Improved Distribution Matching for Dataset Condensation

Ganlong Zhao[1,2]   Guanbin Li[1,3*]   Yipeng Qin[4]   Yizhou Yu[2*]

[1]Sun Yat-sen University   [2]The University of Hong Kong

[3]Research Institute, Sun Yat-sen University, Shenzhen   [4]Cardiff University

zhaogl@connect.hku.hk, liguanbin@mail.sysu.edu.cn, qiny16@cardiff.ac.uk, yizhouy@acm.org

Abstract

Dataset Condensation aims to condense a large dataset into a smaller one while maintaining its ability to train a well-performing model, thus reducing the storage cost and training effort in deep learning applications. However, conventional dataset condensation methods are optimization-oriented and condense the dataset by performing gradient or parameter matching during model optimization, which is computationally intensive even on small datasets and models. In this paper, we propose a novel dataset condensation method based on distribution matching, which is more efficient and promising. Specifically, we identify two important shortcomings of naive distribution matching (i.e., imbalanced feature numbers and unvalidated embeddings for distance computation) and address them with three novel techniques (i.e., partitioning and expansion augmentation, efficient and enriched model sampling, and class-aware distribution regularization). Our simple yet effective method outperforms most previous optimization-oriented methods with much fewer computational resources, thereby scaling data condensation to larger datasets and models. Extensive experiments demonstrate the effectiveness of our method. Codes are available at https://github.com/uitrbn/IDM

1. Introduction

Deep learning [23, 25, 57] is notoriously data-hungry, which poses challenges for both its training and data storage. To improve data storage efficiency, Dataset Condensation (DC) [47, 56] aims to condense large datasets into smaller ones while retaining their validity for model training. Unlike traditional coreset selection methods [38, 44, 49, 50], such dataset condensation is often achieved through image synthesis and yields better performance. If properly condensed, the resulting datasets not only consume less storage space, but can also benefit various downstream tasks
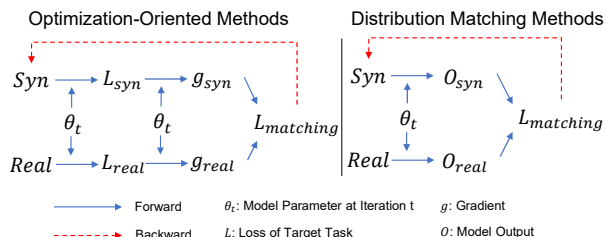

Figure 1. Illustration of optimization-oriented methods and distribution matching methods. L: classification loss; g: gradient; O: output of models; $L_{matching}$: the matching loss for condensation.

such as network architecture search and continual learning by reducing their computational costs.

The pioneering DC approach [56, 56] guarantees the validity of a condensed dataset by imposing a strong assumption that a model trained on it should be identical to that trained on the real dataset. However, naive matching between these converged models can be too challenging due to their large parameter space and long optimization towards convergence. To this end, they impose an even stronger assumption that the two models should share an identical or similar optimization path, which can be achieved by matching either their gradients [53, 56] or their intermediate model parameters [8] during training. We therefore refer to them as *optimization-oriented* methods.

However, despite their success, the unique characteristics of optimization-oriented DC methods imply that they inevitably suffer from high computational costs and thus scale poorly to large datasets and models. Specifically, they all involve the optimization of models (either randomly-initialized [53, 56] or initialized with parameters of pre-trained models [8]) against the condensed dataset, and thus rely on a nested loop that optimizes the condensed dataset and model parameters in turn. Note that the use of pre-trained models require additional computation and storage space [8]. As a result, existing optimization-oriented DC methods are only applicable to "toy" networks (*e.g.*, three-layer convolutional networks) and small datasets (*e.g.*, CIFAR10, CIFAR100). Whether they can be scaled to real-world scenarios is still an open question.

---

*Corresponding authors are Guanbin Li and Yizhou Yu.

To scale DC to large models and datasets, distribution matching (DM) [54] proposes to match the output feature distributions of the real and condensed datasets extracted by randomly-initialized models. This stems from the fact that the validity of a condensed dataset can also be guaranteed if it produces the same feature distribution as the real dataset. Since DM does not involve the optimization of models against the condensed dataset, it avoids the expensive nested loops in optimization-oriented methods, and is thus highly efficient and scalable. However, despite being promising, experimental results show that its performance still lags behind that of the state-of-the-art optimization-oriented methods.

In this paper, we perform an in-depth analysis on DM's unsatisfactory performance and propose a set of remedies that can significantly improve it, namely improved distribution matching (IDM). Specifically, we analyzed the output feature distributions of DM and observed that although their means match, the features of the condensed dataset scatter around, causing severe class misalignment problems, which accounts for its impaired performance. We ascribe such scattered features to two shortcomings of DM as follows:
i) DM suffers from the imbalanced number of features. Intuitively, DM uses the features of *small* condensed datasets to match those of *large* real datasets, which is inherently intractable. Addressing this shortcoming, we propose *Partitioning and Expansion augmentation*, which augments the condensed dataset by evenly splitting each image into $l \times l$ parts and expanding each part to the size of the original image, resulting in l^2 features per image and a better match to the features of the real dataset.
ii) Randomly initialized models are not valid embedding functions for the Maximum Mean Discrepancy (MMD) [18] estimation used in DM. Specifically, DM justifies the validity of randomly-initialized models by their intrinsic classification power observed in tasks such as deep clustering [3,5,6,36]. However, we believe that this does not apply to DM as randomly-initialized models do not satisfy the requirement of embedding functions used in MMD and makes it an invalid measure of distribution distance. Since it is too challenging to design neural network based embedding functions that are valid for MMD, we propose two simple yet effective remedies: 1) *Efficient and enriched model sampling*. We enrich the embedding functions in MMD with semi-trained models as additional feature extractors, and develop a memory-efficient model queue to facilitate their sampling. 2) *Class-aware distribution regularization*. We explicitly regularize the feature distributions of condensed datasets to further alleviate class misalignment.

