DataDAM: Efficient Dataset Distillation with Attention Matching

Ahmad Sajedi[1,*], Samir Khaki[1*], Ehsan Amjadian[2,3], Lucy Z. Liu[2], Yuri A. Lawryshyn[1], and Konstantinos N. Plataniotis[1]

[1]University of Toronto [2]Royal Bank of Canada (RBC) [3]University of Waterloo

{ahmad.sajedi,samir.khaki}@mail.utoronto.ca

Code: https://github.com/DataDistillation/DataDAM

Abstract

Researchers have long tried to minimize training costs in deep learning while maintaining strong generalization across diverse datasets. Emerging research on dataset distillation aims to reduce training costs by creating a small synthetic set that contains the information of a larger real dataset and ultimately achieves test accuracy equivalent to a model trained on the whole dataset. Unfortunately, the synthetic data generated by previous methods are not guaranteed to distribute and discriminate as well as the original training data, and they incur significant computational costs. Despite promising results, there still exists a significant performance gap between models trained on condensed synthetic sets and those trained on the whole dataset. In this paper, we address these challenges using efficient Dataset Distillation with Attention Matching (DataDAM), achieving state-of-the-art performance while reducing training costs. Specifically, we learn synthetic images by matching the spatial attention maps of real and synthetic data generated by different layers within a family of randomly initialized neural networks. Our method outperforms the prior methods on several datasets, including CIFAR10/100, TinyImageNet, ImageNet-1K, and subsets of ImageNet-1K across most of the settings, and achieves improvements of up to 6.5% and 4.1% on CIFAR100 and ImageNet-1K, respectively. We also show that our high-quality distilled images have practical benefits for downstream applications, such as continual learning and neural architecture search.

1. Introduction

Deep learning has been highly successful in various fields, including computer vision and natural language processing, due to the use of large-scale datasets and modern Deep Neural Networks (DNNs) [12, 19, 14, 21].

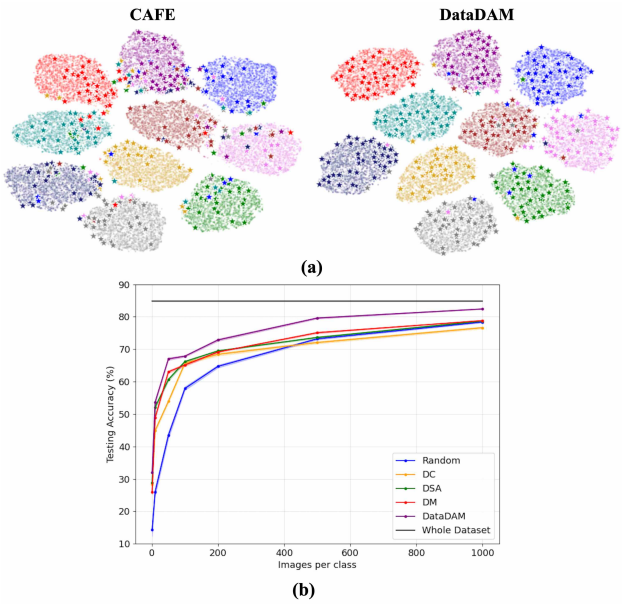


Figure 1: (a) Data distribution of the distilled images on the CIFAR10 dataset with 50 images per class (IPC50) for CAFE [43] and DataDAM. (b) Performance comparison with state-of-the-art methods on the CIFAR10 dataset for varying IPCs.

However, extensive infrastructure resources for training, hyperparameter tuning, and architectural searches make it challenging to reduce computational costs while maintaining comparable performance. Two primary approaches to address this issue are model-centric and data-centric. Model-centric methods involve model compression techniques [20, 47, 1, 49, 35], while data-centric methods concentrate on constructing smaller datasets with enough information for training, which is the focus of this paper. A traditional data-centric approach is the coreset selection method, wherein we select a representative subset of an original dataset [33, 8, 4, 37, 40]; however, these methods have limitations as they rely on heuristics to generate a coarse approximation of the whole dataset, which may lead to a suboptimal solution for downstream tasks like image

*Equal contribution

classification [40, 33]. Dataset distillation (or condensation) [44] is proposed as an alternative, which distills knowledge from a large training dataset into a smaller synthetic set such that a model trained on it achieves competitive testing performance with one trained on the real dataset. The condensed synthetic sets contain valuable information, making them a popular choice for various machine learning applications like continual learning [44, 54, 52], neural architecture search [11, 53, 54], federated learning [48, 56], and privacy-preserving [13, 41] tasks.

Dataset distillation was first proposed by Wang *et al*. [44] where bi-level meta-learning was used to optimize model parameters on synthetic data in the inner loop and refine the data with meta-gradient updates to minimize the loss on the original data in the outer loop. Various methods have been proposed to overcome the computational expense of this method, including approximating the inner optimization with kernel methods [5, 30, 29, 55], surrogate objectives like gradient matching [54, 52, 26], trajectory matching [9], and distribution matching [43, 53]. The kernel-based methods and gradient matching work still require bi-level optimization and second-order derivation computation, making training a difficult task. Trajectory matching [9] demands significant GPU memory for extra disk storage and expert model training. CAFE [43] uses dynamic bi-level optimization with layer-wise feature alignment, but it may generate biased images and incur a significant time cost (Figure 1). Thus, these methods are not scalable for larger datasets such as ImageNet-1K [12]. Distribution matching (DM) [53] was proposed as a scalable solution for larger datasets by skipping optimization steps in the inner loop. However, DM usually underperforms compared to prior methods [9].

In this paper, we propose a new framework called "**Data**set **D**istillation with **A**ttention **M**atching (DataDAM)" to overcome computational problems, achieve an unbiased representation of the real data distribution, and outperform the performance of the existing methods. Due to the effectiveness of randomly initialized networks in generating strong representations that establish a distance-preserving embedding of the data [7, 36, 16, 53], we leverage multiple randomly initialized DNNs to extract meaningful representations from real and synthetic datasets. We align their most discriminative feature maps using the Spatial Attention Matching (SAM) module and minimize the distance between them with the MSE loss. We further reduce the last-layer feature distribution disparities between the two datasets with a complementary loss as a regularizer. Unlike existing methods [54, 43, 9], our approach does not rely on pre-trained network parameters or employ bi-level optimization, making it a promising tool for synthetic data generation. The generated synthetic dataset does not introduce any bias into the data distribution while outperforming concurrent methods, as shown in Figure 1.

The contributions of our study are:

[C1]: We proposed an effective end-to-end dataset distillation method with attention matching and feature distribution alignment to closely approximate the distribution of the real dataset with low computational costs.

[C2]: Our method is evaluated on computer vision datasets with different resolutions, where it achieves state-of-the-art results across multiple benchmark settings. Our approach offers up to a 100x reduction in training costs while simultaneously enabling cross-architecture generalizations.

[C3]: Our distilled data can enhance downstream applications by improving memory efficiency for continual learning and accelerating neural architecture search through a more representative proxy dataset.

