# CAFE: Learning to Condense Dataset by Aligning Features

Kai Wang[1*]    Bo Zhao[2*]    Xiangyu Peng[1]    Zheng Zhu[3]    Shuo Yang[4]    Shuo Wang[5]

Guan Huang [3]    Hakan Bilen[2]    Xinchao Wang [1]    Yang You[1†]

[1]National University of Singapore    [2]The University of Edinburgh    [3]PhiGent Robotics

[4]University of Technology Sydney    [5]Institute of Automation, Chinese Academy of Sciences

Code: https://github.com/kaiwang960112/CAFE

## Abstract

*Dataset condensation aims at reducing the network training effort through condensing a cumbersome training set into a compact synthetic one. State-of-the-art approaches largely rely on learning the synthetic data by matching the gradients between the real and synthetic data batches. Despite the intuitive motivation and promising results, such gradient-based methods, by nature, easily overfit to a biased set of samples that produce dominant gradients, and thus lack a global supervision of data distribution. In this paper, we propose a novel scheme to Condense dataset by Aligning FEatures (CAFE), which explicitly attempts to preserve the real-feature distribution as well as the discriminant power of the resulting synthetic set, lending itself to strong generalization capability to various architectures. At the heart of our approach is an effective strategy to align features from the real and synthetic data across various scales, while accounting for the classification of real samples. Our scheme is further backed up by a novel dynamic bi-level optimization, which adaptively adjusts parameter updates to prevent over-/under-fitting. We validate the proposed CAFE across various datasets, and demonstrate that it generally outperforms the state of the art: on the SVHN dataset, for example, the performance gain is up to 11%. Extensive experiments and analysis verify the effectiveness and necessity of proposed designs.*

## 1. Introduction

Deep neural networks (DNNs) have demonstrated unprecedented results in many if not all applications in computer vision [8, 20, 37, 25, 9, 22, 27, 7, 45, 43, 50, 49, 26, 36, 35, 34]. These gratifying results, nevertheless, come with costs: the training of DNNs heavily rely on the sheer amount of data, sometimes up to tens of millions of sam-

ples, which consequently requires enormous computational resources.
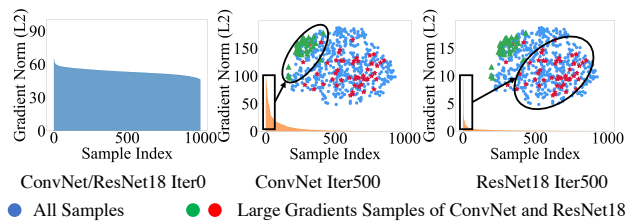


(a) The gradients distribution changes from an uniform to long-tailed during the training. Meanwhile, the overlap of large gradients samples are very small among different architectures.



(b) Visualizations the features distributions and synthetic images of **gradient matching** and **CAFE** with ConvNet.

Figure 1: (a) At the later training stage, most examples do not contribute meaningful gradients, making the synthetic set learned by gradient matching extremely bias towards those large-gradient samples, which downgrades its generalization to unseen architectures. (b) Compared with gradient-based method [54], the synthetic set learned by our approach effectively captures the whole distribution thus generalizes well to other network architectures.

Numerous research endeavours have, therefore, focused on alleviating the cumbersome training process through constructing small training sets [1, 13, 6, 12, 40, 29, 31, 42, 44, 46, 51]. One classic approach is known as coreset or subset selection [1, 29, 11], which aims to obtain a subset of salient data points to represent the original dataset of interest. Nevertheless, coreset selection is typically a NP-hard problem [18], making it computationally intractable over

large-scale datasets. Most existing approaches have thus resorted to greedy algorithms with heuristics [6, 29, 2, 47, 33] to speed up the process by trading-off the optimality.

Recently, dataset condensation [38, 54] has emerged as a competent alternative with promising results. The goal of dataset condensation is, as its name indicates, to condense a large training set into a small synthetic one, upon which DNNs are trained and expected to preserve the performances. Along this line, the pioneering approach of [38] proposed a meta-learning-based strategy; however, the nest-loop optimization precludes its scaling up to large-scale in-the-wild datasets. The work of [54] alleviates this issue by enforcing the batch gradients of the synthetic samples to approach those of the original ones, which bypasses the recursive computations and achieves impressive results. The optimization of the synthetic examples is explicitly supervised by minimizing the distance between the gradients produced by the synthetic dataset and the real dataset.

However, gradient matching method has two potential problems. First, due to the memorization effect of deep neural networks [48], only a small number of hard examples or noises produce dominant gradients over the network parameters. Thus, gradient matching may overlook those representative but easy samples, while overfit to those hard samples or noises. Second, these hard examples that produce large gradients may vary across different architectures; relying solely on gradients, therefore, will yield poor generalization performance to unseen architectures. The distributions of gradients and hard examples are illustrated in Fig. 1a. The synthetic data learned by gradient matching may be highly biased towards a small number of unrepresentative data points, which is illustrated in Fig. 1b.

To go beyond the learning bias and better capture the whole dataset distribution, in this paper, we propose a novel strategy to Condense dataset by Aligning FEatures, termed as CAFE. Unlike the approach of [54], we account for the distribution consistency between synthetic and real datasets by applying distribution-level supervision. Our approach, through matching the features that involve all intermediary layers, expands the attention across all samples and hence provides a much more comprehensive characterization of the distribution while avoiding over-fitting on hard or noisy samples. Such distribution-level supervision will, in turn, endow CAFE with stronger generalization power than gradient-based methods, since the hard examples may easily vary across different architectures.

Specifically, we impose two complementary losses into the objective of CAFE. The first one concerns capturing the data distribution, in which the layer-wise alignment between the features of the real and synthetic samples is enforced and further the distribution is preserved. The second loss, on the other hand, concerns discrimination. Intuitively, the learned synthetic samples from one class should

well represent the corresponding clusters of the real samples. Hence, we may treat each real sample as a testing sample, and classify it based on its affinity to the synthetic clusters. Our second loss is then defined upon the classification result of the real samples, which, effectively, injects the discriminant capabilities into the synthetic samples.

The proposed CAFE is further backed up by a novel bi-level optimization scheme, which allows our network and synthetic data to be updated through a customized number of SGD steps. Such a dynamic optimization strategy, in practice, largely alleviates the under- and over-fitting issues of prior methods. We conduct experiments on several popular benchmarks and demonstrate that, the results yielded by CAFE are significantly superior to the state of the art: on the SVHN dataset, for example, our method outperforms the runner-up by $11\%$ when learning 1 image/class synthetic set. We also especially prove that synthetic set learned by our method has better generalization ability than that learned by [54].