These three novel techniques together help to extract better feature distributions for DM. Our contributions include:

- We deeply analyze the shortcomings of the Distributed Matching [54] algorithm and reveal that the root of its impaired performance lies in the problem of class misalignment.

- We propose improved distribution matching (IDM), consisting of three novel techniques that address the shortcomings of DM and help to learn better feature distributions.

- Experimental results show that our IDM achieves significant improvement over DM and surpasses the performance of most optimization-oriented methods.

- We show that our IDM method is highly efficient and scalable, and can be applied to large datasets such as ImageNet Subset [10,43].

2. Related Works

Dataset Condensation. Dataset condensation aims to condense large datasets to smaller ones while preserving the information to train models. It can benefit various applications including continual learning [53, 54, 56], efficient neural architecture search [53, 54, 56], federated learning [17, 41, 59] and privacy-preserving ML [11, 27]. Data Distillation (DD) [47] pioneered this topic by maximizing the accuracy of models trained by the condensed set with meta-learning techniques [33]. Later methods significantly outperformed DD by introducing more advanced techniques, such as soft-label [4, 42], gradient matching [56], augmentation [53], infinite kernel-limit [31, 32], long-range parameter matching [8], data parameterization [21], contrastive signal [26] and feature alignment [46]. Despite their success, most of the best-performing methods rely on bi-level optimizations involving second-order derivatives and thus require intensive computation resources. Recently, some researchers proposed to use subnet optimization to reduce the computation cost [58], but it still cannot condense large datasets due to its intensive kernel computation.

In contrast, Distribution Matching (DM) [54] discards the bi-level optimization and condenses datasets by matching the feature distributions of the real and condensed sets. This saves a lot of memory and computation, allowing DM to condense large datasets. However, such high efficiency comes at the cost of inferior performance, which hinders the further application of DM. In this work, we address two important shortcomings of DM, thereby scaling it to larger datasets and models without sacrificing performance.

Some other methods adopted a generative modeling approach for data condensation [29, 40, 55]. However, we do not compare with them due to the different settings.

Coreset Selection. Coreset selection methods [1, 9, 12–15, 19, 48, 52], which have been widely used in continual learning [2, 7, 35, 44] and active learning [37], first describe a criterion to measure the representativeness of samples, which

is then used to identify and cluster the coreset. Examples of these criteria include compactness [35], diversity [2, 37], and forgetfulness [44]. However, these heuristic criteria are irrelevant to target tasks and can not guarantee the optimal solution. In addition, the performance of coreset selection is restricted by the quality of the original images.

3. Problem Definition

Dataset Condensation. Given a large training set $\mathcal{T} = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_{|\mathcal{T}|}, y_{|\mathcal{T}|})\}$ containing $|\mathcal{T}|$ images and their labels, dataset condensation aims to synthesize a much smaller set $\mathcal{S} = \{(\boldsymbol{s}_1, y_1^s), ..., (\boldsymbol{s}_{|\mathcal{S}|}, y_{|\mathcal{S}|}^s)\}$, $|\mathcal{S}| \ll |\mathcal{T}|$ such that $\mathcal{S}$ has the same or similar power as $\mathcal{T}$ in terms of model training. Let $\boldsymbol{x}$ be a sample from the real data distribution P_D with label y, $\phi_{\theta^\mathcal{T}}$ and $\phi_{\theta^\mathcal{S}}$ be two variants of the same model ϕ with parameters θ trained on $\mathcal{T}$ and $\mathcal{S}$, respectively, ℓ be the loss function (cross-entropy loss), we have:

$$\mathcal{S}^* = \arg \min_{\mathcal{S}} \mathbb{E}_{\boldsymbol{x} \sim P_D} ||\ell(\phi_{\theta^\mathcal{T}}(\boldsymbol{x}), y) - \ell(\phi_{\theta^\mathcal{S}}(\boldsymbol{x}), y)||, \quad (1)$$

Distribution Matching. Previous methods either solved Eq. 1 as a nested optimization problem directly [47] or converted it to an gradient/parameter matching problem [56], which are all computationally intensive and scale poorly to large datasets. To overcome this computational barrier, Distribution Matching (DM) [54] proposed to match the feature distributions $\phi_\theta(\boldsymbol{x}_i)$ and $\phi_\theta(\boldsymbol{s}_j)$ for $\mathcal{T}$ and $\mathcal{S}$, respectively. Taking maximum mean discrepancy (MMD) [18] as the distance measure, it can be formulated as:

$$\begin{aligned}
\mathcal{S}^* &= \arg\min_{\mathcal{S}} \mathcal{L}_{DM_{\theta \sim P_{\theta_0}}} \\
&= \arg\min_{\mathcal{S}} \mathbb{E}_{\theta \sim P_{\theta_0}} ||\frac{1}{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{T}|} \phi_\theta(\boldsymbol{x}_i) - \frac{1}{|\mathcal{S}|} \sum_{j=1}^{|\mathcal{S}|} \phi_\theta(\boldsymbol{s}_j)||^2,
\end{aligned} \quad (2)$$

where P_{θ_0} denotes the distribution of randomly initialized network parameters. Note that Eq. 2 relies on the random initialization of $\theta \sim P_{\theta_0}$ and avoids its training, thereby reducing the computational cost.

4. Methodology

Recognizing the great potential of Distribution Matching (DM) [54] for data condensation, we use it as the starting point for our research. Specifically, we aim to address two of its shortcomings (SCs) described below.