2. Related Work

Dataset Distillation. Wang *et al*. [44] first introduced dataset distillation by expressing network parameters as a function of synthetic data and optimizing the synthetic set to minimize the training loss on real training data. Later works extended this approach with soft labels [5] and a generator network [39]. Researchers have proposed simplifying the neural network model in bi-level optimization using kernel methods, such as ridge regression, which has a closed-form solution [5, 55], and a kernel ridge regression model with Neural Tangent Kernel [25] (NTK) that approximates the inner optimization [30, 29]. Alternatively, some studies have utilized surrogate objectives to address unrolled optimization problems. Dataset condensation (DC) [54] and DCC [26] generate synthetic images by matching the weight gradients of neural networks on real and distilled training datasets, while Zhao *et al*. [52] improve gradient matching with data augmentation. MTT [9] matches model parameter trajectories trained with synthetic and real datasets, and CAFE [53] and DM [43] match features generated by a model using distilled and real datasets. However, these methods have limitations, including bi-level optimization [54, 52, 43, 25], second-order derivative computation [54], generating biased examples [52, 43], and massive GPU memory demands [9, 55]. In contrast, our approach matches the spatial attention map in intermediate layers, reducing memory costs while outperforming most existing methods on standard benchmarks.

Coreset Selection. Coreset selection is another data-centric approach that chooses a representative subset of an original dataset using heuristic selection criteria. For example, random selection [33] selects samples randomly; Herding [8, 4] selects the samples closest to the cluster center for each class center; K-Center [37] chooses multiple center points of a class to minimize the maximum distance between data points and their nearest center point; and [40] identifies

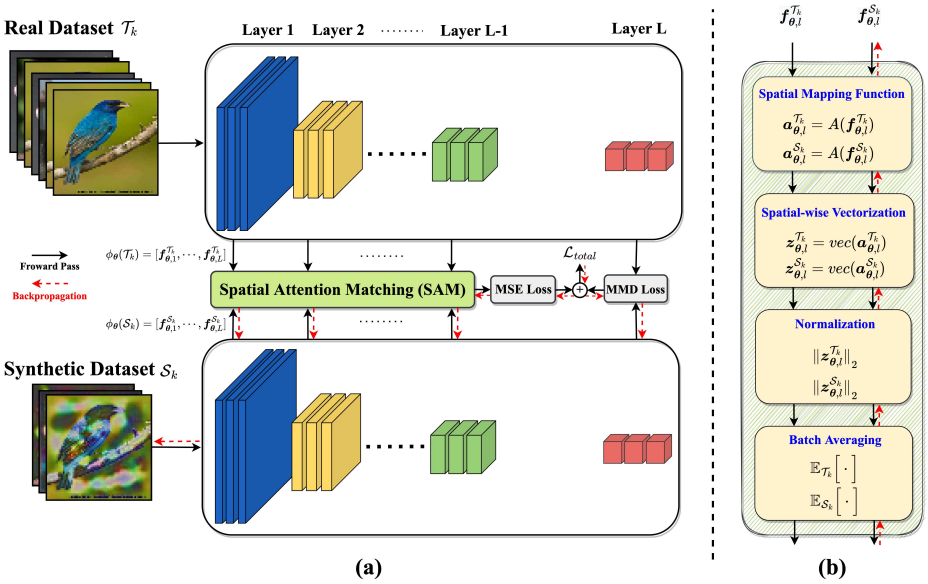


Figure 2: (a) Illustration of the proposed DataDAM method. DataDAM includes a Spatial Attention Matching (SAM) module to capture the dataset's distribution and a complementary loss for matching the feature distributions in the last layer of the encoder network. (b) The internal architecture of the SAM module.

training samples that are easily forgotten during the training process. However, heuristics-based methods may not be optimal for downstream tasks like image classification, and finding an informative corset may be challenging when the dataset's information is not concentrated in a few samples. Instead, our approach learns a computationally efficient synthetic set that is not limited to a subset of the original training samples.

Attention Mechanism. Attention has been widely used in deep learning to improve performance on various tasks [2, 45, 50], with initial applications in natural language processing by Bahdanau *et al*. [2] for language translation. Attention has since been used in computer vision, with global attention models [45] for improved classification accuracy on image datasets and convolutional block attention modules [46] for learning to attend to informative feature maps. Attention has also been used for model compression in knowledge distillation [50]. However, this mechanism has not been explored in the context of dataset distillation. To fill this gap, we propose a spatial attention matching module to approximate the distribution of the real dataset.

3. Methodology

In this section, we propose a novel end-to-end framework called **Data**set **D**istillation with **A**ttention **M**atching (DataDAM), which leverages attention maps to synthesize data that closely approximates the real training data distribution. The high dimensionality of training images makes it difficult to estimate the real data distribution accurately. Therefore, we represent each training image using spatial attention maps generated by different layers within a family of randomly initialized neural networks. These maps effectively highlight the most discriminative regions of the input image that the network focuses on at different layers (early, intermediate, and last layers) while capturing low-, mid-, and high-level representation information of the image. Although each individual network provides a partial interpretation of the image, the family of these randomly initialized networks produces a more comprehensive representation.

3.1. Dataset Distillation with Attention Matching

Given a large-scale dataset $\mathcal{T} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{|\mathcal{T}|}$ containing $|\mathcal{T}|$ real image-label pairs, we first initialize a learnable synthetic dataset $\mathcal{S} = \{(\boldsymbol{s}_j, y_j)\}_{j=1}^{|\mathcal{S}|}$ with $|\mathcal{S}|$ synthetic image and label pairs, by using either random noise or a selection of real images obtained through random sampling or a clustering algorithm such as K-Center [11, 37]. For each class k, we sample a batch of real and synthetic data (*i.e.* $B_k^{\mathcal{T}}$ and $B_k^{\mathcal{S}}$, resp.) and extract features using a neural network $\phi_{\boldsymbol{\theta}}(\cdot)$ with standard network random initialization $\boldsymbol{\theta}$ [18]. Figure 2 shows the proposed approach, where the neural network $\phi_{\boldsymbol{\theta}}(\cdot)$, consisting of L layers, is employed to embed the real and synthetic sets. The network generates feature maps for each dataset, represented as $\phi_{\boldsymbol{\theta}}(\mathcal{T}_k) = [\boldsymbol{f}^{\mathcal{T}_k}_{\boldsymbol{\theta},1}, \cdots, \boldsymbol{f}^{\mathcal{T}_k}_{\boldsymbol{\theta},L}]$ and $\phi_{\boldsymbol{\theta}}(\mathcal{S}_k) = [\boldsymbol{f}^{\mathcal{S}_k}_{\boldsymbol{\theta},1}, \cdots, \boldsymbol{f}^{\mathcal{S}_k}_{\boldsymbol{\theta},L}]$, respectively. The feature $\boldsymbol{f}^{\mathcal{T}_k}_{\boldsymbol{\theta},l}$ is a multi-dimensional array in $\mathbb{R}^{|B_k^{\mathcal{T}}| \times C_l \times W_l \times H_l}$, coming from the real dataset in the l^{th} layer, where C_l represents the

number of channels and $H_l \times W_l$ is the spatial dimensions. Similarly, a feature $\boldsymbol{f}_{\boldsymbol{\theta},l}^{\mathcal{S}_k}$ is extracted for the synthetic set.

