FUNDD: FUNCTA DISTRIBUTION SAMPLING FOR GENERATIVE DATASET DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

A distilled dataset is a small fixed set of synthetic images that leads to similar training performance compared to its original dataset. However, a classifier trained on it sees the same images for hundreds of epochs; its storage grows with every image and with resolution, and a new budget needs a new distillation run. We propose Functa Distribution Distillation (FunDD), which distills a dataset into a distribution instead of a set. Each image is encoded as a latent modulation of a Functa implicit neural representation meta-learned on the dataset, and each class is fitted with a Gaussian over these modulations in closed form. The distilled dataset consists of the class Gaussians and the shared Functa decoder. During classifier training, new modulations are drawn at every step and used once, either decoded into images or fed directly to a classifier in modulation space. On CIFAR-10, classifiers trained this way exceed classifiers trained on fixed distilled sets from GLaD, LD3M, and H-PD by 10.2, 10.4, and 9.9 percentage points on average, respectively. On ImageNet subsets, it exceeds the strongest optimization-based baseline in all 15 settings by 9.7 points on average, and with ten draws per class it is ahead of the training-free diffusion method DMGD on three of five subsets, without a generator pretrained on ImageNet-1K. Because every sample is used once, the training budget is the number of samples drawn, and at a fixed budget accuracy does not depend on how the samples are grouped into updates. The stored representation is 32 MiB for ten classes and does not change with the number of samples drawn or with image resolution, so the amount of training data a distilled dataset can provide is no longer bounded by what it stores.

1 INTRODUCTION

Training data is either scarce or abundant, and both cases are challenging. When data is scarce, a model has too little to learn from (Kaplan et al., 2020). When data is abundant, it has to be stored, moved to every machine that trains on it, and read in full for every hyperparameter setting, architecture, and later task (Guo et al., 2022; Zhao et al., 2021). Two lines of work reduce these costs with synthetic data. Dataset distillation compresses an abundant training set into a few synthetic images that train a model almost as well as the original (Moser et al., 2026; Cazenavette et al., 2023; Yin et al., 2023). Generative modeling produces an unlimited stream of synthetic images from a learned model, and classifiers trained on such streams now approach classifiers trained on real data (Adamkiewicz et al., 2026; Qin et al., 2024). Both lines aim at the same thing, cheap data that trains well, but they deliver opposite objects. One delivers a small fixed set. The other delivers a large sampler. This paper delivers a third object, a distribution that is as small as the set and as unlimited as the sampler.

Given a real dataset $\mathcal{T}$ with N samples, dataset distillation searches for a synthetic set $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{M}$ with $M \ll N$ that minimizes a distillation loss $\mathcal{L}(\mathcal{S}, \mathcal{T})$, for example by matching gradients, feature distributions, or training trajectories (Zhao et al., 2021; Zhao & Bilen, 2023; Cazenavette et al., 2022). The output is a set of $M = C \cdot \mathrm{IPC}$ images for C classes and IPC images per class, and three costs follow from that form. Storage grows with the set. M images of resolution $H \times W$ occupy $3MHWb$ bytes at b bytes per value, linear in the budget and in the number of pixels. Training reuses the set. A classifier f_θ trained on $\mathcal{S}$ minimizes the empirical risk $\hat{R}_{\mathcal{S}}(\theta) = \frac{1}{M}\sum_{i=1}^{M} \ell(f_\theta(x_i), y_i)$ over the same M inputs for hundreds of epochs. Any model that

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

can memorize M inputs can drive $\hat{R}_{\mathcal{S}}$ to zero, and no later epoch adds an input the model has not already seen. The budget is fixed at distillation time, since a set optimized for one value of IPC is not reused for another. Generative priors lower the cost of the search by optimizing latent codes instead of pixels (Cazenavette et al., 2023; Duan et al., 2023; Moser et al., 2026), and distributional distillation replaces the images with per-class latent statistics (Qin et al., 2024), but both obtain their output by optimizing a distillation loss and inherit the cost of the search.

Generative modeling takes the opposite route: a generator G_ψ turns every noise vector $z \sim p(z)$ into a new image $x = G_\psi(z)$, so the number of distinct training samples is unbounded, and the costs move from the samples to the model. Storage is the parameter count $|\psi|$, which reaches hundreds of millions for the diffusion models that produce the best samples (Rombach et al., 2022). Sampling costs T_G sequential evaluations of that network per image, in the tens even for fast samplers (Lu et al., 2022), so a new batch at every classifier step is expensive. The objective is the data distribution, $\min_\psi \mathrm{D}\big(p_\psi \,\|\, p_{\mathrm{data}}\big)$ for some divergence D, not the downstream training task. Dataset distillation shows that the two differ, because the synthetic images that train models best are often abstract rather than realistic (Cazenavette et al., 2023; Moser et al., 2026).

We propose a different paradigm for dataset distillation: the output is a distribution instead of a set. Every training image is encoded as a modulation vector of a Functa network (Dupont et al., 2022) and each class is fitted with a Gaussian whose mean and covariance are simply those of its real modulations, so the distillation step is a closed-form estimate rather than an optimization. The distilled dataset is the shared network with parameters ω plus one Gaussian per class, and a synthetic image is a draw from the class Gaussian, spread by a temperature β and decoded by the shared network. The Gaussians take $C\,(d + d(d+1)/2)\,b$ bytes and the decoder $|\omega|\,b$ bytes, and neither term depends on the number of samples drawn or on H and W. With $d = 512$ and ten classes it is 32 MiB for any number of draws. Sampling is cheap: one Gaussian draw and one forward pass of a 7-million-parameter network per image, at 434 images per second on one GPU. Every step draws a new batch and uses it once, so stochastic gradient descent optimizes the expected risk under the distilled distribution, $R_q(\theta) = \mathbb{E}_y\, \mathbb{E}_{\phi \sim q_\beta(\phi|y)}\, \ell(f_\theta(D_\omega(\phi)), y)$ with q_β the tempered class Gaussian and D_ω the shared decoder, in a single pass, the setting in which stochastic gradient descent optimizes the expected risk directly rather than a proxy on a finite set (Bottou, 2010). After T steps of B samples the model has seen TB distinct inputs, and the stored representation has not grown by a byte. The training budget is chosen by whoever trains the classifier, not by whoever distilled the data. Because the class information lives in the modulation, the same distribution also trains a classifier on modulations directly, without decoding an image at all.