In summary, our contribution is a novel and effective approach for condensing datasets, achieved through aligning layer-wise features between the real and synthetic data, and meanwhile explicitly encoding the discriminant power into the synthetic clusters. In addition, a new bi-level optimization scheme is introduced, so as to adaptively alter the number of SGD steps. These strategies jointly enable the proposed CAFE to well characterize the distribution of the original samples, yielding state-of-the-art performances with strong generalization and robustness across various learning settings.

## 2. Related Work

**Dataset Condensation.** Several methods have been proposed to improve the performance, scalability and efficiency of dataset condensation. Based on the meta-learning method proposed in [38], some works [4, 24] try to simplify the inner-loop optimization of a classification model by training with ridge regression which has a closed-form solution. [32] trains a generative network to produce the synthetic set. The generative network is trained using the same objective as [38]. To improve the data efficiency of [54], differentiable Siamese augmentation is proposed in [52]. They enable the synthetic data train neural networks with data augmentation effectively.

A recent work [53] also learns synthetic set with feature distribution matching. Our method is different from it in three main aspects: 1) we match layer-wise features while [53] only uses final-layer features; 2) we further explicitly enable the synthetic images to be discriminative as the classifier (*i.e.* Sec. 3.3); 3) our method includes a dynamic bi-level optimization which can boost the performance with adaptive SGD steps, while [53] tries to reduce the training cost by dropping the bi-level optimization.

**Coreset Selection.** The classic technique to condense the training set size is coreset or subset selection [1, 6, 14, 39]. Most of these methods incrementally select important data points based on heuristic selection criteria. For example, [29] selects data points that can approach the cluster centers. [2] tries to maximize the diversity of samples in the gradient space. [33] measures the forgetfulness of trained samples during network training and drops those that are not easy to forget. However, these heuristic selection criteria cannot ensure that the selected subset is optimal for training models, especially for deep neural networks. In addition, greedy sample selection algorithms are unable to guarantee that the selected subset is optimal to satisfy the criterion.

**Generative Models** Our work is also closely related to generative model such as auto-encoder [17] and generative adversarial networks (GANs) [15, 23]. The difference is that image generation aims to synthesize real-looking images that can fool human beings, while our goal is to generate informative training samples that can be used to train deep neural networks more efficiently. As shown in [54], concerning training models, the efficiency of these images generated by GANs closes to that of randomly sampled real images. In contrast, our method can synthesize better training images that significantly outperform those selected real images in terms of model training.

## 3. Method

In this section, we first briefly overview the proposed CAFE. Then, we introduce three carefully designed modules: layer-wise feature alignment module, discrimination loss, and dynamic bi-level optimization module.

### 3.1. Overview

Dataset condensation aims to condense a large-scale dataset $\mathcal{T} = \{(\boldsymbol{x}_i, y_i)\}|_{i=1}^{|\mathcal{T}|}$ into small (synthetic) dataset $\mathcal{S} = \{(\boldsymbol{s}_j, y_j)\}|_{j=1}^{|\mathcal{S}|}$ while achieving similar generalization performance. Fig. 2 illustrates the proposed method. First, we sample two data batches from the large-scale dataset $\mathcal{T}$ and the learnable synthetic set $\mathcal{S}$ respectively, and then extract the features using neural network $\phi_{\boldsymbol{\theta}}(\cdot)$ which is parameterized with $\boldsymbol{\theta}$. To capture the distribution of $\mathcal{T}$ accurately, layer-wise feature alignment module is designed, in which we minimize the difference of layer-wise feature maps of real and synthetic images using Mean Square Error (MSE). To enable learning discriminative synthetic images, we use the feature centers of synthetic images of each class to classify the real images by computing their inner-product and cross-entropy loss. The synthetic images are updated by minimizing the above two losses, which is the outer-loop. Then, we update the network $\phi_{\boldsymbol{\theta}}(\cdot)$ by mini-

mizing the cross-entropy loss on synthetic images, which is the inner-loop. The synthetic images and network are alternatively using a novel dynamic bi-level optimization algorithm which avoids the over- or under-fitting on synthetic dataset and breaks the outer- and inner-loop automatically.

### 3.2. Layer-wise Features Alignment

As mentioned above, previous works [54, 52] compare the differences of gradients between real and synthetic data. Such objective produces samples with large gradients, but these samples fail to capture the distribution of the original dataset (illustrated in Fig. 5). Thus, it may have poor performance when generalizing to unseen architectures. To tackle this issue, we design Category-Wise Feature Averaging (CWFA), illustrated in Fig. 2, to measure the feature difference between $\mathcal{T}$ and $\mathcal{S}$ at each convolutional layer. Specifically, we sample a batch of real data $\mathcal{T}_k$ and synthetic data $\mathcal{S}_k$ with the same label $k$ and the batch size $N$ and $M$, from $\mathcal{T}$ and $\mathcal{S}$ respectively. We embed each real and synthetic datum using network $\phi_{\boldsymbol{\theta}}(\cdot)$ with $L$ layers (except the output layer) and obtain the layer-wise features $\boldsymbol{F}_k^{\mathcal{T}} = [\boldsymbol{f}_{k,1}^{\mathcal{T}}, \boldsymbol{f}_{k,2}^{\mathcal{T}}, ..., \boldsymbol{f}_{k,L}^{\mathcal{T}}] = \phi_{\boldsymbol{\theta}}(\mathcal{T}_k)$ and $\boldsymbol{F}_k^{\mathcal{S}} = [\boldsymbol{f}_{k,1}^{\mathcal{S}}, \boldsymbol{f}_{k,2}^{\mathcal{S}}, ..., \boldsymbol{f}_{k,L}^{\mathcal{S}}] = \phi_{\boldsymbol{\theta}}(\mathcal{S}_k)$. The $l^{\text{th}}$ layer feature $\boldsymbol{f}_{k,l}^{\mathcal{T}} \in \mathbb{R}^{N \times C'}$ is reduced to $\bar{\boldsymbol{f}}_{k,l}^{\mathcal{T}} \in \mathbb{R}^{1 \times C'}$ by averaging the $N$ samples in the real data batch, where $C' = C \times H \times W$ and it refers to the feature size of corresponding layer. Similarly, we obtain $\bar{\boldsymbol{f}}_{k,l}^{\mathcal{S}}$ for synthetic data batch.