The **S**patial **A**ttention **M**atching (SAM) module then generates attention maps for the real and synthetic images using a feature-based mapping function $A(\cdot)$. The function takes the feature maps of each layer (except the last layer) as an input and outputs two separate attention maps: $A\big(\phi_{\boldsymbol{\theta}}(\mathcal{T}_k)\big) = [\boldsymbol{a}_{\boldsymbol{\theta},1}^{\mathcal{T}_k}, \cdots, \boldsymbol{a}_{\boldsymbol{\theta},L-1}^{\mathcal{T}_k}]$ and $A(\phi_{\boldsymbol{\theta}}(\mathcal{S}_k)) = [\boldsymbol{a}_{\boldsymbol{\theta},1}^{\mathcal{S}_k}, \cdots, \boldsymbol{a}_{\boldsymbol{\theta},L-1}^{\mathcal{S}_k}]$ for the real and synthetic sets, respectively. Prior studies [50, 51] have shown that the absolute value of a hidden neuron activation can indicate its importance for a given input, thus we create a spatial attention map by aggregating the absolute values of the feature maps across the channel dimension. This means that the feature map $\boldsymbol{f}_{\boldsymbol{\theta},l}^{\mathcal{T}_k}$ of the l^{th} layer is converted into a spatial attention map $\boldsymbol{a}_{\boldsymbol{\theta},l}^{\mathcal{T}_k} \in \mathbb{R}^{|B_k^{\mathcal{T}}| \times W_l \times H_l}$ using the following pooling operation:

$$A(\boldsymbol{f}_{\boldsymbol{\theta},l}^{\mathcal{T}_k}) = \sum_{i=1}^{C_l} \left|(\boldsymbol{f}_{\boldsymbol{\theta},l}^{\mathcal{T}_k})_i\right|^p, \tag{1}$$

where, $(\boldsymbol{f}_{\boldsymbol{\theta},l}^{\mathcal{T}_k})_i = \boldsymbol{f}_{\boldsymbol{\theta},l}^{\mathcal{T}_k}(:,i,:,:)$ is the feature map of channel i from the l^{th} layer and the power and absolute value operations are applied element-wise. The resulting attention map emphasizes the spatial locations associated with neurons with the highest activations. This helps retain the most informative regions and generates a more efficient feature descriptor. In a similar manner, the attention maps for synthetic data can be obtained as $\boldsymbol{a}_{\boldsymbol{\theta},l}^{\mathcal{S}_k}$. The effect of parameter p is studied in the supplementary materials.

To capture the distribution of the original training set at different levels of representations, we compare the normalized spatial attention maps of each layer (excluding the last layer) between the real and synthetic sets using the loss function $\mathcal{L}_{\text{SAM}}$, which is formulated as

$$\mathop{\mathbb{E}}_{\boldsymbol{\theta}\sim P_{\boldsymbol{\theta}}}\left[\sum_{k=1}^{K}\sum_{l=1}^{L-1}\left\|\mathbb{E}_{\mathcal{T}_k}\left[\frac{\boldsymbol{z}_{\boldsymbol{\theta},l}^{\mathcal{T}_k}}{\|\boldsymbol{z}_{\boldsymbol{\theta},l}^{\mathcal{T}_k}\|_2}\right] - \mathbb{E}_{\mathcal{S}_k}\left[\frac{\boldsymbol{z}_{\boldsymbol{\theta},l}^{\mathcal{S}_k}}{\|\boldsymbol{z}_{\boldsymbol{\theta},l}^{\mathcal{S}_k}\|_2}\right]\right\|^2\right], \tag{2}$$

where, $\boldsymbol{z}_{\boldsymbol{\theta},l}^{\mathcal{T}_k} = vec(\boldsymbol{a}_{\boldsymbol{\theta},l}^{\mathcal{T}_k}) \in \mathbb{R}^{|B_k^{\mathcal{T}}| \times (W_l \times H_l)}$ and $\boldsymbol{z}_{\boldsymbol{\theta},l}^{\mathcal{S}_k} = vec(\boldsymbol{a}_{\boldsymbol{\theta},l}^{\mathcal{S}_k}) \in \mathbb{R}^{|B_k^{\mathcal{S}}| \times (W_l \times H_l)}$ are the l^{th} pair of vectorized attention maps along the spatial dimension for the real and synthetic sets, respectively. The parameter K is the number of categories in a dataset, and $P_{\boldsymbol{\theta}}$ denotes the distribution of network parameters. It should be noted that normalization of the attention maps in the SAM module improves performance on the syntactic set (see supplementary materials).

Despite the ability of $\mathcal{L}_{\text{SAM}}$ to approximate the real data distribution, a discrepancy still exists between the synthetic and real training sets. The features in the final layer of neural network models encapsulate the highest-level abstract information of the images in the form of an embedded representation, which has been shown to effectively capture the semantic information of the input data [34, 53, 28, 17]. Therefore, we leverage a complementary loss as a regularizer to promote similarity in the mean vectors of the embeddings between the two datasets for each class. To that end, we employ the widely known Maximum Mean Discrepancy (MMD) loss, $\mathcal{L}_{\text{MMD}}$, which is calculated within a family of kernel mean embeddings in a Reproducing Kernel Hilbert Space (RKHS) [17]. The $\mathcal{L}_{\text{MMD}}$ loss is formulated as

$$\mathop{\mathbb{E}}_{\boldsymbol{\theta}\sim P_{\boldsymbol{\theta}}}\left[\sum_{k=1}^{K}\left\|\mathbb{E}_{\mathcal{T}_k}\left[\tilde{\boldsymbol{f}}_{\boldsymbol{\theta},L}^{\mathcal{T}_k}\right] - \mathbb{E}_{\mathcal{S}_k}\left[\tilde{\boldsymbol{f}}_{\boldsymbol{\theta},L}^{\mathcal{S}_k}\right]\right\|_{\mathcal{H}}^2\right], \tag{3}$$

where $\mathcal{H}$ is a reproducing kernel Hilbert space. The $\tilde{\boldsymbol{f}}_{\boldsymbol{\theta},L}^{\mathcal{T}_k} = vec(\boldsymbol{f}_{\boldsymbol{\theta},L}^{\mathcal{T}_k}) \in \mathbb{R}^{|B_k^{\mathcal{T}}| \times (C_L \times W_L \times H_L)}$ and $\tilde{\boldsymbol{f}}_{\boldsymbol{\theta},L}^{\mathcal{S}_k} = vec(\boldsymbol{f}_{\boldsymbol{\theta},L}^{\mathcal{S}_k}) \in \mathbb{R}^{|B_k^{\mathcal{S}}| \times (C_L \times W_L \times H_L)}$ are the final feature maps of the real and synthetic sets in vectorized form with both the channel and spatial dimensions included. We estimate the expectation terms in Equations 2 and 3 empirically if ground-truth data distributions are not available. Finally, we learn the synthetic dataset by solving the following optimization problem using SGD with momentum:

$$\mathcal{S}^* = \arg\min_{\mathcal{S}} \big(\mathcal{L}_{\text{SAM}} + \lambda \mathcal{L}_{\text{MMD}}\big), \tag{4}$$

where λ is the task balance parameter. Further information on the effect of λ is discussed in Section 4.3. Note that our approach assigns a fixed label to each synthetic sample and keeps it constant during training. A summary of the learning algorithm can be found in Algorithm 1.