Experiments show that classifiers trained on FunDD draws exceed the best of GLaD, LD3M, and H-PD in every setting, by 8.3 points on average on CIFAR-10 and 9.7 points on the ImageNet subsets, under the same optimizer, schedule, and number of steps.

They also show that at a fixed sample budget, with the learning rate scaled to the batch size, accuracy depends on how many draws the classifier sees and not on how they are batched, and that an MLP on the modulations alone, without decoding, reaches 53.2% on CIFAR-10 and 60.4% on five ImageNet subsets on average, so the class information is in the distribution and not only in the images decoded from it.

2 Background

2.1 Dataset Distillation

Dataset distillation condenses a real dataset $\mathcal{T}$ with N samples into a synthetic dataset $\mathcal{S}$ with $M \ll N$ samples such that a model trained on $\mathcal{S}$ performs close to a model trained on $\mathcal{T}$ (Moser et al., 2026; Liu & Du, 2025). For a classification task with C classes the size of $\mathcal{S}$ is given in images per class, $M = C \cdot \mathrm{IPC}$, and the distilled set is the solution of

$$\mathcal{S}^* = \arg\min_{\mathcal{S}} \mathcal{L}(\mathcal{S}, \mathcal{T}), \tag{1}$$

where $\mathcal{L}$ is a distillation loss that rewards $\mathcal{S}$ for preserving the training behavior of $\mathcal{T}$. Methods differ mainly in the choice of $\mathcal{L}$ (Liu & Du, 2025). Dataset Condensation (DC) matches the gradients that real and synthetic batches induce in a network (Zhao et al., 2021), Distribution Matching (DM) matches their mean features under random extractors (Zhao & Bilen, 2023), and Matching

Training Trajectories (MTT) matches the parameter trajectory of a network trained on $\mathcal{S}$ to trajectories recorded on $\mathcal{T}$ (Cazenavette et al., 2022). Several works add plug-and-play components to these objectives to improve quality and scalability (Du et al., 2023; Cui et al., 2025; Shang et al., 2023; Zhong et al., 2025a; Lu et al., 2023; Wang et al., 2026). FunDD does not solve Eq. 1 at all. It replaces $\mathcal{S}$ by a distribution whose parameters are estimated in closed form (Section 3).

Generative-prior methods optimize the distilled set in the latent space of a pretrained generator instead of in pixel space. GLaD (Cazenavette et al., 2023) optimizes intermediate feature maps of StyleGAN-XL under the DC, DM, or MTT loss. LD3M (Moser et al., 2026) replaces the GAN with a latent diffusion model and makes both the initial latent and the class conditioning learnable. H-PD (Zhong et al., 2025b) optimizes progressively through the layers of StyleGAN-XL and selects the latents of each layer with a class-relevant feature distance. DMGD (Wang et al., 2026) optimizes nothing: it guides the sampling of a pretrained diffusion transformer toward the class semantics and, with an optimal-transport loss, toward the distribution of the real data. All four output a finite set of IPC images per class that is stored and reused, and on the ImageNet subsets all four rely on a generator pretrained on ImageNet-1K. FunDD stores a distribution that is sampled at every step. D3 (Qin et al., 2024) also stores per-class statistics with a decoder, but obtains them by minimizing a distillation objective; FunDD estimates them in closed form.

2.2 Functa

An implicit neural representation (INR) models a signal as a continuous function computed by a neural network. For an image, the network maps a pixel coordinate to a color, and the image is the network evaluated on the pixel grid (Sitzmann et al., 2020; Dupont et al., 2022). Functa (Dupont et al., 2022) fits a whole dataset with one such network, a SIREN (Sitzmann et al., 2020), whose parameters capture what the images have in common, while each image is specified by a modulation. Each data point is associated with a low-dimensional latent modulation ϕ, which is mapped linearly to shift modulations $s = W'\phi + b'$, with W' and b' part of the shared network. With $s = [s^{(1)}, \dots, s^{(n)}]$, the i-th layer maps its input h to

$$h \mapsto \sin\left(\omega_0 \left(W^{(i)}h + b^{(i)} + s^{(i)}\right)\right), \tag{2}$$

where $\{W^{(i)}, b^{(i)}\}_{i=1:n}$ are the weights and biases of the n-layer SIREN. The shared network is meta-learned so that the modulation of a new image can be fitted in a few gradient steps: the inner loop updates only the modulation and the outer loop updates the shared parameters (Dupont et al., 2022). After meta-learning, the modulations of the training images can be treated as data in their own right, and Functa has been used this way for generative modeling, imputation, novel view synthesis, and classification (Dupont et al., 2022; Bauer et al., 2023). The structure of the modulation space itself has received less attention (Wolleb et al., 2026; Dupont et al., 2022), and it is the space in which FunDD distills.

