Data-Free Learning of Student Networks

Hanting Chen[1*], Yunhe Wang[2], Chang Xu[3†], Zhaohui Yang[1*], Chuanjian Liu[2], Boxin Shi[4,5], Chunjing Xu[2], Chao Xu[1], Qi Tian[2]

[1] Key Lab of Machine Perception (MOE), CMIC, School of EECS, Peking University, China

[2] Huawei Noah's Ark Lab, [3] School of Computer Science, Faculty of Engineering, The University of Sydney, Australia

[4] National Engineering Laboratory for Video Technology, Peking University, [5] Peng Cheng Laboratory

{chenhanting, zhaohuiyang, shiboxin}@pku.edu.cn, c.xu@sydney.edu.au

{yunhe.wang, liuchuanjian, xuchunjing, tian.qi1}@huawei.com, xuchao@cis.pku.edu.cn

Abstract

*Learning portable neural networks is very essential for computer vision for the purpose that pre-trained heavy deep models can be well applied on edge devices such as mobile phones and micro sensors. Most existing deep neural network compression and speed-up methods are very effective for training compact deep models, when we can directly access the training dataset. However, training data for the given deep network are often unavailable due to some practice problems (*e.g. *privacy, legal issue, and transmission), and the architecture of the given network are also unknown except some interfaces. To this end, we propose a novel framework for training efficient deep neural networks by exploiting generative adversarial networks (GANs). To be specific, the pre-trained teacher networks are regarded as a fixed discriminator and the generator is utilized for derivating training samples which can obtain the maximum response on the discriminator. Then, an efficient network with smaller model size and computational complexity is trained using the generated data and the teacher network, simultaneously. Efficient student networks learned using the proposed Data-Free Learning (DAFL) method achieve* 92.22% *and* 74.47% *accuracies using ResNet-18 without any training data on the CIFAR-10 and CIFAR-100 datasets, respectively. Meanwhile, our student network obtains an* 80.56% *accuracy on the CelebA benchmark.*

1. Introduction

Deep convolutional neural networks (CNNs) have been successfully used in various computer vision applications such as image classification [24, 11], object detection [21] and semantic segmentation [15]. However, launching most of the widely used CNNs requires heavy computation and storage, which can only be used on PCs with modern GPU cards. For example, over $500MB$ of memory and over $10^{10}\times$ multiplications are demanded for processing one image using VGGNet [24], which is almost impossible to be applied on edge devices such as autonomous cars and micro robots. Although these pre-trained CNNs have a number of parameters, Han *et al*. [6] showed that discarding over 85% of weights in a given neural network would not obviously damage its performance, which demonstrates that there is a significant redundancy in these CNNs.

In order to compress and speed-up pre-trained heavy deep models, various effective approaches have been proposed recently. For example, Gong *et al*. [5] utilized vector quantization approach to represent similar weights as cluster centers. Denton *et al*. [3] exploited low-rank decomposition to process the weight matrices of fully-connected layers. Chen *et al*. [1] proposed a hashing based method to encode parameters in CNNs. Han *et al*. [6] employed pruning, quantization and Huffman coding to obtain a compact deep CNN with lower computational complexity. Hinton *et al*. [8] proposed the knowledge distillation approach, which distills the information of the pre-trained teacher network for learning a portable student network, *etc*.

Although the above mentioned methods have made tremendous efforts on benchmark datasets and models, an important issue has not been widely noticed, *i.e*. most existing network compression and speed-up algorithms have a strong assumption that training samples of the original network are available. However, the training dataset is routinely unknown in real-world applications due to privacy and transmission limitations. For instance, users do not want to let their photos leaked to others, and some of the training datasets are too huge to quickly upload to the cloud. In addition, parameters and architecture of pre-trained networks are also unknown sometimes except the input and output layers. Therefore, conventional methods cannot be