Algorithm 1 Dataset Distillation with Attention Matching

Input: Real training dataset $\mathcal{T} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{|\mathcal{T}|}$
Required: Initialized synthetic samples for K classes, Deep neural network $\phi_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$, Probability distribution over randomly initialized weights $P_{\boldsymbol{\theta}}$, Learning rate $\eta_{\mathcal{S}}$, Task balance parameter λ, Number of training iterations I.

```
Initialize synthetic dataset S
for i = 1, 2, ..., I do
    Sample θ from P_θ
    Sample mini-batch pairs B_k^T and B_k^S from the real
      and synthetic sets for each class k
    Compute L_SAM and L_MMD using Equations 2 and 3
    Calculate L = L_SAM + λ L_MMD
    Update the synthetic dataset using S ← S − η_S ∇_S L
end for
```

Output: Synthetic dataset $\mathcal{S} = \{(\boldsymbol{s}_i, y_i)\}_{i=1}^{|\mathcal{S}|}$

4. Experiments

In this section, we demonstrate the effectiveness of DataDAM in improving the performance of dataset distillation. We introduce the datasets and implementation details

	IPC	Ratio%	Resolution	Coreset Selection				Training Set Synthesis									Whole Dataset
				Random	Herding	K-Center	Forgetting	DD† [44]	LD† [5]	DC [54]	DSA [52]	DM [53]	CAFE [43]	KIP [30]	MTT [9]	DataDAM	
CIFAR-10	1	0.02	32	14.4 ± 2.0	21.5 ± 1.2	21.5 ± 1.3	13.5 ± 1.2	-	25.7 ± 0.7	28.3 ± 0.5	28.8 ± 0.7	26.0 ± 0.8	31.6 ± 0.8	29.8 ± 1.0	31.9 ± 1.2	**32.0 ± 1.2**	84.8 ± 0.1
	10	0.2	32	26.0 ± 1.2	31.6 ± 0.7	14.7 ± 0.9	23.3 ± 1.0	36.8 ± 1.2	38.3 ± 0.4	44.9 ± 0.5	52.1 ± 0.5	48.9 ± 0.6	50.9 ± 0.5	46.1 ± 0.7	**56.4 ± 0.7**	54.2 ± 0.8	
	50	1	32	43.4 ± 1.0	40.4 ± 0.6	27.0 ± 1.4	23.3 ± 1.1	-	42.5 ± 0.4	53.9 ± 0.5	60.6 ± 0.5	63.0 ± 0.4	62.3 ± 0.4	53.2 ± 0.7	65.9 ± 0.6	**67.0 ± 0.4**	
CIFAR-100	1	0.2	32	4.2 ± 0.3	8.3 ± 0.3	8.4 ± 0.3	4.5 ± 0.2	-	11.5 ± 0.4	12.8 ± 0.3	13.9 ± 0.3	11.4 ± 0.3	14.0 ± 0.3	12.0 ± 0.2	13.8 ± 0.6	**14.5 ± 0.5**	56.2 ± 0.3
	10	2	32	14.6 ± 0.5	17.3 ± 0.3	17.3 ± 0.3	15.1 ± 0.3	-	-	25.2 ± 0.3	32.3 ± 0.3	29.7 ± 0.3	31.5 ± 0.2	29.0 ± 0.3	33.1 ± 0.4	**34.8 ± 0.5**	
	50	10	32	30.0 ± 0.4	33.7 ± 0.5	30.5 ± 0.3	-	-	-	30.6 ± 0.6	42.8 ± 0.4	43.6 ± 0.4	42.9 ± 0.2	-	42.9 ± 0.3	**49.4 ± 0.3**	
Tiny ImageNet	1	0.2	64	1.4 ± 0.1	2.8 ± 0.2	1.6 ± 0.1	-	-	-	5.3 ± 0.1	5.7 ± 0.1	3.9 ± 0.2	-	-	6.2 ± 0.4	**8.3 ± 0.4**	37.6 ± 0.4
	10	2	64	5.0 ± 0.2	6.3 ± 0.2	5.1 ± 0.2	-	-	-	12.9 ± 0.1	16.3 ± 0.2	12.9 ± 0.4	-	-	17.3 ± 0.2	**18.7 ± 0.3**	
	50	10	64	15.0 ± 0.4	16.7 ± 0.3	15.0 ± 0.3	-	-	-	12.7 ± 0.4	5.1 ± 0.2	25.3 ± 0.2	-	-	26.5 ± 0.3	**28.7 ± 0.3**	

Table 1: The performance (testing accuracy %) comparison to state-of-the-art methods. We distill the given number of images per class using the training set, train a neural network on the synthetic set from scratch, and evaluate the network on the testing data. IPC: image(s) per class. Ratio (%): the ratio of distilled images to the whole training set. The works DD† and LD† use AlexNet [23] for CIFAR-10 dataset. All other methods use a 128-width ConvNet for training and evaluation. **Bold entries** are the best results. Note: some entries are marked as absent due to scalability issues or unreported values. For more information, refer to the supplementary materials.

for reproducibility (Section 4.1), compare our method with state-of-the-art benchmarks (Section 4.2), conduct ablation studies to evaluate each component's efficacy and transferability across various architectures (Section 4.3), and show some visualizations (Section 4.4). Finally, we demonstrate the applicability of our method to the common tasks of continual learning and neural architecture search (Section 4.5).

4.1. Experimental Setup

Datasets. Our method was evaluated on CIFAR10/100 datasets [22], which have a resolution of 32 × 32, in line with state-of-the-art benchmarks. For medium-resolution data, we resized the Tiny ImageNet [24] and ImageNet-1K [12] datasets to 64 × 64. Previous work on dataset distillation [9] introduced subsets of ImageNet-1K that focused on categories and aesthetics, including assorted objects, dog breeds, and birds. We utilized these subsets, namely ImageNette, ImageWoof, and ImageSquawk, which consist of 10 classes, as high-resolution (128 × 128) datasets in our experimental studies. For more detailed information on the datasets, please refer to the supplementary materials.

Network Architectures. We use a ConvNet architecture [15] for the distillation task, similar to prior research. The default ConvNet has three identical convolutional blocks and a linear classifier. Each block includes a 128-kernel 3 × 3 convolutional layer, instance normalization, ReLU activation, and 3 × 3 average pooling with a stride of 2. We adjust the network for medium- and high-resolution data by adding a fourth and fifth convolutional block to account for the higher resolutions, respectively. In all experiments, we initialize the network parameters using normal initialization [18].

Evaluation. We evaluate the methods using standard measures from prior studies [53, 54, 43, 52]. We generate five sets of small synthetic images using 1, 10, and 50 images per class (IPC) from a real training dataset. Next, we train 20 neural network models on each synthetic set using an SGD optimizer with a learning rate of 0.01. We report the mean and standard deviation over 100 models for each experiment to assess the effectiveness of the performance of distilled datasets. Additionally, we evaluate computational costs using run-time expressed per step, averaged over 100 iterations, and peak GPU memory usage during 100 iterations of training. Finally, we visualize the unbiasedness of state-of-the-art methods using t-SNE visualization [42].

	IPC	Ratio%	Resolution	Random	DM [53]	DataDAM	Whole Dataset
ImageNet-1K	1	0.078	64	0.5 ± 0.1	1.3 ± 0.1	**2.0 ± 0.1**	33.8 ± 0.3
	2	0.156	64	0.9 ± 0.1	1.6 ± 0.1	**2.2 ± 0.1**	
	10	0.780	64	3.1 ± 0.2	5.7 ± 0.1	**6.3 ± 0.0**	
	50	3.902	64	7.6 ± 1.2	11.4 ± 0.9	**15.5 ± 0.2**	
ImageNette	1	0.105	128	23.5 ± 4.8	32.8 ± 0.5	**34.7 ± 0.9**	87.4 ± 1.0
	10	1.050	128	47.7 ± 2.4	58.1 ± 0.3	**59.4 ± 0.4**	
ImageWoof	1	0.110	128	14.2 ± 0.9	21.1 ± 1.2	**24.2 ± 0.5**	67.0 ± 1.3
	10	1.100	128	27.0 ± 1.9	31.4 ± 0.5	**34.4 ± 0.4**	
ImageSquawk	1	0.077	128	21.8 ± 0.5	31.2 ± 0.7	**36.4 ± 0.8**	87.5 ± 0.3
	10	0.770	128	40.2 ± 0.4	50.4 ± 1.2	**55.4 ± 0.9**	

Table 2: The performance (testing accuracy %) comparison to state-of-the-art methods on ImageNet-1K [12] and ImageNet subsets [9].