SC1: Imbalanced Number of Features between $\mathcal{T}$ and $\mathcal{S}$. DM aims to match the feature distribution of $\mathcal{T}$ with that of $\mathcal{S}$ with $|\mathcal{S}| \ll |\mathcal{T}|$. However, this is inherently intractable given such a small $|\mathcal{S}|$. Specifically, in naive DM, one feature is extracted for each image in either $\mathcal{S}$ or $\mathcal{T}$. Thus, the above aim indicates that DM works well if and only if the large number of features (*e.g.*, thousands) extracted from $\mathcal{T}$

can be well approximated by a much smaller number of features (*e.g.*, ten) extracted from $\mathcal{S}$. This would only be the case when $\mathcal{T}$ is highly redundant, which is not true for most real-world scenarios.

SC2: Unvalidated Embeddings in MMD Computation. DM used maximum mean discrepancy (MMD) to measure the distance between the feature distributions of real and synthetic datasets (Eq. 2). However, instead of carefully designing the mapping functions, DM proposed to use randomly initialized networks ϕ_θ to get various embeddings for MMD estimation directly, whose validity has not been verified. Specifically, DM claimed that randomly initialized models, *i.e.*, $\theta \sim P_{\theta_0}$, are sufficient for feature extraction and comparable to trained models. However, randomly initialized models are inadequate as their parameters are sampled from a simple pre-defined distribution, which has a limited number of patterns and occupies only a small fraction of the hypothesis space. Besides, the equivalence between gradient and distribution matching [54] reveals the necessity of distribution matching throughout the entire optimization procedure, not just the initialization stage.

4.1. Partitioning and Expansion Augmentation

Addressing **SC1**, we propose a simple yet effective technique, namely partitioning and expansion augmentation, to increase the number of features extracted from $\mathcal{S}$ without increasing its size $|\mathcal{S}|$. Specifically, for each image $s_i \in \mathcal{S}$, we first partition it into $l \times l$ equal pieces, and then expand each piece to the size of s_i using differentiable augmentation [53]:

$$s_i^1, s_i^2, ... s_i^{l \times l} = \text{Expand}(\text{Partition}(s_i, l)). \quad (3)$$

In this way, we increase the number of features extracted from $\mathcal{S}$ from $|\mathcal{S}|$ to $l^2|\mathcal{S}|$ without increasing its size, facilitating DM by alleviating the imbalance in feature numbers identified in SC1.

Remark. The rationale of this augmentation stems from our observation that the potential of synthetic images in $\mathcal{S}$ was underutilized. Specifically, it is well known that many fine details of an image are discarded during feature extraction and contribute little to the final results. Based on this observation, we discard the fine details with our partitioned pieces before feature extraction but retain their power in semantic representation of an image, which makes better use of the synthetic images. We observed some concurrent works [21] with similar ideas. However, they employed an optimization-oriented approach and did not recognize the unique benefits of this augmentation to DM.

4.2. Efficient and Enriched Model Sampling

Addressing **SC2**, we propose to extend the $\theta \sim P_{\theta_0}$ used in Eq. 2 to $\theta \sim P_{\theta(T)}$ where $P_{\theta(T)} = P_{\theta_0} \cup P_{\theta_1} \cup ... \cup P_{\theta_T}$,
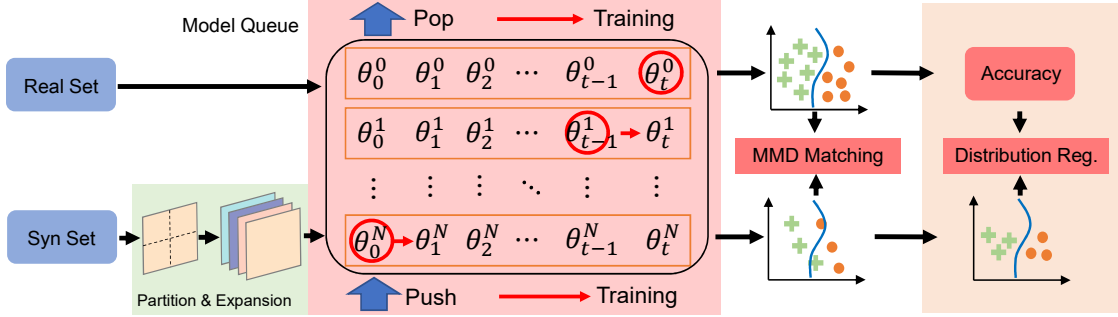
Figure 2. The overall framework of our method. Images in the synthetic set are first partitioned and expanded (Sec. 4.1) and then fed with those of the real set into the model sampled from our efficient and enriched model queue (Sec. 4.2). The matching of output distributions is regularized with our class-aware distribution regularization (Sec. 4.3). Push, Pop, and Training: model queue operations.

P_{θ_T} denotes the parameter distribution of models initialized with P_{θ_0} and trained for T iterations. Intuitively, this extension *enriched* the sampling of θ with an additional dimension "training iterations", allowing for a significant diversification of the parameter distribution so that more informative features can be extracted for data condensation.

In practice, a naive way to construct $P_\theta(T)$ is to initialize models with samples from P_{θ_0} and record their intermediate parameters during training as estimations of P_{θ_t}, ($0 \le t \le T$). However, this may not be feasible as it suffers from the trade-off between T and the computational resources required to pre-train and store the model parameter samples: the larger T, the more diverse $P_\theta(T)$, but also the more computation costs. To mitigate this trade-off, a straightforward idea is to identify and remove the redundancies in $P_\theta(T)$. Nevertheless, according to previous studies [56], the intermediate model parameters θ_t is highly dependent on its initial value θ_0 and different from each other, which implies that the elements in $P_\theta(T)$ are mostly necessary and rarely redundant. Addressing this challenge, we propose a novel data structure to facilitate *efficient* model sampling as follows.

Memory-efficient Model Queue. As an effective estimation of $P_\theta(T)$, our memory-efficient model queue Q has four operations:

- Push. Push a newly initialized model ϕ_θ to Q, where $\theta \sim P_{\theta_0}$.

