Slimmable Dataset Condensation

Songhua Liu, Jingwen Ye, Runpeng Yu, Xinchao Wang*
National University of Singapore
songhua.liu@u.nus.edu, jingweny@nus.edu.sg, r.yu@u.nus.edu, xinchao@nus.edu.sg

Abstract

Dataset distillation, also known as dataset condensation, aims to compress a large dataset into a compact synthetic one. Existing methods perform dataset condensation by assuming a fixed storage or transmission budget. When the budget changes, however, they have to repeat the synthesizing process with access to original datasets, which is highly cumbersome if not infeasible at all. In this paper, we explore the problem of slimmable dataset condensation, to extract a smaller synthetic dataset given only previous condensation results. We first study the limitations of existing dataset condensation algorithms on such a successive compression setting and identify two key factors: (1) the inconsistency of neural networks over different compression times and (2) the underdetermined solution space for synthetic data. Accordingly, we propose a novel training objective for slimmable dataset condensation to explicitly account for both factors. Moreover, synthetic datasets in our method adopt a significance-aware parameterization. Theoretical derivation indicates that an upper-bounded error can be achieved by discarding the minor components without training. Alternatively, if training is allowed, this strategy can serve as a strong initialization that enables a fast convergence. Extensive comparisons and ablations demonstrate the superiority of the proposed solution over existing methods on multiple benchmarks.

1. Introductions

The success of deep learning is largely attributed to the enormous amount of training data [5,8,12,15,21,36,37,41,50]. However, the massive data not only inevitably introduces heavy burdens on storage and transmission but also incommodes many applications that require training over datasets multiple times, such as hyper-parameter optimization [3, 9, 16, 29, 30] and neural architecture search [10, 24, 43, 49]. Moreover, it raises concerns on privacy and copyright if raw datasets are published and accessed directly [7, 40, 51]. These issues can be largely alleviated by using smaller datasets containing only a few synthetic samples but with performance similar to the original ones. How to compress a given real dataset into such a synthetic dataset is the main focus of *dataset distillation*, also known as *dataset condensation* (DC), whose concept is introduced by Wang *et al.* [45] and further developed by a series of following works recently [2,17,28,33,34,44,54,55,57,58].

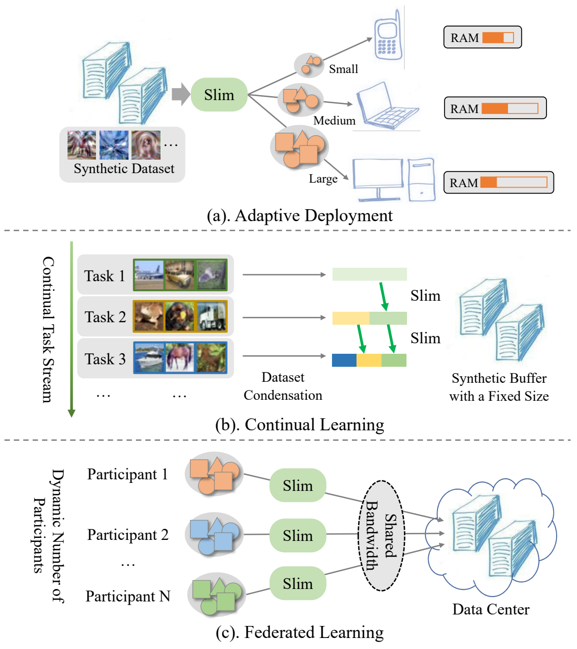

Figure 1. Scenarios where slimmable dataset condensation is useful: (a) adapting to devices with different storage budgets, (b) continual learning using a synthetic buffer with a fixed size, and (c) federated learning with synthetic data where a dynamic number of participants share the network bandwidth.

Specifically, existing DC approaches work under a predefined storage budget, *e.g.*, the number of images per class. Although it has been demonstrated that most performance of original datasets can be recovered by the synthetic ones with only a few synthetic samples, the fixed storage budget does not take the variations of the storage budget into consideration. Some examples are shown in Fig. 1. On the one hand, different devices may have different storage and transmission resources. On the other hand, in applications

*Corresponding Author.

like continual learning [4,31,32,38,39,46,52] and federated learning [11, 13, 19, 25, 26, 42, 48, 59], the storage and transmission budgets may change on different occasions, since a replay buffer with a static memory size needs to take account for more and more historical data, and the bandwidth allocated to each participant is smaller with the increasing number of participants, respectively. In these scenarios where it is necessary to adapt to different capacities on storage and transmission, or the budget is changed, existing algorithms have to repeat the synthesizing process for the newly-defined budget with access to original datasets, which is largely cumbersome if not infeasible at all due to the lack of original data.

In this paper, we phrase the task of re-condensing a synthetic dataset, derived from dataset distillation per se, as *slimmable dataset condensation* (slimmable DC). In fact, it even remains unclear whether a valid synthetic dataset can be re-condensed from *only* previously condensed samples. Unfortunately, we find that the answer is negative for existing state-of-the-art methods [28, 33, 34, 58]. The basic idea of these methods is to optimize the validation error on real datasets for models trained by synthetic ones. Although the solution is effective for the original DC setting, it is not the case for slimmable DC. Specifically, we reveal that since the synthetic data for re-condensation are much less than the original ones, existing methods suffer from two main issues: (1) the performance is sensitive to the inconsistency of neural networks adopted on different occasions of compression, and (2) solution spaces for re-condensed datasets become underdetermined, which triggers deviations in training results and further leads to inferior performances.

To address these drawbacks, we propose to explicitly regulate the consistency between the training effects using synthetic datasets before and after a condensation step for slimmable DC. Specifically, the proposed objective is composed of two terms: first-order and infinity-order parameter matching, which are designed to explicitly account for the two aforementioned issues. The former encourages a unified embedding space over different training iterations, while the latter enforces the consistency of final training parameters in such a space. Optimized with the proposed objective function, we achieve favorable results for slimmable DC: the performance of a further condensed dataset from a previously condensed one effectively approaches that obtained with access to the real dataset.

Moreover, for an efficient slimming procedure, we explore a significance-aware synthetic dataset parameterization, which explicitly embeds a linear space with orthogonal bases and askew-distributed singular values during training. Theoretical derivation indicates an upper-bounded error by discarding the minor components, *i.e.*, bases with the smallest singular values. This strategy may serve as either a learning-free slimmable DC solution or a strong initialization in learning-based settings to accelerate convergence.