Then, MSE is applied to calculate the feature distribution matching loss $\mathcal{L}_{\text{f}}$ for every layer, which is formulated as

$$\mathcal{L}_{\text{f}} = \sum_{k=1}^{K} \sum_{l=1}^{L} |\bar{\boldsymbol{f}}_{k,l}^{\mathcal{S}} - \bar{\boldsymbol{f}}_{k,l}^{\mathcal{T}}|^2, \qquad (1)$$

where $K$ is the number of categories in a dataset.

### 3.3. Discrimination Loss

Though the layer-wise feature alignment can capture the distribution of original dataset, it may overlook the discriminative sample mining. We hold the view that an informative synthetic set could be used as a classifier to classify real samples. Based on this, we calculate the classification loss in the last-layer feature space. we obtain the synthetic feature center $\bar{\boldsymbol{f}}_{k,L}^{\mathcal{S}} \in \mathbb{R}^{1 \times C'}$ of each category $k$ by averaging the batch. We concatenate the feature centers $\bar{\boldsymbol{F}}_L^{\mathcal{S}} = [\bar{\boldsymbol{f}}_{1,L}^{\mathcal{S}}, \bar{\boldsymbol{f}}_{2,L}^{\mathcal{S}}, ..., \bar{\boldsymbol{f}}_{K,L}^{\mathcal{S}}]$ and also real data $\boldsymbol{F}_L^{\mathcal{T}} = [\boldsymbol{f}_{1,L}^{\mathcal{T}}, \boldsymbol{f}_{2,L}^{\mathcal{T}}, ..., \boldsymbol{f}_{K,L}^{\mathcal{T}}]$ from all classes. The real data is classified using the inner-product between real data and the synthetic centers

$$\mathbf{O} = \left\langle \boldsymbol{F}_L^{\mathcal{T}}, (\bar{\boldsymbol{F}}_L^{\mathcal{S}})^{\text{T}} \right\rangle, \qquad (2)$$
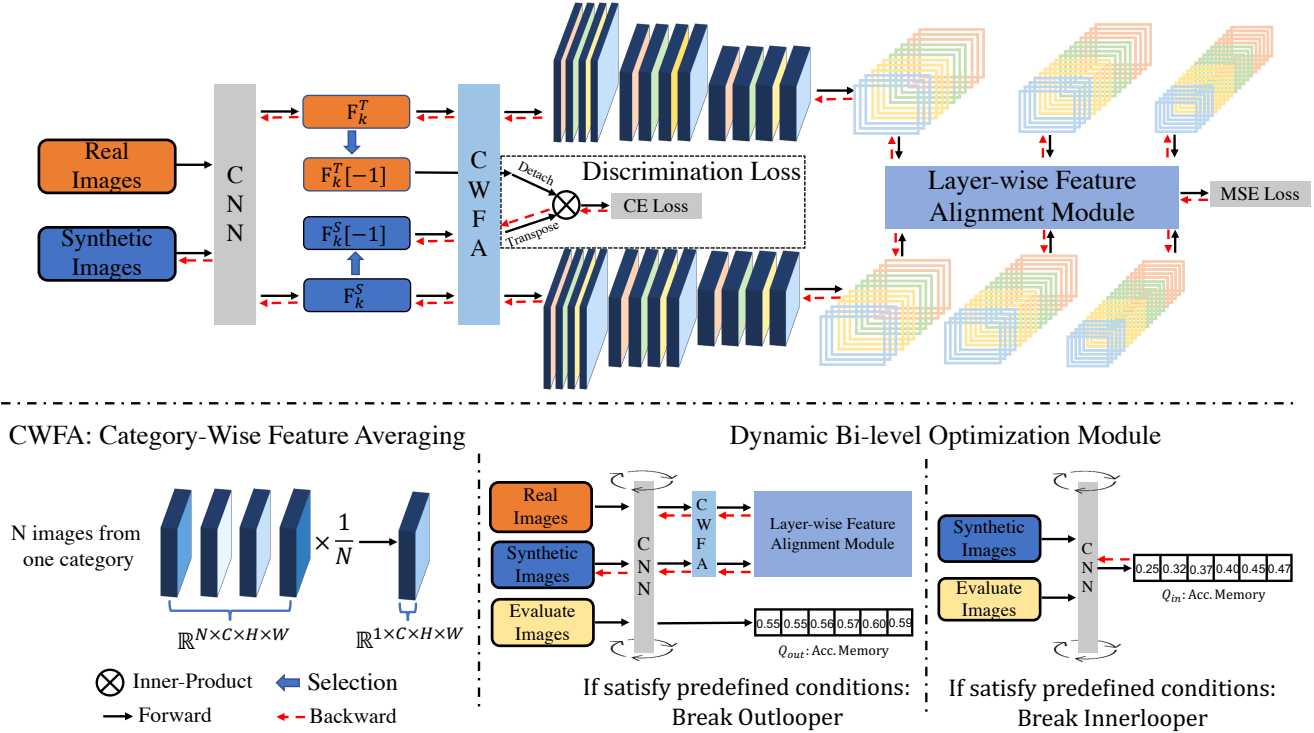
Figure 2: Illustration of the proposed CAFE pipeline. The CAFE consists of a layer-wise feature alignment module to capture the accurate distribution with the original large-scale dataset, a discrimination loss for mining the discriminate samples from real dataset, a dynamic bi-level optimization module to reduce the influence of under- and over-fitting on synthetic images.

where $\mathbf{O} \in \mathbb{R}^{N' \times K}$ contains the logists of $N' = K \times N$ real data points. The classification loss is

$$\mathcal{L}_{\mathrm{d}} = -\frac{1}{N'} \Sigma_{i=1}^{N'} \log p_i, \qquad (3)$$

where the probability $p_i$ is the softmax value corresponding to its ground-truth label over all classes $p_i = \mathrm{softmax}(\mathbf{O}_i)$. The total loss for learning synthetic images is

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{f}} + \beta \mathcal{L}_{\mathrm{d}}, \qquad (4)$$

where $\beta$ is a positive scalar weight of $\mathcal{L}_{\mathrm{d}}$. We study the influence of $\beta$ in Sec. 4.3. The synthetic set is updated by minimizing $\mathcal{L}_{\mathrm{total}}$:

$$\mathcal{S} \leftarrow \underset{\mathcal{S}}{\arg\min}\, \mathcal{L}_{\mathrm{total}} \qquad (5)$$