Implementation Details. We employ the SGD optimizer with a fixed learning rate of 1 to learn synthetic datasets with 1, 10, and 50 IPCs. We learn low- and medium/high-resolution synthetic images in 8000 iterations with a task balance (λ) of 0.01 and 0.02, respectively. Following from [52], we apply the differentiable augmentation strategy for learning and evaluating the synthetic set. For dataset reprocessing, we utilized the Kornia implementation of Zero Component Analysis (ZCA) with default parameters, following previous works [30, 9]. All experiments are conducted on two Nvidia A100 GPUs. Further details on hyperparameters are available in the supplementary materials.

4.2. Comparison to State-of-the-art Methods

Competitive Methods. We evaluate DataDAM against four corset selection approaches and eight advanced methods for training set synthesis. The corset selection methods include Random selection [33], Herding [8, 4], K-Center [37], and Forgetting [40]. We also compare our approach with state-of-the-art distillation methods, including Dataset Distillation [44] (DD), Flexible Dataset Distillation [5] (LD), Dataset Condensation [54] (DC), Dataset Condensation with

Differentiable Siamese Augmentation [52] (DSA), Distribution Matching [53] (DM), Aligning Features [43] (CAFE), Kernel Inducing Points [30, 29] (KIP), and Matching Training Trajectories [9] (MTT). To ensure reproducibility, we downloaded publicly available distilled data for each baseline method and trained models using our experimental setup. We make minor adjustments to some methods to ensure a fair comparison, and for those that did not conduct experiments on certain data, we implemented them using the released author codes. For details on the implementation of baselines and comparisons to other methods such as generative models [31, 6, 27], please refer to the supplementary materials.

Performance Comparison. We compare our method with selection- and synthesis-based approaches in Tables 1 and 2. The results demonstrate that training set synthesis methods outperform coreset methods, especially when the number of images per class is limited to 1 or 10. This is due to the fact that synthetic training data is not limited to a specific set of real images. Moreover, our method consistently outperforms all baselines in most settings for low-resolution datasets, with improvements on the top competitor, MTT, of 1.1% and 6.5% for the CIFAR10/100 datasets when using IPC50. This indicates that our DataDAM can achieve up to 88% of the upper-bound performance with just 10% of the training dataset on CIFAR100 and up to 79% of the performance with only 1% of the training dataset on CIFAR10. For medium- and high-resolution datasets, including Tiny ImageNet, ImageNet-1K, and ImageNet subsets, DataDAM also surpasses all baseline models across all settings. While existing methods fail to scale up to the ImageNet-1K due to memory or time constraints, DataDAM achieved accuracies of 2.0%, 2.2%, 6.3%, and 15.5% for 1, 2, 10, and 50 IPC, respectively, surpassing DM and Random by a significant margin. This improvement can be attributed to our methodology, which captures essential layer-wise information through spatial attention maps and the feature map of the last layer. Our ablation studies provide further evidence that the performance gain is directly related to the discriminative ability of the method in the synthetic image learning scheme.

	T\E	ConvNet	AlexNet	VGG-11	ResNet-18
DC [54]	ConvNet	53.9±0.5	28.8±0.7	38.8±1.1	20.9±1.0
CAFE [43]	ConvNet	62.3±0.4	43.2±0.4	48.8±0.5	43.3±0.7
DSA [52]	ConvNet	60.6±0.5	53.7±0.6	51.4±1.0	47.8±0.9
DM [53]	ConvNet	63.0±0.4	60.1±0.5	57.4±0.8	52.9±0.4
KIP [30]	ConvNet	56.9±0.4	53.2±1.6	53.2±0.5	47.6±0.8
MTT [9]	ConvNet	66.2±0.6	43.9±0.9	48.7±1.3	60.0±0.7
DataDAM	ConvNet	**67.0±0.4**	**63.9±0.9**	**64.8±0.5**	**60.2±0.7**
	AlexNet	61.8±0.6	60.6±0.9	61.8±0.6	56.4±0.7
	VGG-11	56.5±0.4	53.7±1.5	56.2±0.6	52.0±0.7

Table 3: Cross-architecture testing performance (%) on CIFAR10 with 50 images per class. The synthetic set is trained on one architecture (T) and then evaluated on another architecture (E).

Cross-architecture Generalization. In this section, we test our learned synthetic data across different unseen neural architectures, consistent with state-of-the-art benchmarks [54, 53]. To that end, synthetic data was generated from CIFAR10 using one architecture (T) with IPC50 and then transferred to a new architecture (E), where it was trained from scratch and tested on real-world data. Popular CNN architectures like ConvNet [15], AlexNet [23], VGG-11 [38], and ResNet-18 [19] are used to examine the generalization performance.

Table 3 shows that DataDAM outperforms state-of-the-art across unseen architectures when the synthetic data is learned with ConvNet. We achieve a margin of 3.8% and 7.4% when transferring to AlexNet and VGG-11, respectively, surpassing the best method, DM. Additionally, the remaining architectures demonstrate improvement due to the robustness of our synthetic images and their reduced architectural bias, as seen in the natural appearance of the distilled images (Figure 6).

Training Cost Analysis. In dataset distillation, it is crucial to consider the resource-time costs of various methods, particularly in terms of scalability. This study compares our method to state-of-the-art benchmarks presented in Table 4. We demonstrate a significantly lower run-time by almost 2 orders of magnitude compared to most state-of-the-art results. Our method, like DM, has an advantage over methods such as DC, DSA, and MTT that require costly inner-loop bi-level optimization. It should be noted that DataDAM can leverage information from randomly initialized neural networks without training and consistently achieve superior performance.

Method	run time(sec)			GPU memory(MB)		
	IPC1	IPC10	IPC50	IPC1	IPC10	IPC50
DC[54]	0.16 ± 0.01	3.31 ± 0.02	15.74 ± 0.10	3515	3621	4527
DSA[52]	0.22 ± 0.02	4.47 ± 0.12	20.13 ± 0.58	3513	3639	4539
DM[53]	0.08 ± 0.02	0.08 ± 0.02	0.08 ± 0.02	3323	3455	3605
MTT[9]	0.36 ± 0.23	0.40 ± 0.20	OOM	2711	8049	OOM
DataDAM	0.09 ± 0.01	0.08 ± 0.01	0.16 ± 0.04	3452	3561	3724

Table 4: Training time and GPU memory comparisons for state-of-the-art synthesis methods. Run time is expressed per step, averaged over 100 iterations. GPU memory is expressed as the peak memory usage during 100 iterations of training. All methods were run on an A100 GPU for CIFAR-10. OOM (out-of-memory) is reported for methods that are unable to run within the GPU memory limit.

4.3. Ablation Studies

In this section, we evaluate the robustness of our method under different experimental configurations. All experiments averaged performance over 100 randomly initialized ConvNets across five synthetic sets. The CIFAR10 dataset is used for all studies. The most relevant ablation studies to our method are included here; further ablative experiments are included in the supplementary materials.