3 Functa Distribution Distillation (FunDD)

FunDD takes a labeled dataset $\mathcal{T} = \{(x_i, y_i)\}_{i=1}^{N}$ with C classes and returns a distilled dataset that is a distribution rather than a set: one Gaussian per class over the modulation space of a Functa network, together with the network itself. Figure 1 shows the pipeline. A Functa network is meta-learned on $\mathcal{T}$ and every image is encoded as a modulation vector (Section 3.1). Each class is fitted with a Gaussian whose parameters are computed in closed form (Section 3.2), and a temperature sets how far draws spread from the class mean (Section 3.3). During classifier training, modulations are drawn at every step, used once, and either decoded into images or passed directly to a classifier in modulation space (Section 3.4). The stored object has the same size for any number of draws and any image resolution (Section 3.5).

3.1 Modulations as the Distillation Space

Section 2.2 describes Functa (Dupont et al., 2022). FunDD uses three of its properties. First, one network D_ω with parameters ω is shared by all images, and an image x is represented by a modulation vector $\phi \in \mathbb{R}^d$ such that $x(p) \approx D_\omega(p, \phi)$ at every pixel coordinate p. Second, the modulation

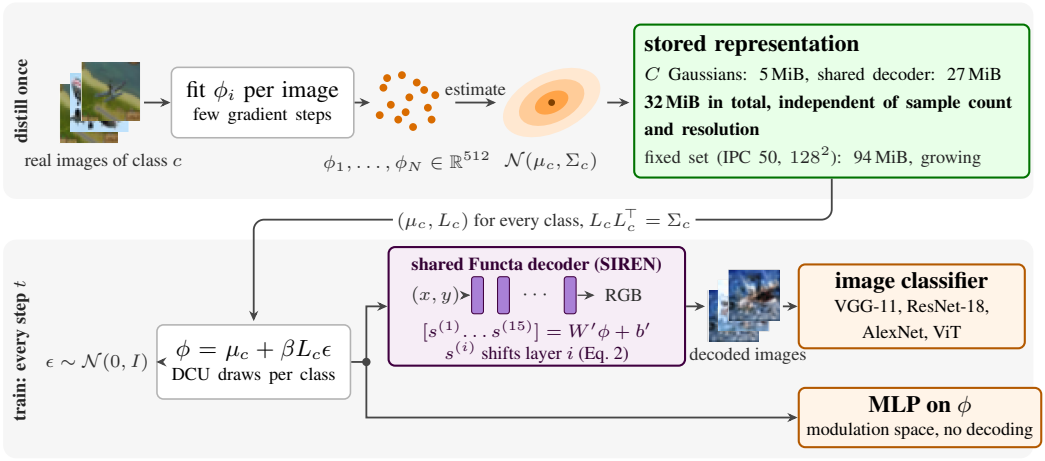

Figure 1: **FunDD.** *Top:* every real image of class c is encoded once into a Functa latent modulation $\phi_i \in \mathbb{R}^{512}$, and one Gaussian $\mathcal{N}(\mu_c, \Sigma_c)$ is fitted per class (Eq. 3). Only the C Gaussians and the shared Functa decoder are stored. *Bottom:* the classifier never sees a fixed synthetic set. At every optimizer step, DCU (draws per class per update) new modulations per class are drawn independently at temperature β (Eq. 6) and either decoded through the shared SIREN into images or classified directly in modulation space. The key difference is that the stored representation is a distribution, so its size is independent of how many samples the classifier processes and of image resolution.

of an image is obtained by a few gradient steps on the reconstruction loss with ω held fixed, because ω was meta-learned so that few steps suffice. Third, the image size enters only through the coordinate grid at which D_ω is evaluated, so $|\omega|$ does not depend on H or W. After meta-learning on $\mathcal{T}$, every training image becomes a point $\phi_i \in \mathbb{R}^d$. With $d = 512$ and 128×128 images, this is a $96\times$ reduction from the $3HW = 49{,}152$ pixel values. FunDD distills in this space.

3.2 Class-Conditional Gaussians in Closed Form

Let $I_c = \{i : y_i = c\}$ be the indices of class c and $N_c = |I_c|$. The class is fitted with the Gaussian

$$q(\phi \mid y = c) = \mathcal{N}(\mu_c, \Sigma_c), \qquad \mu_c = \frac{1}{N_c} \sum_{i \in I_c} \phi_i, \qquad \Sigma_c = \frac{1}{N_c - 1} \sum_{i \in I_c} (\phi_i - \mu_c)(\phi_i - \mu_c)^\top. \quad (3)$$

Both parameters are computed in one pass over the modulations. No distillation loss is minimized, no classifier is involved, and no budget is chosen. The distilled dataset is the triple $(\omega, \{\mu_c\}_{c=1}^C, \{\Sigma_c\}_{c=1}^C)$.

Two facts justify the Gaussian. First, among all distributions on $\mathbb{R}^d$ with mean μ_c and covariance Σ_c, the Gaussian has the largest entropy (Cover & Thomas, 2006); consequently, Eq. 3 keeps the first two moments of the real modulations of class c and adds no structure beyond them. Second, it is the natural end point of distribution matching: DM (Zhao & Bilen, 2023) optimizes a synthetic set until its mean feature under a random extractor equals that of the real data. FunDD makes the first two moments of the real data the distilled dataset itself, so the match is exact by construction and costs one pass rather than an optimization. Every class has $N_c = 5000$ modulations on CIFAR-10 and up to $N_c = 1300$ on the ImageNet subsets, both larger than $d = 512$, so Σ_c has full rank. We add εI with $\varepsilon = 10^{-6}$ before taking the Cholesky factor L_c with $L_c L_c^\top = \Sigma_c + \varepsilon I$, for numerical stability.