- Pop. Pop a model from Q if Q's size exceeds $N_{\max}$.

- Train. Randomly fetch a model from Q and train it using real data for K iterations.

- Sample. Randomly sample a model from Q for use.

Among them, Sample implements the sampling $\theta \sim P_\theta(T)$ for data condensation and the rest three are used to maintain Q for the estimation of $P_\theta(T)$. Specifically, Train is used to obtain the intermediate model parameters θ_t during

training. Push is used to initialize Q (*i.e.*, Push N times) and to maintain the diversity of models in Q by periodically adding newly initialized models into Q. On one hand, this ensures the diversity of randomly initialized models whose parameters are sampled from P_{θ_0}; On the other hand, this ensures the diversity of training iterations, *i.e.*, the models in Q always have different training iterations. Pop is used to control the size of Q for the sake of computational cost.

As shown in Alg. 1, after initialization, we loop through Sample and DM, Train, Push, Pop, which implements the proposed memory-efficient model sampling scheme for data condensation. Intuitively, we can visualize $P_\theta(T)$ as a matrix (Fig. 2) with its rows as samples from P_{θ_0} and its columns as training iterations, and our scheme allows for a novel way of sampling $P_\theta(T)$ by **"scanning"** through the matrix. Unlike DM that only samples from the first column of the matrix and MTT [8] that stores the entire matrix for sampling, our model queue is essentially a dynamic "band" that traverses the matrix from left to right and top to bottom, thereby achieving a better balance between T and computational resources. Specifically, the left-to-right traverse is implemented by Train, and the top-to-bottom traverse is implemented by Push together with Pop.

Remark. Note that we implicitly assume that on average a model can only be trained for at most $K \times N_{\max}$ times before being popped from Q and replaced by a newly initialized model. This implies that $K \times N_{\max}$ should be large enough so that the trained models can extract all informative features of the real data for data condensation. Please see Sec. 5.3 for an empirical verification of this claim.

4.3. Class-aware Distribution Regularization

As mentioned above, we enrich the model sampling in DM with an additional dimension "training iterations" and propose an efficient data structure, namely the memory-efficient model queue Q, to organize models with different training iterations at low computational costs. In short, we have enriched P_{θ_0} in Eq. 2 to $P_\theta(T)$.

However, we observed that **SC2** still holds, *i.e.*, the enriched embeddings provided by $P_\theta(T)$ is still not enough for the MMD estimation. Specifically, we observed that the extracted features are scattered around and only their means (*i.e.*, first-order moment) are matched, which indicates that the minimization of MMD loss failed to match higher order moments of the real and synthetic distributions [28] (Sec. 5.3). As a result, such scattered features are mixed with each other and become less distinguishable in terms of classification, which impairs the performance of data condensation.

Addressing this issue, we propose to add a classification loss (*i.e.*, cross-entropy loss) as a regularization term to the synthetic distribution to make the extracted features more distinguishable. We conjecture this implicitly helps the matching of higher order moments of the two distributions [28]. However, since all our models are sampled from the model queue Q and thus have different training iterations, their classification accuracy varies and we should only require the synthetic data to achieve the same classification accuracy as the real data. To this end, let ϕ be a model sampled from Q, Acc_ϕ be its accuracy on real data, $\mathcal{L}_{CE}$ be the cross-entropy loss, we have:

$$\arg\min_\mathcal{S} Acc_\phi \mathcal{L}_{CE}(\mathcal{S}), \qquad (4)$$

Remark. We did not attempt to improve MMD as it is too challenging to design a neural network based mapping function with tractable kernels. We also abandoned the clustering approach (*i.e.*, clustering features of the same class towards their mean) as it is less relevant to the classification task and thus less effective for dataset condensation.

4.4. Overall Loss Function and Pseudocode

In summary, the overall loss function of our method is as follows[1]:

$$\mathcal{L}_{overall} = \mathcal{L}_{DM_{\phi \sim P_\theta(T)}} + \lambda_{reg} Acc_\phi \mathcal{L}_{CE}, \qquad (5)$$

where $\mathcal{L}_{DM_{\phi \sim P_\theta(T)}}$ is the modified version of the distribution matching loss depicted in Eq. 2 which samples model parameter ϕ from our enriched distribution $P_\theta(T)$, $Acc_\phi \mathcal{L}_{CE}$ is our class-aware distribution regularizer with a weighting parameter λ_{reg}. The pseudocode of our algorithm is shown in Alg. 1.

5. Experiment

5.1. Experimental Setup

Following the evaluation protocol of previous dataset condensation studies, we use image classification as a proxy task for evaluation and report the classification accuracy of

---

[1]Standard cross-entropy loss is used to train models in the model queue.

Algorithm 1: Improved Distribution Matching for Dataset Condensation

---
Input: Training set $\mathcal{T}$
Params: ϕ: network; P_{θ_0}: distribution of randomly initialized network weights; Q: model queue; N: initial queue size; N_{max}: maximum queue size; K: step number; M: training iterations.

1 Initialize Q with N randomly initialized models ϕ_{θ_i} that $\theta_i \sim P_{\theta_0}$.
2 **for** $m = 0, \cdots, M-1$ **do**
3   Sample ϕ_θ and from Q and calculate its Acc_{ϕ_θ}
    /* Improved DM   */
4   Calculate $\mathcal{L}_{DM}$ using Eq. 2 but with ϕ_θ
5   Calculate $\mathcal{L}_{CE}$ according to Eq. 4
6   Update condensed set $\mathcal{S}$ by minimizing Eq. 5
    /* Model Queue Q Maintenance   */
7   Train ϕ_θ for K steps to $\phi_{\theta'}$ and put back in Q
8   Push a new model ϕ_{θ_0} that $\theta_0 \sim P_{\theta_0}$ to Q
9   Pop a model from Q if $|Q| > N_{max}$
Output: Condensed set $\mathcal{S}$

---

deep neural networks trained on the condensed set synthesized by our method.