Exploring the importance of different initialization methods for synthetic images. In dataset distillation, syn-

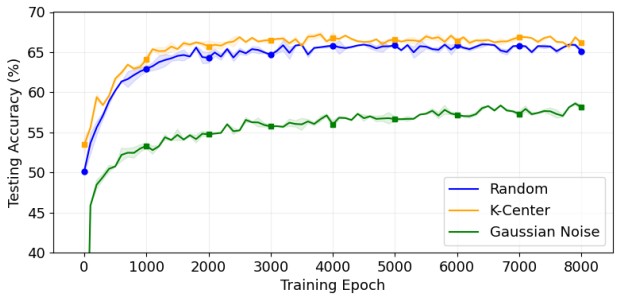


Figure 3: Test accuracy evolution of synthetic image learning on CIFAR10 with IPC50 under three different initializations: Random, K-Center, and Gaussian noise.

thetic images are usually initialized through Gaussian noise or sampled from the real data; however, the choice of initialization method has proved to be crucial to the overall performance [11]. To assess the robustness of DataDAM, we conducted an empirical evaluation with an IPC50 under three initialization conditions: Random selection, K-Center [11, 37], and Gaussian noise (Figure 3). As reported in [11], other works including [53, 52, 54] have seen benefits to testing performance and convergence speed by leveraging K-Center as a smart selection. Empirically, we show that our method is robust across both random and K-Center with only a minute performance gap, and thus the initialization of synthetic data is not as crucial to our final performance. Finally, when comparing with noise, we notice a performance reduction; however, based on the progression over the training epochs, it appears our method is successful in transferring the information from the real data onto the synthetic images. For further detailed experimental results, please refer to the supplementary materials.

Evaluation of task balance λ in DataDAM. It is common in machine learning to use regularization to prevent overfitting and improve generalization. In the case of DataDAM, the regularizing coefficient λ controls the trade-off between the attention matching loss $\mathcal{L}_{\text{SAM}}$ and the maximum mean discrepancy loss $\mathcal{L}_{\text{MMD}}$, which aims to reduce the discrepancy between the synthetic and real training distributions. The experiments conducted on the CIFAR10 dataset with IPC 10 showed that increasing the value of λ improved the performance of DataDAM up to a certain point (Figure 4). This is because, at lower values of λ, the attention matching loss dominates the training process, while at higher values of λ, the regularizer contributes more effectively to the overall performance. The results in Figure 4 also indicate that the method is robust to larger regularization terms, as shown by the plateau to the right of 0.01. Therefore, a task balance of 0.01 is chosen for all experiments on low-resolution data and 0.02 on medium- and high-resolution data.

Evaluation of loss components in DataDAM. We conducted an ablation study to evaluate the contribution of

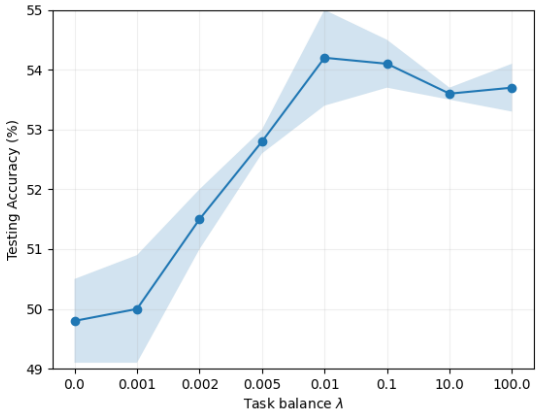


Figure 4: The effect of task balance λ on the testing accuracy (%) for CIFAR10 dataset with IPC10 configuration.

each loss component, namely spatial attention matching loss ($\mathcal{L}_{\text{SAM}}$) and the complementary loss ($\mathcal{L}_{\text{MMD}}$), to the final performance of DataDAM. As seen in table 5, the joint use of $\mathcal{L}_{\text{MMD}}$ and $\mathcal{L}_{\text{SAM}}$ led to state-of-the-art results, while using $\mathcal{L}_{\text{MMD}}$ alone resulted in significant underperformance, as it emphasizes the extraction of high-level abstract data but fails to capture different level representations of the real training distribution. On the other hand, $\mathcal{L}_{\text{SAM}}$ alone outperformed the base complementary loss, indicating the extracted discriminative features contain significant information about the training but still have room for improvement. To highlight the importance of intermediate representations, we compared our attention-based transfer approach with the transfer of layer-wise feature maps, similar to CAFE [43], and demonstrated a significant performance gap (see "Feature Map Transfer" in Table 5). Overall, our findings support the use of attention to match layer-wise representations and a complementary loss to regulate the process.

$\mathcal{L}_{\text{MMD}}$	$\mathcal{L}_{\text{SAM}}$	Feature Map Transfer	Testing Performance (%)
✓	-	-	48.9 ± 0.6
-	✓	-	49.8 ± 0.7
-	-	✓	47.2 ± 0.3
✓	✓	-	$\mathbf{54.2 \pm 0.8}$

Table 5: Evaluation of loss components in DataDAM.

Exploring the effect of each layer in DataDAM. Following the previous ablation, it is equally important to examine how each layer affects the final performance. As shown in Table 6, different layers perform differently since each provides different levels of information about the data distributions. This finding supports the claim that matching spatial attention maps in individual layers alone cannot obtain promising results. As a result, to improve the overall performance of the synthetic data learning process, it is crucial to transfer

different levels of information about the real data distribution using the SAM module across all intermediate layers.

Layer 1	Layer 2	Last Layer	Testing Performance (%)
-	-	✓	48.9 ± 0.6
✓	-	✓	50.2 ± 0.4
-	✓	✓	51.5 ± 1.0
✓	✓	-	49.8 ± 0.7
✓	✓	✓	$\mathbf{54.2 \pm 0.8}$

Table 6: Evaluation of each layer's impact in ConvNet (3-layer). The output is transferred under $\mathcal{L}_{\text{MMD}}$ while the effects of the specified layers are measured through $\mathcal{L}_{\text{SAM}}$. We evaluate the performance of the CIFAR10 dataset with IPC10.

Network Distributions. We investigate the impact of network initialization on DataDAM's performance by training 1000 ConvNet architectures with random initializations on the original training data and categorizing their learned states into five buckets based on testing performance. We sampled networks from each bucket and trained our synthetic data using IPCs 1, 10, and 50. As illustrated in Table 7, our findings indicate that DataDAM is robust across various network initializations. This is attributed to the transfer of attention maps that contain relevant and discriminative information rather than the entire feature map statistics, as shown in [43]. These results reinforce the idea that achieving state-of-the-art performance does not require inner-loop model training.

IPC	Random	0-20	20-40	40-60	60-80	≥80
1	$\mathbf{32.0 \pm 2.0}$	30.8 ± 1.1	30.7 ± 1.7	31.5 ± 1.9	26.2 ± 1.8	26.9 ± 1.3
10	$\mathbf{54.2 \pm 0.8}$	54.0 ± 0.7	53.1 ± 0.5	52.1 ± 0.8	51.2 ± 0.7	51.7 ± 0.7
50	$\mathbf{67.0 \pm 0.4}$	66.2 ± 0.4	66.4 ± 0.4	$\mathbf{67.0 \pm 0.5}$	65.8 ± 0.5	65.3 ± 0.6

Table 7: Performance of synthetic data learned with IPCs 1, 10, and 50 for different network initialization. Models are trained on the training set and grouped by their respective accuracy levels.