3.3 Temperature

A temperature $\beta > 0$ controls the spread of the draws. Following distribution tempering (Neal, 1996), the tempered distribution is

$$q_\beta(\phi \mid y = c) \propto q(\phi \mid y = c)^{1/\beta^2}, \quad (4)$$

which for the Gaussian of Eq. 3 is again a Gaussian with the covariance scaled,

$$q_\beta(\phi \mid y = c) = \mathcal{N}(\mu_c, \beta^2 \Sigma_c), \qquad \phi = \mu_c + \beta L_c \epsilon, \quad \epsilon \sim \mathcal{N}(0, I). \tag{5}$$

For $\beta < 1$ the draws concentrate around the class mean, and for $\beta > 1$ they spread beyond the real modulations. The temperature trades diversity against staying in the region on which D_ω was meta-learned: small β gives cleaner decoded images from a narrower distribution, large β gives more varied draws that D_ω has not seen. We select β per dataset by validation accuracy of the downstream classifier (Appendix A.4) and use $\beta = 0.85$ on CIFAR-10 and $\beta = 0.7$ on the ImageNet subsets.

3.4 Drawing Samples During Training

A classifier f_θ is trained on FunDD without ever seeing a stored synthetic set. At step t, for every class c, we draw DCU modulations, where DCU denotes the number of *draws per class per update*. Each is sampled from Eq. 5 with independent noise,

$$\phi_{t,c,j} = \mu_c + \beta L_c \epsilon_{t,c,j}, \qquad \epsilon_{t,c,j} \sim \mathcal{N}(0, I), \qquad j = 1, \ldots, \mathrm{DCU}, \tag{6}$$

giving a batch of size $B = C \cdot \mathrm{DCU}$. The parameter update uses the batch gradient

$$g_t = \frac{1}{B} \sum_{c=1}^{C} \sum_{j=1}^{\mathrm{DCU}} \nabla_\theta \, \ell\big(f_\theta(D_\omega(\phi_{t,c,j})), c\big), \tag{7}$$

after which the batch is discarded. Because the draws are independent of θ_t and of all earlier draws, g_t is an unbiased estimate of the gradient of the expected risk under the distilled distribution,

$$R_q(\theta) = \mathbb{E}_y \, \mathbb{E}_{\phi \sim q_\beta(\phi \mid y)} \Big[\ell\big(f_\theta(D_\omega(\phi)), y\big) \Big], \tag{8}$$

with y uniform over the C classes,

$$\mathbb{E}\big[g_t \mid \theta_t\big] = \nabla_\theta R_q(\theta_t). \tag{9}$$

Stochastic gradient descent on independent draws therefore minimizes R_q directly. This is the single-pass regime, in which every gradient is computed on inputs the model has never seen and no input is visited twice (Bottou, 2010). After T steps the classifier has processed $S = TB$ distinct synthetic samples. We call S the sample budget.

DCU replaces IPC as the unit of a distilled dataset, but the two measure different things. IPC counts images that are stored, and changing it means distilling again. DCU counts draws per update, and changing it means changing a batch size at training time. The stored object of Eq. 3 is the same for every DCU and every T.

We train in two settings. In the image-space setting, every draw is decoded, $x = D_\omega(\phi)$, and the classifier is a standard image network (AlexNet, VGG-11, ResNet-18, or ViT). Alternatively, in the modulation-space setting, the classifier is an MLP on ϕ itself, D_ω drops out of Eq. 7, and no image is ever decoded. Draws can also be generated ahead of time and buffered to amortize decoding, at the cost of temporary memory proportional to the buffer. As a control, a fixed variant draws K modulations per class once, decodes them, and reuses the resulting set like a conventional distilled dataset (Section 4.1).

3.5 Storage

The distilled dataset has two parts, and neither grows with the number of draws or with the image resolution. The class Gaussians take

$$M_{\mathrm{dist}} = C \left(d + \frac{d(d+1)}{2} \right) b \tag{10}$$

bytes at b bytes per value, using the symmetry of Σ_c, which is $\mathcal{O}(Cd^2)$ and independent of K, H, and W. The shared network takes $M_{\mathrm{dec}} = |\omega|\, b$ bytes, and $|\omega|$ is set by d, the width, and the depth of the network, not by the image size, because D_ω maps coordinates to colors. In the modulation-space setting the network is not needed at training time and the distilled dataset is M_{dist} alone. A fixed set of K images per class takes

$$M_{\mathrm{img}} = 3CKHWb, \tag{11}$$

which is $\mathcal{O}(CKHW)$ and grows linearly with K and with the number of pixels. For our configuration the whole representation is 32.09 MiB. Appendix A.3 gives the breakdown and the budgets at which a fixed set overtakes it.

4 EXPERIMENTS

We ask three questions. Does a classifier trained on draws from FunDD beat one trained on a fixed distilled set of the same per-step size (Section 4.2)? Where does the gain come from, the draws themselves or the fact that they are never reused (Sections 4.2 and 4.3)? And does the modulation space carry the class information on its own (Section 4.4)? Section 4.5 looks at the draws and the temperature, and Section 4.6 reports the cost.

4.1 SETUP

We use CIFAR-10 at 32×32 and the ten-class ImageNet subsets of Cazenavette et al. (2023) at 128×128 (class lists in Appendix A.8). We meta-learn one Functa network on CIFAR-10 and a separate Functa network for each ImageNet subset on the training split with $d = 512$, width 512, depth 15 and 3 inner-loop steps per image, and for the latent-dimension ablation we also train $d = 64$ and $d = 128$ on CIFAR-10. The temperature is $\beta = 0.85$ on CIFAR-10 and $\beta = 0.7$ on ImageNet (Section 3.3 and Appendix A.4). We compare against GLaD (Cazenavette et al., 2023), LD3M (Moser et al., 2026), and H-PD (Zhong et al., 2025b), which combine a generative prior with the DC, DM, and MTT losses, at IPC 1 and 10, and against DMGD (Wang et al., 2026) at IPC 10. For each method we take the best of its published variants under the GLaD evaluation protocol.