Datasets. Given the saturating performance of dataset condensation (DC) on simple datasets like MNIST [25] and FashionMNIST [51], we evaluate our method on four larger and more complex datasets, including CIFAR-10, CIFAR-100 [22], TinyImageNet [24] and a subset of ImageNet [10]. Following [43], ImageNet Subset selects 100 categories from the ImageNet dataset, whose high-resolution images (224×224) contain more realistic patterns and are thus closer to real-world application scenarios.

Experiment Settings. Following previous studies [53, 54, 56], we evaluate DC methods with three different settings for each dataset: condensing it to different synthetic sets of 1, 10, and 50 images per class, respectively. Unless specified, we follow DM [54] and use the same ConvNet architecture [16] in all the experiments on CIFAR-10, CIFAR-100 and TinyImageNet. This ConvNet consists of three blocks and each block is made up of a 128-kernel convolution layer, an instance normalization layer [45], a ReLU activation function [30], and an average pooling layer. The instance normalization is used to facilitate training on small condensed sets. For ImageNet Subset, we increase the number of blocks in the abovementioned ConvNet to 6 to cope with its higher resolution and more complex patterns. For the evaluation, we use the same network architectures used in DC and report the mean accuracy and standard deviation of 5 runs where the models are randomly initialized and trained for 1000 epochs using the condensed set.

Implementation Details. We follow the implementation of DM [54] to set most hyper-parameters of our method. Specifically, for model training, we use the same SGD optimizer setting in both DC and evaluation, where the learning rate is 0.01, momentum is 0.9 and weight decay is 0.0005; for the optimization of the condensed set, we use a learning rate of 0.2 with a momentum of 0.5. For the different experimental settings mentioned above, we set different λ_{reg} for the cross-entropy loss, *i.e.*, 0.5 for the cases of 1 and 10 condensed images per class, and 0.1 for the case of 50 images per class. The hyper-parameters of Model Parameter Sampling are set according to the computation demand for different datasets. For CIFAR-10/100, we set the maximum queue size N_{max} as 100 and the training step K as 10. Following DM, we perform distribution matching for each class separately. We initialize the condensed set by randomly sampling images from the training set. The partition number l is set as 2. We run all our experiments in Table 1 with a single GTX 3090 GPU with 24GB memory. Please see the supplementary material for more details.

5.2. Comparison with Previous Methods

Table 1 shows the comparison of our method with previous coreset selection and data condensation methods. Following previous studies, we compare our method to three *coreset selection* methods, Random, Herding [7, 35], and Forgetting [44]. Specifically, Random refers to randomly selects images from the training set $\mathcal{T}$ as the condensed set $\mathcal{S}$; Herding selects the samples in $\mathcal{T}$ that are closest to the clustering center for each class as $\mathcal{S}$; Forgetting selects the samples in $\mathcal{T}$ that are more frequently forgotten during the model training as $\mathcal{S}$. For previous *dataset condensation* methods, we compare to: DD [47] and LD [4], two pioneering dataset condensation methods whose performance are evaluated on different architectures; DC [56], DSA [53] and CAFE [46], three typical optimization-oriented dataset condensation methods that achieved significant performance improvement on small and lower-resolution datasets; DM [54], the first method that approached data condensation via distribution matching, which works as our baseline. Following DM [54], we do not provide the results of optimization-oriented method on TinyImageNet and ImageNet Subset due to limited computational resources.

The improvement of our IDM over DM is significant. For example, our IDM surpasses DM by 9.7% in CIFAR-10 10 Img/Cls and 15.4% in CIFAR-100 10 Img/Cls. As a result, unlike DM that lags behind most optimization-oriented methods (*e.g.*, DC, DSA, CAFE), our IDM outperforms all of them, showing the competitiveness of distribution matching based methods and thus shedding light on follow-up research. In addition, thanks to our partitioning and expansion augmentation, our IDM produces higher numbers of features, leading to better distribution estima-
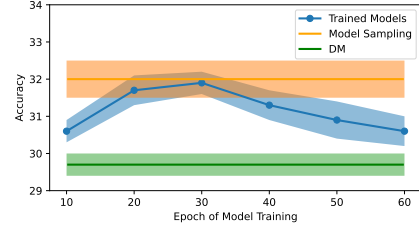

Figure 3. The performance of DM using models trained for different numbers of epochs on CIFAR-100.
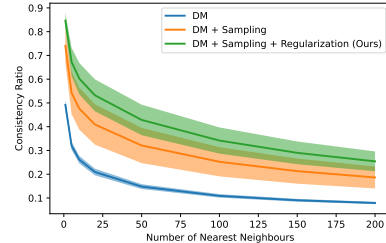

Figure 4. The proposed model sampling (Sec. 4.2) and regularization (Sec. 4.3) techniques effectively alleviate DM's class misalignment problem. y-axis: consistency ratio, *i.e.*, the ratio of each synthetic sample's real image neighbours that belong to the same class of the synthetic sample in the feature space. x-axis: the number of nearest neighbours used in consistency ratio calculation.

tions and greater improvements over DM in challenging settings (*e.g.*, CIFAR-10 1 Img/Cls). For example, we achieve a 19.6% performance improvement over DM in CIFAR-10 1 Img/Cls. For TinyImageNet and ImageNet Subset, IDM achieved similar performance improvements.

5.3. Justification of Algorithmic Motivation

Trained Models vs. Random Models. We would like to point out that an important conclusion of DM [54], which claims that the use of trained models does not significantly improve the performance of dataset condensation, only applies to small and simple datasets like CIFAR-10 and does not generalize to larger datasets like CIFAR-100.