We conduct extensive experiments on multiple benchmarks and applications, including continual learning and federated learning, and demonstrate the effectiveness of the proposed solution. Results suggest that our method outperforms all state-of-the-art baselines by a large margin on slimmable DC. Our contributions are summarized below:

- We introduce the task of slimmable dataset condensation beyond the typical DC setting, which alleviates the dilemma of existing DC methods when the budget changes for storage or transmission.
- We delve into the limitations of existing algorithms for typical DC and propose a novel first-order and infinity-order matching-based training objective pertinently for slimmable DC.
- We propose a significance-aware synthetic dataset parameterization with a theoretical guarantee for learning-free slimmable DC or initialization to accelerate convergence in learning-based settings.
- Experiments on multiple benchmarks and applications demonstrate the effectiveness of the proposed method and its superiority over state-of-the-art baselines.

2. Related Works

The target of dataset distillation, or dataset condensation, is to synthesize a much smaller dataset whose training performance on deep neural networks can be comparable with the original one. The seminal work by Wang *et al.* [45] propose a bi-level learning framework, to minimize the loss on real datasets for networks trained by synthetic ones, which is known as the *performance matching* objective [53]. Since the GPU memory required for the bi-level optimization increases proportionally with the number of inner updates, this method can only take training effects for a few steps with synthetic datasets into consideration, which bottlenecks its performance. Works in recent years leverage linear regression, by either approximating neural networks as linear models [33, 34] or mapping samples to a linear embedding space with non-linear neural networks [28, 58]. Due to the analytical optimal solution for linear regression, these methods can access the optimal parameters trained by synthetic datasets without computing higher-order gradients, which achieves state-of-the-art DC performance.

On another line, Zhao *et al.* [57] propose the *gradient matching* objective for DC, to overcome the inconvenience of higher-order gradients in the seminal work [45], given that gradients in neural networks can reflect the training effect of adopted datasets, whose performance is further improved by a lot of following works [14, 17, 23, 54]. In particular, Cazenavette *et al.* [2] propose to match training trajectories between synthetic datasets and real ones, which can be viewed as a more general gradient matching framework considering higher-order gradients.

Some works adopt the *distribution matching* objective [44, 55]. Although they often achieve inferior performance, these methods enjoy overall fast optimization and low GPU memory consumption, which makes them scale up for larger synthetic datasets efficiently.

In this paper, we aim to study whether existing DC approaches are suitable for slimmable DC, *i.e.*, a successive condensation fashion. Thanks to the superior baseline performance of the performance matching objective, the analysis mainly focuses on works in this direction. We find that although they achieve state-of-the-art results for typical DC, the performance would drop significantly during further condensation compared to retraining a synthetic set with the same size using original datasets. We will also discuss the performance of other methods in the experiments.

There are also a series of works focusing on *synthetic data parameterization* [6, 17, 22, 27, 56]. The essential idea lies in that synthetic samples are stored in more parameter-efficient ways other than formats of raw samples. In this paper, the way of storing different components along with their significance scores can also be viewed as an alternative parameterization. However, our main focus is on slimmable DC, while the emphasis of other works is on how to incorporate as many samples as possible given a fixed storage budget, which is dramatically different.

3. Methods

This section introduces the proposed approach for slimmable dataset condensation, *i.e.*, how to extract a smaller synthetic dataset from only a previously synthesized one for a real target dataset.

3.1. Preliminary of Typical DC

Let $\mathcal{T}=(X_t,Y_t)$, $X_t\in\mathbb{R}^{n_t\times d}$ and $Y_t\in\mathbb{R}^{n_t\times c}$, denote a target real dataset, where X_t denotes samples, Y_t denotes the corresponding labels, n_t is the number of samples in the dataset, and each sample has d dimensions and c label entries. For example, for RGB image classification tasks, $d=h\times w\times 3$ and c is the number of classes. Dataset condensation aims at a synthetic dataset $\mathcal{S}=(X_s,Y_s)$, $X_s\in\mathbb{R}^{n_s\times d}$ and $Y_s\in\mathbb{R}^{n_s\times c}$, where $n_s\ll n_t$ is the size of the synthetic dataset $\mathcal{S}$.

A typical optimization objective for $\mathcal{S}$ is to minimize the loss function on $\mathcal{T}$ for deep neural networks trained by $\mathcal{S}$, which is known as the performance matching objective. To this end, recent approaches [28, 58] first embed all samples with a pool of non-linear neural networks and then consider the linear regression problem in embedding spaces, which achieves state-of-the-art DC performance and ensures the efficiency simultaneously. Formally, let $X^\theta\in\mathbb{R}^{n\times f}$ denote the output embedding of a neural network parameterized by θ taking X as input, where f is the embedding dimension, and the parameter θ is drawn from a distribution Θ. The objective can be written as:

$$\min_{X_s,Y_s}\mathbb{E}_{\theta\sim\Theta_s}[\|X_t^\theta X_s^{\theta\dagger}Y_s-Y_t\|_F^2],\tag{1}$$

where $X_s^{\theta\dagger}Y_s$ is the optimal linear regression parameter with respect to (X_s^θ,Y_s). Since the number of synthetic samples is typically less than the dimension of feature embedding, the pseudo inverse $X_s^{\theta\dagger}$ should adopt the form of $X_s^{\theta\top}(X_s^\theta X_s^{\theta\top})^{-1}$. For simplicity, we omit the regularization term λI added before the matrix inversion for numerical stability since the weight λ is typically a small constant in practice and makes little difference to the final result.

Θ_s in Eq. 1 denotes a pool of neural networks trained by synthetic datasets $\mathcal{S}$. In practice, Θ_s and $\mathcal{S}$ are trained alternately [58]. Specifically, in each iteration, a network is sampled from the pool, and the synthetic dataset is updated with Eq. 1 while the network is trained with the currently synthetic dataset for one step.

3.2. Drawbacks in Slimmable DC

For slimmable DC, only given a previously synthesized dataset $\mathcal{S}=(X_s,Y_s)$, we aim at a even smaller synthetic dataset $\mathcal{S}'=(X_s',Y_s')$, which is expected to hold the optimality of Eq. 1 when (X_s,Y_s) is substituted with (X_s',Y_s'). In formal, using the same paradigm for further condensation is to optimize the following objective:

$$\min_{X_s',Y_s'}\mathbb{E}_{\theta\sim\Theta_s'}[\|X_s^\theta X_s'^{\theta\dagger}Y_s'-Y_s\|_F^2],\tag{2}$$

which is expected to achieve:

$$\min_{X_s',Y_s'}\mathbb{E}_{\theta\sim\Theta_s}[\|X_t^\theta X_s'^{\theta\dagger}Y_s'-Y_t\|_F^2].\tag{3}$$

Comparing Eqs. 2 and 3, we find that for further condensation in slimmable DC, the distribution of neural networks is desired to be consistent with that used in the first condensation with real data and that the loss on unseen real data is desired to be minimized for linear regression in embedding spaces of these neural networks. Unfortunately, simply adopting original methods for further compression can meet neither goal through the following analysis.