### 3.4. Dynamic Bi-level Optimization

Similar to previous work [38, 54], we also learn the synthetic set with a bi-level optimization, in which the synthetic set $\mathcal{S}$ is updated using Eq. 5 in the outer-loop and network parameters $\boldsymbol{\theta}$ is updated using

$$\boldsymbol{\theta} \leftarrow \underset{\boldsymbol{\theta}}{\arg\min}\, J(\mathcal{S}, \boldsymbol{\theta}) \qquad (6)$$

in the inner-loop alternatively. $J(\mathcal{S}, \boldsymbol{\theta})$ calculates the cross-entropy classification loss on the synthetic set $\mathcal{S}$. In this way, the synthetic set can be trained on many different $\boldsymbol{\theta}$ so that it can generalize to them. We initialize $\mathcal{S}$ and $\boldsymbol{\theta}$ from random noise and standard network random initialization [16]. Previous work [54, 52] sets a fixed number of outer-loop and inner-loop optimization steps, which takes too much time to adjust the hyper-parameters and may lead to networks' over- or under-fitting on synthetic set. To address these issues, we design a new bi-level optimization algorithm that can break the outer- and inner-loop automatically. Fig. 2 illustrates the proposed dynamic bi-level optimization module. To monitor the changing of network parameters $\boldsymbol{\theta}$, we randomly sample some images from real training set as a query set to evaluate the network. Then, a queue $Q$ is used to store the performance on the query set. We expect to learn synthetic data on more diverse network parameters. Hence, we sample inner-loop networks to optimize synthetic images when remarkable performance improvement is achieved on the query set. The optimization will be stopped when the performance on the query set is converged. $\lambda_1$ and $\lambda_2$ are two hyper parameters of dynamic bi-level optimization. We implement ablation study to show that the performance is not sensitive to $\lambda_1$ and $\lambda_2$. The train-

**Algorithm 1** Dynamic Bi-level Optimization

$\mathcal{T}$ and $\mathcal{S}$ are the real the synthetic datasets. $\epsilon$ is a random sampling function for selecting $N$ images from $K$ categories. $Q_{out}$ and $Q_{in}$ are the queues to save the performance on real dataset in outer- and inner-loop, respectively. $\text{div}(\cdot)$ is a function that calculates the difference between maximum and minimum values of $Q_{out}$ and $Q_{in}$. $\gamma$ is the maximum length of queues. The default loop numbers of DC are $l_{out}$ and $l_{in}$. $l_c$ represents the loop number of CAFE.

1: **while** not converged **do**
2:    randomly initialize $\boldsymbol{\theta}$, $Q_{out}$ = []; $Q_{in}$ = []; $l_c^{out}=l_c^{in}=0$.
3:    **while** True **do**
4:       updating $\mathcal{S}$ using Eq. 5; $l_c^{out}$ += 1. {outer-loop}
5:       acc. = get_acc($\epsilon(K, N)$); $Q_{out}$.append(acc.).
6:       **if** $|Q_{out}| == \gamma$ **and** $\text{div}(Q_{out}) < \lambda_1$ **or** $l_c^{out} > l_{out}$ **then**
7:          $l_c^{out} = 0$, $Q_{out}$ = [].
8:          **Break**
9:       **else**
10:         $Q_{out}$.pop[0].
11:       **end if**
12:       **while** True **do**
13:          updating $\boldsymbol{\theta}$ using Eq. 6. {inner-loop}
14:          acc. = get_acc($\epsilon(K, N)$); $Q_{in}$.append(acc.)
15:          **if** $|Q_{in}| == \gamma$ **and** $\text{div}(Q_{in}) > \lambda_2$ **or** $l_c^{in} > l_{in}$ **then**
16:             $l_c^{in} = 0$, $Q_{in}$ = [].
17:             **Break**
18:          **else**
19:             $Q_{in}$.pop[0].
20:          **end if**
21:       **end while**
22:    **end while**
23: **end while**

ing algorithm is summarized in Alg. 1.

# 4. Experiments

In this section, we first introduce the used datasets and implementation details. Then, we compare the proposed method to the state-of-the-art methods. After that, we conduct sufficient ablation studies to analyze the significant components and the influence of hyper parameters. Finally, the visualizations of synthetic images and feature distributions are provided to show the superiority of our CAFE.

## 4.1. Datasets & Implementation Details

**MNIST [21].** The MNIST is a handwritten digits dataset that is commonly used for validating image recognition models. It contains 60,000 training images and 10,000 testing images with the size of $28{\times}28$.

**FashionMNIST [41].** FashionMNIST is a dataset of Zalando's article images, consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example

is a $28{\times}28$ gray-scale image, associated with a label from 10 classes.

**SVHN [30].** SVHN is a real-world image dataset for developing machine learning and object recognition algorithms. It consists of over 600,000 digit images coming from real world data. The images are cropped to $32{\times}32$.

**CIFAR10/100 [19].** The two CIFAR datasets consist of tiny colored natural images with the size of $32{\times}32$ from 10 and 100 categories, respectively. In each dataset, 50,000 images are used for training and 10,000 images for testing.

**Implementation Details.** We present the experiments details of the outer-loop and inner-loop, respectively. In outer-loop, we optimize 1/10/50 Images Per Class (IPC) synthetic sets for all the five datasets using three-layer Convolutional Network (ConvNet) as same as [54]. The ConvNet includes three repeated "Conv-InstNorm-ReLU-AvgPool" blocks. The channel number of each convolutional layer is 128. The initial learning rate of synthetic images is 0.1, which is divided by 2 in 1,200, 1,400, and 1,800 iterations. We stop training in 2,000 iterations. For inner-loop, we train the ConvNet on synthetic sets for 300 epochs and evaluate the performances on 20 randomly initialized networks. The initial learning rate of network is 0.01. Following [54], we perform 5 experiments and report the mean and standard deviation on 100 networks. The default $N$ is 256, $\lambda_1$ is 0.05 and $\lambda_2$ is 0.05. We assess the sensitiveness of $\lambda_1$ and $\lambda_2$ in the Sec. 4.3.

