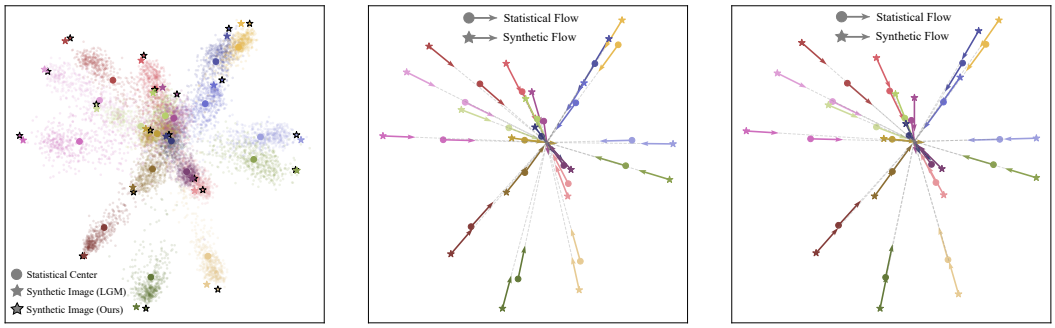


Figure 5: A comparison of images synthesized by LGM versus our SFM under a single augmentation, alongside LGM* and SFM* under multiple augmentations. The dataset is Stanford Dogs.



(a) Distribution (b) LGM (c) Ours

Figure 6: The (a) distribution, along with the flows of (b) LGM and (c) our SFM, on ImageNet-100. We perform PCA for dimensionality reduction, visualizing 20 classes. Both LGM and our SFM synthesize samples on the distribution boundary with discriminative attributes.

Visualization and Explainability. As shown in Figure 5, our synthetic images are clearer than those of LGM and exhibit clear structures with no noticeable artifacts, indicating the stability of our global-level SFM during the optimization process. As shown in Figure 6 (a), synthetic images reside on the distribution boundary that carries discriminative information and is distant from the distribution center, which is also why this paradigm is more effective. As shown in Figures 6 (b) and (c), compared to the dynamic process of LGM, our synthetic flow is closer to the statistical flow. This further confirms that our supervised paradigm effectively captures the globally optimal objective.

5 Conclusion

In this work, we are the first to propose the concept of "flow" in DD, from which we derive a novel paradigm termed Statistical Flow Matching that encourages the synthetic images to efficiently learn the statistical flow of the original dataset through the pre-trained self-supervised model. Based on this, we efficiently achieve superior downstream performance using only one image per class, which has a positive impact on edge environments with limitations in computation, storage, and communication.

Limitation and Future Work. As the number of augmentations increases, our method currently processes all classes simultaneously, which still results in greater GPU memory requirements. Notably, each of our loading steps can be class-independent. Therefore, in future work, we will process classes in blocks of a fixed size, keeping memory usage constant and allowing flexible adjustment of the class block size based on available computational resources.

Acknowledgments and Disclosure of Funding

I acknowledge the support of the China Scholarship Council (CSC) (Grant No. 202506070011) for my doctoral studies. This work is supported by the Sichuan Science and Technology Program (Granted No. 2024ZDZX0011, No. 2025ZHCG0002 and No. 2026NSFSC1482). This work is also supported by Xin Zhang's A*STAR Career Development Fund (CDF) (Project No. H26-KSR0051), as well as by the Japan Science and Technology Agency (JST) and the Agency for Science, Technology and Research (A*STAR) under the Japan-Singapore Joint Call (Project No. R24I6IR133).

References

[1] Y. Bao, Y. Liu, Z. Chen, Y. Liang, M. Li, and K. Ma. Dataset distillation as data compression: A rate-utility perspective. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 519–529, 2025.

[2] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[3] G. Cazenavette, A. Torralba, and V. Sitzmann. Dataset distillation for pre-trained self-supervised vision models. *NeurIPS*, 2025.

[4] G. Cazenavette, T. Wang, A. Torralba, A. A. Efros, and J.-Y. Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4750–4759, 2022.

[5] G. Cazenavette, T. Wang, A. Torralba, A. A. Efros, and J.-Y. Zhu. Generalizing dataset distillation via deep generative prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3739–3748, 2023.

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

[7] D. Chen, J.-P. Mei, H. Zhang, C. Wang, Y. Feng, and C. Chen. Knowledge distillation with the reused teacher classifier. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11933–11942, 2022.

[8] X. Chen, S. Xie, and K. He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9640–9649, 2021.

[9] Y. Chen, M. Welling, and A. Smola. Super-samples from kernel herding. *arXiv preprint arXiv:1203.3472*, 2012.

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[11] M. Ding, Y. Xu, T. Rabbani, X. Liu, B. Gravelle, T. Ranadive, T.-C. Tuan, and F. Huang. Calibrated dataset condensation for faster hyperparameter search. *arXiv preprint arXiv:2405.17535*, 2024.

[12] J. Du, Y. Jiang, V. Y. Tan, J. T. Zhou, and H. Li. Minimizing the accumulated trajectory error to improve dataset distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3749–3758, 2023.