To support our claim, we first pre-train 600 ConvNets using the entire training set of CIFAR-100 for 10, 20, 30, 40, 50, and 60 epochs respectively. Then we perform DM using models trained with the same number of epochs for 10 Img/Cls condensation. As Fig. 3 shows, i) the performance of DM using pre-trained models significantly outperform that using random models (*i.e.*, DM) across all six settings; ii) our model sampling strategy, using both trained and random models, achieves the best performance. Note that our model sampling avoids the space-consuming preparation of trained models and the time-consuming tuning of the number of training epochs.

Class Misalignment. To justify that our IDM effectively alleviate the class misalignment problem of DM [54],

Table 1. Comparison with previous coreset selection and dataset condensation methods. As in previous work, we evaluate our method on four different datasets with different numbers of synthetic images per class. Img/Cls: number of images per class. Ratio(%): the ratio of condensed images to the whole training set. Whole Dataset: the accuracy of the model trained on the whole training set. Note: DD[†] and LD[†] use different architectures *i.e.* LeNet [25] for MNIST and AlexNet [23] for CIFAR10. The rest of the methods all use ConvNet [16].

	CIFAR-10			CIFAR100			TinyImageNet			ImageNet Subset		
Img/Cls	1	10	50	1	10	50	1	10	50	1	10	50
Ratio (%)	0.02	0.2	1	0.2	2	10	0.2	2	10	0.08	0.8	4
Random	14.4±2.0	26.0±1.2	43.4±1.0	4.2±0.3	14.6±0.5	30.0±0.4	1.4±0.1	5.0±0.2	15.0±0.4	2.4±0.3	6.2±0.0	10.0±0.1
Herding	21.5±1.2	31.6±0.7	40.4±0.6	8.4±0.3	17.3±0.3	33.7±0.5	2.8±0.2	6.3±0.2	16.7±0.3	3.0±0.2	8.3±0.1	14.8±0.4
Forgetting	13.5±1.2	23.3±1.0	23.3±1.1	4.5±0.2	15.1±0.3	30.5±0.3	1.6±0.1	5.1±0.2	15.0±0.3	1.4±0.2	4.5±0.4	9.0±0.6
DD	-	36.8±1.2	-	-	-	-	-	-	-	-	-	-
LD	25.7±0.7	38.3±0.4	42.5±0.4	11.5±0.4	-	-	-	-	-	-	-	-
DC	28.3±0.5	44.9±0.5	53.9±0.5	12.8±0.3	25.2±0.3	-	-	-	-	-	-	-
DSA	28.8±0.7	52.1±0.5	60.6±0.5	13.9±0.3	32.3±0.3	42.8±0.4	-	-	-	-	-	-
CAFE	30.3±1.1	46.3±1.6	55.5±0.6	12.9±0.3	27.8±0.3	37.9±0.3	-	-	-	-	-	-
CAFE+DSA	31.6±0.8	50.9±0.5	62.3±0.4	14.0±0.3	31.5±0.2	42.9±0.2	-	-	-	-	-	-
DM	26.0±0.8	48.9±0.6	63.0±0.4	11.4±0.3	29.7±0.3	43.6±0.4	3.9±0.2	12.9±0.4	24.1±0.3	4.5±0.4	11.9±0.3	22.5±0.3
IDM (Ours)	**45.6±0.7**	**58.6±0.1**	**67.5±0.1**	**20.1±0.3**	**45.1±0.1**	**50.0±0.2**	**10.1±0.2**	**21.9±0.2**	**27.7±0.3**	**11.2±0.5**	**17.1±0.6**	**26.3±0.4**
Whole Dataset	84.8±0.1			56.2±0.3			37.6±0.4			46.8±0.6		

we report the "consistency ratio", which is the ratio of each synthetic sample's real image neighbours that belong to the same class of the synthetic sample in the feature space, on three settings: DM, DM + our model sampling (Sec. 4.2), and DM + our model sampling + our regularization (Sec. 4.3), *i.e.*, Ours. Specifically, we use k-nearest neighbors algorithm (k-NN) to get k real image neighbours and L2 norm as the similarity metric following the distribution matching loss. Fig. 4 shows the experimental results on CIFAR-100 with 10 Img/Cls. It can be observed that both our model sampling and regularization improve the consistency ratio with different k, suggesting that our IDM has effectively alleviated DM's class misalignment problem.

5.4. Ablation Study

Effectiveness of Each Component. As Table 2 shows, the three components of our method, *i.e.*, model sampling (Sec. 4.2), distribution regularization (Sec. 4.3) and augmentation (Sec. 4.1) improve the performance of DM on CIFAR-100 by 2.3%, 2.3% and 5.4% respectively. Interestingly, our model sampling lowers the performance of DM on ImageNet Subset by 1.1% while our distribution regularization improves the performance by a large margin of 3.7%. We ascribe this to our limited computational resources against large datasets, which greatly reduces the number of models affordable in our model queue, making them insufficient for training. Please note that distribution regularization is infeasible without model sampling.

Sensitivity of Hyper-parameters. Fig. 5, Fig. 6 and Fig. 7 show how the performance of our IDM change with different choices of regularization weight λ_{reg} (Sec. 4.4), the number of steps a model is trained in each training iteration K (Sec. 4.2), and the size of the model queue N_{max}

Table 2. Ablation study on CIFAR-100 and ImageNet Subset.

Dataset	CIFAR100	ImageNet Subset
Img/Cls	10	10
DM	29.7±0.3	11.9±0.3
+ Model Sampling	32.0±0.5	10.8±0.7
+ Distribution Regularization	34.3±0.3	15.6±0.4
+ Augmentation	45.1±0.1	17.1±0.6

Table 3. Ablation study of the number of partition l^2 in our augmentation (Sec. 4.1) on ImageNet Subset with 1 Img/Cls.