*This work was done while visiting Huawei Noah's Ark Lab

†corresponding author

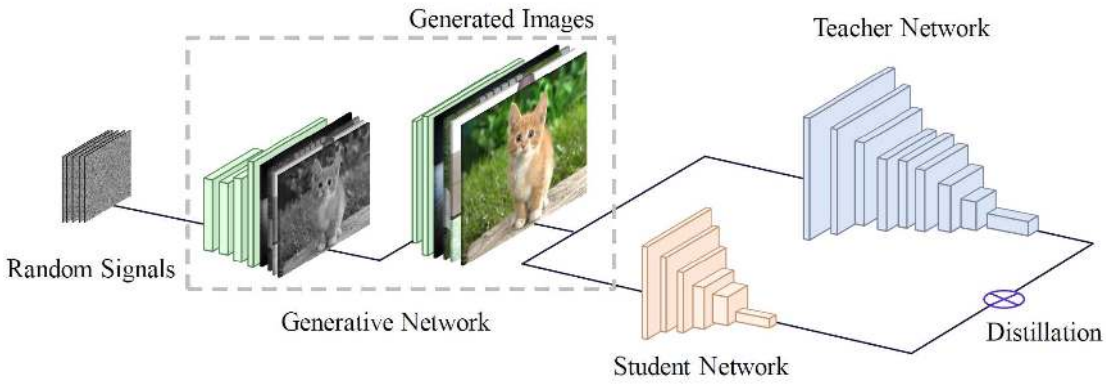


Figure 1. The diagram of the proposed method for learning efficient deep neural networks without the training dataset. The generator is trained for approximating images in the original training set by extracting useful information from the given network. Then, the portable student network can be effective learned by using generated images and the teacher network

directly used for learning portable deep models under these practice constrains.

Nevertheless, only a few works have been proposed for compressing deep models without training data. Lopes *et al*. [16] utilized the "meta-data" (*e.g*. means and standard deviation of activations from each layer) recorded from the original training dataset, which is not provided for most well-trained CNNs. Srinivas and Babu [26] compressed the pre-trained network by merging similar neurons in fully-connected layers. However, the performance of compressed networks using these methods is much lower than that of the original network, due to they cannot effectively utilize the pre-trained neural networks. To address the aforementioned problem, we propose a novel framework for compressing deep neural networks without the original training dataset. To be specific, the given heavy neural network is regarded as a fixed discriminator. Then, a generative network is established for alternating the original training set by extracting information from the network during the adversarial procedure, which can be utlized for learning smaller networks with acceptable performance. The superiority of the proposed method is demonstrated through extensive experiments on benchmark datasets and models.

Rest of this paper is organized as follows. Section 2 investigates related works on CNN compression algorithms. Section 3 proposes the data-free teacher-student paradigm by exploiting GAN. Section 4 illustrates experimental results of the proposed method on benchmark datasets and models and Section 5 concludes the paper.

2. Related Works

Based on different assumptions and applications, existing portable network learning methods can be divided into two categories, *i.e*. data-driven and data-free methods.

2.1. Data-Driven Network Compression

In order to learn efficient deep neural networks, a number of methods have been proposed to eliminate redundancy in pre-trained deep models. For example, Gong *et al*. [5] employed the vector quantization scheme to represent similar weights in neural networks. Denton *et al*. [3] exploited the singular value decomposition (SVD) approach to decompose weight matrices of fully-connected layers. Han *et al*. [6] proposed the pruning approach for removing subtle weights in pre-trained neural networks. Wang *et al*. [27] further introduced the discrete cosine transform (DCT) bases and converted convolution filters into the frequency domain to achieve higher compression and speed-up ratios. Yang *et al*. [28] used a set of Lego filters to build efficient CNNs.

