Up to 100× Faster Data-free Knowledge Distillation

Gongfan Fang[1,3*], **Kanya Mo**[1*], **Xinchao Wang**[2], **Jie Song**[1]
Shitao Bei[1], **Haofei Zhang**[1], **Mingli Song**[1†]
[1]Zhejiang University
[2]National University of Singapore
[3]Alibaba-Zhejiang University Joint Research Institute of Frontier Technologies
{fgf, sjie, best, haofeizhang, brooksong}@zju.edu.cn
Kanya.18@intl.zju.edu.cn, xinchao@nus.edu.sg

Abstract

Data-free knowledge distillation (DFKD) has recently been attracting increasing attention from research communities, attributed to its capability to compress a model only using synthetic data. Despite the encouraging results achieved, state-of-the-art DFKD methods still suffer from the inefficiency of data synthesis, making the data-free training process extremely time-consuming and thus inapplicable for large-scale tasks. In this work, we introduce an efficacious scheme, termed as FastDFKD, that allows us to accelerate DFKD by a factor of orders of magnitude. At the heart of our approach is a novel strategy to reuse the shared common features in training data so as to synthesize different data instances. Unlike prior methods that optimize a set of data independently, we propose to learn a meta-synthesizer that seeks common features as the initialization for the fast data synthesis. As a result, FastDFKD achieves data synthesis within only a few steps, significantly enhancing the efficiency of data-free training. Experiments over CIFAR, NYUv2, and ImageNet demonstrate that the proposed FastDFKD achieves 10× and even 100× acceleration while preserving performances on par with state of the art.

1 Introduction

Knowledge distillation (KD) has recently emerged as a popular paradigm to reuse pre-trained models that are nowadays prevalent online. KD aims to train a compact student model for efficient inference by imitating the behavior of the pre-trained teacher (Hinton, Vinyals, and Dean 2015; Yang et al. 2020; Fang et al. 2021a). The conventional setup of KD requires possessing the original training data as input so as to train the student. Unfortunately, due to confidential or copyright reasons, in many cases the original data cannot be released and only the pre-trained models are available to users (Kolesnikov et al. 2020; Shen et al. 2019; Ye et al. 2019), which, in turn, imposes a major obstacle towards applying KD for a broader domain.

To remedy this issue, data-free knowledge distillation (DFKD) approaches have been proposed by assuming that no access to training data is available at all (Lopes, Fenu, and Starner 2017). Due to the much relaxed constraint on training data, DFKD has been receiving increasing attention from research communities including computer vision (Chen et al. 2019), natural language processing (Ma et al. 2020), and graph learning (Deng and Zhang 2021). Typically, DKFD follows a distilling-by-generating paradigm, where a synthetic dataset is often crafted for training by "inverting" the pre-trained teacher (Yin et al. 2019). To learn a comparable student model, the synthetic set should contain sufficient samples to enable a comprehensive knowledge transfer from teachers. Consequentially, this poses a significant challenge to DFKD, since synthesizing a large-scale dataset is inevitably time-consuming, especially for sophisticated tasks like ImageNet recognition (Yin et al. 2019) and COCO detection (Chawla et al. 2021).

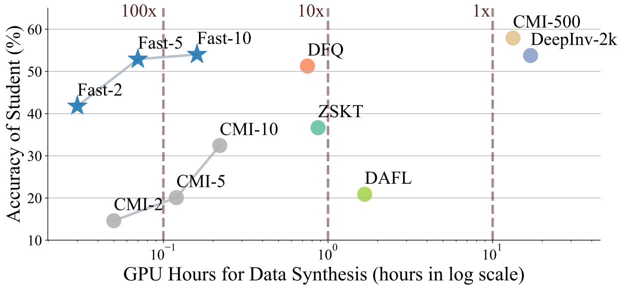


Figure 1: Accuracy (%) of student models v.s. GPU hours of data synthesis on CIFAR-100 dataset. Our method, denoted as "Fast", achieves 10× to even 100× acceleration and performance on par with existing methods.

In this paper, we introduce a novel approach, termed as FastDFKD, to accelerate data synthesis process so as to make data-free knowledge distillation more applicable for large-scale tasks. Our motivation stems from the fact that, data instances from the same domain usually share some common features, and hence such shared features should be explicitly utilized for data synthesis. For example, the textures of "fur" may frequently emerge in an animal dataset, and thus can be reused to create different instances. Unfortunately, existing DFKD approaches have mainly focused on synthesizing samples independently and, in fact, none of the existing DFKD approaches has explored taking advantage

[*]Equal contributions

[†]Corresponding author

of feature sharing, making the DFKD process very cumbersome.

The proposed FastDFKD approach, on the other hand, expressively explores the common features between samples for synthesizing. FastDFKD follows the batch-based strategy for data synthesis (Yin et al. 2019; Fang et al. 2021b); yet unlike prior methods that synthesizes different samples independently, FastDFKD focuses on a "learning to synthesize" problem, where an efficient synthesizer is explicitly trained to capture common features for fast adaptation. The advantage of common features sharing is that we do not need to synthesize them repeatedly for each instance, which significantly improves the synthesis efficiency.

Specifically, we develop FastDFKD under a meta-learning framework (Finn, Abbeel, and Levine 2017), which aims to learn a meta-generator in the synthesis process. FastDFKD comprises two optimization loops, the outer loop, and the inner loop. The inner loop accounts for the data synthesis process, where a set of samples are created by adapting and re-organizing common features. The outer loop, on the other hand, updates the common features using the results of inner loops for a better meta-initialization. As illustrated in Figure 1, such a meta synthesizer dramatically improves the efficiency of data synthesis in data-free distillation while preserving a performance on par with state of the art. As will be demonstrated in our experiments, FastDFKD is able to achieve $10\times$ and in some cases even more than $100\times$ speed up compared to the state of the art.