We follow the evaluation of GLaD. A classifier is trained from scratch on the distilled data and tested on the real test set, and we report mean and standard deviation of the best test accuracy over five runs. The classifiers are AlexNet, VGG-11, ResNet-18, and ViT, none of which is used during distillation. Training uses SGD with momentum and weight decay for T steps, half of them linear warm-up and half cosine decay. In the released GLaD code the batch is the whole distilled set, so one epoch is one step. FunDD uses the same optimizer, schedule, and T, and differs only in the batch: at every step it draws $B = C \cdot \mathrm{DCU}$ new modulations (Eq. 6) instead of reusing the same $C \cdot \mathrm{IPC}$ images, so DCU matches IPC in per-step size. To separate the value of the draws from the value of not reusing them, we also evaluate a fixed control, FunDD (fixed), in which K modulations per class are drawn once, decoded, and reused for all T steps exactly like a distilled set. FunDD (fixed) and other distillation methods differ in where the stored images come from, and FunDD (fixed) and FunDD (dynamic) differ only in whether the images are reused.

4.2 FUNDD AGAINST FIXED DISTILLED SETS

Tables 1 and 2 report the main comparison. On the ImageNet subsets, FunDD (dynamic) exceeds the best of GLaD, LD3M, and H-PD by 9.7 points on average, with the smallest margin of 3.5 points on ImageNet-Cats and the largest of 18.0 on ImageNet-Fruits at DCU 1. At IPC 10, DMGD is the only baseline above FunDD on two ImageNet subsets. On CIFAR-10, FunDD exceeds the best baseline variants by 8.3 points on average over the four architectures. The largest gain, 11.1 points, is on AlexNet over GLaD, and the smallest, 6.4 points, is on VGG-11 over H-PD.

Table 3 isolates the cause on CIFAR-10 with VGG-11 at a fixed number of steps. Ten decoded draws per class, stored and reused, train to 31.1%, within 2 points of ten real images per class stored and reused, 29.1%. The same draws, redrawn at every step, train to 52.9%. At $K = 100$ the pattern repeats: 49.1% for stored draws, 44.4% for stored real images, 58.8% for redrawn ones. A stored draw is therefore worth at least as much as a stored real image. The gain comes from never reusing a sample. This is also what the FunDD (fixed) rows of Tables 1 and 2 show: when the decoded draws are stored and reused, FunDD falls below every baseline on every dataset and architecture. The distribution is not a better set. It is a better object, because the same images that give set-level accuracy when stored give FunDD-level accuracy when redrawn.

The comparison above matches the per-step batch and the number of steps, which is the protocol of the field. It does not match storage: at 128×128 an image set at IPC 10 takes 18.75 MiB and FunDD takes 32.09 MiB (Table 6 in Appendix A.3), and it does not match the number of distinct images seen, which is $C \cdot \mathrm{IPC}$ for a set and $T \cdot C \cdot \mathrm{DCU}$ for FunDD. Both differences are the point of the paradigm, not artifacts of it. A set cannot show the classifier more distinct images without storing them, and at 128×128 a set that stores as much as FunDD holds 17 images per class, while FunDD at DCU 1 already exceeds GLaD at IPC 10 on four out of five subsets. In modulation space (Section 4.4) the stored object is 5.03 MiB, less than a GLaD set at IPC 3.

Table 1: Test accuracy (%) on ImageNet subsets, mean ± std over five runs, averaged over AlexNet, VGG-11, ResNet-18, and ViT trained from scratch. Results for GLaD, LD3M, and H-PD baselines are taken from Cazenavette et al. (2023), Moser et al. (2026), and Zhong et al. (2025b), respectively; for each, we report the best of its MTT, DC, and DM variants. Results for are DMGD taken from Wang et al. (2026). All rows use the same optimizer, schedule, and steps. Baselines reuse a stored set of IPC images per class. FunDD (fixed) reuses IPC decoded draws per class. FunDD (dynamic) draws $\mathrm{DCU} = \mathrm{IPC}$ new modulations per class at every step. Best in bold, second best underlined.