[13] Y. Fang, Q. Sun, X. Wang, T. Huang, X. Wang, and Y. Cao. Eva-02: A visual representation for neon genesis. *Image and Vision Computing*, 149:105171, 2024.

[14] S. Fort and J. Whitaker. Direct ascent synthesis: Revealing hidden generative capabilities in discriminative models. *arXiv preprint arXiv:2502.07753*, 2025.

[15] J. Howard. A smaller subset of 10 easily classified classes from imagenet, and a little more french. *URL https://github. com/fastai/imagenette*, 4, 2019.

[16] M. Huh, B. Cheung, T. Wang, and P. Isola. Position: The platonic representation hypothesis. In *Forty-first International Conference on Machine Learning*, 2024.

[17] Y. Jia, S. Vahidian, J. Sun, J. Zhang, V. Kungurtsev, N. Z. Gong, and Y. Chen. Unlocking the potential of federated learning: The symphony of dataset distillation via deep generative latents. In *European Conference on Computer Vision*, pages 18–33. Springer, 2024.

[18] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei. Novel dataset for fine-grained image categorization. In *CVPR*, 2011.

[19] S. Lei and D. Tao. A comprehensive survey of dataset distillation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(1):17–32, 2023.

[20] H. Li, Y. Zhou, X. Gu, B. Li, and W. Wang. Diversity-enhanced distribution alignment for dataset distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3747–3756, 2025.

[21] P. Liao, X. Li, X. Liu, and K. Keutzer. The artbench dataset: Benchmarking generative models with artworks. *arXiv preprint arXiv:2206.11404*, 2022.

[22] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[23] X. Liu, C. Gong, and Q. Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[24] D. Medvedev and A. D'yakonov. Learning to generate synthetic training data using gradient matching and implicit differentiation. In *International Conference on Analysis of Images, Social Networks and Texts*, pages 138–150. Springer, 2021.

[25] M. Oh and J.-Y. Sim. Asymmetric synthetic data update for domain incremental dataset distillation. In *The Fourteenth International Conference on Learning Representations*, 2026.

[26] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[27] T. Qin, Z. Deng, and D. Alvarez-Melis. A label is worth a thousand images in dataset distillation. *Advances in Neural Information Processing Systems*, 37:131946–131971, 2024.

[28] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[29] I. Redko, E. Morvant, A. Habrard, M. Sebban, and Y. Bennani. A survey on domain adaptation theory: learning bounds and theoretical guarantees. *arXiv preprint arXiv:2004.11829*, 2020.

[30] S. Shao, Z. Yin, M. Zhou, X. Zhang, and Z. Shen. Generalized large-scale data condensation via various backbone and statistical matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16709–16718, 2024.

[31] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa, F. Massa, D. Haziza, L. Wehrstedt, J. Wang, T. Darcet, T. Moutakanni, L. Sentana, C. Roberts, A. Vedaldi, J. Tolan, J. Brandt, C. Couprie, J. Mairal, H. Jégou, P. Labatut, and P. Bojanowski. DINOv3, 2025.

[32] D. Su, J. Hou, W. Gao, Y. Tian, and B. Tang. D^4m: Dataset distillation via disentangled diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5809–5818, 2024.

[33] P. Sun, B. Shi, D. Yu, and T. Lin. On the diversity and realism of distilled dataset: An efficient dataset distillation paradigm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9390–9399, 2024.

[34] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. Caltech-UCSD birds 200. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[35] S. Wang, Y. Yang, Z. Liu, C. Sun, X. Hu, C. He, and L. Zhang. Dataset distillation with neural characteristic function: A minmax perspective. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 25570–25580, 2025.

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

[37] K. Wei, R. Iyer, and J. Bilmes. Submodularity in data subset selection and active learning. In *International conference on machine learning*, pages 1954–1963. PMLR, 2015.

[38] X. Wu, B. Zhang, Z. Deng, and O. Russakovsky. Vision-language dataset distillation. *arXiv preprint arXiv:2308.07545*, 2023.

[39] Q. Xia, Z. Shu, W. Jiang, J. Du, J. Wang, and G. Lu. Diver: Diving deeper into distilled data via expressive semantic recovery. *arXiv preprint arXiv:2605.12649*, 2026.

[40] G. Yan, L. Xie, X. Gao, W. Zhang, Q. Shen, Y. Fang, and Z. Wu. Fedvck: Non-iid robust and communication-efficient federated learning via valuable condensed knowledge for medical image analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 21904–21912, 2025.

[41] E. Yang, L. Shen, Z. Wang, T. Liu, and G. Guo. An efficient dataset condensation plugin and its application to continual learning. *Advances in Neural Information Processing Systems*, 36:67625–67642, 2023.

[42] H. Yang, D. Lv, S. Ma, W. Xi, and J. Zhao. Sneakdoor: Stealthy backdoor attacks against distribution matching-based dataset condensation. *arXiv preprint arXiv:2603.28824*, 2026.

[43] Z. Yin, E. Xing, and Z. Shen. Squeeze, recover and relabel: Dataset condensation at imagenet scale from a new perspective. *Advances in Neural Information Processing Systems*, 36:73582–73603, 2023.