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

We compare our method to four coreset selection methods, namely Random [6, 28], Herding [5, 3], K-Center [10, 29] and Forgetting [33]. We also make comparisons to recent state-of-the-art condensation methods, namely Dataset Distillation (DD) [38], LD [4], Dataset Condensation (DC) [54] and DSA (adding differentiable Siamese augmentation for DC) [52]. We report the performances of our method and competitors on five datasets in Tab. 1. When learning 1 image per class, our method achieves the best results on all the 5 datasets. In particular, the improvements on SVHN and FashionMNIST are 11% and 6.5% over other methods. Condensation-based methods outperforms than coreset selection methods with a large margin. Among coreset selection methods, Herding and K-Center outperform Random and Forgetting with a large margin. When learning 10 and 50 images/class, the performance of our method exceeds DC with 0.7%~2.6% on most datasets. Compared with DSA, Our CAFE+DSA achieves comparable results with DSA on most datasets on CIFAR10/100. For 50 images/class learning on CIFAR10, our CAFE+DSA outperforms DSA by 1.7%.

Table 1: The performance comparison to coreset selection and recent state-of-the-art methods. This table shows the testing accuracies (%) of different methods on five datasets. LD† and DD† use LeNet for MNIST and AlexNet for CIFAR10, while the rest use ConvNet for training and testing. IPC: Images Per Class, Ratio (%): the ratio of condensed images to whole training set.

| | IPC | Ratio % | Coreset Selection | | | | DD† | LD† | Condensation | | | | Whole Dataset |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Random | Herding | K-Center | Forgetting | | | DC | DSA | CAFE | CAFE+DSA | |
| MNIST | 1 | 0.017 | 64.9±3.5 | 89.2±1.6 | 89.3±1.5 | 35.5±5.6 | - | 60.9±3.2 | 91.7±0.5 | 88.7±0.6 | **93.1±0.3** | 90.8±0.5 | |
| | 10 | 0.17 | 95.1±0.9 | 93.7±0.3 | 84.4±1.7 | 68.1±3.3 | 79.5±8.1 | 87.3±0.7 | 97.4±0.2 | 97.8±0.1 | 97.2±0.2 | 97.5±0.1 | 99.6±0.0 |
| | 50 | 0.83 | 97.9±0.2 | 94.8±0.2 | 97.4±0.3 | 88.2±1.2 | - | 93.3±0.3 | 98.8±0.2 | 99.2±0.1 | 98.6±0.2 | 98.9±0.2 | |
| FashionMNIST | 1 | 0.017 | 51.4±3.8 | 67.0±1.9 | 66.9±1.8 | 42.0±5.5 | - | - | 70.5±0.6 | 70.6±0.6 | **77.1±0.9** | 73.7±0.7 | |
| | 10 | 0.17 | 73.8±0.7 | 71.1±0.7 | 54.7±1.5 | 53.9±2.0 | - | - | 82.3±0.4 | 84.6±0.3 | **83.0±0.4** | 83.0±0.3 | 93.5±0.1 |
| | 50 | 0.83 | 82.5±0.7 | 71.9±0.8 | 68.3±0.8 | 55.0±1.1 | - | - | 83.6±0.4 | 88.7±0.2 | **84.8±0.4** | 88.2±0.3 | |
| SVHN | 1 | 0.014 | 14.6±1.6 | 20.9±1.3 | 21.0±1.5 | 12.1±1.7 | - | - | 31.2±1.4 | 27.5±1.4 | **42.6±3.3** | 42.9±3.0 | |
| | 10 | 0.14 | 35.1±4.1 | 50.5±3.3 | 14.0±1.3 | 16.8±1.2 | - | - | 76.1±0.6 | 79.2±0.5 | 75.9±0.6 | 77.9±0.6 | 95.4±0.1 |
| | 50 | 0.7 | 70.9±0.9 | 72.6±0.8 | 20.1±1.4 | 27.2±1.5 | - | - | 82.3±0.3 | 84.4±0.4 | **81.3±0.3** | 82.3±0.4 | |
| CIFAR10 | 1 | 0.02 | 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 | **30.3±1.1** | 31.6±0.8 | |
| | 10 | 0.2 | 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 | **46.3±0.6** | 50.9±0.5 | 84.8±0.1 |
| | 50 | 1 | 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 | **55.5±0.6** | 62.3±0.4 | |
| CIFAR100 | 1 | 0.2 | 4.2±0.3 | 8.4±0.3 | 8.3±0.3 | 4.5±0.3 | - | 11.5±0.4 | 12.8±0.3 | 13.9±0.3 | **12.9±0.3** | 14.0±0.3 | |
| | 10 | 2 | 14.6±0.5 | 17.3±0.3 | 7.1±0.2 | 9.8±0.2 | - | - | 25.2±0.3 | 32.3±0.3 | **27.8±0.3** | 31.5±0.2 | 56.17±0.3 |
| | 50 | 10 | 30.0±0.4 | 33.7±0.5 | 30.5±0.3 | - | - | - | - | 42.8±0.4 | **37.9±0.3** | 42.9±0.2 | |

Table 2: Evaluation of the three components in CAFE

| DL | LFA | Dynamic Bi-level Opt. | Performance |
|---|---|---|---|
| ✓ | | | 49.78 |
| | ✓ | | 53.96 |
| ✓ | ✓ | | 54.53 |
| ✓ | | ✓ | 50.92 |
| | ✓ | ✓ | 54.98 |
| ✓ | ✓ | ✓ | **55.50** |

## 4.3. Ablation Studies

In this subsection, we study ablations using CIFAR10 (IPC = 50) to investigate the effectiveness of each module and the influence of the hyper parameters.

**Evaluation of the three components in CAFE.** To explore the effect of each component in our method, we design ablation studies of Discrimination Loss (DL), Layerwise Feature Alignment (LFA) and Dynamic Bi-level Optimization on CIFAR10. As shown in Tab. 2, DL, LFA and Dynamic Bi-level Opt. are complementary with each other. CAFE performs poorly when using DL individually (49.78%), as DL focuses more on classifying the real samples but ignores the distribution consistency with real images. The result of using LFA individually outperforms DL with 4.18%, which implies considering of distribution consistency is more important for dataset condensation. However, utilizing LFA independently means the importance of all the images in real dataset are equal, which may overlook the information from discriminative samples (*i.e.* samples nearby the decision boundaries). Jointly using DL and LFA

can obtain better result than using DC on CIFAR10 testing set. Adding the Dynamic Bi-level Opt. can further improve the performance of DL and LFA, which indicates breaking out-/inner-looper automatically can reduce the over-/underfitting effectively. Using these three components together achieves the highest result. To understand the effect of DL and LFA more intuitively, we also visualize the synthetic images feature distributions of using DL or LFA independently in Sec. 4.4.