Inconsistent Embedding Space: Analyzed from Eqs. 2 and 3, the adopted neural network space is trained with $\mathcal{S}'$ while the desired space is with $\mathcal{S}$. Given that the pool of neural networks is not allowed to be stored for future use, we find that the performance is sensitive to such inconsistency without an explicit regulation in existing approaches[1]. We use linear embedding for illustration. Assume that (X_s,Y_s) minimizes the performance matching

[1]Since $\mathcal{S}$ is available, we can recover the neural network pool with $\mathcal{S}$ instead of using $\mathcal{S}'$ as the original version does. However, this operation does not support multiple-time slimmable condensation where $\mathcal{S}$ in the first condensation becomes unavailable.

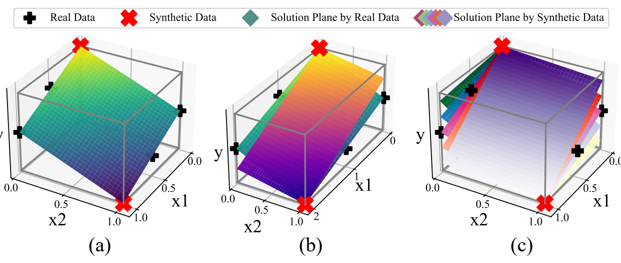

Figure 2. Limitations of directly using existing DC approaches on slimmable DC. (a) The solution plane by synthetic data can perfectly minimize the error on real data in the original linear space. (b) Projecting the original space to another, *e.g.*, 2× scaling up for axis x1 in this case, the solution plane by synthetic data shifts from the optimal one. (c) If current synthetic data are used for further condensation, the solution plane becomes underdetermined: any plane passing through the two synthetic points can minimize the error on current synthetic data, but cannot for real data.

loss for linear regression under a linear embedding function parameterized by θ, *i.e.*, $\|X_t^\theta {X_s^\theta}^\dagger Y_s - Y_t\|_F^2 = 0$, and $P \in \mathbb{R}^{f\times f}$ is a projection matrix transforming this linear space to another. The performance matching loss $\mathcal{L}_{pm}$ in the new embedding space is then:

$$\begin{aligned}\mathcal{L}_{pm} &= \|X_t^\theta P (X_s^\theta P)^\dagger Y_s - Y_t\|_F^2 \\ &= \|X_t^\theta P (X_s^\theta P)^\top (X_s^\theta P P^\top {X_s^\theta}^\top)^{-1} Y_s - Y_t\|_F^2 \\ &= \|X_t^\theta P P^\top ({X_s^\theta}^\dagger X_s^\theta)^\top (PP^\top)^{-1} {X_s^\theta}^\dagger Y_s - Y_t\|_F^2.\end{aligned} \tag{4}$$

Note that ${X_s^\theta}^\dagger X_s^\theta$ cannot be canceled out since $n_s < f$ typically, and thus the loss can be amplified by inconsistent embedding functions, which negatively affects slimmable DC. An illustrative example is shown in Fig. 2(b).

Underdetermined Solution Space: Even if the neural networks for embedding, Θ'_s and Θ_s, are the same in Eqs. 2 and 3, we find that solutions by Eq. 2 cannot guarantee the optimality of Eq. 3 due to the underdetermined solution space. Specifically, the performance matching objective in Eq. 2 enforces the parameters of linear regression with the new synthetic dataset to minimize the error on the previous one. Note that for the previously synthesized dataset, the number of samples n_s is less than the embedding dimension f. Thus, there are multiple, or an infinite number of parameters as optimal solutions of linear regression in slimmable DC, but most of them cannot minimize the error on original real datasets. In other words, the resultant (X'_s, Y'_s) by optimizing Eq. 2 also becomes underdetermined and fails to achieve the goal in Eq. 3. Fig. 2(c) provides a visualized example demonstrating this effect.

3.3. Parameter Matching for Slimmable DC

Based on the above analysis, inconsistent embedding space and underdetermined solution space are the two major obstacles to existing state-of-the-art DC approaches in the setting of slimmable DC. Accordingly, in this paper, we propose first-order and infinity-order parameter-matching objectives specifically for slimmable DC.

First-Order Parameter Matching: The intuitive target of first-order parameter matching is to enforce a consistent neural network pool Θ_s in Eq. 1. Given that neural networks typically adopts gradient-decent-based solutions like Adam [18], we consider the first-order gradient and encourage the consistency between parameters of a neural network trained by previously condensed data and targeting further condensed data in each iteration. This workflow is similar to the gradient matching objective [57]. However, in this paper, we empirically find that it is sufficient to only take gradients of the final linear layer into consideration, which has an analytical form and can be computed more efficiently without the necessity of second-order derivative computation. Specifically, given the parameters of the final linear layer w^θ, the embedding before this layer X^θ, and the corresponding label Y, the prediction, training loss, and gradient with respect to the parameters can be computed by:

$$\begin{aligned}\hat{Y} &= X^\theta w^\theta, \\ \mathcal{L}_{train} &= \frac{1}{2}\|\hat{Y} - Y\|_F^2, \\ \frac{\partial \mathcal{L}_{train}}{\partial w} &= {X^\theta}^\top (\hat{Y} - Y) = {X^\theta}^\top (X^\theta w^\theta - Y).\end{aligned} \tag{5}$$

The parameters are updated by a gradient decent step: $w^1 \leftarrow w - \eta \frac{\partial \mathcal{L}_{train}}{\partial w}$, where η is the learning rate. The first-order parameter matching loss with respective to the further condensed data (X'_s, Y'_s) in slimmable DC is:

$$\begin{aligned}\mathcal{L}_{pm}^1 &= \|w_s'^1 - w_s^1\|_F^2 \\ &= \|{X_s'^\theta}^\top (X_s'^\theta w^\theta - Y'_s) - {X_s^\theta}^\top (X_s^\theta w^\theta - Y_s)\|_F^2.\end{aligned} \tag{6}$$

Infinity-Order Parameter Matching: The analysis in Sec. 3.2 and the phenomenon in Fig. 2(c) indicate that it is important to maintain the solution plane solved with original real data during different times of condensation. To this end, we explicitly enforce the consistency of optimal linear regression parameters using their analytical form:

$$\begin{aligned}\mathcal{L}_{pm}^\infty &= \|{X_s'^\theta}^\dagger Y'_s - {X_s^\theta}^\dagger Y_s\|_F^2 \\ &= \|X_s'^{\theta\top} (X_s'^\theta X_s'^{\theta\top})^{-1} Y'_s - X_s^{\theta\top} (X_s^\theta X_s^{\theta\top})^{-1} Y_s\|_F^2.\end{aligned} \tag{7}$$

Since linear regression is a convex-optimization problem and the solution would converge to the optimal analytical one by sufficient gradient decent steps, we term this objective as infinity-order parameter matching denoted as $\mathcal{L}_{pm}^\infty$.