[44] X. Zhang, J. Du, P. Liu, and J. T. Zhou. Breaking class barriers: Efficient dataset distillation via inter-class feature compensator. In *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2025.

[45] Z. Zhang, J. DU, Z. Liu, J. T. Zhou, et al. Beyond modality collapse: Representation blending for multimodal dataset distillation. *Advances in Neural Information Processing Systems*, 38:114809–114829, 2026.

[46] B. Zhao and H. Bilen. Dataset condensation with distribution matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6514–6523, 2023.

[47] B. Zhao, K. R. Mopuri, and H. Bilen. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*, 2020.

[48] R. Zheng, V. A. Dasu, Y. O. Wang, H. Wang, and F. De la Torre. Improving noise efficiency in privacy-preserving dataset distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4838–4847, 2025.

[49] W. Zhong, H. Tang, Q. Zheng, M. Xu, Y. Hu, and W. Guan. Towards stable and storage-efficient dataset distillation: Matching convexified trajectory. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 25581–25589, 2025.

Appendix

A Theoretical Proof

A.1 The Proof of Equation 1

The cross-entropy loss ℓ is:

$$\boldsymbol{\ell} = -\frac{1}{B}\sum_{i=1}^{B}\sum_{c=1}^{C} y_{ic}\log(p_{ic}), \tag{12}$$

where y_{ic} is the label value of the i-th sample and the c-th category. The predicted probability p_{ic} is:

$$p_{ic} = \text{softmax}(z_{ic}) = \frac{\exp\big(z_{ic}\big)}{\sum_{j=1}^{C}\exp\big(z_{ij}\big)}, \tag{13}$$

where $\boldsymbol{p}_i = [p_{i1}, p_{i2}, \cdots, p_{ic}, \cdots, p_{iC}]$ is the probability distribution of the i-th sample, $\boldsymbol{z}_i = [z_{i1}, z_{i2}, \cdots, z_{ic}, \cdots, z_{iC}] = \boldsymbol{W}\boldsymbol{\phi}(\boldsymbol{x}_i)$ is the logits of the i-th sample. $\boldsymbol{W}$ is the linear classifier matrix.

The gradient of the loss function $\boldsymbol{\ell}$ with respect to p_{ic} is:

$$\frac{\partial \boldsymbol{\ell}}{\partial p_{ic}} = -\frac{y_{ic}}{p_{ic}}. \tag{14}$$

The gradient of p_{ic} with respect to z_{ic} has two cases.

When $c = c'$ (the same class), the derivative is:

$$\begin{aligned}\frac{\partial p_{ic}}{\partial z_{ic'}} &= \partial\Big(\frac{\exp\big(z_{ic}\big)}{\sum_{j=1}^{C}\exp\big(z_{ij}\big)}\Big)/\partial z^{s}_{ic'} \\ &= \frac{\exp\big(z_{ic}\big)}{\sum_{j=1}^{C}\exp\big(z_{ij}\big)}\frac{\sum_{j=1}^{C}\exp\big(z_{ic}\big) - \exp\big(z_{ic}\big)}{\sum_{j=1}^{C}\exp\big(z_{ij}\big)} \\ &= p_{ic}(1-p_{ic}).\end{aligned} \tag{15}$$

When $c \neq c'$ (different classes), the derivative is:

$$\begin{aligned}\frac{\partial p_{ic}}{\partial z_{ic'}} &= \partial\Big(\frac{\exp\big(z_{ic}\big)}{\sum_{j=1}^{C}\exp\big(z_{ij}\big)}\Big)/\partial z_{ic'} \\ &= -\frac{\exp\big(z_{ic'}\big)\exp\big(z_{ic}\big)}{(\sum_{j=1}^{C}\exp\big(z_{ij}\big))^2} \\ &= -p_{ic}\cdot p_{ic'},\end{aligned} \tag{16}$$

For $\boldsymbol{\ell}$, the gradient for each part of the logits z_{ic} is calculated by chain rule:

$$\begin{aligned}\frac{\partial \ell}{\partial z_{ic}} &= \sum_{j=1}^{C}\frac{\partial \ell}{\partial p_{ij}}\cdot\frac{\partial p_{ij}}{\partial z_{ic}} \\ &= -\frac{y_{ic}}{p_{ic}}p_{ic}(1-p_{ic}) + \sum_{j\neq i}^{C} y_{ij}p_{ic} \\ &= (-y_{ic} + y_{ic}p_{ic} + \sum_{j\neq i}^{C} y_{ij}p_{ic}) \\ &= (-y_{ic} + p_{ic}\sum_{j=1}^{C} y_{ij}) \\ &= (p_{ic} - y_{ic})\end{aligned} \tag{17}$$

where $z_{ic} = W_c\boldsymbol{\phi}(\boldsymbol{x}_i)$, the gradient of the cross-entropy loss $\boldsymbol{\ell}$ with respect to W_c is:

$$\frac{\partial \boldsymbol{\ell}}{\partial W_c} = \frac{1}{B}\sum_{i=1}^{B}\frac{\partial \boldsymbol{\ell}}{\partial z_{ic}}\cdot\frac{\partial z_{ic}}{\partial W_c} = \frac{1}{B}\sum_{i=1}^{B}(p_{ic} - y_{ic})\boldsymbol{\phi}(x_i)^\top, \tag{18}$$