Table 3: Evaluation of the importance of layer-wise feature alignment. The layer1 is closest to the output layer while layer4 is closest to input layer. Note that, layer4 represents the last average pooling layer in ConvNet.

| Layer1 | Layer2 | Layer3. | Layer4 | Performance/+DL |
|---|---|---|---|---|
| ✓ | | | | 50.74/**52.78** |
| | ✓ | | | 43.45/49.30 |
| | | ✓ | | 44.52/49.08 |
| | | | ✓ | **51.30**/52.05 |

**Exploring the importance of layer-wise feature alignment in each layer.** To investigate the importance of feature alignment, we apply the feature alignment operation to each layer individually. As shown in Tab. 3, the performances of different layers vary remarkably. Applying feature alignment operation in layer1 or layer4 obtains better results than in layer2 or layer3, as the supervision in layer2 or layer3 is far from the input and output layers. Applying feature alignment in each layer individually can not obtain promising results. To demonstrate the effectiveness of DL in each layer, we also show the results of adding DL loss. The addition of DL can consistently improve the per-

formances in all layers.

Table 4: Evaluation of complementarity of layer-wise feature alignment. The indexes of layers are same as Tab. 3.

| Layer1 | Layer2 | Layer3. | Layer4 | Performance/+DL |
|:---:|:---:|:---:|:---:|:---:|
| ✓ | | | | 50.74/52.78 |
| ✓ | ✓ | | | 51.27/53.28 |
| ✓ | ✓ | ✓ | | 53.16/53.96 |
| ✓ | ✓ | ✓ | ✓ | **54.98/55.50** |

**Exploring the complementarity of layer-wise feature alignment among all layers.** After evaluating the importance of the LFA in each layer, exploring the complementarity of LFA in all layers is also very important. We first utilize the feature alignment in the layer1 to update the synthetic images. Then, we apply the same feature alignment to other layers (*i.e.* layer2, layer3, layer4). Here, we also consider the effect of DL and report the results with and without using DL in Tab. 4. When adding feature alignment to more layers, the performances on testing set become better. Meanwhile, the DL can further improve the performance in all cases. Specifically, the performance difference between using LFA in all layers and using in the first layer is about 4% (w/o DL) and 3% (with DL). The average performance boost of adding each layer is about 1% (w/o DL) and 0.7% (with DL), which indicates the strong complementarity of layer-wise feature alignment in all layers.

**Evaluation of $\lambda_1$ and $\lambda_2$.** $\lambda_1$ and $\lambda_2$ are the thresholds to control whether break the out-looper and inner-looper or not. As shown in Fig. 3a and 3b, we study different values of $\lambda_1$ and $\lambda_2$ ranging from 0.01 to 0.08. Our default $\lambda_1 = 0.05$ and $\lambda_2 = 0.05$ achieve the best results, which outperforms DC 1.6%. For out-looper, too large $\lambda_1$ may reduce the iterations of updating the synthetic images, which leads to the worse results. Too small $\lambda_1$ increases the optimization difficulties and even makes the model unable to break the out-looper normally. As for inner-looper, the model diversity is not large enough when $\lambda_2$ is too small, whereas it would be tricky to break the inner-looper when $\lambda_2$ is very large. Furthermore, it is worth noting that our method outperforms DC with a large margin at almost all settings. Meanwhile, the performance is not sensitive to $\lambda_1$ and $\lambda_2$.

**Evaluation of the ratio $\beta$.** In Fig. 3c, we evaluate the effect of different ratios between the $\mathcal{L}_f$ and $\mathcal{L}_d$. We find that setting equal weight for each loss achieves the best results. The performance gets promoted as $\beta$ increase from 0.1 to 1, while increasing the weight of $\mathcal{L}_d$ from 1 to 10 dramatically degrades performance.

**Evaluation of $\gamma$ and the training time.** $\gamma$ is hyper parameter of the maximum length of queues in dynamic bi-level optimization. We show the performances and training time of different $\gamma$ in Tab. 5. One can find the default $\gamma = 10$ achieves the best result and requires less time than *DC*. Too small and too large $\gamma$ may lead to under- or over-fitting.

Table 5: Evaluation of $\gamma$ and the training time.

| $\gamma$ | 5 | 10 | 15 | 20 | DC |
|:---:|:---:|:---:|:---:|:---:|:---:|
| Accuracy (%) | 53.16 | **55.50** | 54.10 | 53.63 | 53.9 |
| Time (minutes) | ≈117 | ≈367 | ≈463 | ≈463 | ≈460 |

**Evaluation of the generalization to unseen architectures** To evaluate the generalization ability of synthetic data on unseen architectures, we first condense CIFAR10 dataset with ConvNet to generate synthetic images. Then, we train different architectures, including AlexNet, VGG11, ResNet18, and MLP (3 layers), on the synthetic images. As shown in Tab. 6, our method achieves better generalization performance than DC obviously. Specifically, our method outperforms DC with 5.25%, 1.79%, 4.42%, and 7.96% when testing on AlexNet, VGG11, ResNet18, and MLP (3 layers).

Table 6: The testing performance (%) on unseen architectures. The 50 IPC synthetic set is learned on one architecture (C), and then tested on another architecture (T).

| C\T | | ConvNet | AlexNet | VGG11 | ResNet18 | MLP |
|:---:|:---:|:---:|:---:|:---:|:---:|:---:|
| DC | ConvNet | 53.9±0.5 | 28.77±0.7 | 38.76±1.1 | 20.85±1.0 | 28.71±0.7 |
| CAFE | ConvNet | **55.50±0.4** | **34.02±0.6** | **40.55±0.8** | **25.27±0.9** | **36.67±0.6** |

### 4.4. Visualizations

In this subsection, we visualize the synthetic images as well as data distribution to show the effectiveness of our proposed CAFE.