Besides eliminating redundant weights or filters, Hinton *et al*. [8] proposed a knowledge distillation (KD) paradigm for transferring useful information from a given teacher network to a portable student network. Yim *et al*. [29] introduced the FSP (Flow of Solution Procedure) matrix to inherit the relationship between features from two layers. Li *et al*. [13] further presented a feature mimic framework to train efficient convolutional networks for objective detection. In addition, Rastegari *et al*. [20] and Courbariaux *et al*. [2] explored binarized neural networks to achieve considerable compression and speed-up ratios, which weights are -1/1 or -1/0/1, *etc*.

Although the above mentioned algorithms obtained promising results on most of benchmark datasets and deep models, they cannot be effectively launched without the original training dataset. In practice, the training dataset could be unavailable for some reasons, *e.g*. transmission limitations and privacy. Therefore, it is necessary to study the data-free approach for compressing neural networks.

2.2. Data-Free Network Compression

There are only a few methods that are proposed for compressing deep neural networks without the original training dataset. Srinivas and Babu [26] proposed to directly merge similar neurons in fully-connected layers, which cannot be applied on convolutional layers and networks which detail architectures and parameters information are unknown. In addition, Lopes *et al*. [16] attempted to reconstruct the original data from "meta-data" and utilize the knowledge distillation scheme to learn a smaller network.

Since the fine-tuning procedure cannot be accurately conducted without the original training dataset, performance of compressed methods by existing algorithms is worse than that of baseline models. Therefore, an effective data-free approach for learning efficient CNNs with comparable performance is highly required.

3. Data-free Student Network learning

In this section, we will propose a novel data-free framework for compressing deep neural networks by embedding a generator network into the teacher-student learning paradigm.

3.1. Teacher-Student Interactions

As mentioned above, the original training dataset is not usually provided by customers for various concerns. In addition, parameters and detailed architecture information could also be unavailable sometimes. Thus, we propose to utilized the teacher-student learning paradigm for learning portable CNNs.

Knowledge Distillation (KD) [8] is a widely used approach to transfer the output information from a heavy network to a smaller network for achieving higher performance, which does not utilize parameters and the architecture of the given network. Although the given deep models may only be provided with limited interfaces (*e.g*. input and output interfaces), we can transfer the knowledge to inherit the useful information from the teacher networks. Let $\mathcal{N}_T$ and $\mathcal{N}_S$ denote the original pre-trained convolutional neural network (teacher network) and the desired portable network (student network), the student network can be optimized using the following loss function based on knowledge distillation:

$$\mathcal{L}_{KD} = \frac{1}{n}\sum_{i}\mathcal{H}_{cross}(\mathrm{y}_S^i, \mathrm{y}_T^i). \tag{1}$$

where $\mathcal{H}_{cross}$ is the cross-entropy loss, $\mathrm{y}_T^i = \mathcal{N}_T(\mathrm{x}^i)$ and $\mathrm{y}_S^i = \mathcal{N}_S(\mathrm{x}^i)$ are the outputs of the teacher network $\mathcal{N}_T$ and student network $\mathcal{N}_S$, respectively. Therefore, utilizing the knowledge transfer technique, a portable network can be optimized without the specific architecture of the given network.

3.2. GAN for Generating Training Samples

In order to learn portable network without original data, we exploit GAN to generate training samples utilizing the available information of the given network.

Generative adversarial networks (GANs) have been widely applied for generating samples. GANs consist of a generator G and a discriminator D. G is expected to generate desired data while D is trained to identify the differences between real images and those produced by the generator. To be specific, given an input noise vector z, G maps z to the desired data x, *i.e*. $G : \mathrm{z} \rightarrow \mathrm{x}$. On the other hand, the goal of D is to distinguish the real data from synthetic data $G(\mathrm{z})$. For an aribitrary vanilla GAN, the objective function can be formulated as

$$\begin{aligned}\mathcal{L}_{GAN} =& \mathbb{E}_{y\sim p_{data}(y)}[\log D(y)] \\ &+\mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))].\end{aligned} \tag{2}$$

In the adversarial procedure, the generator is continuously upgraded according to the training error produced by D. The optimal G is obtained by optimizing the following problem

$$G^* = \arg\min_G \mathbb{E}_{z\sim p_z(z)}[\log(1-D^*(G(z)))], \tag{3}$$

where D^* is the optimal discriminator. Adversarial learning techniques can be naturally employed to synthesize training data. However according to Eq. (2), the discriminator requires real images for training. In the absence of training data, it is thus impossible to train the discriminator as vanilla GANs.