Therefore, the gradient of the cross-entropy loss $\boldsymbol{\ell}$ with respect to $\boldsymbol{W}$ is:

$$\frac{\partial \boldsymbol{\ell}}{\partial \boldsymbol{W}} = \frac{1}{B}\sum_{i=1}^{B}(\boldsymbol{p}_i - \boldsymbol{y}_i)\boldsymbol{\phi}(\boldsymbol{x}_i)^\top, \tag{19}$$

A.2 The Proof of Theorem 1

We aim to compute the expectation of the predicted probability p_{ic} for class c. The softmax function for class c is defined as:

$$p_{ic} = \frac{e^{W_c^T\boldsymbol{\phi}(x_i)}}{\sum_{c'=1}^{C} e^{W_{c'}^T\boldsymbol{\phi}(x_i)}}, \tag{20}$$

where W_c is the weight vector for class c and $\boldsymbol{\phi}(x_i)$ is the feature vector for sample x_i. The weight vectors W_c are independently initialized from a Gaussian distribution $W_c \sim N(0, \sigma_c^2 I)$

We now calculate the expectation of p_{ic}, which is given by:

$$\mathbb{E}[p_{ic}] = \mathbb{E}\left[\frac{e^{W_c^T\boldsymbol{\phi}(x_i)}}{\sum_{c'=1}^{C} e^{W_{c'}^T\boldsymbol{\phi}(x_i)}}\right]. \tag{21}$$

Since the weight vectors W_c are independent and identically distributed (i.i.d.) from the same Gaussian distribution, the scoring terms $W_c^T\boldsymbol{\phi}(x_i)$ are symmetric in expectation. Therefore, due to the exchangeability of the weight vectors, the expectation of the class probabilities will be the same across all classes. This implies:

$$\mathbb{E}[e^{W_c^T x_i}] = \mathbb{E}[e^{W_{c'}^T\boldsymbol{\phi}(x_i)}] \quad \text{for all} \quad c, c' \in \{1, 2, \ldots, C\}. \tag{22}$$

Now, considering the symmetry of the weight vectors, the softmax function will normalize the probabilities such that each class will have the same expected probability. Hence, the expected probability for each class is:

$$E[p_{ic}] = \frac{\mathbb{E}[e^{W_c^T\boldsymbol{\phi}(x_i)}]}{\mathbb{E}\left[\sum_{c'=1}^{C} e^{W_{c'}^T\boldsymbol{\phi}(x_i)}\right]} = \frac{1}{C}. \tag{23}$$

Therefore, for all classes $c \in \{1, 2, \ldots, C\}$, we have:

$$\mathbb{E}[p_{i1}] = \mathbb{E}[p_{i2}] = \cdots = \mathbb{E}[p_{iC}] = \frac{1}{C}. \tag{24}$$

A.3 The Proof of Theorem 2

1. Derivation of the Expectation of X

The expected value of X is given by $\mathbb{E}[X] = \mathbb{E}[e^Y]$. Since $Y \sim N(\mu, \sigma^2)$, the probability density function of Y is:

$$f_Y(y) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right), \tag{25}$$

Thus, the expectation of X is:

$$\begin{aligned}\mathbb{E}[e^Y] &= \int_{-\infty}^{\infty} e^y f_Y(y)\, dy \\ &= \int_{-\infty}^{\infty} e^y \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right) dy \\ &= \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{\infty} \exp\left(y - \frac{(y-\mu)^2}{2\sigma^2}\right) dy,\end{aligned} \tag{26}$$

To simplify the integral, we shift the variable by setting $z = y - \mu$, so $dz = dy$ and $y = z + \mu$. This gives:

$$\mathbb{E}[e^Y] = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{\infty} \exp\left((z+\mu) - \frac{z^2}{2\sigma^2}\right) dz, \tag{27}$$

Now, factor out $\exp(\mu)$:

$$\mathbb{E}[e^Y] = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(\mu) \int_{-\infty}^{\infty} \exp(z) \exp\left(-\frac{z^2}{2\sigma^2}\right) dz, \tag{28}$$

We know that the Gaussian integral evaluates to:

$$\int_{-\infty}^{\infty} \exp(z) \exp\left(-\frac{z^2}{2\sigma^2}\right) dz = \sqrt{2\pi\sigma^2} \exp(\frac{\sigma^2}{2}), \tag{29}$$

Thus, we obtain the expectation:

$$\mathbb{E}[X] = \mathbb{E}[e^Y] = \exp(\mu) \cdot \exp\left(\frac{\sigma^2}{2}\right) = \exp\left(\mu + \frac{\sigma^2}{2}\right). \tag{30}$$

2. Derivation of the Variance of X

The variance of X is given by:

$$\mathrm{Var}[X] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2, \tag{31}$$