IPC = 1 (baselines) and DCU = 1 (FunDD)					
Method	**ImNet-A**	**ImNet-B**	**ImNet-C**	**ImNet-D**	**ImNet-E**
GLaD	41.8 ± 1.7	42.1 ± 1.2	35.8 ± 1.4	30.4 ± 1.5	29.3 ± 1.3
LD3M	42.3 ± 1.3	42.0 ± 1.1	37.1 ± 1.8	31.5 ± 1.2	31.4 ± 1.1
H-PD	42.4 ± 1.2	42.6 ± 1.1	36.1 ± 1.1	31.2 ± 1.0	29.6 ± 0.9
FunDD (fixed)	30.0 ± 1.6	28.7 ± 2.4	24.6 ± 1.5	21.1± 1.4	20.0 ± 2.4
FunDD (dynamic)	**57.2 ± 1.3**	**53.2 ± 1.3**	**51.5 ± 1.8**	**42.3 ± 1.1**	**35.6 ± 1.6**
	ImNette	**ImNet-Woof**	**ImNet-Birds**	**ImNet-Fruits**	**ImNet-Cats**
GLaD	31.0 ± 1.6	17.8 ± 1.1	29.1 ± 1.0	22.3 ± 1.6	21.2 ± 1.4
LD3M	32.9 ± 1.2	19.9 ± 1.2	30.4 ± 1.5	24.5 ± 1.9	22.1 ± 1.0
H-PD	–	–	–	–	–
FunDD (fixed)	22.1 ± 1.1	14.7 ± 0.8	19.7 ± 1.2	17.4 ± 1.6	14.2 ± 1.4
FunDD (dynamic)	**43.8 ± 1.2**	**25.7 ± 0.6**	**35.0 ± 1.6**	**42.5 ± 1.5**	**25.6 ± 1.7**
IPC = 10 (baselines) and DCU = 10 (FunDD)					
Method	**ImNet-A**	**ImNet-B**	**ImNet-C**	**ImNet-D**	**ImNet-E**
GLaD	53.1 ± 1.4	51.3 ± 0.6	49.7 ± 0.4	38.9 ± 1.0	38.6 ± 0.7
LD3M	57.0 ± 1.3	52.3 ± 1.1	49.9 ± 1.3	39.5 ± 1.0	39.4 ± 1.8
DMGD	65.4 ± 0.3	**70.2 ± 0.9**	62.2 ± 1.0	46.8 ± 1.5	**51.3 ± 0.3**
H-PD	55.1 ± 0.5	54.2 ± 0.5	50.8 ± 0.4	39.8 ± 0.7	40.1 ± 0.7
FunDD (fixed)	50.2 ± 1.7	46.7 ± 1.7	44.8 ± 1.8	34.8 ± 1.8	34.2 ± 1.6
FunDD (dynamic)	**67.7 ± 1.3**	62.1 ± 1.2	**63.0 ± 0.8**	**49.1 ± 0.7**	47.9 ± 0.6

Table 2: Test accuracy (%) on CIFAR-10 per architecture at IPC = 1 for the baselines and DCU = 1 for FunDD, mean ± std over five runs, same protocol as Table 1. Best in bold, second best underlined. Results for GLaD, LD3M, and H-PD baselines are taken from Cazenavette et al. (2023), Moser et al. (2026), and Zhong et al. (2025b), respectively; for each, we report the best of its MTT, DC, and DM variants.

Method	**AlexNet**	**ResNet-18**	**VGG-11**	**ViT**	**Average**
GLaD	30.1 ± 0.5	30.2 ± 0.6	31.3 ± 0.7	23.4 ± 0.2	28.8 ± 0.5
LD3M	27.2 ± 0.4	26.6 ± 0.9	31.5 ± 0.3	29.0 ± 0.2	28.6 ± 0.5
H-PD	28.5 ± 0.3	31.4 ± 0.4	32.2 ± 0.2	24.4 ± 0.5	29.1 ± 0.4
FunDD (fixed)	15.1 ± 1.2	16.9 ± 0.8	15.9 ± 1.5	17.8 ± 1.7	16.4 ± 1.3
FunDD (dynamic)	**41.2 ± 0.6**	**39.1 ± 0.7**	**38.6 ± 1.2**	**37.1 ± 0.5**	**39.0 ± 0.8**

4.3 Sample Budget versus Batch Size

Section 3.4 argued that training on independent draws minimizes the expected risk, so the quantity a classifier should care about is the number of draws it has seen, $S = TB$, and not how they are grouped into batches. In Table 4 we fix $T = 1000$ steps and raise DCU. Going from DCU 10 to 100 raises accuracy on every architecture, by 5.8 points on ResNet-18 up to 12.9 points on ViT, and going from 100 to 200 adds at most 4.6 more. Ten times the draws help a great deal, twenty times helps little more. On ImageNet-A, increasing DCU from 10 to 100 improves accuracy by up to 4.4 points for three architectures. Then we fix $S = 350{,}000$ draws and vary DCU from 10 to 200, so that

Table 3: The effect of reuse, CIFAR-10, VGG-11, test accuracy (%) at a fixed number of steps. Stored real: K real images per class reused throughout training. Stored draws: K decoded FunDD draws per class reused throughout training, the FunDD (fixed) control. Redrawn: FunDD (dynamic) with $\mathrm{DCU} = K$.

K	**Stored real**	**Stored draws**	**Redrawn every step**
10	29.1 ± 2.5	31.1 ± 0.9	$\mathbf{52.9 \pm 0.2}$
100	44.4 ± 0.6	49.1 ± 0.9	$\mathbf{58.8 \pm 0.3}$

Table 4: Best Test accuracy (%) on decoded draws as a function of DCU and the learning rate is scaled with DCU. $T = 1000$ steps fixed, so the sample budget $S = T \cdot C \cdot \mathrm{DCU}$ grows with DCU. Mean $\pm$ std over five runs.

	DCU	**AlexNet**	**ResNet-18**	**VGG-11**	**ViT**
Fixed steps, $T = 1000$					
CIFAR-10	10	47.1 ± 0.4	54.2 ± 0.1	52.9 ± 0.2	33.3 ± 0.5
	100	57.3 ± 0.1	60.0 ± 0.3	58.8 ± 0.3	46.2 ± 0.3
	200	58.7 ± 0.3	58.9 ± 0.4	59.8 ± 0.1	50.8 ± 0.1
ImageNet-A	10	66.9 ± 1.9	72.5 ± 1.4	66.6 ± 1.7	65.1 ± 0.3
	100	68.2 ± 0.5	71.6 ± 0.3	70.1 ± 0.4	69.5 ± 0.5

T shrinks from 3500 to 175 steps. Figure 4 in Appendix A.7 plots test accuracy against the number of draws seen. For each architecture, the four curves overlap within their standard-deviation bands along the whole run, despite a twentyfold change in batch size. Together these two experiments show that accuracy is a function of S. This is the practical meaning of Eq. 9: with independent draws every batch is an unbiased sample of the same objective, so with the learning rate scaled to the batch size, the batch size only sets the variance of a step. The curves also show where the returns stop. The draws add no information beyond μ_c and Σ_c (Cover & Thomas, 2006), and the curves flatten once the classifier has seen enough of them. A user of FunDD picks S from the curve and DCU from the hardware, and neither choice touches the stored 32 MiB.