**Synthetic images.** To make fair comparison, the synthetic set is initialized by the same random noise (IPC = 50). After that, we apply DC and CAFE to optimize the synthetic set on CIFAR10 dataset. Finally, the partial (only show 10 images per class) optimized synthetic images and original images of CIFAR10 are shown in Fig. 4. There are several observations can be summarized as follows: 1). It is easy to find that the synthetic images generated by our method is more visually similar to original CIFAR10 images than DC. 2). The synthetic images have more semantic information than DC, which illustrates the effectiveness of LFA and DL modules. 3). A certain ratio of images generated by DC are
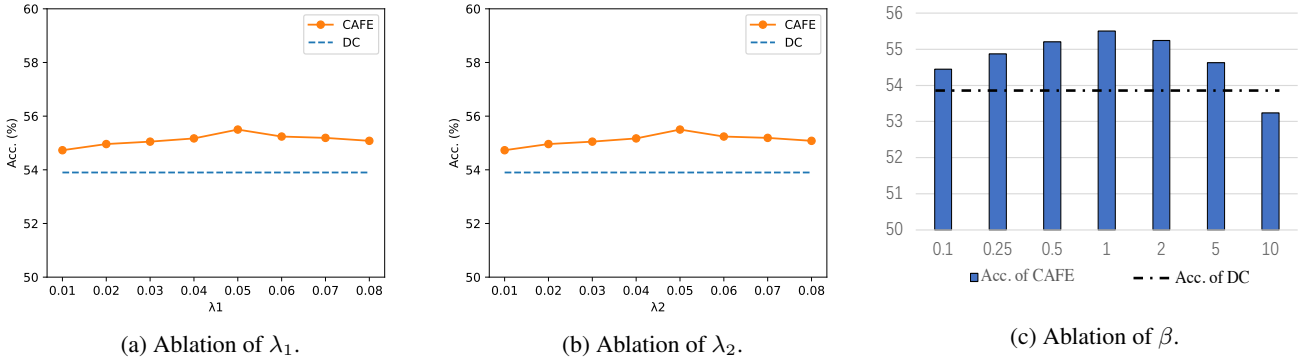
(a) Ablation of $\lambda_1$.  (b) Ablation of $\lambda_2$.  (c) Ablation of $\beta$.

Figure 3: $\lambda_1$ and $\lambda_2$ are the hyper parameters in dynamic bi-level optimization module. $\beta$ is the ratio between $\mathcal{L}_\mathrm{f}$ and $\mathcal{L}_\mathrm{d}$.



(a) Original CIFAR10 images.  (b) The synthetic images using **CAFE**.  (c) The synthetic images using DC.
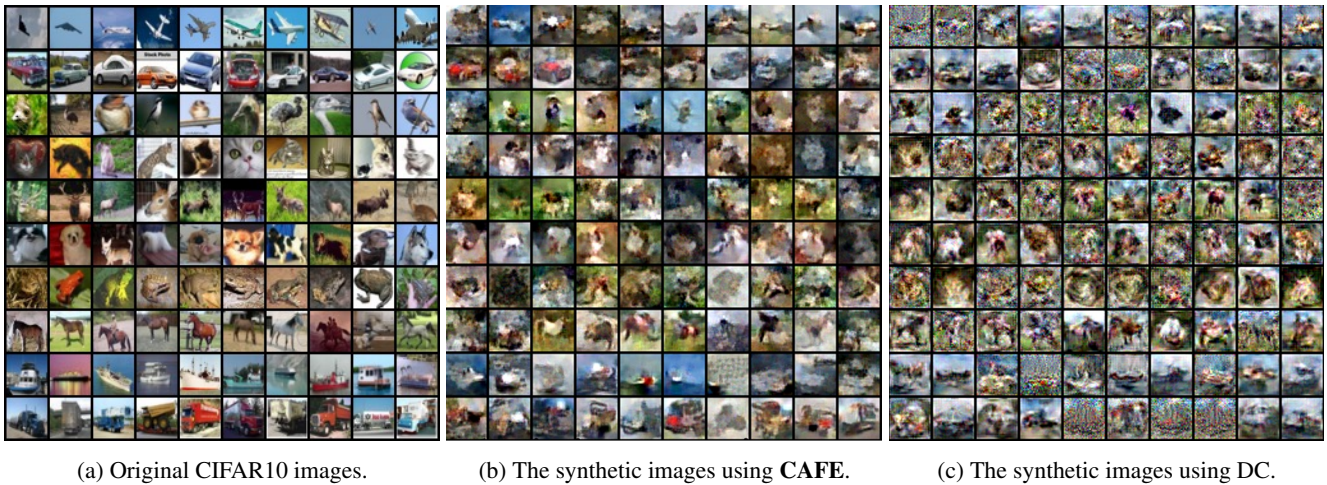
Figure 4: Visualizations of original images, and synthetic images using CAFE and DC. Both CAFE and DC are initialized from random noise.

not very clear, which could not provide enough discriminative features for classification.
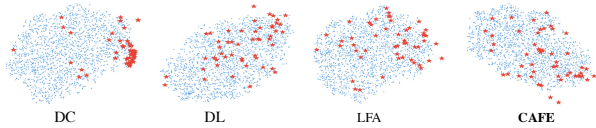


Figure 5: The data distribution of real images and synthetic images learned by DC [54], DL, LFA, and CAFE for one category in CIFAR10.

**Data distribution.**  To evaluate whether the synthetic images using our method can capture more accurate distribution from original dataset, we utilize t-SNE to visualize the features of real set and synthetic sets generated by DC, DL, LFA and CAFE. As shown in Fig. 5, the "points"

and "stars" represent the real and synthetic features. The synthetic images of DC gather around a small area of decision boundary, which indicates using DC can not capture the original distribution well. Our methods DL, LFA, and CAFE effectively capture useful information across the whole real dataset, which possesses a good generalization among different CNN architectures.

## 5. Conclusion

In this work, we propose a novel scheme to Condense dataset by Aligning FEatures (CAFE), which explicitly attempts to preserve the real-feature distribution as well as the discriminant power of the resulting synthetic data, lending itself to strong generalization capability to unseen architectures. The CAFE consists of three carefully designed modules, namely layer-wise feature alignment module, discrimination loss, and dynamic bi-level optimization module. The feature alignment module and discrimination loss con-

cern capturing distribution consistency between synthetic and real sets, while bi-level optimization enables CAFE to learn customized SGD steps to avoid over-/under-fitting. Experimental results across various datasets demonstrate that, CAFE consistently outperforms the state of the art with less computation cost, making it readily applicable to in-the-wild scenarios. As the future work, we plan to explore the use of dataset condensation on more challenging datasets such as ImageNet [8].