First, we compute $\mathbb{E}[X^2] = \mathbb{E}[e^{2Y}]$. The expectation is:

$$\begin{aligned}\mathbb{E}[e^{2Y}] &= \int_{-\infty}^{\infty} e^{2y} f_Y(y)\, dy \\ &= \int_{-\infty}^{\infty} e^{2y} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right) dy \\ &= \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{\infty} \exp\left(2y - \frac{(y-\mu)^2}{2\sigma^2}\right) dy,\end{aligned} \tag{32}$$

Shifting the variable again by setting $z = y - \mu$, we get:

$$\mathbb{E}[e^{2Y}] = \exp(2\mu) \cdot \exp(2\sigma^2), \tag{33}$$

Now, we compute the variance:

$$\begin{aligned}\mathrm{Var}[X] &= \mathbb{E}[e^{2Y}] - \left(\mathbb{E}[e^Y]\right)^2 \\ &= \exp(2\mu) \cdot \exp(2\sigma^2) - \left(\exp(\mu) \cdot \exp\left(\frac{\sigma^2}{2}\right)\right)^2 \\ &= \exp\left(2\mu + \sigma^2\right)\left(\exp(\sigma^2) - 1\right).\end{aligned} \tag{34}$$

B More Implementation Details

Following LGM [3], we train the pyramid representations [14] with the Adam optimizer at a learning rate of 0.002. We apply RandomHorizontalFlip (default parameters), RandomResizedCrop (224×224), and RandomGaussianNoise (σ=0.2) for differentiable augmentation of synthetic images. We distill the synthetic images for 5000 iterations, adding a new level to the pyramid every 200 iterations. During evaluation, the linear classifier is trained for 1000 epochs with a batch size of 100, optimized with Adam. The initial learning rate is set to 0.001/256, which follows the DINO-v1 [2] evaluation protocol, and employs a cosine decay schedule. An early stopping criterion is applied, halting training if test accuracy shows no improvement for 50 epochs. We calculate the statistical center using the total number of images per class as the batch size. If GPU memory is limited, a smaller batch size can be used and yields the same result, demonstrating that our center computation is resource-friendly. We use one H100 GPU for a single augmentation, and two H100 GPUs otherwise.

In this paper, we use an earlier version of LGM, where the Gaussian noise has a mean of 0.5 instead of 0. See here for details. This would likely result in darker distilled images since they would have to compensate for the brighter mean noise. In fact, we observe this difference in our experiments. As shown in Table 7, changing the mean to 0 indeed leads to a performance improvement.

Table 7: Effect of different gaussian noise means on performance. The distillation model is EVA-02.

Mean	CLIP	DINO-v2	EVA-02	MoCo-v3
0.5	81.0±0.2	88.0±0.2	88.9±0.2	70.5±0.0
0	82.2±0.0	88.8±0.0	88.6±0.3	76.6±0.0

C Practical Applications or Scenarios

In our view, dataset distillation, especially in the era of powerful pre-trained self-supervised backbones, matters as a form of data compression and distribution. It transforms part of the learning problem into a resource-aware "how to package and deliver training signals" problem, which becomes valuable when storage, communication, and training costs are bottlenecks. In our setting, which involves very low IPC and frozen pre-trained backbones, concrete scenarios that can benefit include:

Edge-device deployment and weak-connectivity settings: Synthetic datasets are extremely lightweight to store and transmit, enabling faster model updates when bandwidth, storage, or compute is limited.

Cross-site sharing and fast reproducibility: Distilled data can serve as a compact surrogate training set, reducing the burden of dataset management, transfer, and storage for quick verification and reproduction.

Fast task bootstrap with strong backbones: When downstream training is largely “train a lightweight head”, pre-computed distilled sets can shorten iteration cycles by reducing the need to repeatedly curate and distribute large datasets.

Privacy-friendly collaboration: Distilled data does not originate from the original dataset and is not as authentic as the original dataset, thus protecting user privacy to some extent.

Extension to richer tasks (future work): We plan to extend the paradigm to object detection and segmentation, adapting to their task-specific heads, where a small distilled dataset is synthesized on a self-supervised specific backbone and then trained on other downstream backbones.

D Classifier Inheritance

The goal of a neural network in the classification task is to learn a decision boundary that separates different classes in the feature space. In the evaluation of our downstream tasks, as the self-supervised model is frozen, the classifier uniquely learns task-specific information and determines the decision boundary. However, given that our synthetic dataset is highly compressed, such limited data makes it difficult to learn the decision boundary or discriminant information originally acquired by the classifier on the original dataset.