Recent works [19] have proved that the discriminator D can learn the hierarchy of representations from samples, which encourages the generalization of D in other tasks like image classification. Odena [18] further suggested that the tasks of discrimination and classification can improve each other. Instead of training a new discriminator as vanilla GANs, the given deep neural network can extract semantic features from images as well, since it has already been well trained on large-scale datasets. Hence, we propose to regard this given deep neural network (*e.g*. ResNet-50 [7]) as a fixed discriminator. Therefore, G can be optimized directly without training D together, *i.e*. the parameters of original network D are fixed during training G. In addition, the output of the discriminator is a probability indicating whether an input image is real or fake in vanilla GANs. However, given the teacher deep neural network as the discriminator, the output is to classify images to different concept sets, instead of indicating the reality of images. The loss function in vanilla GANs is therefore inapplicable for approximating the original training set. Thus, we conduct thorough analysis on real images and their responses on this teacher network. Several new loss functions will be devised to reflect our observations.

On the image classification task, the teacher deep neural network adopts the cross entropy loss in the training stage, which enforces the outputs to be close to ground-truth labels of inputs. Specifically for multi-class classification, the outputs are encouraged to be one-hot vectors, where only one entry is 1 and all the others are 0s. Denote the generator and the teacher network as G and $\mathcal{N}_T$, respectively. Given a set of random vector $\{\mathrm{z}^1, \mathrm{z}^2, \cdots, \mathrm{z}^n\}$, images generated from these vectors are $\{\mathrm{x}^1, \mathrm{x}^2, \cdots, \mathrm{x}^n\}$, where $\mathrm{x}^i = G(\mathrm{z}^i)$. Inputting these images into the teacher network, we can obtain the outputs $\{\mathrm{y}_T^1, \mathrm{y}_T^2, \cdots, \mathrm{y}_T^n\}$ with $\mathrm{y}_T^i = \mathcal{N}_T(\mathrm{x}^i)$. The predicted labels $\{\mathrm{t}^1, \mathrm{t}^2, \cdots, \mathrm{t}^n\}$ are then calculated by $\mathrm{t}^i = \arg\max_j (\mathrm{y}_T^i)_j$. If images generated by G follow the same distribution as that of the training data of the teacher network, they should also have similar outputs as the training data. We thus introduce the one-hot loss, which encourages the outputs of generated images by the teacher network to be close to one-hot like vectors. By taking $\{\mathrm{t}^1, \mathrm{t}^2, \cdots, \mathrm{t}^n\}$ as pseudo ground-truth labels, we formulate the one-hot loss function as

$$\mathcal{L}_{oh} = \frac{1}{n} \sum_i \mathcal{H}_{cross}(\mathrm{y}_T^i, \mathrm{t}^i), \tag{4}$$

where $\mathcal{H}_{cross}$ is the cross-entropy loss function. By introducing the one-hot loss, we expect that a generated image can be classified into one particular category concerned by the teacher network with a higher probability. In other words, we pursue synthetic images that are exclusively compatible with the teacher network, rather than general real images for any scenario.

Besides predicted class labels by DNNs, intermediate features extracted by convolution layers are also important representations of input images. A large number of works have investigated the interpretability of deep neural networks [30, 22, 4]. Features extracted by convolution filters are supposed to contain valuable information about the input images. In particular, Zhang *et al*. [31] assigned each filter in a higher convolution layer with a part of object, which demonstrates that each filter stands for different semantics. We denote features of x^i extracted by the teacher network as f_T^i, which corresponds to the output before the fully-connected layer. Since filters in the teacher DNNs have been trained to extract intrinsic patterns in training data, feature maps tend to receive higher activation value if input images are real rather than some random vectors. Hence, we define an activation loss function as:

$$\mathcal{L}_a = -\frac{1}{n} \sum_i \|f_T^i\|_1, \tag{5}$$

where $\|\cdot\|_1$ is the conventional l_1 norm.