4.4. Visualization

Data Distribution. To evaluate whether our method can capture a more accurate distribution from the original dataset, we use t-SNE [42] to visualize the features of real and synthetic sets generated by DM, DSA, CAFE, and DataDAM in the embedding space of the ResNet-18 architecture. Figure 5 shows that methods such as DSA and CAFE are biased towards the edges of their clusters and not representative of the training data. Much like DM, our results indicate a more equalized distribution, allowing us to better capture the data distribution. Preserving dataset distributions is of utmost importance in fields like ethical machine learning since methods that cannot be impartial in capturing data distribution can lead to bias and discrimination. Our method's capacity to capture the distribution of data makes it more appropriate than other approaches in these conditions, particularly in fields such as facial detection for privacy [10].

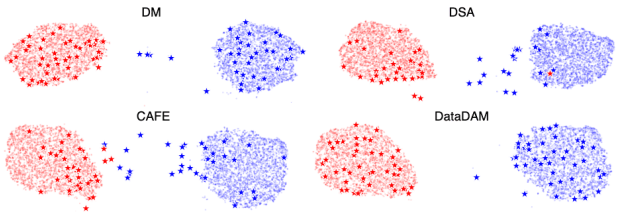


Figure 5: Distributions of synthetic images learned by four methods on CIFAR10 with IPC50. The stars represent the synthetic data dispersed amongst the original training dataset.

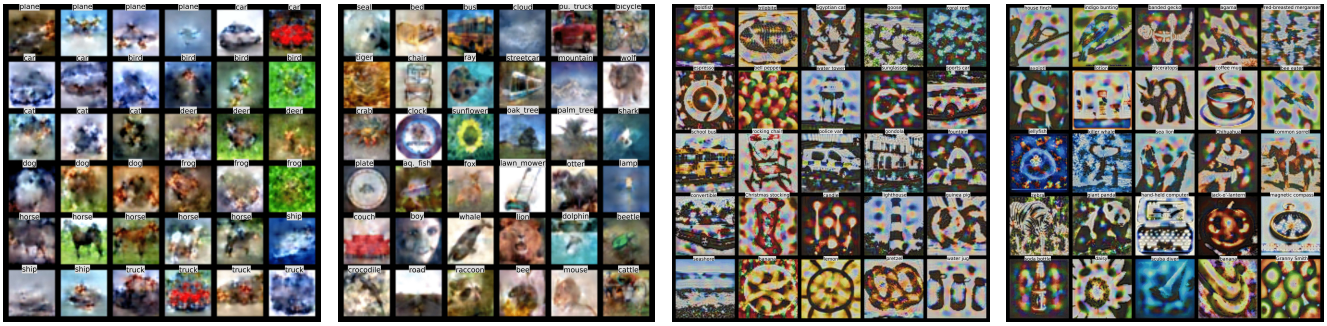
Synthetic Images. We have included samples from our learned synthetic images for different resolutions in Figure 6. In low-resolution images, the objects are easily distinguishable, and their class labels can be recognized intuitively. As we move to higher-resolution images, the objects become more outlined and distinct from their backgrounds. These synthetic images have a natural look and can be transferred well to different architectures. Moreover, the high-resolution images accurately represent the relevant colors of the objects and provide more meaningful data for downstream tasks. For more visualizations, refer to the supplementary materials.

4.5. Applications

We assess the effectiveness of DataDAM's performance through the use of two prevalent applications involving dataset distillation algorithms: *continual learning* and *neural architecture search*.

Continual Learning. Continual learning trains a model incrementally with new task labels to prevent catastrophic forgetting [33]. One approach is to maintain a replay buffer that stores balanced training examples in memory and train the model exclusively on the latest memory, starting from scratch [33, 3, 32]. Efficient storage of exemplars is crucial for optimal continual learning performance, and condensed data can play a significant role. We use the class-incremental setting from [53] with an augmented buffer size of 20 IPC to conduct class-incremental learning on the CIFAR100 dataset. We compare our proposed memory construction approach with random [32], herding [8, 4, 33], DSA [52], and DM [53] methods at 5 and 10 learning steps. In each step, including the initial one, we added 400 and 200 distilled images to the replay buffer, respectively, following the class split of [53]. The test accuracy is the performance metric, and default data preprocessing and ConvNet are used for each approach.

Figure 7 shows that our memory construction approach consistently outperforms others in both settings. Specifically, DataDAM achieves final test accuracies of 39.7% and 39.7% in 5-step and 10-step learning, respectively, outperforming DM (34.4% and 34.7%), DSA (31.7% and 30.3%), herding (28.1% and 27.4%), and random (24.8% and 24.8%). Notably, the final performance of DataDAM, DM, and random selection methods remains unchanged upon increasing the



(a) CIFAR10 (b) CIFAR100 (c) Tiny ImageNet (d) ImageNet-1K

Figure 6: Example distilled images from 32x32 CIFAR10/100 (IPC10), 64x64 Tiny ImageNet (IPC1), and 64x64 ImageNet-1K (IPC1).

number of learning steps, as these methods independently learn the synthetic datasets for each class. Our findings reveal that DataDAM provides more informative training to the models than other baselines, resulting in more effective prevention of memory loss associated with past tasks.

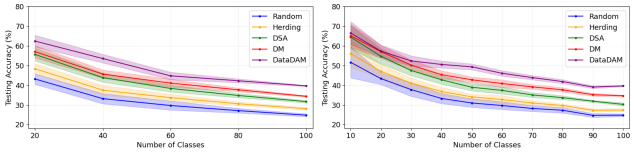


Figure 7: (Left): Showcases 5-step and (Right): Showcases 10-step continual learning with tolerance region.

Neural Architecture Search. Our synthetic sets can be used as a proxy set to accelerate model evaluation in Neural Architecture Search (NAS). Following [54], we establish a 720 ConvNet search space on CIFAR10 with a grid varying in network depth, width, activation, normalization, and pooling layers. We compared our method with Random, DSA, CAFE, early stopping, and DM. Each architecture was trained on the proxy set (synthetic 50 IPC) for 200 epochs and the whole dataset for 100 epochs to establish a baseline performance metric. Early stopping still uses the entire dataset, but we limit the iterations to those of the proxy set, as in [53]. For each method, we rank all the architectures based on the validation performance and report the testing accuracy of the best-selected model when trained on the whole dataset in Table 8. DataDAM achieved the best accuracy among the competitors, with an accuracy of 89.0%, which is very similar to the original training data at 89.2%, indicating the potential of our proxy set to accurately represent the training data. Furthermore, we calculated Spearman's correlation over the entire search space to evaluate the robustness of our learned data in architecture searching. The correlation is calculated between the testing performances of each method when trained on the proxy versus the original training data. Our method achieves the highest correlation (0.72), indicating that it generates a suitable proxy set that is generalizable across the entire search space and encodes the most important and relevant information from the training data into a condensed form. For more experimentation with NAS, refer to the supplementary materials.

	Random	DSA	DM	CAFE	Ours	Early-stopping	Whole Dataset
Performance (%)	88.9	87.2	87.2	83.6	**89.0**	88.9	89.2
Correlation	0.70	0.66	0.71	0.59	**0.72**	0.69	1.00
Time cost (min)	206.4	206.4	206.6	206.4	206.4	206.2	5168.9
Storage (imgs)	**500**	**500**	**500**	**500**	**500**	5×10^4	5×10^4

Table 8: Neural architecture search on CIFAR10.