Partition ($l \times l$)	1×1	2×2	3×3
Accuracy	5.4±0.1	11.2±0.5	10.7±0.7

(Sec. 4.2), respectively. The experiments are conducted on CIFAR-100 with 10 Img/Cls. Note that the best performance achieved are higher than those in Table 1 because we further optimized the choices of hyper-parameters.

Number of Partition l^2 in Augmentation (Sec. 4.1). As Table 3 shows, using a partition of 2×2 achieves the best performance on ImageNet Subset with 1 Img/Cls. The performance of 3×3 partition is slightly worse as it discarded too much image details in each partition. This indicates that partitions with high l are not applicable to datasets of low-resolution images (*e.g.*, CIFAR-10/100).

5.5. Ablation Study on CIFAR-10 Architectural Generalization

Following previous studies [54, 56], we verify the cross-architectural transferability of the condensed sets on CIFAR-10 with 10 Img/Cls. Specifically, we perform data condensation with one architecture (denoted as C) and evaluate the effectiveness of the condensed set obtained with
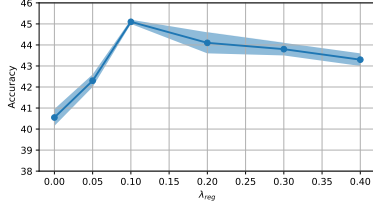
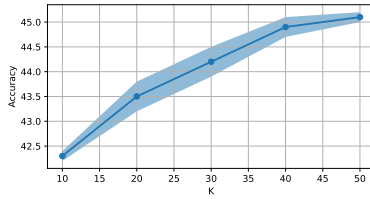Figure 5. Ablation of λ_{reg} (Sec. 4.4).
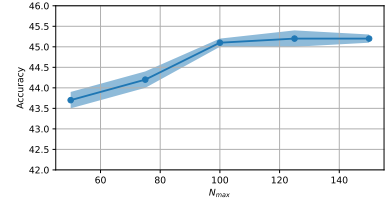

Figure 6. Ablation of K (Sec. 4.2).

Figure 7. Ablation of $N_{\max}$ (Sec. 4.2).

Table 4. Cross-architectural performance of our IDM method on CIFAR-10 with 10 Img/Cls. Our IDM achieves a significant improvement over DM.

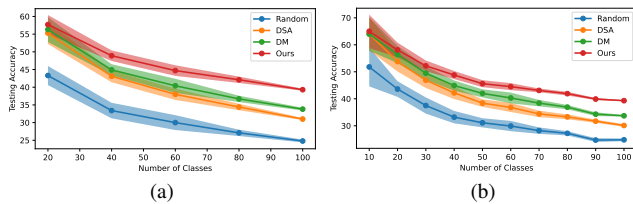|	C\T	ConvNet	AlexNet	VGG	ResNet
DM	ConvNet	48.9±0.6	38.8±0.5	42.1±0.4	41.2±1.1
Ours	ConvNet	53.0±0.3	44.6±0.8	47.8±1.1	44.6±0.4
	AlexNet	44.8±0.5	41.4±1.4	43.1±0.6	41.0±0.1
	VGG	41.2±0.4	37.4±0.3	41.7±0.4	38.8±0.8
	ResNet	38.3±0.4	37.0±0.7	39.0±0.1	39.0±0.4

Figure 8. Application in continual learning. (a) 5-step and (b) 10-step continual learning on CIFAR-100.

another architecture (denoted as T). Following DM [54], we evaluate our method on four different architectures, including ConvNet, AlexNet [23], VGG11 [39] and ResNet18 [20]. The experiments setting and all hyper-parameters are the same as Table 1, except that we set the partition parameter $l = 1$ to allow for a fair comparison with DM. As Table 4 shows, i) dataset condensation using our IDM method and ConvNet performs the best on all four evaluation architectures and significantly outperforms those of DM; ii) similar to the discussion in DM, dataset condensation with complex architectures generally performs worse due to the difficulties in optimization and noisy extracted features. A detailed comparison is provided in the supplementary material.

5.6. Continual Learning

Dataset condensation facilitates continual learning [35] by alleviating its catastrophic forgetting problem by storing more efficient training samples in the memory. Following previous studies, we build a baseline based on GDumb [34] which stores training samples greedily in the memory and maintains their class-balance. The model is trained from scratch on the latest memory only to make the continual learning performance a valid metric to measure the quality of memory construction and condensed data. Following DM [54], we compare our method with three competitors,

including Random, DSA [53] and DM. The continual learning experiments are conducted on CIFAR-100 with 5-step and 10-step settings, where the "step" indicates the number of stages in the continual training. All the experiments are repeated five times with different class order in continual learning, and we then report the mean and standard deviation of performance. The synthetic image budget for the experiment is 20 Img/Cls following DM, and we use the same class order to DM in every step to make the result comparable. The condensed images are synthesized with the same hyper-parameters of the experiment of CIFAR-100 with 10 Img/Cls, except that we set the partition parameter $l = 1$ to allow for a fair comparison with other methods.

As shown in Fig. 8a and Fig. 8b, our method outperforms Random, DSA and DM in both settings, indicating that the condensed data generated by our method are of the highest quality for continual learning. The final performance of our method is 39.3%/39.3% for 5/10 steps, while the performance of Random, DSA and DM are 24.8%/25.2%, 31.0%/29.8% and 33.8%/33.7%, respectively.

6. Conclusion

We propose an improved distribution matching (IDM) method for dataset condensation by alleviating two important shortcomings of naive distribution matching, *i.e.*, imbalanced feature numbers and unvalidated embeddings for distance computation, with three novel techniques, *i.e.*, partitioning and expansion augmentation, efficient and enriched model sampling, and class-aware distribution regularization. As a result, our method achieves significant improvements over previous methods while requiring fewer computational resources. This also allows our method to be applied to larger datasets with more categories with minimal extra cost. Extensive experiments demonstrate the effectiveness of our method, showing the competitiveness of distribution matching based methods and thus shedding light on follow-up research.