4.4 Classification Without Decoding

If the class information lives in ϕ, a classifier should not need the images. Table 5 trains an MLP directly on drawn modulations, with D_ω removed from the loop. We employ a simple yet effective three-layer MLP with layer normalization and dropout. On CIFAR-10 at d = 512 and DCU 200, the MLP reaches 53.2%, above ViT (50.8%) and 5.5 to 6.6 points below the three convolutional classifiers trained on decoded draws with the same T and DCU (Table 4), and it does so with a distilled dataset of 5.03 MiB and no decoder at training time. On the ImageNet subsets the MLP reaches 64 to 70% on A to C, above most image classifiers at DCU 10 and 0.2 to 3.7 points below them at DCU 100 (Tables 4 and 7).

Table 5 also shows why $d = 512$. Reconstruction PSNR and modulation-space accuracy rise together with d, from 21.8 dB and 47.6% at $d = 64$ to 31.2 dB and 53.2% at $d = 512$ on CIFAR-10. A modulation that reconstructs the image better also separates the classes better, which is the property FunDD relies on. The ImageNet network reconstructs its training images at 22.5 dB on average, far below the 31.4 dB of CIFAR-10, which is one reason the ImageNet draws look coarser (Section 4.5).

4.5 Decoded Draws and Temperature

Figure 3 in Appendix A.6 shows real images next to decoded draws. The draws are not photographs. They keep the colors, the layout, and the coarse shape of the class and lose the fine detail, more so on ImageNet, where one network has to cover 128×128 images and reconstruct the training set at 22.5 dB. This is the expected look of a sample from a Gaussian around a class mean, and it is not a defect for the task: the distilled sets that train models best are often abstract (Moser et al., 2026). The temperature sets how far the draws leave the mean. Appendix A.4 describes the selection, and its

Table 5: Modulation space. Left: CIFAR-10, Functa reconstruction PSNR (dB) of the training and test images and test accuracy (%) of an MLP trained on drawn modulations without decoding, for three latent dimensions, $T = 1000$ steps. Right: the same MLP on the ImageNet subsets at $d = 512$. Mean $\pm$ std over five runs.

CIFAR-10					**ImageNet subsets,** $d = 512$ PSNR: 22.5 / 21.5		
d	PSNR	DCU 10	DCU 100	DCU 200	Subset	DCU 10	DCU 100
64	23.5 / 21.8	47.0 ± 0.4	47.7 ± 0.3	47.6 ± 0.4	A	67.3 ± 1.8	68.0 ± 1.3
128	25.4 / 24.3	49.3 ± 0.4	49.0 ± 0.4	48.8 ± 0.2	B	66.8 ± 1.3	69.6 ± 1.9
512	31.4 / 31.2	52.6 ± 0.3	53.1 ± 0.5	$\mathbf{53.2 \pm 0.3}$	C	63.9 ± 1.7	64.5 ± 1.5
					D	47.2 ± 1.2	48.0 ± 1.0
					E	52.1 ± 1.9	51.7 ± 2.1

Figure 2 shows draws of one ImageNet class across a grid of β. At small β a draw is a smooth field close to the class mean with little structure, raising β adds texture and contrast, and above $\beta = 1$ the draws gain high-frequency artifacts and colors that do not belong to the class, because they leave the region on which D_ω was meta-learned. We select β per dataset by a grid search on the validation accuracy of the downstream classifier.

4.6 Cost

On one NVIDIA A100 with 40 GB, drawing a batch of modulations takes about 129 ms independent of batch size and resolution, and decoding takes 2.3 ms per 128×128 image, or 434 images per second. A FunDD step therefore costs the classifier step plus a decoding cost linear in B, and the decoding can run ahead of the classifier and be buffered, at the price of temporary memory for the buffer. Since accuracy does not depend on DCU at a fixed budget (Section 4.3), DCU can be chosen for the hardware alone: larger batches make better use of the accelerator until it saturates (Goyal et al., 2017; Shallue et al., 2019). Distillation itself is one meta-learning run of the Functa network followed by one pass over the modulations to compute μ_c and Σ_c. On the same A100 and dataset, meta-training takes approximately 7.5 hours, followed by about 7 minutes to fit the modulations.

5 Conclusion and Limitations

FunDD proposes a new paradigm for generative dataset distillation by providing a distribution over the Functa modulation space, fitted in closed form from the real data and stored as one Gaussian per class beside the shared decoder. We have shown that classifiers trained on draws from this distribution exceed classifiers trained on the distilled sets of GLaD, LD3M, and H-PD in every setting, by 8.3 points over the strongest of them on CIFAR-10 and 9.7 points on ImageNet subsets, under the same optimizer, schedule, and number of steps, and that the gain comes from never reusing a sample rather than from the draws themselves, since stored draws are worth about as much as stored real images. At a fixed sample budget and with the learning rate scaled to the batch size, accuracy does not depend on how the draws are batched, so the budget and the batch size are two separate choices, both made at training time and neither touching the 32 MiB that is stored. A classifier on the modulations alone, without decoding, reaches 53.2% on CIFAR-10, so the modulation space carries the class information by itself. FunDD also has limitations. We evaluate it on ten-class datasets and on classification only, and the class covariances grow as $O(Cd^2)$. One Gaussian per class keeps only the first two moments of the modulations, which bounds the information the draws can carry; richer per-class distributions such as mixtures could capture more, and a stronger INR backbone could raise the reconstruction quality of the decoded draws. The storage advantage holds at high resolution and large budgets but not at 32×32, where a fixed set of up to 274 images per class is smaller. Finally, the Functa network is class-agnostic: it is meta-learned for reconstruction alone, so the modulation space is not shaped to separate classes, and meta-learning the decoder with a label-aware objective, for example an auxiliary classification loss on the modulations, could make the class Gaussians more separable.