5. Conclusion and Limitations

Our proposed method, Dataset Distillation with Attention Matching (DataDAM), efficiently captures real datasets' most informative and discriminative information. It consists of two modules, spatial attention matching (SAM) and last-layer feature alignment, that match attention maps and embedded representations generated by different layers in randomly initialized neural networks, respectively. We conduct extensive experiments on datasets with different resolutions to show that DataDAM could lower CNN training costs while maintaining superior generalization performance. We also offer two applications that take advantage of our distilled set: continual learning and neural architecture search. In the future, we plan to apply DataDAM to more fine-grained datasets and explore the analytical concepts behind them.

Limitations. DataDAM exhibits robust generalization across various CNN architectures, but it is limited to convolutional networks due to its formulation. For example, it, along with other data distillation algorithms, faces challenges in achieving successful cross-architecture generalization on ViT (Vision Transformer) models. Additionally, all data distillation methods, including DataDAM, need to be re-optimized when the distillation ratio changes, which can limit efficiency in some applications.

References

[1] Hossam Amer, Ahmed H Salamah, Ahmad Sajedi, and Enhui Yang. High performance convolution using sparsity and patterns for inference in deep convolutional neural networks. *arXiv preprint arXiv:2104.08314*, 2021. 1

[2] Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations*, 2015. 3

[3] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8218–8227, 2021. 8

[4] Eden Belouadah and Adrian Popescu. Scail: Classifier weights scaling for class incremental learning. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1266–1275, 2020. 1, 2, 5, 8

[5] Ondrej Bohdal, Yongxin Yang, and Timothy Hospedales. Flexible dataset distillation: Learn labels instead of images. *arXiv preprint arXiv:2006.08572*, 2020. 2, 5

[6] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019. 6

[7] Weipeng Cao, Xizhao Wang, Zhong Ming, and Jinzhu Gao. A review on neural networks with random weights. *Neurocomputing*, 275:278–287, 2018. 2

[8] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248, 2018. 1, 2, 5, 8

[9] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4750–4759, 2022. 2, 5, 6

[10] Umur A Ciftci, Gokturk Yuksek, and Ilke Demir. My face my choice: Privacy enhancing deepfakes for social media anonymization. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1369–1379, 2023. 8

[11] Justin Cui, Ruochen Wang, Si Si, and Cho-Jui Hsieh. Dc-bench: Dataset condensation benchmark. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2, 3, 7

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 1, 2, 5

[13] Tian Dong, Bo Zhao, and Lingjuan Lyu. Privacy for free: How does dataset condensation help privacy? In *International Conference on Machine Learning*, pages 5378–5396. PMLR, 2022. 2

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1

[15] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4367–4375, 2018. 5, 6

[16] Raja Giryes, Guillermo Sapiro, and Alex M Bronstein. Deep neural networks with random gaussian weights: A universal classification strategy? *IEEE Transactions on Signal Processing*, 64(13):3444–3457, 2016. 2

[17] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012. 4

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 3, 5

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 6

[20] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1

[21] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019. 1

[22] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. 5, 6

[24] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 5

[25] Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *Advances in neural information processing systems*, 32, 2019. 2

[26] Saehyung Lee, Sanghyuk Chun, Sangwon Jung, Sangdoo Yun, and Sungroh Yoon. Dataset condensation with contrastive signals. In *International Conference on Machine Learning*, pages 12352–12364. PMLR, 2022. 2

[27] Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *International conference on machine learning*, pages 1718–1727. PMLR, 2015. 6

[28] Chao Ma, Jia-Bin Huang, Xiaokang Yang, and Ming-Hsuan Yang. Hierarchical convolutional features for visual tracking. In *Proceedings of the IEEE international conference on computer vision*, pages 3074–3082, 2015. 4

[29] Timothy Nguyen, Zhourong Chen, and Jaehoon Lee. Dataset meta-learning from kernel-ridge regression. In *International Conference on Learning Representations*, 2021. 2, 6

[30] Timothy Nguyen, Roman Novak, Lechao Xiao, and Jaehoon Lee. Dataset distillation with infinitely wide convolutional networks. *Advances in Neural Information Processing Systems*, 34:5186–5198, 2021. 2, 5, 6

[31] Gaurav Parmar, Dacheng Li, Kwonjoon Lee, and Zhuowen Tu. Dual contradistinctive generative autoencoder. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 823–832, 2021. 6

[32] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 524–540. Springer, 2020. 8

[33] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017. 1, 2, 5, 8

[34] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3723–3732, 2018. 4

[35] Ahmad Sajedi, Yuri A Lawryshyn, and Konstantinos N Plataniotis. Subclass knowledge distillation with known subclass labels. In *2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*, pages 1–5. IEEE, 2022. 1

[36] Andrew M Saxe, Pang Wei Koh, Zhenghao Chen, Maneesh Bhand, Bipin Suresh, and Andrew Y Ng. On random weights and unsupervised feature learning. In *Icml*, volume 2, page 6, 2011. 2

[37] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018. 1, 2, 3, 5, 7

[38] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

[39] Felipe Petroski Such, Aditya Rawal, Joel Lehman, Kenneth Stanley, and Jeffrey Clune. Generative teaching networks: Accelerating neural architecture search by learning to generate synthetic training data. In *International Conference on Machine Learning*, pages 9206–9216. PMLR, 2020. 2

[40] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019. 1, 2, 5

[41] Nikolaos Tsilivis, Jingtong Su, and Julia Kempe. Can we achieve robustness from data alone? *arXiv preprint arXiv:2207.11727*, 2022. 2

[42] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 5, 8

[43] Kai Wang, Bo Zhao, Xiangyu Peng, Zheng Zhu, Shuo Yang, Shuo Wang, Guan Huang, Hakan Bilen, Xinchao Wang, and Yang You. Cafe: Learning to condense dataset by aligning features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12196–12205, 2022. 1, 2, 5, 6, 7, 8

[44] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018. 2, 5

[45] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018. 3

[46] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 3

[47] Jiaxiang Wu, Cong Leng, Yuhang Wang, Qinghao Hu, and Jian Cheng. Quantized convolutional neural networks for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4820–4828, 2016. 1

[48] Yuanhao Xiong, Ruochen Wang, Minhao Cheng, Felix Yu, and Cho-Jui Hsieh. Feddm: Iterative distribution matching for communication-efficient federated learning. In *Workshop on Federated Learning: Recent Advances and New Challenges (in Conjunction with NeurIPS 2022)*. 2

[49] Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7370–7379, 2017. 1

[50] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016. 3, 4

[51] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014. 4

[52] Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. In *International Conference on Machine Learning*, pages 12674–12685. PMLR, 2021. 2, 5, 6, 7, 8

[53] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6514–6523, 2023. 2, 4, 5, 6, 7, 8, 9

[54] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *Ninth International Conference on Learning Representations 2021*, 2021. 2, 5, 6, 7, 9

[55] Yongchao Zhou, Ehsan Nezhadarya, and Jimmy Ba. Dataset distillation using neural feature regression. In *Advances in Neural Information Processing Systems*, 2022. 2

[56] Yanlin Zhou, George Pu, Xiyao Ma, Xiaolin Li, and Dapeng Wu. Distilled one-shot federated learning. *arXiv preprint arXiv:2009.07999*, 2020. 2