Interestingly, if we pre-multiply the content in the expectation of Eq. 1 by ${X_t^\theta}^\dagger$, given that $n_t > f$ for real datasets, ${X_t^\theta}^\dagger X_t^\theta$ would result in an identity matrix and we derive $\|{X_s^\theta}^\dagger Y_s - {X_t^\theta}^\dagger Y_t\|_F^2$, which indicates a theoretical equivalency between performance matching and our infinity-order parameter matching objectives in the special case. However, for further condensation using only (X_s, Y_s), ${X_s^\theta}^\dagger X_s^\theta$

cannot be canceled out since $n_s < f$, the performance matching objective fails to maintain the optimal parameters, which makes it inapplicable for slimmable DC.

3.4. Significance-Aware Parameterization

In slimmable DC, it is expected that smaller synthetic datasets could be established as efficiently as possible, with only a few or even no training steps required. To this end, we explore a significance-aware parameterization for synthetic datasets. Motivated by singular value decomposition (SVD), in this paper, we propose to learn a joint parameterization of different components and their corresponding singular values, which represent their contribution to the whole synthetic dataset. In specific, a synthetic dataset (X_s, Y_s) is parameterized by (U, Σ, V_x, V_y): $V_x \in \mathbb{R}^{b\times d}$ and $V_y \in \mathbb{R}^{b\times c}$ denote orthogonal bases for constructing samples and labels respectively, where b is the total number of components; $\Sigma = \mathrm{diag}(s_1, \dots, s_b)$ with $s_1 \geq \cdots \geq s_b$ is a diagonal matrix, where each s_i denotes the significance of the i-th component; and $U \in \mathbb{R}^{n_s\times b}$ is an orthogonal matrix representing coefficients of different components for constructing each data. In this way, the synthetic samples and corresponding labels are constructed by:

$$X_s = U\Sigma V_x, \quad Y_s = U\Sigma V_y. \tag{8}$$

One significant benefit of such a significance-aware parameterization is that it is possible to simply discard less important components when we need to slim a synthetic dataset, *i.e.*, deleting the entries with least singular values in Σ, the corresponding columns in U, and the corresponding rows in V_x and V_y, which has the potential to serve as a learning-free slimmable DC strategy. Alternatively, if learning is allowed, it can also become a favorable initialization for the slimmed datasets, which encourages fast convergence in only a few training steps. Theoretically, in the case of linear regression, the error on the resultant solution plane satisfies the following proposition:

Proposition 1 *In linear regression, if a synthetic dataset (X_s, Y_s) takes the parameterization in Eq. 8, and rows in V_x and V_y corresponding to the least singular values in Σ, denoted as $\tilde{V_x}$ and $\tilde{V_y}$, are removed for slimmable DC, the first-order parameter distance between parameters before and after slimming is bounded by:*

$$\|w_s'^1 - w_s^1\|_2^2 \leq s_2^2 \|X_s w - Y_s\|_2^2, \tag{9}$$

and the infinity-order parameter distance is bounded by:

$$\|X_s'^{\theta\dagger} Y_s' - X_s^{\theta\dagger} Y_s\|_2^2 = \|\tilde{V_x}^\top \tilde{V_y}\|_2^2 \leq 1. \tag{10}$$

The proof can be found in the supplement.

The bound in Eq. 9 indicates that maximizing the contribution of the first component and minimizing others are of great importance for maintaining the solution plane in slimmable DC. Given that we choose components with the least singular values for deletion, a skewed distribution for all singular values would result in a smaller parameter distance compared with a uniform one. Thus, we add the following objective $\mathcal{L}_{skew}$ on skewness:

$$\mathcal{L}_{skew} = 1 - \frac{s_1^2}{\sum_{i=1}^b s_i^2}. \tag{11}$$

In this paper, U, V_x, and V_y are orthogonal matrices. The orthogonality of U is enforced by conducting SVD to a learnable matrix. For V_x and V_y, we add the following regularization to encourage their orthogonality:

$$\mathcal{L}_{ortho} = \|V_x V_x^\top - I\|_F^2 + \|V_y^\top V_y - I\|_F^2, \tag{12}$$

given that the number of bases is smaller than the feature dimension and larger than the number of label entries.

Algorithm 1 Slimmable Dataset Condensation.

Input: V_x, V_y, Σ, and U: parameterization of a synthetic dataset by Eq. 8; K: the number of components for the slimmed dataset; T: the number of allowed training steps; T': the number of maximal updated steps for each neural network; M: the size of the neural network pool.

Output: V_x', V_y', Σ', and U': parameterization of the slimmed dataset.

```
1:  Initialize $V_x'$, $V_y'$, $\Sigma'$, and $U'$ by deleting components
    with significance scores less than the $K$-th largest one;
2:  Initialize the neural network pool $\Theta$;
3:  Compute $X_s = U\Sigma V_x$ and $Y_s = U\Sigma V_y$;
4:  for $1 \leq i \leq T$ do
5:      Compute $X_s' = U'\Sigma' V_x'$ and $Y_s' = U'\Sigma' V_y'$;
6:      Sample a neural network with parameter $\theta$ from $\Theta$;
7:      Compute embedding $X_s'^\theta$ and $X_s^\theta$ for $X_s'$ and $X_s$;
8:      Gradient decent with Eq. 13 for $V_x'$, $V_y'$, $\Sigma'$, and $U'$;
9:      Update $\theta$ with $(X_s', Y_s')$;
10:     if $\theta$ has been updated $T'$ times then
11:         Reinitialize $\theta$;
12:     end if
13: end for
```

3.5. Overall Pipeline

Overall, given a previously synthesized dataset S organized by Eq. 8 and a smaller storage budget K, we initially discard components with singular values less than the K-th largest one to fit the budget and then perform optimization for a limited number of step T, with a weighted combination of objectives in Eqs. 6, 7, 11, and 12:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{pm}^1 + \lambda_\infty \mathcal{L}_{pm}^\infty + \lambda_{skew}\mathcal{L}_{skew} + \lambda_{ortho}\mathcal{L}_{ortho}, \tag{13}$$

where each λ is the hyper-parameter controlling the weight of the corresponding term. Alg. 1 summarizes the pipeline.