Our contribution is therefore a novel DFKD scheme, termed as FastDFKD, that allows us to significantly accelerate the data-free training through common feature reusing. Experimental results over the CIFAR, ImageNet, and NYUv2 datasets demonstrate that, FastDFKD yields performances comparable to the state of the art, while achieving a speedup factor of 10 and sometimes even more than 100 over existing approaches.

2 Related Works

Data-Free Knowledge Distillation. Data-free knowledge distillation aims to train a compact student model from a pre-trained teacher model without access to original training data. It typically follows a distilling-by-generating paradigm, where a fake dataset will be synthesized and used for student training. In the literature, Lopes *et al.* proposes the first data-free approach for knowledge distillation, which utilizes statistical information of original training data to reconstruct a synthetic set during knowledge distillation (Lopes, Fenu, and Starner 2017). This seminal work has spawned several works, which has achieved impressive progressive on several tasks including detection (Chawla et al. 2021), segmentation (Fang et al. 2019), text classification (Ma et al. 2020), graph classification (Deng and Zhang 2021) and Federated Learning (Zhu, Hong, and Zhou 2021). Despite the impressive progress, a vexing problem remains in DFKD, i.e., the inefficiency of data synthesis, which makes data-free training extraordinarily time-consuming. For example, (Luo et al. 2020) trains 1,000 generators to compress an ImageNet-pretrained ResNet-50 and (Yin et al. 2019) optimizes a large number of mini-batches for data synthesis. In this work, we focus on this under-studied problem, i.e., the efficiency of DFKD, and proposes the first approach to accelerate data-free training.

Meta Learning. Meta-learning is a popular framework for few-shot learning (Hospedales et al. 2020), which follows a "learning to learn" paradigm to find a helpful initialization for target tasks. Among various meta-learning algorithms, MAML is one of the most influential methods owning to its impressive results on several benchmarks (Finn, Abbeel, and Levine 2017; Nichol, Achiam, and Schulman 2018). As an optimization-based method for meta-learning, MAML introduces two optimization loops to handle a set correlated tasks: an inner loop for task learning and an outer loop for training a meta-learner. The inner and outer loops are collaboratively trained to find a meta-initialization that can be adapted to different tasks quickly, where some general knowledge across tasks are captured by the meta-learner (Finn, Abbeel, and Levine 2017). Inspired by the "learning to learn" paradigm of meta-learning, we develop a fast approach to train a meta-synthesizer for DFKD problems, which can be quickly adapted for fast data synthesis.

3 Method

Problem Setup

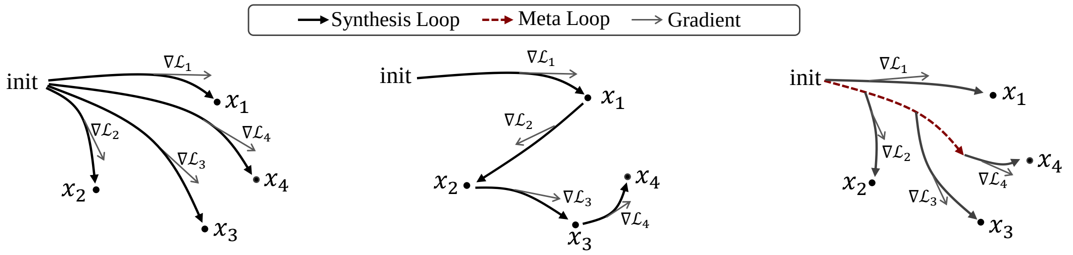
Given a teacher model $f_t(x;\theta_t)$ pre-trained on a labeled but inaccessible training set $D^t = \sum_i^N \{(x_i, y_i)|x_i \in \mathcal{X}, y_i \in \mathcal{Y}\}$, the goal of data-free knowledge distillation (DFKD) is to craft a synthetic dataset $D = \sum_i^N \{x_i|x_i \in \mathcal{X}\}$ with N samples by inverting the pre-trained model, on which a comparable student model $f_s(x;\theta_s)$ can be trained by imitating the behaviour of the teacher. Typically, The synthesis of D is driven by a pre-trained inversion loss $\mathcal{L}: \mathcal{X} \to \mathbb{R}$, which indicates whether a input sample x comes from the training domain according to some statistical information in the pre-trained teacher model (Yin et al. 2019). Therefore, the optimization of a single data point x can be formalized as follows:

$$x^* = \arg\min_x \mathcal{L}(x) \tag{1}$$

To obtain a complete synthetic set $D' = \{x_1, x_2, ..., x_N\}$ of size N, DFKD repeats the above optimization to construct a set of samples, which leads to a series of optimization problems in the form of Equation 1. Notably, the loss function for different instances x_i, denoted as $\mathcal{L}_i$, can be different, so that a diverse dataset D can be constructed to carry out comprehensive knowledge from the teacher. To this end, we consider a generalized data synthesis problem for DFKD, which leverages a set of inversion losses $\mathcal{L} = \{\mathcal{L}_1, \mathcal{L}_2, ..., \mathcal{L}_N\}$ to craft the synthetic dataset as follows:

$$D' = \{x_1^*, x_2^*, ..., x_N^*\} = \underset{x_1, x_2, ..., x_N}{\arg\min} \sum_i^N \mathcal{L}_i(x_i) \tag{2}$$

In DFKD, a popular way to solve Equation 2 is to optimize different samples directly in a batch-by-batch manner, (Yin et al. 2019; Fang et al. 2019; Chawla et al. 2021). As illustrated in Figure 2 (a), batch-based approaches synthesize