References

[1] Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. Approximating extent measures of points. *Journal of the ACM (JACM)*, 51(4):606–635, 2004. 2

[2] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32, 2019. 2, 3

[3] Ehsan Amid, Rohan Anil, Wojciech Kotłowski, and Manfred K Warmuth. Learning from randomly initialized neural network features. *arXiv preprint arXiv:2202.06438*, 2022. 2

[4] Ondrej Bohdal, Yongxin Yang, and Timothy Hospedales. Flexible dataset distillation: Learn labels instead of images. *arXiv preprint arXiv:2006.08572*, 2020. 2, 6

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

[6] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pages 132–149, 2018. 2

[7] 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. 2, 6

[8] 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. 1, 2, 4

[9] Yutian Chen, Max Welling, and Alex Smola. Super-samples from kernel herding. *arXiv preprint arXiv:1203.3472*, 2012. 2

[10] 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. 2, 5

[11] Tian Dong, Bo Zhao, and Lingjuan Lyu. Privacy for free: How does dataset condensation help privacy? *arXiv preprint arXiv:2206.00240*, 2022. 2

[12] Dan Feldman. Introduction to core-sets: an updated survey. *arXiv preprint arXiv:2011.09384*, 2020. 2

[13] Dan Feldman, Matthew Faulkner, and Andreas Krause. Scalable training of mixture models via coresets. *Advances in neural information processing systems*, 24, 2011. 2

[14] Dan Feldman, Morteza Monemizadeh, and Christian Sohler. A ptas for k-means clustering based on weak coresets. In *Proceedings of the twenty-third annual symposium on Computational geometry*, pages 11–18, 2007. 2

[15] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca, and projective clustering. *SIAM Journal on Computing*, 49(3):601–657, 2020. 2

[16] 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, 7

[17] Jack Goetz and Ambuj Tewari. Federated learning via synthetic data. *arXiv preprint arXiv:2008.04489*, 2020. 2

[18] 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. 2, 3

[19] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 291–300, 2004. 2

[20] 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. 8

[21] Jang-Hyun Kim, Jinuk Kim, Seong Joon Oh, Sangdoo Yun, Hwanjun Song, Joonhyun Jeong, Jung-Woo Ha, and Hyun Oh Song. Dataset condensation via efficient synthetic-data parameterization. In *Proceedings of the 39th International Conference on Machine Learning*, pages 11102–11118, 2022. 2, 3

[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. 1, 7, 8

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

[25] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1, 5, 7

[26] Saehyung Lee, Sanghyuk Chun, Sangwon Jung, Sangdoo Yun, and Sungroh Yoon. Dataset condensation with contrastive signals. *Proceedings of the 39th International Conference on Machine Learning*, 2022. 2

[27] Guang Li, Ren Togo, Takahiro Ogawa, and Miki Haseyama. Soft-label anonymous gastric x-ray image distillation. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 305–309. IEEE, 2020. 2

[28] Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *International conference on machine learning*, pages 1718–1727. PMLR, 2015. 5

[29] Wojciech Masarczyk and Ivona Tautkute. Reducing catastrophic forgetting with learning on synthetic data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 252–253, 2020. 2

[30] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pages 807–814, 2010. 5

[31] Timothy Nguyen, Zhourong Chen, and Jaehoon Lee. Dataset meta-learning from kernel ridge-regression. In *International Conference on Learning Representations*, 2020. 2

[32] 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

[33] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018. 2

[34] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *European conference on computer vision*, pages 524–540. Springer, 2020. 8

[35] 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. 2, 3, 6, 8

[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 *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 1089–1096, 2011. 2

[37] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017. 2, 3

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

[39] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, May 2015. 8

[40] 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

[41] Ilia Sucholutsky and Matthias Schonlau. Secdd: Efficient and secure method for remotely training neural networks. *arXiv preprint arXiv:2009.09155*, 2020. 2

[42] Ilia Sucholutsky and Matthias Schonlau. Soft-label dataset distillation and text dataset distillation. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2021. 2

[43] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *European conference on computer vision*, pages 776–794. Springer, 2020. 2, 5

[44] 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. *arXiv preprint arXiv:1812.05159*, 2018. 1, 2, 3, 6

[45] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 5

[46] 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. 2, 6

[47] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018. 1, 2, 3, 6

[48] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning*, pages 1954–1963. PMLR, 2015. 2

[49] Max Welling. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1121–1128, 2009. 1

[50] Gert W Wolf. Facility location: concepts, models, algorithms and case studies. series: Contributions to management science. *International Journal of Geographical Information Science*, 25(2):331–333, 2011. 1

[51] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 5

[52] Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. *arXiv preprint arXiv:2106.01085*, 2021. 2

[53] Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. In *International Conference on Machine Learning*, pages 12674–12685. PMLR, 2021. 1, 2, 3, 5, 6, 8

[54] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. *arXiv preprint arXiv:2110.04181*, 2021. 2, 3, 5, 6, 7, 8

[55] Bo Zhao and Hakan Bilen. Synthesizing informative training samples with gan. *arXiv preprint arXiv:2204.07513*, 2022. 2

[56] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *Ninth International Conference on Learning Representations 2021*, 2021. 1, 2, 3, 4, 5, 6, 7

[57] H.-Y. Zhou, X. Chen, Y. Zhang, R. Luo, L. Wang, and Y. Yu. Generalized radiograph representation learning via cross-supervision between images and free-text radiology reports. *Nature Machine Intelligence*, 4:32–40, 2022. 1

[58] Yongchao Zhou, Ehsan Nezhadarya, and Jimmy Ba. Dataset distillation using neural feature regression. *arXiv preprint arXiv:2206.00719*, 2022. 2

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