Dataset		FashionMNIST						CIFAR10						CIFAR100				
	IPC	50	20	10	5	2	1	50	20	10	5	2	1	20	10	5	2	1
	RS	81.77	77.88	76.02	72.47	66.41	55.32	55.10	44.97	38.17	32.71	21.88	20.95	23.23	17.74	13.97	8.41	6.51
DC [57]	RT	82.89	84.37	83.38	80.54	76.07	70.27	53.43	49.73	43.74	39.88	38.89	28.20	28.93	25.08	21.29	16.46	12.44
	LBS	-	79.60	75.42	67.92	61.34	57.35	-	45.59	39.83	35.45	29.65	23.00	-	21.28	17.07	12.63	9.34
	Gap↓	-	4.77	7.96	12.62	14.73	12.92	-	4.14	3.91	4.43	9.24	5.20	-	3.80	4.22	3.83	3.10
	LFS	-	72.25	70.22	56.39	54.79	34.84	-	41.82	32.66	25.88	18.76	17.37	-	21.78	13.30	6.74	4.64
DSA [54]	RT	88.73	86.68	85.27	81.99	76.66	70.33	60.58	57.11	52.15	47.31	34.23	28.10	36.35	32.49	27.35	20.47	13.81
	LBS	-	86.08	83.32	79.17	70.29	51.58	-	52.77	46.55	39.51	30.07	20.48	-	29.89	24.34	17.61	11.62
	Gap↓	-	0.60	1.95	2.82	6.37	18.75	-	4.34	5.60	7.80	4.16	7.62	-	2.60	3.01	2.86	2.19
	LFS	-	79.86	74.14	71.27	54.63	43.81	-	41.54	29.29	27.56	20.10	14.05	-	23.69	14.92	8.06	4.95
DM [55]	RT	88.20	86.21	83.84	80.89	74.42	71.45	62.94	55.41	48.80	42.89	33.50	27.08	34.39	29.33	23.91	15.98	11.51
	LBS	-	85.92	83.21	80.21	73.78	70.69	-	56.47	49.89	43.57	34.35	26.67	-	30.84	24.74	16.47	11.62
	Gap↓	-	0.29	0.63	0.68	0.64	0.76	-	-1.06	-1.09	-0.68	-0.85	0.41	-	-1.51	-0.83	-0.49	-0.11
	LFS	-	81.05	**78.56**	68.04	59.22	58.48	-	46.76	35.35	25.34	16.05	13.81	-	26.72	15.69	7.95	5.22
IDC [17]	RT	89.06	86.81	85.16	83.13	77.96	70.64	69.32	62.01	58.50	52.13	44.12	35.34	41.99	36.08	30.68	23.34	17.93
	LBS	-	84.81	83.36	81.16	76.52	67.73	-	58.77	54.24	47.83	38.61	29.16	-	35.16	28.29	18.39	13.40
	Gap↓	-	2.00	1.80	1.97	1.44	2.91	-	3.24	4.26	4.30	5.51	6.18	-	0.92	2.39	4.95	4.53
	LFS	-	82.57	77.02	74.41	60.86	52.75	-	51.91	42.17	30.20	22.84	17.68	-	30.25	19.50	10.96	7.63
FRePo [58]	RT	**89.15**	87.44	85.54	**83.80**	**79.91**	**75.44**	**71.03**	**68.63**	**65.76**	**61.07**	**53.24**	43.24	40.57	39.97	36.34	31.63	**27.07**
	LBS	-	86.60	81.53	67.74	33.44	29.24	-	65.64	53.76	38.02	17.31	11.01	-	35.53	32.08	26.51	19.27
	Gap↓	-	0.84	4.01	16.06	46.47	46.20	-	2.99	12.00	23.05	35.93	32.23	-	4.44	4.26	5.12	7.80
	LFS	-	82.59	75.65	71.76	61.94	44.00	-	59.14	**50.48**	38.34	29.60	18.22	-	35.18	**30.00**	19.94	13.63
Ours	RT	88.68	**87.50**	**86.65**	83.54	79.63	74.14	70.33	67.60	64.57	59.49	52.88	**43.56**	**42.47**	**40.29**	**36.42**	**32.28**	26.75
	LBS	-	**86.81**	**85.18**	**83.62**	**78.58**	**72.74**	-	**67.93**	**63.96**	**61.05**	**55.82**	**47.77**	-	**36.23**	**33.49**	**29.27**	**26.04**
	Gap↓	-	0.69	1.47	-0.08	1.05	1.40	-	-0.33	0.61	-1.56	-2.94	-4.21	-	4.06	2.93	3.01	0.71
	LFS	-	**82.96**	76.71	**74.72**	**69.52**	**66.43**	-	**62.05**	48.89	**40.48**	**36.51**	**33.09**	-	**35.39**	28.58	**23.69**	**20.34**

Table 1. Comparisons with typical DC methods on the performance of slimmable DC. IPC: number of images per class. RT: retraining using original datasets. LBS: learning-based slimming. LFS: learning-free slimming. RS: randomly selected real images.

4. Experiments

4.1. Settings and Implementation Details

To evaluate the performance of our method in slimmable DC, we mainly consider the following successive compression setting in experiments. A given real dataset is first condensed to a synthetic one with relatively large size. Then, it is shrunk to a smaller set step by step where given only the set before the current step. We consider two fashions: learning-based and learning-free slimmable DC in this paper, where the former allows using a limited number of optimization steps to slim a dataset, while the latter does not. The evaluation metric consists of two parts: the absolute accuracy for downstream models trained by slimmed datasets and the performance gap with an equal-size synthetic dataset condensed directly from the real dataset.

Following previous works on DC, we conduct experiments on benchmarked datasets including FashionMNIST [47], CIFAR10, and CIFAR100 [20]. The number of classes is 10, 10, and 100, and the total number of images is 70,000, 60,000, and 60,000 respectively. For FashionMNIST and CIFAR10, sizes of synthetic datasets are $50 \rightarrow 20 \rightarrow 10 \rightarrow 5 \rightarrow 2 \rightarrow 1$ images per class (IPC) for successive compression, and for CIFAR100, the sequence is $20 \rightarrow 10 \rightarrow 5 \rightarrow 2 \rightarrow 1$. Experiments on more datasets and settings can be found in the supplement.

We start the implementation of the main pipeline of slimmable dataset condensation in Alg. 1 following the JAX [1] framework of FRePo [58], which is a performance-matching based dataset condensation solution and achieves state-of-the-art performance on typical DC settings. The hyper-parameters λ_1, λ_∞, λ_{skew}, and λ_{ortho} are empirically set as $1e-2$, 1, $1e-2$, and $1e-4$ respectively by default and the sensitivity analysis is conducted in Sec. 4.3. The training iteration T is $30,000$. For the significance-aware parameterization, samples of each class share the same U and Σ for memory efficiency and we simply set $n_s = b$ for $U \in \mathbb{R}^{n_s \times b}$ in each class and initialize it with an identity matrix when training with real datasets. Other configurations like network architectures for training and evaluation hold the same as FRePo if not specified. All quantitative results are based on the average of 5 repeated evaluations. Full results can be found in the supplement.