(a) No feature reuse (b) Sequential feature reuse (c) Meta feature reuse

Figure 2: Diagram of the proposed meta feature reuse, as well as its difference to other synthesis strategies. (a) Data instances are synthesized independently without feature reuse; (b) Data instances are synthesized sequentially reusing previous results as initialization. (c) The proposed common feature reuse that learns a meta-generator for fast adaptation.

different instances independently and merely take the relation between samples into account. Despite the encouraging results achieved, DFKD approaches usually suffer from the inefficiency of data synthesis, since crafting a large-scale dataset requires solving a large number of optimization problems in Equation 1, each takes thousands of steps to converge (Yin et al. 2019). Typically, data from the same domain are likely to share some common features, which can be reused to synthesize different samples. In this work, we present FastDFKD, a novel and efficacious approach to learning common features to accelerate the optimization of Equation 2.

Fast Data-Free Knowledge Distillation

Overview. At the heart of our proposed approach is the reusing of common features. Our motivation stems from the fact that data from the same domain typically share some reusable patterns, which can be reused to synthesize different instances. The following sections develop a novel definition for common features from a generative perspective and propose FastDFKD to capture common features for fast synthesis through a meta-learning process.

Common Feature. As the key step towards fast data-free training, a clear definition of the common feature is required to construct an optimizable objective for network training. As illustrated in Figure 2 (b), a naive reusing strategy would be the sequential feature reusing, where features learned in the previous synthesis are directly used as the initialization to craft new samples. However, such a naive scheme would be problematic since the learned features only come from a single data point, which may not always be reusable for other samples. To address this problem, we develop a more natural definition for common features from a generative perspective. Let's consider a generative network $\mathcal{G}(z;\theta)$ with latent code z and trainable parameters θ, which satisfies that for each sample $x_i \in D'$, a latent code z_i can be found to generate $x_i = \mathcal{G}(z_i;\theta)$. The generator describes the generation process of different samples x_i. To some extent, whether there are common features between a set of samples $D' = \{x_1, x_2, ..., x_N\}$ is usually highly correlated with the similarity of different data instances, which means that the generator can implicitly capture the common features if we can find the optimal parameter θ making the code $z = \{z_1, z_2, ..., z_N\}$ of different samples close in the latent space. Based on this, common features can be learned by solving the following problems:

$$\min_{z,\theta} \underbrace{\frac{1}{N^2}\sum_i\sum_j d_z(z_i,z_j)}_{\textit{close in latent space}} + \underbrace{\frac{1}{N}\sum_i d_x(\mathcal{G}(z_i,\theta),x_i)}_{\textit{generation}} \tag{3}$$

where d_z and d_x refers to distance metrics in latent space and input space. The above optimization aims at finding a generation process for dataset D', whose z distance in latent space is as small as possible so that, with the learned common features, different samples can be efficiently obtained by navigating in the latent code z. However, in data-free settings, the synthetic dataset D' is not available until we synthesize it. Thus, We replace the second term defined on d_x of Equation 3 with the inversion loss $\mathcal{L}$ for DFKD, which leads to a data-free objective for common feature learning:

$$\min_{z,\theta} \underbrace{\frac{1}{N^2}\sum_i\sum_j d_z(z_i,z_j)}_{\textit{close in latent space}} + \underbrace{\frac{1}{N}\sum_i \mathcal{L}_i(\mathcal{G}(z_i,\theta))}_{\textit{data-free generation}} \tag{4}$$

However, due to the limited capacity of the generative model and the difficulty in GAN training, it is almost intractable to learn a perfect generator to synthesize the full synthetic set D' at the same time (Luo et al. 2020). To remedy this problem, we make some relaxation on Equation 4 and does not force the generator $\mathcal{G}$ to capture all features for the whole dataset. Instead, we train a generator that allows fast adaptation to different samples within k-steps gradient descents, which naturally leads to a meta-learning problem.

Meta Generator. Equation 4 is challenging to optimize because it requires generating the full dataset D' with a single generative network, including a lot of non-reusable features. Instead, we can train a generator that only contains common features and synthesize other missing features on the fly for the data synthesis process as illustrated in Figure 2 (c). Specifically, we relax the objective of common feature learning to train a lightweight generator that can be adapted to synthesize different instances within k-step, formalized as a meta-learning problem:

$$\min_{z,\theta} \frac{1}{N}\sum_i \mathcal{L}_i(\mathcal{G}(\underbrace{U^k_{\mathcal{L}_i}(\hat{z},\hat{\theta})}_{k\text{-step adaptation}})) \tag{5}$$

where $U^k_{\mathcal{L}_i}$ is the inner loop of meta-learning, which refers to a k-step optimization initialized from $\hat{\theta}$ and code $\hat{z}$ for the synthesis of x_i. The inner loop can be unrolled as follows:

$$\begin{aligned} z^0_i, \theta^0_i &= \hat{z}, \hat{\theta} \\ z^k_i &= z^{k-1}_i - \alpha \nabla_{z^{k-1}_i} \mathcal{L}_i(\mathcal{G}(z^{k-1}_i; \theta^{k-1}_i)) \\ \theta^k_i &= \theta^{k-1}_i - \alpha \nabla_{\theta^{k-1}_i} \mathcal{L}_i(\mathcal{G}(z^{k-1}_i; \theta^{k-1}_i)) \end{aligned} \tag{6}$$

Notably, Equation 5 plays a similar role as the common feature loss in Equation 4. The inner loop, i.e., the k-step adaptation, aims to learn a generator for synthesis by explicitly optimizing the second term of Equation 4. On the other hand, the outer loop tries to make different samples reachable within k-step optimization by implicitly optimizing the first term of Equation 4.