Moreover, to ease the training procedure of a deep neural network, the number of training examples in each category is usually balanced, *e.g*. there are 6,000 images in each class in the MNIST dataset. We employ the information entropy loss to measure the class balance of generated images. Specifically, given a probability vector $\mathrm{p} = (p_1, p_2, \cdots, p_k)$, the information entropy, which measures the degree of confusion, of p is calculated as $\mathcal{H}_{info}(\mathrm{p}) = -\frac{1}{k} \sum_i p_i \log(p_i)$. The value of $\mathcal{H}_{info}(\mathrm{p})$ indicates the amount of information that p owns, which will take the maximum when all variables equal to $\frac{1}{k}$. Given a set of output vectors $\{\mathrm{y}_T^1, \mathrm{y}_T^2, \cdots, \mathrm{y}_T^n\}$, where $\mathrm{y}_T^i = \mathcal{N}_T(\mathrm{x}^i)$, the frequency distribution of generated images for every class is $\frac{1}{n} \sum_i \mathrm{y}_T^i$. The information entropy loss of generated images is therefore defined as

$$\mathcal{L}_{ie} = -\mathcal{H}_{info}(\frac{1}{n} \sum_i \mathrm{y}_T^i). \tag{6}$$

When the loss takes the minimum, every element in vector $\frac{1}{n} \sum_i \mathrm{y}_S^i$ would equal to $\frac{1}{k}$, which implies that G could generate images of each category with roughly the same probability. Therefore, minimizing the information entropy of generated images can lead to a balanced set of synthetic images.

By combining the aforementioned three loss functions, we obtain the final objective function

$$\mathcal{L}_{Total} = \mathcal{L}_{oh} + \alpha \mathcal{L}_a + \beta \mathcal{L}_{ie}, \tag{7}$$

Algorithm 1 DAFL for learning portable student networks.

Input: A given teacher network $\mathcal{N}_T$, parameters of different objects: α and β.

Initialize the generator G, the student network $\mathcal{N}_S$ with fewer memory usage and computational complexity;
repeat
  Module 1: Training the Generator.
  Randomly generate a batch of vector: $\{\mathrm{z}^i\}_{i=1}^n$;
  Generate the training samples: $x \leftarrow G(\mathrm{z})$;
  Employ the teacher network on the mini-batch:
    $[\mathrm{y}_T, \mathrm{t}, f_T] \leftarrow \mathcal{N}_T(\mathrm{x})$;
  Calculate the loss function $\mathcal{L}_{Total}$ (Fcn.7):
  Update weights in G using back-propagation;
  Module 2: Training the student network.
  Randomly generate a batch of vector $\{\mathrm{z}^i\}_{i=1}^n$;
  Utlize the generator on the mini-batch: $x \leftarrow G(\mathrm{z})$;
  Employ the teacher network and the student network on the mini-batch simultaneously:
    $\mathrm{y}_S \leftarrow \mathcal{N}_S(\mathrm{x})$, $\mathrm{y}_T \leftarrow \mathcal{N}_T(\mathrm{x})$;
  Calculate the knowledge distillation loss:
    $\mathcal{L}_{KD} \leftarrow \frac{1}{n} \sum_i \mathcal{H}(\mathrm{y}_S^i, \mathrm{y}_T^i)$;
  Update weights in $\mathcal{N}_S$ according to the gradient;
until convergence

Output: The student network $\mathcal{N}_S$.

Table 1. Classification result on the MNIST dataset.