4.2. Comparisons with Typical DC Methods

In this section, we are interested in the question that whether existing algorithms for typical DC settings support slimmable condensation, *i.e.*, taking a previously compressed dataset as the input for further condensation and producing a slimmed set representative for the original real one. Thus, we adopt all three categories of DC approaches as candidates, including three gradient-matching-based methods: DC [57], DSA [54], and IDC [17], one distribution-matching based method: DM [55], and one performance-matching based method: FRePo [58]. Their performances on slimmable DC settings mentioned above are listed in Tab. 1. We can figure out that: (1) **Gradient-matching-based methods** in slimmable DC demonstrate

	IPC	50	20	10	5	2	1
ResNet	FRePo-RT	62.11	58.07	51.71	44.69	34.65	25.28
	FRePo-LFS	-	53.22	39.34	23.97	14.09	11.60
	Gap↓	-	4.85	12.37	20.72	20.56	13.68
	Ours-RT	62.05	55.76	49.64	43.26	34.33	24.63
	Ours-LFS	-	56.24	49.41	43.73	32.74	25.56
	Gap↓	-	-0.48	0.23	-0.47	1.59	-0.93
AlexNet	FRePo-RT	68.11	64.41	59.68	55.27	46.78	38.03
	FRePo-LFS	-	59.65	44.03	29.24	14.85	11.76
	Gap↓	-	4.76	15.65	26.03	31.93	26.27
	Ours-RT	68.67	63.37	58.57	52.85	47.55	37.13
	Ours-LFS	-	63.91	57.33	53.74	45.27	36.27
	Gap↓	-	-0.54	1.24	-0.89	2.28	0.86
VGG	FRePo-RT	60.89	54.30	46.71	42.22	34.02	28.50
	FRePo-LFS	-	50.72	34.79	23.11	12.93	10.69
	Gap↓	-	3.58	11.92	19.11	21.09	17.81
	Ours-RT	60.27	52.35	44.52	38.16	33.75	28.92
	Ours-LFS	-	54.78	43.78	37.43	32.31	28.02
	Gap↓	-	-2.43	0.74	0.73	1.44	0.90

Table 2. Comparisons with the performance-matching baseline FRePo [58] on cross-architecture performance on slimmable DC.

a moderate amount of performance dropping compared with the benchmarked result of retraining using the original dataset. The possible reason is that only single-step gradient matching is insufficient to preserve consistent neural networks used in different compression times. (2) **Distribution-matching-based DM** can recover or even surpass its benchmark, which is not surprising given the analysis in Sec. 3.2. It adopts random neural networks without any learning bias and thus ensures a consistent embedding space. The mean vector in this space is also readily preserved during multi-time compression. Nevertheless, its inferior absolute accuracy makes it a less satisfactory slimmable DC algorithm. (3) **Performance-matching based FRePo** is incompetent for slimmable DC with dramatic performance drop from the benchmark based on the analysis in Sec. 3.2. (4) The proposed method based on first-order and infinity-order parameter matching accounts for the above factors and performs well on both absolute accuracy and performance gap with the benchmark[2].

We also report results of our learning-free slimmable DC based on the significance-aware parameterization in Tab. 1, where results of other methods are based on a random selection of synthetic samples. We find that our method achieves significantly superior performance for relatively small storage budgets and at least comparable results in general.

Moreover, Tab. 2 shows the comparisons with the performance-matching based FRePo on the cross-architecture performance of slimmable DC on CIFAR10, with three different architectures: ResNet-18 [12], AlexNet [21], and VGG-11 [41]. We can observe that the results are still comparable with the benchmark. Therefore, the superiority of our method also holds given different architectures during training and evaluation.

[2]We omit MTT [2] here since it requires caching hundreds of training trajectories beforehand, which fulfills neither memory nor time constraints imposed by slimmable DC. More analysis can be found in the supplement.

4.3. Ablation Studies

Learning-Based Slimmable DC: The proposed first-order parameter matching objective $\mathcal{L}^1_{pm}$ and infinity-order parameter matching objective $\mathcal{L}^\infty_{pm}$ are expected to address problems of inconsistent embedding space and underdetermined solution space respectively. To demonstrate their effectiveness, on CIFAR10, we try removing $\mathcal{L}^1_{pm}$ and replacing $\mathcal{L}^\infty_{pm}$ with the original performance-matching objective in Fig. 3(left). We can observe that $\mathcal{L}^\infty_{pm}$ is a fatal constraint to maintain the performance during successive compression. Moreover, removing $\mathcal{L}^1_{pm}$ would also cause a performance drop compared with the full version.

To figure out how this objective works, we use slimmed datasets trained with and without this objective, denoted as $\mathcal{S}'_w$ and $\mathcal{S}'_{wo}$ respectively, and the synthetic dataset before slimmed $\mathcal{S}$, to train the same network ϕ_0 separately, and the trained networks are denoted as ϕ'_w, ϕ'_{wo}, and ϕ. We then send $\mathcal{S}$ to ϕ'_w, ϕ'_{wo}, and ϕ, and calculate the embedding distances: $\|\phi'_w(\mathcal{S}) - \phi(\mathcal{S})\|^2_F$ and $\|\phi'_{wo}(\mathcal{S}) - \phi(\mathcal{S})\|^2_F$, which are plotted as Fig. 3(right) for each iteration on CIFAR10 with 1 IPC. It turns out that the embedding distance for $\mathcal{S}_w$ does not increase with training using different data, unlike $\mathcal{S}_{wo}$. Thus, $\mathcal{L}^1_{pm}$ encourages a consistent embedding space and thus benefits slimmable DC.

Learning-Free Slimmable DC: According to Prop. 1, objectives of $\mathcal{L}_{skew}$ and $\mathcal{L}_{ortho}$ would contribute to minimizing the first-order and infinity-order parameter distances and encourage promising results of learning-free slimmable DC. As shown in Fig. 4(left), we also witness this effect in the experiments on CIFAR10, where removing any of them would decrease performances. Also, the results of learning-free slimming can serve as a strong initialization to benefit the learning-based setting. As shown in Fig. 4(right), on CIFAR10 with 1 IPC, this initialization strategy can accelerate the training convergence to the final performance compared with not using this parameterization. Here, for a fair comparison, we also provide results of the same parameterization, but without $\mathcal{L}_{skew}$ or $\mathcal{L}_{ortho}$, which achieves inferior performance. In other words, the success of the proposed learning-free slimming solution relies on both the parameterization and the constraints imposed by the loss terms.