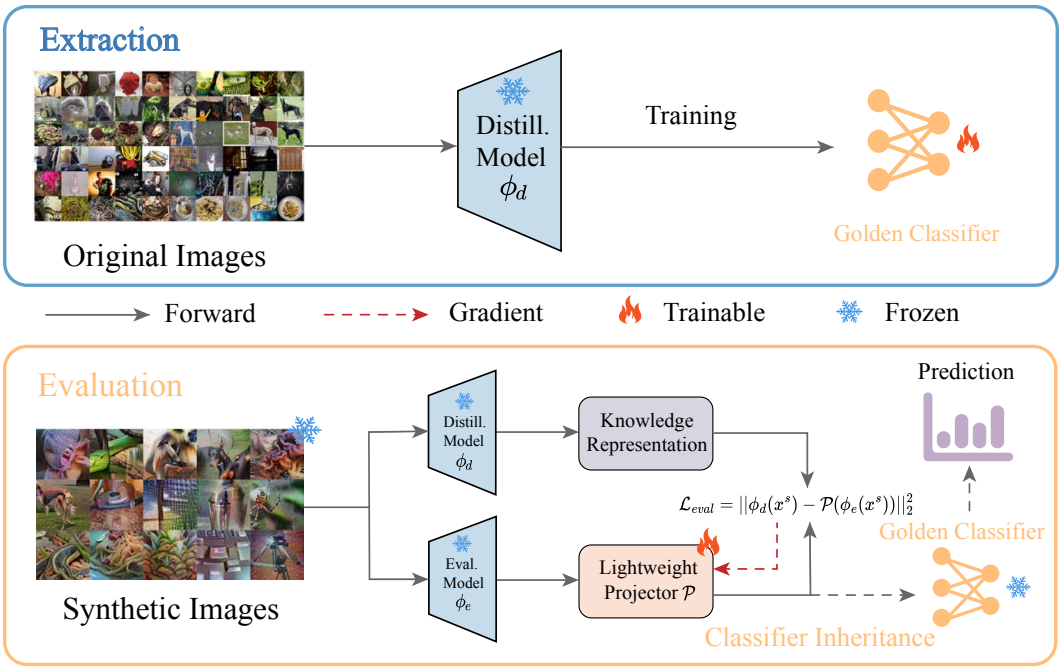


Figure 7: An overview of our Classifier Inheritance (CI). We first train a classifier on the original dataset using a frozen distillation model, then freeze it. During evaluation on synthetic images, we directly drive the evaluation model to align its feature representation with that of the distillation model, and use the aforementioned classifier for inference and prediction.

Table 8: The performance of the gold classifier trained on various full datasets.

Epoch	ImageNet-100					ImageNet-1k				
	CLIP	DINO-v2	EVA-02	MoCo-v3	Average	CLIP	DINO-v2	EVA-02	MoCo-v3	Average
10	92.8±0.1	95.3±0.1	94.5±0.0	88.2±0.1	92.7±0.1	79.0±0.0	82.0±0.0	82.4±0.1	73.9±0.0	79.3±0.0
Epoch	Stanford Dogs					CUB-200				
	CLIP	DINO-v2	EVA-02	MoCo-v3	Average	CLIP	DINO-v2	EVA-02	MoCo-v3	Average
10	72.0±0.1	88.8±0.0	83.2±0.1	70.4±0.3	78.6±0.1	72.5±0.4	89.7±0.1	83.2±0.1	45.3±0.4	72.7±0.3

In our experiments, the projector $\mathcal{P}$ is extremely lightweight, consisting only of a single linear layer. On the one hand, it aligns the input dimensionality of the golden classifier, and on the other hand, it learns architecture-specific knowledge from the distillation model. Such specific knowledge is crucial for cross-architecture generalization, as the images synthesized according to Equation 9 are inherently biased toward the distillation model ϕ_d, and this cannot be eliminated during evaluation. Therefore, we embrace this bias and directly learn the representations of the distillation model.

Our Classifier Inheritance (CI) is simple and storage-efficient, requiring only an additional classifier and one image per class to be stored. This design is well-suited to bandwidth-limited, weakly connected settings and resource-constrained edge environments.

We only train our golden classifier for 10 epochs, and its performance is shown in Table 8. Remarkably, it attains high performance with just 10 epochs, highlighting the strong feature extraction ability of pre-trained self-supervised models.

Table 9: The cross-model generalization performance on ImageNet-1k.

Distill. Model	CLIP					DINO-v2				
Eval. Model	CLIP	DINO-v2	EVA-02	MoCo-v3	Average	CLIP	DINO-v2	EVA-02	MoCo-v3	Average
LGM*	63.0±0.0	56.4±0.1	59.7±0.1	39.5±0.0	54.7±0.1	54.1±0.0	75.0±0.1	65.4±0.1	**60.0±0.0**	63.7±0.1
LGM	55.5±0.0	45.0±0.0	48.5±0.0	25.8±0.0	43.7±0.0	46.7±0.0	73.5±0.2	61.2±0.1	54.7±0.1	59.0±0.1
SFM	**67.1±0.0**	**64.2±0.1**	**64.7±0.0**	**46.6±0.0**	**60.7±0.1**	**56.6±0.0**	**75.1±0.1**	**65.6±0.0**	59.9±0.0	**64.3±0.0**
Distill. Model	EVA-02					MoCo-v3				
Eval. Model	CLIP	DINO-v2	EVA-02	MoCo-v3	Average	CLIP	DINO-v2	EVA-02	MoCo-v3	Average
LGM*	55.5±0.1	65.9±0.1	70.3±0.1	**51.8±0.0**	60.9±0.1	41.4±0.0	**66.9±0.1**	57.2±0.1	63.2±0.0	57.2±0.1
LGM	47.8±0.1	61.1±0.1	67.6±0.0	38.9±0.0	53.9±0.1	32.9±0.0	61.5±0.1	47.5±0.0	61.6±0.0	50.9±0.1
SFM	**58.8±0.1**	**70.0±0.2**	**71.6±0.1**	51.2±0.0	**62.9±0.1**	**47.0±0.1**	65.2±0.1	**57.5±0.1**	**63.4±0.0**	**58.3±0.1**
Full Dataset	78.7±0.0	83.0±0.0	81.7±0.1	76.5±0.0	80.0±0.0	78.7±0.0	83.0±0.0	81.7±0.1	76.5±0.0	80.0±0.0

Table 10: The cross-model generalization performance on CUB-200.