Optimization. Optimizing Equation 6 naturally leads to a meta-learning problem, where a useful initialization $(\hat{z}, \hat{\theta})$ is demanded for fast adaptation. After k-step gradient descent, we obtain the a set of new parameter $(z^*_i, \theta^*_i) = U^k_{\mathcal{L}_i}(\hat{z}, \hat{\theta}))$ under the guidance of loss function $\mathcal{L}_i$, which provides gradient w.r.t $\hat{\theta}$ as follows:

$$\begin{aligned} g_{\hat{\theta}} &= \nabla_{\hat{\theta}} \mathcal{L}_i(\mathcal{G}(U_{\mathcal{L}_i}(\hat{z}; \hat{\theta}))) \\ &= U'_{\mathcal{L}_i}(\hat{\theta}) \mathcal{G}'(\theta^*) \mathcal{L}'_i(x^*_i) \end{aligned} \tag{7}$$

where $\theta^* = U^k_{\mathcal{L}_i}(\hat{z}; \hat{\theta})$ refers to the optimization results of k-step adaptation using Equation 6 and $x^*_i = \mathcal{G}(z^*_i; \theta^*_i) = \mathcal{G}(U^k_{\mathcal{L}_i}(\hat{z}; \hat{\theta}))$ refers to the synthesis results under the guidance of loss $\mathcal{L}_i$. However, note that the k-step adaptation in Equation 6 involves k gradient updates:

$$(z^*, \theta^*) = U^k_{\mathcal{L}_i}(\hat{z}; \hat{\theta}) = (\hat{z}, \hat{\theta}) + g_1 + g_2 + ... + g_k \tag{8}$$

where g_k refers to the gradient computed at the k-th step of Equation 6, which introduces high-order gradients to the generator training and makes the back-propagation very inefficient. Inspired by prior works in meta learning (Nichol, Achiam, and Schulman 2018), we apply a first-order approximation to further accelerate the gradient computing, which treats high-order gradients in 3 as constants and replace the $U'_i(\hat{\theta})$ with an identity mapping. In this case, the gradient computing in Equation 7 only involves first-order gradient and can be simplified as follows:

$$\nabla_{\hat{\theta}} \mathcal{L}_i(\mathcal{G}(U^k_{\mathcal{L}_i}(\hat{z}; \hat{\theta}))) = \mathcal{G}'(\theta^*) \mathcal{L}'_i(x^*) \tag{9}$$

The first-order approximation directly uses the gradient computed on the adapted generator in the inner loops to update the meta generator. Further, a more efficient gradient approximation, known as reptile (Nichol, Achiam, and Schulman 2018), can be achieved by approximating the gradient in Equation 9 with the parameter difference between the adapted generator and meta generator, which further simplifies Equation 9 as:

$$\nabla_{\hat{\theta}} \mathcal{L}_i(\mathcal{G}(U^k_{\mathcal{L}_i}(\hat{z}; \hat{\theta}))) = \hat{\theta}_i - \theta^*_i \tag{10}$$

In conclusion, the optimization of meta generator can be presented as follows:

$$\begin{aligned} \hat{\theta} &= \hat{\theta} - \eta \sum_i \nabla_{\hat{\theta}} \mathcal{L}_i(\mathcal{G}(U^k_{\mathcal{L}_i}(\hat{z}; \hat{\theta}))) \\ \hat{z} &= \hat{z} - \eta \sum_i \nabla_{\hat{z}} \mathcal{L}_i(\mathcal{G}(U^k_{\mathcal{L}_i}(\hat{z}; \hat{\theta}))) \end{aligned} \tag{11}$$

Algorithm 1 FastDFKD

```
Input: Pretrained teacher $\mathcal{F}_t(x; \theta_t)$, student $\mathcal{F}_s(x; \theta_s)$.
Output: An optimized student $\mathcal{F}_s(x; \theta_s)$

Randomly initialize a generator $\mathcal{G}(\hat{z}; \hat{\theta})$
Initialize an empty dataset $D' = \{\}$
for each synthesis loss $\mathcal{L}_i$ do:
    // 1. $k$-step adaptation for synthesis (Eq. 6)
    Periodically re-initialize $\hat{z}$ for diversity
    Reuse meta feature $z, \theta \leftarrow \hat{z}, \hat{\theta}$
    for k steps do:
        Update $(z, \theta) \leftarrow (z, \theta) - \alpha \nabla_{(z,\theta)} \mathcal{L}_i(\mathcal{G}(z; \theta))$
    end for
    Generate the synthetic data $x^* = \mathcal{G}(z; \theta)$
    Update the synthetic set $D' = D' \cup \{x^*\}$

    // 2. Meta update (Eq. 10)
    Update $(\hat{z}, \hat{\theta}) \leftarrow (\hat{z}, \hat{\theta}) - \eta \nabla_{(\hat{z}, \hat{\theta})} \mathcal{L}_i(\mathcal{G}(U^k_i(\hat{z}; \hat{\theta})))$

    // 3. KD update (Eq. 12)
    for t steps do:
        sample a mini-batch $\mathcal{B}$ from $D'$
        $\theta_s \leftarrow \theta_s - \gamma \sum_{x \in \mathcal{B}} \nabla_{\theta_s} KL(f_t(x) \| f_s(x))$
    end for
end for
```

Method Summary. The proposed method is summarized in Algorithm 1, which consists of three stages: 1) a k-step adaptation for data synthesis; 2) a meta-learning step for common feature learning; 3) several KD steps to update the student model by optimizing the KL divergence as follows:

$$\theta_s \leftarrow \theta_s - \gamma \sum_{x \in \mathcal{B}} \nabla_{\theta_s} KL(f_t(x) \| f_s(x)) \tag{12}$$

where $\mathcal{B}$ is a mini-batch sampled from the synthetic set D'. The proposed approach allows a small k for data synthesis, which significantly improves the efficiency of DFKD.