Qualitative Visualization: We visualize results of our learning-based and learning-free slimmable DC methods in Figs. 5 and 6 respectively. In the learning-based case, compared with FRePo [58], our method can still preserve clear contents of target semantics, while the content structures by FRePo are nearly indistinguishable. Given that there is some positive correlation between performance and quality of synthetic images [58], our method performs better in slimmable DC. In the learning-free case, we observe that the deleted components contain mainly minor structures and

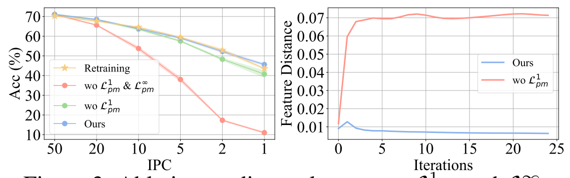

Figure 3. Ablation studies on loss terms $\mathcal{L}^1_{pm}$ and $\mathcal{L}^\infty_{pm}$.

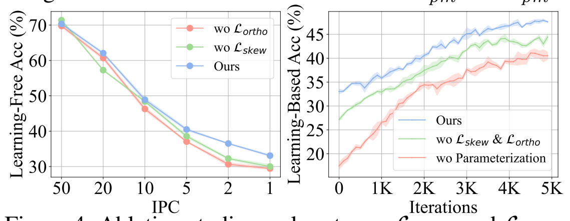

Figure 4. Ablation studies on loss terms $\mathcal{L}_{skew}$ and $\mathcal{L}_{ortho}$.

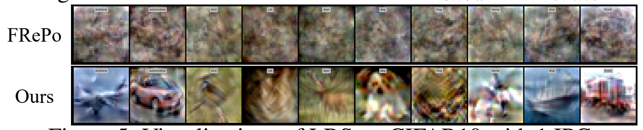

Figure 5. Visualizations of LBS on CIFAR10 with 1 IPC.

less-significant information, while major contours and semantics are preserved in components with the largest singular values. It turns out that reconstructed results using top-20 components only lose some details and become vague.

4.4. Applications

We demonstrate the practical applications of slimmable DC with two examples: continual learning and federated learning. Continual learning aims at learning from a sequence of tasks. Data of previous tasks are unavailable when learning the current task. DC is useful in this area to compress data into a small buffer for future use [35]. Following the configuration of previous works [17, 55, 58], we conduct experiments on CIFAR100 with 5 sequentially-coming tasks, where each task includes random 20 classes. However, previous works assume that the size of the synthetic buffer is proportional to the number of seen classes. In this paper, we consider a more practical case where the size of the buffer is fixed and we do not know the total number of classes in advance. Thus, when new classes come and the buffer has been used up, we have to conduct slimmable condensation first. Here, if the size of the buffer is 500 images in total, then the number of synthetic images per class rounded up should be 25 $\rightarrow$ 13 $\rightarrow$ 9 $\rightarrow$ 7 $\rightarrow$ 5 respectively when each new task comes. The performance of our method and FRePo baseline is shown in Fig. 7(left).

In federated learning, multiple users jointly train a model with data privacy protection. DC achieves this goal by uploading synthetic samples rather than raw data [7]. Similarly, assume there are 5 participants in maximal and each has 20 classes, with 25 synthetic images per class. The maximal network bandwidth is 500 images. If there is more than 1 participant, they have to slim their synthetic dataset to adapt to the per-user bandwidth, *e.g.*, the case of 3 participants can support 9 images per class. The performance

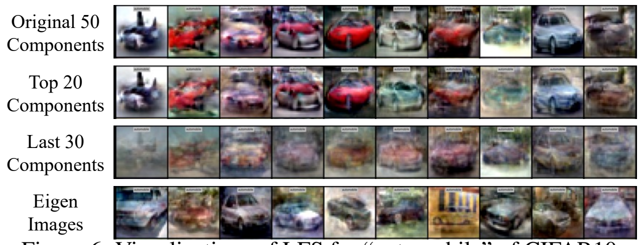

Figure 6. Visualizations of LFS for "automobile" of CIFAR10.

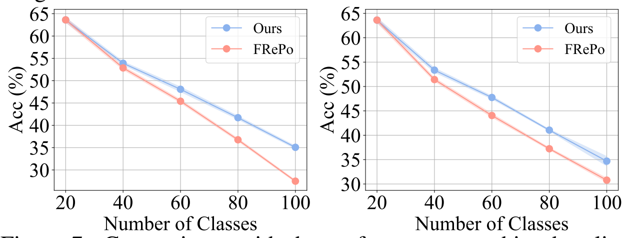

Figure 7. Comparisons with the performance matching baseline on applications of slimmable DC: continual learning using a synthetic buffer with a fixed size (left) and federated learning with a dynamic number of participant (right).

is shown in Fig. 7(right). Unlike continual learning, we do not need to consider successive compression in this case but start from initial datasets given a slimming request. Thus, the performance of FRePo does not drop too much compared with that of continual learning. Please refer to the supplement for more different settings of these applications.

5. Conclusions

In this paper, we focus on slimmable dataset condensation, a topic overlooked by existing DC approaches but important in practice, to extract a slimmed synthetic dataset only given a previously synthesized one, without access to the original real dataset. We identify the limitation of directly applying existing DC methods on slimmable DC, and accordingly propose an appropriate training objective to tackle the drawbacks of inconsistent network space and underdetermined solution space. Moreover, we devise a significance-aware synthetic data parameterization with an upper-bounded error for slimmable DC. Discarding the less important components may serve as either a learning-free slimming solution or a strong initialization to boost the training efficiency in learning-based slimmable DC. Experiments on multiple benchmarks and applications demonstrate the superiority of our method on slimmable DC over state-of-the-art typical DC algorithms.

Acknowledgment

This work is supported by the National Research Foundation, Singapore under its Medium Sized Center for Advanced Robotics Technology Innovation, and by the Advanced Research and Technology Innovation Centre (ARTIC), the National University of Singapore (project number: A-0005947-21-00, project reference: ECT-RP2).