Distill. Model	CLIP					DINO-v2				
Eval. Model	CLIP	DINO-v2	EVA-02	MoCo-v3	Average	CLIP	DINO-v2	EVA-02	MoCo-v3	Average
LGM	16.4±0.0	15.2±0.0	13.6±0.0	2.7±0.1	12.0±0.0	22.1±0.2	81.1±0.1	50.3±0.1	26.0±0.1	44.9±0.1
SFM	**67.6±0.2**	**51.4±0.3**	**48.3±0.3**	**16.2±0.3**	**45.9±0.3**	**37.8±0.1**	**85.7±0.1**	**61.7±0.1**	**29.5±0.1**	**53.7±0.1**
Distill. Model	EVA-02					MoCo-v3				
Eval. Model	CLIP	DINO-v2	EVA-02	MoCo-v3	Average	CLIP	DINO-v2	EVA-02	MoCo-v3	Average
LGM	17.8±0.1	37.4±0.1	51.5±0.2	8.9±0.1	28.9±0.1	6.4±0.0	36.4±0.5	17.5±0.3	25.5±0.0	21.5±0.2
SFM	**43.4±0.0**	**66.8±0.3**	**72.5±0.2**	18.3±0.1	**50.3±0.2**	**14.8±0.1**	**51.7±0.0**	**33.1±0.0**	**41.4±0.1**	**35.3±0.1**
Full Dataset	77.5±0.7	90.2±0.2	84.0±0.3	43.7±0.8	73.8±0.5	77.5±0.7	90.2±0.2	84.0±0.3	43.7±0.8	73.8±0.5

E More Experiments

Cross-model Generalization Performance. We compare the generalization performance of LGM and our SFM on ImageNet-1k, CUB-200-2011, and Stanford Dogs. As shown in Table 9, Table 10, and Table 11, SFM still demonstrates remarkable competitiveness. For example, on ImageNet-1k, when using CLIP as the distillation model, the average generalization performance of SFM is 6% higher than that of LGM. This also confirms that our method can be seamlessly extended to larger-scale datasets. Distilling ImageNet-1k is memory-intensive. LGM employs 3 rounds of augmentation on synthetic images, which leads to Out-of-Memory (OOM) issues on two H100 GPUs with 90GB of memory each, while our SFM can function normally. However, simultaneously optimizing 1000 synthetic images at each distillation step still imposes high memory demands. We plan to adopt a bi-level optimization paradigm in the future, where the inner loop optimizes a subset of synthetic images and the outer loop iterates through all images, to further reduce memory consumption. This approach may potentially increase training time.

Table 11: The cross-model generalization performance on Stanford Dogs.

Distill. Model	CLIP					DINO-v2				
Eval. Model	CLIP	DINO-v2	EVA-02	MoCo-v3	Average	CLIP	DINO-v2	EVA-02	MoCo-v3	Average
LGM	15.0±0.0	17.7±0.2	16.3±0.0	6.2±0.1	13.8±0.1	27.5±0.1	79.8±0.2	59.7±0.0	56.7±0.6	55.9±0.2
SFM	**55.0±0.3**	**62.6±0.7**	**60.0±0.3**	**45.6±0.8**	**55.8±0.5**	**39.2±0.2**	**81.8±0.2**	**66.0±0.4**	**62.3±0.2**	**62.3±0.3**
Distill. Model	EVA-02					MoCo-v3				
Eval. Model	CLIP	DINO-v2	EVA-02	MoCo-v3	Average	CLIP	DINO-v2	EVA-02	MoCo-v3	Average
LGM	16.0±0.0	53.8±0.3	65.0±0.2	32.5±0.2	41.8±0.2	11.3±0.3	66.1±0.4	48.9±0.1	63.5±0.2	47.5±0.3
SFM	**38.6±0.0**	**72.7±0.4**	**72.9±0.2**	**53.2±0.1**	**59.4±0.2**	**14.6±0.3**	70.7±0.3	**57.4±0.0**	**67.2±0.3**	**52.5±0.2**
Full Dataset	76.9±0.1	88.6±0.1	82.6±0.1	72.3±0.5	80.1±0.2	76.9±0.1	88.6±0.1	82.6±0.1	72.3±0.5	80.1±0.2



Figure 8: Distilling the ArtBench [21] dataset.

Table 12: Performance of our SFM and CI under homogeneous architectures with different scales.