AI USE STATEMENT

In this work, we used generative AI tools to draft, polish and improve the clarity of the text. We have not used generative AI tools to develop the research ideas, interpret the results, or generate data. We have reviewed all AI-assisted suggestions and take full responsibility for the final content of this work, including all scientific content, code, results, and interpretations.

REFERENCES

Krzysztof Adamkiewicz, Brian B Moser, Stanislav Frolov, Tobias Christian Nauen, Federico Raue, and Andreas Dengel. When pretty isn't useful: Investigating why modern text-to-image models fail as reliable training data generators. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 36660–36669, 2026.

Matthias Bauer, Emilien Dupont, Andy Brock, Dan Rosenbaum, Jonathan Richard Schwarz, and Hyunjik Kim. Spatial functa: Scaling functa to imagenet classification and generation. *arXiv preprint arXiv:2302.03130*, 2023.

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*, pp. 177–186. Springer, 2010.

George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10708–10717. IEEE, 2022.

George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Generalizing dataset distillation via deep generative prior. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3739–3748. IEEE, 2023.

Thomas M Cover and Joy A Thomas. *Elements of Information Theory*. John Wiley & Sons, 2nd edition, 2006.

Xiao Cui, Yulei Qin, Wengang Zhou, Hongsheng Li, and Houqiang Li. Optical: Leveraging optimal transport for contribution allocation in dataset distillation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15245–15254. IEEE, 2025.

Jiawei Du, Qin Shi, and Joey Tianyi Zhou. Sequential subset matching for dataset distillation. *Advances in Neural Information Processing Systems*, 36:67487–67504, 2023.

Yuxuan Duan, Jianfu Zhang, and Liqing Zhang. Dataset distillation in latent space. *arXiv preprint arXiv:2311.15547*, 2023.

Emilien Dupont, Hyunjik Kim, S. M. Ali Eslami, Danilo Jimenez Rezende, and Dan Rosenbaum. From data to functa: Your data point is a function and you can treat it like one. In *39th International Conference on Machine Learning (ICML)*, 2022.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Chengcheng Guo, Bo Zhao, and Yanbing Bai. Deepcore: A comprehensive library for coreset selection in deep learning. In *International Conference on Database and Expert Systems Applications*, pp. 181–195. Springer, 2022.

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

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Ping Liu and Jiawei Du. The evolution of dataset distillation: Toward scalable and generalizable solutions. *arXiv preprint arXiv:2502.05673*, 2025.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

Yao Lu, Xuguang Chen, Yuchen Zhang, Jianyang Gu, Tianle Zhang, Yifan Zhang, Xiaoniu Yang, Qi Xuan, Kai Wang, and Yang You. Can pre-trained models assist in dataset distillation? *arXiv preprint arXiv:2310.03295*, 2023.

Brian Moser, Federico Raue, Sebastian Palacio, Stanislav Frolov, and Andreas Dengel. Unlocking dataset distillation with diffusion models. *Advances in Neural Information Processing Systems*, 38:72581–72614, 2026.

Radford M Neal. Sampling from multimodal distributions using tempered transitions. *Statistics and computing*, 6(4):353–366, 1996.

Tian Qin, Zhiwei Deng, and David Alvarez-Melis. Distributional dataset distillation with subtask decomposition. *arXiv preprint arXiv:2403.00999*, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Christopher J Shallue, Jaehoon Lee, Joseph Antognini, Jascha Sohl-Dickstein, Roy Frostig, and George E Dahl. Measuring the effects of data parallelism on neural network training. *Journal of Machine Learning Research*, 20(112):1–49, 2019.

Yuzhang Shang, Zhihang Yuan, and Yan Yan. Mim4dd: Mutual information maximization for dataset distillation. *Advances in Neural Information Processing Systems*, 36:11205–11218, 2023.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020.

Qichao Wang, Yunhong Lu, Hengyuan Cao, Junyi Zhang, and Min Zhang. Dmgd: Train-free dataset distillation with semantic-distribution matching in diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12417–12427, June 2026.

Julia Wolleb, Cristiana Baloescu, Alicia Durrer, Hemant D Tagare, and Xenophon Papademetris. Low-rank-modulated functa: Exploring the latent space of implicit neural representations for interpretable ultrasound video analysis. *arXiv preprint arXiv:2603.25951*, 2026.

Zeyuan Yin, Eric Xing, and Zhiqiang Shen. Squeeze, recover and relabel: Dataset condensation at imagenet scale from a new perspective. *Advances in Neural Information Processing Systems*, 36: 73582–73603, 2023.

Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 6503–6512. IEEE, 2023.

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL `https://openreview.net/forum?id=mSAKhLYLSsl`.

Xinhao Zhong, Bin Chen, Hao Fang, Xulin Gu, Shu-Tao Xia, and En-Hui Yang. Going beyond feature similarity: Effective dataset distillation based on class-aware conditional mutual information. In *International Conference on Learning Representations*, volume 2025, pp. 36598–36615, 2025a.

Xinhao Zhong, Hao Fang, Bin Chen, Xulin Gu, Meikang Qiu, Shuhan Qi, and Shu-Tao Xia. Hierarchical features matter: A deep exploration of progressive parameterization method for dataset distillation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 30462–30471. IEEE, 2025b.