References

[1] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. 6

[2] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. *arXiv preprint arXiv:2203.11932*, 2022. 1, 2, 7

[3] Can Chen, Yingxue Zhang, Jie Fu, Xue Liu, and Mark Coates. Bidirectional learning for offline infinite-width model-based optimization. In *Thirty-Sixth Conference on Neural Information Processing Systems*, 2022. 1

[4] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021. 2

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

[6] Zhiwei Deng and Olga Russakovsky. Remember the past: Distilling datasets into addressable memories for neural networks. *arXiv preprint arXiv:2206.02916*, 2022. 3

[7] Tian Dong, Bo Zhao, and Lingjuan Lyu. Privacy for free: How does dataset condensation help privacy? *arXiv preprint arXiv:2206.00240*, 2022. 1, 8

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1

[9] Jiawei Du, Yidi Jiang, Vincent TF Tan, Joey Tianyi Zhou, and Haizhou Li. Minimizing the accumulated trajectory error to improve dataset distillation. *arXiv preprint arXiv:2211.11004*, 2022. 1

[10] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017, 2019. 1

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

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 7

[13] Shengyuan Hu, Jack Goetz, Kshitiz Malik, Hongyuan Zhan, Zhe Liu, and Yue Liu. Fedsynth: Gradient compression via synthetic data in federated learning. *arXiv preprint arXiv:2204.01273*, 2022. 2

[14] Zixuan Jiang, Jiaqi Gu, Mingjie Liu, and David Z. Pan. Delving into effective gradient matching for dataset condensation. *arXiv preprint arXiv:2208.00311*, 2022. 2

[15] Yongcheng Jing, Yiding Yang, Xinchao Wang, Mingli Song, and Dacheng Tao. Amalgamating knowledge from heterogeneous graph neural networks. In *CVPR*, 2021. 1

[16] Yongcheng Jing, Yiding Yang, Xinchao Wang, Mingli Song, and Dacheng Tao. Meta-aggregator: Learning to aggregate for 1-bit graph neural networks. In *ICCV*, 2021. 1

[17] 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. *arXiv preprint arXiv:2205.14959*, 2022. 1, 2, 3, 6, 8

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4

[19] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016. 2

[20] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012. 1, 7

[22] Hae Beom Lee, Dong Bok Lee, and Sung Ju Hwang. Dataset condensation with latent space knowledge factorization and sharing. *arXiv preprint arXiv:2208.00719*, 2022. 3

[23] Saehyung Lee, Sanghyuk Chun, Sangwon Jung, Sangdoo Yun, and Sungroh Yoon. Dataset condensation with contrastive signals. *arXiv preprint arXiv:2202.02916*, 2022. 2

[24] Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. In *Uncertainty in artificial intelligence*, pages 367–377. PMLR, 2020. 1

[25] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020. 2

[26] Ping Liu, Xin Yu, and Joey Tianyi Zhou. Meta knowledge condensation for federated learning. *arXiv preprint arXiv:2209.14851*, 2022. 2

[27] Songhua Liu, Kai Wang, Xingyi Yang, Jingwen Ye, and Xinchao Wang. Dataset distillation via factorization. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 3

[28] Noel Loo, Ramin Hasani, Alexander Amini, and Daniela Rus. Efficient dataset distillation using random feature approximation. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 1, 2, 3

[29] Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1540–1552, 2020. 1

[30] Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 2113–2122, 2015. 1

[31] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469:28–51, 2022. 2

[32] 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 (CVPR), Workshop*, 2020. 2

[33] Timothy Nguyen, Zhourong Chen, and Jaehoon Lee. Dataset meta-learning from kernel ridge-regression. *arXiv preprint arXiv:2011.00050*, 2020. 1, 2

[34] Timothy Nguyen, Roman Novak, Lechao Xiao, and Jaehoon Lee. Dataset distillation with infinitely wide convolutional networks. *Advances in Neural Information Processing Systems*, 34, 2021. 1, 2

[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. 8

[36] Sucheng Ren, Fangyun Wei, Zheng Zhang, and Han Hu. Tinymim: An empirical study of distilling mim pre-trained models. June 2023. 1

[37] Sucheng Ren, Daquan Zhou, Shengfeng He, Jiashi Feng, and Xinchao Wang. Shunted self-attention via multi-scale token aggregation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10853–10862, June 2022. 1

[38] Andrea Rosasco, Antonio Carta, Andrea Cossu, Vincenzo Lomonaco, and Davide Bacciu. Distilled replay: Overcoming forgetting through synthetic samples. *arXiv preprint arXiv:2103.15851*, 2021. 2

[39] Mattia Sangermano, Antonio Carta, Andrea Cossu, and Davide Bacciu. Sample condensation in online continual learning. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2022. 2

[40] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321, 2015. 1

[41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1, 7

[42] Rui Song, Dai Liu, Dave Zhenyu Chen, Andreas Festag, Carsten Trinitis, Martin Schulz, and Alois Knoll. Federated learning via decentralized dataset distillation in resource-constrained edge environments. *arXiv preprint arXiv:2208.11311*, 2022. 2

[43] 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 *Proceedings of the International Conference on Machine Learning (ICML)*, pages 9206–9216, 2020. 1

[44] 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. *arXiv preprint arXiv:2203.01531*, 2022. 1, 3

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

[46] Felix Wiewel and Bin Yang. Condensed composite memory continual learning. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2021. 2

[47] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 6

[48] Yuanhao Xiong, Ruochen Wang, Minhao Cheng, Felix Yu, and Cho-Jui Hsieh. Feddm: Iterative distribution matching for communication-efficient federated learning. *arXiv preprint arXiv:2207.09653*, 2022. 2

[49] Xingyi Yang, Zhou Daquan, Songhua Liu, Jingwen Ye, and Xinchao Wang. Deep model reassembly. In *Conference on Neural Information Processing Systems*, 2022. 1

[50] Xingyi Yang, Jingwen Ye, and Xinchao Wang. Factorizing knowledge in neural networks. In *European Conference on Computer Vision*, 2022. 1

[51] Jingwen Ye, Yifang Fu, Jie Song, Xingyi Yang, Songhua Liu, Xin Jin, Mingli Song, and Xinchao Wang. Learning with recoverable forgetting. In *ECCV*, 2022. 1

[52] Jingwen Ye, Songhua Liu, and Xinchao Wang. Partial network cloning. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 2

[53] Ruonan Yu, Songhua Liu, and Xinchao Wang. Dataset distillation: A comprehensive review. *arXiv preprint arXiv:2301.07014*, 2023. 2

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

[55] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. *arXiv preprint arXiv:2110.04181*, 2021. 1, 3, 6, 8

[56] Bo Zhao and Hakan Bilen. Synthesizing informative training samples with gan. *arXiv preprint arXiv:2204.07513*, 2022. 3

[57] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*, 2020. 1, 2, 4, 6

[58] Yongchao Zhou, Ehsan Nezhadarya, and Jimmy Ba. Dataset distillation using neural feature regression. *arXiv preprint arXiv:2206.00719*, 2022. 1, 2, 3, 6, 7, 8

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