Size	ViT-S(dim=384)	ViT-B(dim=768)	ViT-L(dim=1024)
SFM	84.9 ±0.3	90.6 ±0.0	90.4 ±0.3
SFM+CI	86.0 ±0.1	95.1 ±0.1	92.4 ±0.1

Performance on Homogeneous Architecture. In previous experiments, we mainly investigated distillation performance on the original model and generalization performance across architectures with large discrepancies (heterogeneous architectures). Beyond that, we also explore generalization across different scales within homogeneous architectures. As shown in Table 12, images are distilled using DINO-v2/ViT-B, while the evaluation models are DINO-v2/ViT-S and DINO-v2/ViT-L. As these models are architecturally homogeneous and trained under similar paradigms, they still achieve strong performance with our SFM and CI.

Style Transfer. We distill the art dataset ArtBench [21] and then perform style transfer on top of that. As shown in Figure 8, we observe that distillation goes beyond simply "copying" samples from the dataset by comparing them to the nearest real neighbor in the model's embedding space. Instead, distillation involves a more complex transformation, where the model learns to extract and represent the essential features and patterns of the original data. This process leads to a substantial divergence from mere sample replication, as evidenced by the significant differences observed between the distilled data and the original samples in the embedding space. Additionally, we use the synthetic images as a style library for style transfer, transforming the style features of the source image (such as color, texture, brushstrokes, etc.) to generate an image with a new style. We also enhance its artistic expression and visual appeal, creating works that are both realistic and expressive.

Table 13: Quantitative comparison between LGM and our SFM in terms of distillation time and GPU memory usage. The distillation model is EVA-02.

Augmentation		1	3	5	7	10
LGM	GPU(GB)	16.6	48.5	84.5	107.3	165.2
SFM		16.3	48.1	84.2	106.8	164.5
LGM	Time(min)	21	35	48	63	81
SFM		18	28	38	48	62

Quantitative Results of Efficiency. We report the quantitative runtime and GPU consumption under different numbers of Augmentations Per Batch (APB), with EVA-02 as the distillation model. In terms of GPU consumption, SFM requires about 0.5 GB less memory than LGM. This mainly stems from the fact that LGM keeps the original-data dataloader resident in memory and retrieves samples from it at every distillation step, whereas SFM relies only on globally fixed statistics. In terms of runtime, LGM is substantially slower than SGM due to frequent data fetching and heavier gradient computations, and this gap further widens as APB increases.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the paper's scope and contributions: analyzing linear gradient matching under frozen self-supervised vision backbones, introducing statistical flow matching, and evaluating its efficiency and accuracy. The main claims are supported by the theoretical analysis and the experimental results reported in Sections 3 and 4.

   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: We discuss the limitations of SFM in Section 5.

   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: [Yes]

Justification: The paper states the assumptions and theoretical results used to analyze linear gradient matching in Section 3, including the exchangeability and lognormal-distribution results. Appendix A provides detailed derivations and proofs for the main equations and theorems used in the method.

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: The paper describes the datasets, backbone models, baselines, evaluation protocol, number of trials, augmentation settings, and key implementation details in Section 4.1 and Appendix B. These details provide a reasonable path to reproducing the main experimental comparisons, as well as the code is available in the supplementary materials.

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: The distilled data are presented in the Appendix. The code is included in the supplementary materials and will be released to the GitHub repository after the paper is accepted.

   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: Section 4.1 specifies the datasets, backbones, baselines, evaluation setting, number of trials, and augmentation settings. Appendix B further provides optimization details, including optimizer, learning rate, augmentation operators, number of distillation iterations, evaluation epochs, batch size, learning-rate schedule, and early stopping.

   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: The paper reports the average results of three trials, and all main tables list the mean and standard deviation.

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: The paper reports GPU memory usage and distillation time for the key comparisons. And the paper uses one H100 GPU for a single augmentation, and two H100 GPUs otherwise.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The research uses existing public image datasets and public pre-trained vision models, and does not involve human-subject experiments, deception, or collection of sensitive personal data. To the best of our knowledge, the work conforms to 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: [N/A]

    Justification: This work is a foundational study on improving the efficiency of dataset distillation for pre-trained self-supervised vision models, and it does not introduce new data collection, human-subject experiments, or high-risk released models; therefore, we do not identify a direct societal impact beyond general methodological implications for efficient machine learning.

    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 current submission does not release new datasets, model checkpoints, code, or other standalone assets. The paper introduces a new method and reports generated synthetic images for evaluation, but it does not present them as released assets.

    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: The paper cites the datasets, pre-trained models, and baseline methods used in the experiments. Additionally, the supplementary materials indicate the licenses used and provide links to the corresponding publicly available data and methods within the code.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The current submission does not release new datasets, model checkpoints, code, or other standalone assets. The paper introduces a new method and reports generated synthetic images for evaluation, but it does not present them as released assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing experiments or research with human subjects. All experiments are conducted using existing image datasets and pre-trained vision models.

    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 experiments or research with human subjects. Therefore, IRB approval or equivalent human-subject review is not applicable.

    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, experiments, and analysis do not involve LLMs as important, original, or non-standard components. The work is based on dataset distillation for pre-trained self-supervised vision models.

    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.