4 Experiments

Inversion Loss for FastDFKD

In this section, we provide more details about the inversion loss $\mathcal{L}$ used in our method. This work mainly focus on classification and segmentation problems, which have been widely studied in the data-free literature (Yin et al. 2019; Fang et al. 2019; Chen et al. 2019). We utilize the prevalent Deep Inversion loss proposed by (Yin et al. 2019) as criteria for data synthesis and demonstrate how to accelerate the data synthesis with FastDFKD. Note that Deep Inversion loss consists of three terms: a class confidence loss $\mathcal{L}_{cls}$, an adversarial loss $\mathcal{L}_{adv}$ and a feature regularization loss $\mathcal{L}_{feat}$, summarized as follows:

$$\begin{cases} \mathcal{L}_{cls}(x) = CE(f_t(x), c) \\ \mathcal{L}_{adv}(x) = -JSD(f_t(x)/\tau \| f_s(x)/\tau) \\ \mathcal{L}_{feat}(x) = \sum_l \|\mu^l_{feat} - \mu^l_{bn}\| + \|\sigma^l_{feat} - \sigma^l_{bn}\| \end{cases} \tag{13}$$

where both $\mathcal{L}_{cls}(x)$ and $\mathcal{L}_{adv}(x)$ provide dynamic learning targets for data synthesis and thus can be directly used to construct different synthesis tasks for meta-learning. For example, choosing different pseudo labels can lead to various target categories. However, the loss for feature regularization is unchanged during synthesis since the batch normalization layer only encodes the global statistical information on the whole dataset. To fully leverage the power of meta-learning, which requires a set of different but related losses $\mathcal{L}$, we modify the Deep Inversion loss by decomposing the feature regularization loss. Note that the feature regularization $\mathcal{L}_{feat}$ aims to solve a distribution matching problem, where the synthetic distribution is will be aligned with the mean and variance $(\mu_{bn}, \sigma^2_{bn})$ stored in BN layers. The BN statistics is estimated on the whole dataset using the following rules:

$$\begin{aligned} \mu_{bn} &= (1-m)\cdot\mu_{bn} + m\cdot\mu_{feat} \\ \sigma^2_{bn} &= (1-m)\cdot\sigma^2_{bn} + m\cdot\sigma^2_{feat} \end{aligned} \tag{14}$$

where m is a momentum parameter for estimating BatchNorm statistics (BNS). Inspired by the momentum estimation of BNS, we propose to construct a dynamic and adaptive feature regularization in a momentum way, too. Specifically, we introduce two accumulative variables to store the mean and variance of already synthesized data and train the inputs to approximate the global BNS as follows:

$$\begin{aligned} \mathcal{L}_{feat}(x) = \sum\nolimits_l [&\|(1-m)\cdot\mu^l_a + m\cdot\mu^l_{feat} - \mu^l_{bn}\| + \\ &\|(1-m)\cdot\sigma^l_a + m\cdot\sigma^l_{feat} - \sigma^2_{bn}\|] \end{aligned} \tag{15}$$

After synthesis, the accumulative variables (μ_a, σ^2_a) will be updated with Equation 14, so as to provide a different learning target for meta-learning.

Experimental Settings

Datasets and models. We evaluate the proposed method on both classification and semantic segmentation tasks. For image classification, we conduct data-free knowledge distillation on three widely used datasets: CIFAR-10, CIFAR-100 (Krizhevsky, Hinton et al. 2009) and ImageNet (Deng et al. 2009). We use the pretrained models from (Fang et al. 2021b) and follow the same training protocol for comparison, where 50,000 synthetic images are synthesized for distillation. For ImageNet, we use an off-the-shelf ResNet-50 as the teacher and train student models by synthesizing 224×224 images. For semantic segmentation, we use Deeplab models (Chen et al. 2017) trained on NYUv2 (Nathan Silberman and Fergus 2012) dataset for training and evaluation, which contains 1449 densely labeled pairs of aligned RGB images and a 13-class segmentation map.

Baselines. Two types of DFKD methods are compared in our experiments: (1) generative methods that train a generative model for synthesis, including DAFL (Chen et al. 2019), ZSKT (Micaelli and Storkey 2019), DFQ (Choi et al. 2020), and Generative DFD (Luo et al. 2020) (2) non-generative methods that craft transfer set in a batch-by-batch manner including DeepInv (Yin et al. 2019) and CMI (Fang et al. 2021b).

Evaluation Metrics. In addition to the standard metrics like classification accuracy and mean IoU for classification and segmentation, we also focus on the efficiency of data synthesis in DFKD, where GPU hours taken by data synthesis is collected and reported. Note that the time cost of student training is omitted since we only focus on the data synthesis process and adopt the vanilla KD (Hinton, Vinyals, and Dean 2015) in all DFKD methods. For fair comparisons, all GPU hours are estimated on a single GPU.