## 6. Acknowledge

## References

[1] Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. Approximating extent measures of points. *Journal of the ACM*, 2004. 1, 3

[2] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *NeurIPS*, 2019. 2, 3

[3] Eden Belouadah and Adrian Popescu. Scail: Classifier weights scaling for class incremental learning. In *WACV*, 2020. 5

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

[5] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *ECCV*, 2018. 5

[6] Yutian Chen, Max Welling, and Alex Smola. Super-samples from kernel herding. *UAI*, 2010. 1, 2, 3, 5

[7] Dima Damen, Hazel Doughty, Giovanni Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. The epic-kitchens dataset: Collection, challenges and baselines. *IEEE TPAMI*, 2020. 1

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 1, 9

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 1

[10] Reza Zanjirani Farahani and Masoud Hekmatfar. *Facility location: concepts, models, algorithms and case studies*. Springer Science & Business Media, 2009. 5

[11] Dan Feldman. Introduction to core-sets: an updated survey. *arXiv preprint arXiv:2011.09384*, 2020. 1

[12] Dan Feldman, Matthew Faulkner, and Andreas Krause. Scalable training of mixture models via coresets. In *NeurIPS*, 2011. 1

[13] Dan Feldman, Morteza Monemizadeh, and Christian Sohler. A ptas for k-means clustering based on weak coresets. In *SoCG*, 2007. 1

[14] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering. In *SODA*, 2013. 3

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 3

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 4

[17] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 3

[18] Jeremias Knoblauch, Hisham Husain, and Tom Diethe. Optimal continual learning has perfect memory and is np-hard. In *ICML*, 2020. 1

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

[20] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *IJCV*, 2020. 1

[21] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998. 5

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 1

[23] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 3

[24] Timothy Nguyen, Zhourong Chen, and Jaehoon Lee. Dataset meta-learning from kernel-ridge regression. In *ICLR*, 2021. 2

[25] Xiaojiang Peng, Kai Wang, Zhaoyang Zeng, Qing Li, Jianfei Yang, and Yu Qiao. Suppressing mislabeled data via grouping and self-attention. In *European Conference on Computer Vision*, pages 786–802. Springer, 2020. 1

[26] Xiangyu Peng, Kai Wang, Zheng Zhu, and Yang You. Crafting better contrastive views for siamese representation learning. *arXiv preprint arXiv:2202.03278*, 2022. 1

[27] Esteban Real, Jonathon Shlens, Stefano Mazzocchi, Xin Pan, and Vincent Vanhoucke. Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video. In *CVPR*, 2017. 1

[28] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017. 5

[29] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *ICLR*, 2018. 1, 2, 3, 5

[30] Pierre Sermanet, Soumith Chintala, and Yann LeCun. Convolutional neural networks applied to house numbers digit classification. In *ICPR*, 2012. 5

[31] Samarth Sinha, Han Zhang, Anirudh Goyal, Yoshua Bengio, Hugo Larochelle, and Augustus Odena. Small-gan: Speeding up gan training using core-sets. In *ICML*, 2020. 1

[32] Felipe Petroski Such, Aditya Rawal, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Generative teaching networks: Accelerating neural architecture search by learning to generate synthetic training data. *ICML*, 2020. 2

[33] 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. *ICLR*, 2019. 2, 3, 5

[34] Kai Wang, Xiaojiang Peng, Jianfei Yang, Shijian Lu, and Yu Qiao. Suppressing uncertainties for large-scale facial expression recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6897–6906, 2020. 1

[35] Kai Wang, Xiaojiang Peng, Jianfei Yang, Debin Meng, and Yu Qiao. Region attention networks for pose and occlusion robust facial expression recognition. *IEEE Transactions on Image Processing*, 29:4057–4069, 2020. 1

[36] Kai Wang, Shuo Wang, Jianfei Yang, Xiaobo Wang, Baigui Sun, Hao Li, and Yang You. Mask aware network for masked face recognition in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1456–1461, 2021. 1

[37] Kai Wang, Shuo Wang, Zhipeng Zhou, Xiaobo Wang, Xiaojiang Peng, Baigui Sun, Hao Li, and Yang You. An efficient training approach for very large scale face recognition. *arXiv preprint arXiv:2105.10375*, 2021. 1

[38] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018. 2, 4, 5

[39] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *ICML*, 2015. 3

[40] G W Wolf. Facility location: concepts, models, algorithms and case studies. 2011. 1

[41] 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

[42] Shuo Yang, Lu Liu, and Min Xu. Free lunch for few-shot learning: Distribution calibration. In *ICLR*, 2021. 1

[43] Shuo Yang, Peize Sun, Yi Jiang, Xiaobo Xia, Ruiheng Zhang, Zehuan Yuan, Changhu Wang, Ping Luo, and Min Xu. Objects in semantic topology. *arXiv preprint arXiv:2110.02687*, 2021. 1

[44] Shuo Yang, Songhua Wu, Tongliang Liu, and Min Xu. Bridging the gap between few-shot and many-shot learning via distribution calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 1

[45] Shuo Yang, Min Xu, Haozhe Xie, Stuart Perry, and Jiahao Xia. Single-view 3d object reconstruction from shape priors in memory. In *CVPR*, 2021. 1

[46] Shuo Yang, Wei Yu, Ying Zheng, Hongxun Yao, and Tao Mei. Adaptive semantic-visual tree for hierarchical embeddings. In *ACM MM*, 2019. 1

[47] 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

[48] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017. 2

[49] David Junhao Zhang, Kunchang Li, Yunpeng Chen, Yali Wang, Shashwat Chandra, Yu Qiao, Luoqi Liu, and Mike Zheng Shou. Morphmlp: A self-attention free, mlp-like backbone for image and video. *arXiv preprint arXiv:2111.12527*, 2021. 1

[50] Junhao Zhang, Yali Wang, Zhipeng Zhou, Tianyu Luan, Zhe Wang, and Yu Qiao. Learning dynamical human-joint affinity for 3d pose estimation in videos. *IEEE Transactions on Image Processing*, 30:7914–7925, 2021. 1

[51] Ruiheng Zhang, Shuo Yang, Qi Zhang, Lixin Xu, Yang He, and Fan Zhang. Graph-based few-shot learning with transformed feature propagation and optimal class allocation. *Neurocomputing*, 2022. 1

[52] Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. In *ICML*, 2021. 2, 3, 4, 5

[53] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. *arXiv preprint arXiv:2110.04181*, 2021. 2

[54] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *ICLR*, 2021. 1, 2, 3, 4, 5, 8

10