Results on Classification

CIFAR-10 & CIFAR-100. The student accuracy obtained on CIFAR-10 and CIFAR-100 datasets are reported in Table 1. In the table, baseline "Teacher", "Student" and "KD" train networks with the original training data, which does not require data synthesis. To verify the effectiveness of FastDFKD, we compare it with two types of data-free algorithms, including generative methods that train generators for synthesis and non-generative methods that optimize mini-batches iteratively. As shown in Table 1, generative methods are usually 10× faster than non-generative methods like DeepInv and CMI since they only need to train a single generator for synthesis. However, due to the limited capacity of the generative network, it is almost challenging to synthesize a diverse dataset for training, which may degrade the student performance. We find that the performance of generative methods tends to degrade as the complexity of the dataset increases from CIFAR-10 to CIFAR-100. By contrast, non-generative is usually more flexible than generative ones and thus more applicable to different tasks.

Like non-generative methods, the proposed FastDFKD also optimizes mini-batches for data synthesis yet optimizes a generative network for adaptation. As shown in Table 1, the 5-step FastDFKD, i.e., Fast$_5$, can achieve 10× acceleration compared to existing generative methods and even more than 100× acceleration compared to non-generative methods. For example, DeepInv$_{2k}$ synthesizes images by optimizing mini-batches, each of which requires 2,000 iterations to converge (Yin et al. 2019). To obtain 50,000 training samples for CIFAR, DeepInv$_{2k}$ would take 42.1 hours for data synthesis on a single GPU. by contrast, our method, i.e., Fast-5, adopts the same inversion loss as DeepInv but

Dataset	Method	ResNet-34 ResNet-18	VGG-11 ResNet-18	WRN40-2 WRN16-1	WRN40-2 WRN40-1	WRN40-2 WRN16-2	Average Speed Up
CIFAR-10	Teacher	95.70	92.25	94.87	94.87	94.87	-
	Student	95.20	95.20	91.12	93.94	93.95	-
	KD	95.20	95.20	95.20	95.20	95.20	-
	DeepInv$_{2k}$	93.26 (42.1h)	90.36 (20.2h)	83.04 (16.9h)	86.85 (21.9h)	89.72 (18.2h)	1.0×
	CMI$_{500}$	94.84 (19.0h)	91.13 (11.6h)	90.01 (13.3h)	92.78 (14.1h)	92.52 (13.6h)	1.6×
	DAFL	92.22 (2.73h)	81.10 (0.73h)	65.71 (1.73h)	81.33 (1.53h)	81.55 (1.60h)	15.7×
	ZSKT	93.32 (1.67h)	89.46 (0.33h)	83.74 (0.87h)	86.07 (0.87h)	89.66 (0.87h)	30.4×
	DFQ	94.61 (8.79h)	90.84 (1.50h)	86.14 (0.75h)	91.69 (0.75h)	92.01 (0.75h)	18.9×
	Fast$_2$	92.62 (0.06h)	84.67 (0.03h)	88.36 (0.03h)	89.56 (0.03h)	89.68 (0.03h)	655.0×
	Fast$_5$	93.63 (0.14h)	89.94 (0.08h)	88.90 (0.08h)	92.04 (0.09h)	91.96 (0.08h)	247.1×
	Fast$_{10}$	94.05 (0.28h)	90.53 (0.15h)	89.29 (0.15h)	92.51 (0.17h)	92.45 (0.17h)	126.7×
CIFAR-100	Teacher	78.05	71.32	75.83	75.83	75.83	-
	Student	77.10	77.10	65.31	72.19	73.56	-
	KD	77.87	75.07	64.06	68.58	70.79	-
	DeepInv$_{2k}$	61.32 (42.1h)	54.13 (20.1h)	53.77 (17.0h)	61.33 (21.9h)	61.34 (18.2h)	1.0×
	CMI$_{500}$	77.04 (19.2h)	70.56 (11.6h)	57.91 (13.3h)	68.88 (14.2h)	68.75 (13.9h)	1.6×
	DAFL	74.47 (2.73h)	54.16 (0.73h)	20.88 (1.67h)	42.83 (1.80h)	43.70 (1.73h)	15.2×
	ZSKT	67.74 (1.67h)	54.31 (0.33h)	36.66 (0.87h)	53.60 (0.87h)	54.59 (0.87h)	30.4×
	DFQ	77.01 (8.79h)	66.21 (1.54h)	51.27 (0.75h)	54.43 (0.75h)	64.79 (0.75h)	18.8×
	Fast$_2$	69.76 (0.06h)	62.83 (0.03h)	41.77 (0.03h)	53.15 (0.04h)	57.08 (0.04h)	588.2×
	Fast$_5$	72.82 (0.14h)	65.28 (0.08h)	52.90 (0.07h)	61.80 (0.09h)	63.83 (0.08h)	253.1×
	Fast$_{10}$	74.34 (0.27h)	67.44 (0.16h)	54.02 (0.16h)	63.91 (0.17h)	65.12 (0.17h)	124.7×

Table 1: Student accuracy (%) on 32×32 CIFAR. The methods "Teacher", "Student" and "KD" is conducted on original training data and do not require data synthesis. DAFL, ZSKT and DFQ train generative networks for synthesis, while DeepInv, CMI and Fast (ours) optimizes batches to craft different samples.

Method	Data Amount	Syn. Time	Speed Up	ResNet-50 ResNet-50	ResNet-50 ResNet-18	ResNet-50 MobileNetv2
Scratch	1.3M	-	-	75.45	68.45	70.01
Places365+KD	1.8M	-	-	55.74	45.53	39.89
Generative DFD	-	∼300h	1×	69.75	54.66	43.15
DeepInv$_{2k}$	140k	166h	1.8×	68.00	-	-
Fast$_{50}$	140k	6.28h	47.8×	68.61	53.45	43.02

Table 2: Student accuracy (%) on 224×224 ImageNet. We use an off-the-shelf ResNet-50 as the teacher model and train student models from scratch following the training protocol of (Yin et al. 2019).

only requires 5 steps for each batch owning to the proposed common feature reusing, which is much more efficient than Deep Inversion. In addition to the improvements in efficiency, FastDFKD also achieves comparable or even superior performance compared to the state-of-arts.

ImageNet. To verify the effectiveness of FastDFKD, we further evaluate our method on a more challenging dataset, i.e., ImageNet, which contains 1.3 million training images of 224×224 resolutions from 1,000 categories. ImageNet is obviously much more complicated than CIFAR and thus much more time-consuming for data-free training. As shown in Table 2, we compare our methods with a data-driven KD that uses Places365 as alternative data, and two data-free methods, i.e., Generative DFD (Luo et al. 2020) and DeepInv (Yin et al. 2019). Notably, Generative DFD (Luo et al. 2020) trains one generator of 224×224 resolutions for each category, which leads to 1,000 generators in total. Although each generator can be optimized within one hour, the whole training process for 1,000 generators is still cumbersome. By contrast, our method only requires 6.28 hours for image synthesis and preserves comparable performance to existing methods.

Results on Segmentation

In this work, we further conduct data-free training on semantic segmentation tasks to show the effectiveness of our method. In segmentation, we only use the feature regularization loss and adversarial loss of 13 for data synthesis. The mIoU of the student model, as well as the data amount and synthesis time, are reported in Table 3. We compare our method with DFAD (Fang et al. 2019), DAFL (Chen et al. 2019) and DFND (Chen et al. 2021). DFND is a data-driven method and assumes that a sufficient unlabeled set is available for in-domain data retrieval (Chen et al. 2021). DFAD and DAFL refer to data-free methods that train generative networks for knowledge distillation. In comparison, our method successfully synthesizes a training set only in 0.82 hours, which is much more efficient than DAFL (3.99 hours) and DFAD (6.0 hours).

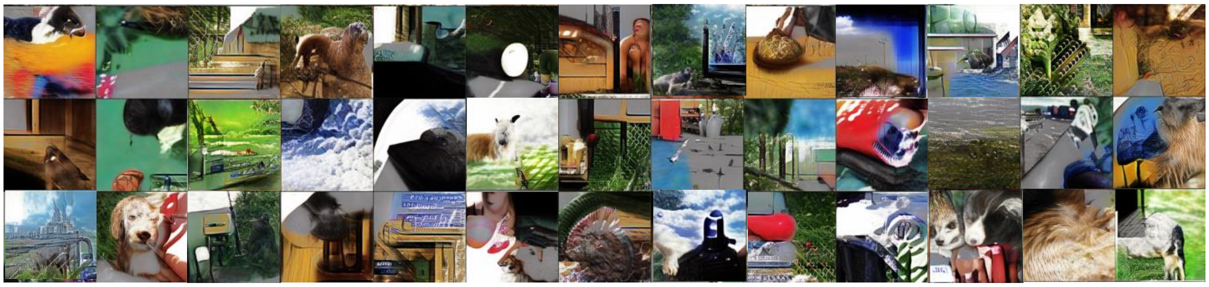


Figure 3: Visualization of synthetic data, inverted from an off-the-shelf ResNet-50 classifier pre-trained on ImageNet. All samples are obtained using the 50-step FastDFKD.

Method	**Data Amount**	**Syn. Time**	**mIoU**
Teacher	1,449 (NYUv2)	-	0.519
Student	1,449 (NYUv2)	-	0.375
KD	1,449 (NYUv2)	-	0.380
DFND	14 M (ImageNet)	-	0.378
DFAD	960k (GAN)	6.0h	0.364
DAFL	960k (GAN)	3.99h	0.105
Fast$_{\mathbf{10}}$	17K (Synthetic)	0.82h	0.366

Table 3: Mean IoU the student model trained on NYUv2 Segmentation dataset. The teacher model is a Deeplabv3-ResNet50 network with ImageNet pre-training, and the student model is a freshly initialized Deeplabv3-Mobilenetv2.

Steps	**2 steps**	**5 steps**	**10 step**
Teacher	75.83	75.83	75.83
Student	65.31	65.31	65.31
DeepInv	2.61 (0.03h)	4.84 (0.06h)	6.77 (0.11h)
CMI	14.62 (0.05h)	20.08 (0.12h)	32.43 (0.22h)
Fast	41.77 (0.03h)	52.90 (0.07h)	54.02 (0.16h)

Table 4: Few-step experiments on CIFAR-100.

Quantitative Analysis

Few-step synthesis. As aforementioned, FastDFKD allows efficient data synthesis within only a few steps. This experiment validates our method by comparing it to the "few-step" versions of existing non-generative methods. For example, we reduce the optimization steps of original DeepInv (Yin et al. 2019) from 2,000 to $\{10, 5, 2\}$ for CIFAR, which leads to the "efficient" DeepInv. As shown in Table 4, the student accuracy of DeepInv and CMI severely degrades when the optimization steps are reduced, which means that existing methods fail to complete the data synthesis in only a few steps. By contrast, the proposed FastDFKD works well even when a 2-step optimization is deployed, which provides strong evidence for the effectiveness of FastDFKD.

Settings	**CIFAR-10**	**CIFAR-100**
No Reuse + Pixel	44.74	3.11
No Reuse + GAN	87.33	35.48
Seq. + Pixel	71.97	10.93
Seq. + GAN	90.91	59.38
Meta + GAN	91.79	61.43
Meta + GAN + MMT	91.96	63.83

Table 5: Ablation study for FastDFKD, where MMT means deep inversion with momentum feature regularization.

Ablation Study. In Table 5, we make an ablation study to make a systematic exploration for the proposed method, where three reusing strategies in Figure 2 are considered: 1) no feature reuse; 2) sequential feature reuse; 3) the proposed common feature reuse. Further, the effectiveness of the generative network is also verified by replacing it with the pixel updating proposed in (Yin et al. 2019).

Visualization. The synthetic results on ImageNet are visualized in Figure 3, where all samples are obtained by deploying the 50-step FastDFKD on an off-the-shelf ResNet-50 classifier. Compared with existing methods that either require solving a 2000-step mini-batch optimization (Yin et al. 2019) or train 1000 generative models for synthesis (Luo et al. 2020), our proposed method can craft plausible samples within a few steps.

5 Conclusions

In this work, we propose a novel approach, termed as FastDFKD, to learn a meta-generator for fast data-free knowledge distillation, which is able to achieve 10 × and even 100× acceleration on CIFAR, NYUv2, and ImageNet. As the first attempt to improve the efficiency of data-free learning, the proposed approach successfully crafted a synthetic ImageNet in 6.28 hours, making data-free KD more applicable in real-world applications.

Acknowledgements. This work is supported by National Natural Science Foundation of China (U20B2066, 61976186,), Key Research and Development Program of Zhejiang Province (2020C01023), the Major Scientific Research Project of Zhejiang Lab (No. 2019KD0AC01), the Fundamental Research Funds for the Central Universities, Alibaba-Zhejiang University Joint Research Institute of Frontier Technologies, NUS ARTIC (Project Reference: ECT-RP2), and Faculty Research Committee Grant (R-263-000-E95-133).

References

Chawla, A.; Yin, H.; Molchanov, P.; and Alvarez, J. 2021. Data-Free Knowledge Distillation for Object Detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 3289–3298.

Chen, H.; Guo, T.; Xu, C.; Li, W.; Xu, C.; Xu, C.; and Wang, Y. 2021. Learning Student Networks in the Wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6428–6437.

Chen, H.; Wang, Y.; Xu, C.; Yang, Z.; Liu, C.; Shi, B.; Xu, C.; Xu, C.; and Tian, Q. 2019. Data-free learning of student networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3514–3522.

Chen, L.-C.; Papandreou, G.; Schroff, F.; and Adam, H. 2017. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587* .

Choi, Y.; Choi, J.; El-Khamy, M.; and Lee, J. 2020. Data-free network quantization with adversarial knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 710–711.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Deng, X.; and Zhang, Z. 2021. Graph-Free Knowledge Distillation for Graph Neural Networks. *arXiv preprint arXiv:2105.07519* .

Fang, G.; Bao, Y.; Song, J.; Wang, X.; Xie, D.; Shen, C.; and Song, M. 2021a. Mosaicking to Distill: Knowledge Distillation from Out-of-Domain Data. In *Thirty-Fifth Conference on Neural Information Processing Systems*.

Fang, G.; Song, J.; Shen, C.; Wang, X.; Chen, D.; and Song, M. 2019. Data-free adversarial distillation. *arXiv preprint arXiv:1912.11006* .

Fang, G.; Song, J.; Wang, X.; Shen, C.; Wang, X.; and Song, M. 2021b. Contrastive Model Inversion for Data-Free Knowledge Distillation. *arXiv preprint arXiv:2105.08584* .

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 1126–1135. PMLR.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* .

Hospedales, T.; Antoniou, A.; Micaelli, P.; and Storkey, A. 2020. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439* .

Kolesnikov, A.; Beyer, L.; Zhai, X.; Puigcerver, J.; Yung, J.; Gelly, S.; and Houlsby, N. 2020. Big Transfer (BiT): General Visual Representation Learning.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images. *technical report* .

Lopes, R. G.; Fenu, S.; and Starner, T. 2017. Data-free knowledge distillation for deep neural networks. *arXiv preprint arXiv:1710.07535* .

Luo, L.; Sandler, M.; Lin, Z.; Zhmoginov, A.; and Howard, A. 2020. Large-scale generative data-free distillation. *arXiv preprint arXiv:2012.05578* .

Ma, X.; Shen, Y.; Fang, G.; Chen, C.; Jia, C.; and Lu, W. 2020. Adversarial Self-Supervised Data-Free Distillation for Text Classification. *arXiv preprint arXiv:2010.04883* .

Micaelli, P.; and Storkey, A. 2019. Zero-shot knowledge transfer via adversarial belief matching. *arXiv preprint arXiv:1905.09768* .

Nathan Silberman, Derek Hoiem, P. K.; and Fergus, R. 2012. Indoor Segmentation and Support Inference from RGBD Images. In *ECCV*.

Nichol, A.; Achiam, J.; and Schulman, J. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999* .

Shen, C.; Wang, X.; Song, J.; Sun, L.; and Song, M. 2019. Amalgamating knowledge towards comprehensive classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3068–3075.

Yang, Y.; Qiu, J.; Song, M.; Tao, D.; and Wang, X. 2020. Distilling knowledge from graph convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7074–7083.

Ye, J.; Ji, Y.; Wang, X.; Ou, K.; Tao, D.; and Song, M. 2019. Student becoming the master: Knowledge amalgamation for joint scene parsing, depth estimation, and more. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2829–2838.

Yin, H.; Molchanov, P.; Li, Z.; Alvarez, J. M.; Mallya, A.; Hoiem, D.; Jha, N. K.; and Kautz, J. 2019. Dreaming to Distill: Data-free Knowledge Transfer via DeepInversion. *arXiv preprint arXiv:1912.08795* .

Zhu, Z.; Hong, J.; and Zhou, J. 2021. Data-Free Knowledge Distillation for Heterogeneous Federated Learning. *arXiv preprint arXiv:2105.10056* .