Algorithm	Required data	LeNet-5 [12]			HintonNet [8]		
		Accuracy	FLOPs	#params	Accuracy	FLOPs	#params
Teacher	Original data	98.91%	∼436K	∼62K	98.39%	∼2.39M	∼2.4M
Standard back-propagation	Original data	98.65%	∼144K	∼16K	98.11%	∼1.28M	∼ 1.28M
Knowledge Distillation [8]	Original data	98.91%	∼144K	∼16K	98.39%	∼1.28M	∼ 1.28M
Normal distribution	No data	88.01%	∼144K	∼16K	87.58%	∼1.28M	∼ 1.28M
Alternative data	USPS dataset	94.56%	∼144K	∼16K	93.99%	∼1.28M	∼ 1.28M
Meta data [16]	Meta data	92.47%	∼144K	∼16K	91.24%	∼1.28M	∼ 1.28M
Data-Free Learning (DAFL)	No data	98.20%	∼144K	∼16K	97.91%	∼1.28M	∼ 1.28M

where α and β are hyper parameters for balancing three different terms. By minimizing the above function, the optimal generator can synthesize images that have the similar distribution as that of the training data previously used for training the teacher network (*i.e*. the discriminator network).

It is noted that some previous works [23, 17] could synthesize images by optimizing the input of the neural network using back-propagation. But it is difficult to generate abundant images for the subsequent student network training, for each synthetic image leads to an independent optimization problem solved by back-propagation. In contrast, the proposed method can imitate the distribution of training data directly, which is more flexible and efficient to generate new images.

3.3. Optimization

The learning procedure of our algorithm can be divided into two stages of training. First, we regard the well-trained teacher network as a fixed discriminator. Using the loss function $\mathcal{L}_{Total}$ in Eq. 7, we optimize a generator G to generate images that follow the similar distribution as that of the original training images for the teacher network. Second, we utilize the knowledge distillation approach to directly transfer knowledge from the teacher network to the student network. The student network with fewer parameters is then optimized using the KD loss $\mathcal{L}_{KD}$ in Eq. 1. The diagram of the proposed method is shown in Figure 1.

We use stochastic gradient descent (SGD) method to optimize the image generator G and the student network $\mathcal{N}_S$. In the training of G, the first term of $\mathcal{L}_{Total}$ is the cross entropy loss, which can be trained traditionally. The second term $\mathcal{L}_a$ in Eq. 7 is exactly a linear operation, and the gradient of $\mathcal{L}_a$ with respect to f_T^i can be easily calculated as:

$$\frac{\partial \mathcal{L}_a}{\partial f_T^i} = -\frac{1}{n}\text{sgn}(f_T^i), \tag{8}$$

where $\text{sgn}(\cdot)$ denotes sign function. Parameters W_G in G will be updated by:

$$\frac{\partial \mathcal{L}_a}{\partial W_G} = \sum_i \frac{\partial \mathcal{L}_a}{\partial f_T^i} \cdot \frac{\partial f_T^i}{\partial W_G}, \tag{9}$$

where $\frac{\partial f_T^i}{\partial W_G}$ is the gradient of the feature f_T^i. The gradient of the final term $\mathcal{L}_{ie}$ with respect to y_T^i can be easily calculated as:

$$\frac{\partial \mathcal{L}_{ie}}{\partial \text{y}_T^i} = -\frac{1}{n}\text{y}^i[\log(\frac{1}{n}\sum_j \text{y}_T^j) + \mathbf{1}], \tag{10}$$

where $\mathbf{1}$ denotes n-dimensional vector with all values as 1. Parameters in G will be additionally updated by:

$$\frac{\partial \mathcal{L}_{ie}}{\partial W_G} = \sum_i \frac{\partial \mathcal{L}_{ie}}{\partial \text{y}_T^i} \cdot \frac{\partial \text{y}_T^i}{\partial W_G}. \tag{11}$$

Detailed procedures of the proposed Data-Free Learning (DAFL) scheme for learning efficient student neural networks is summarized in Algorithm 1.

4. Experiments

In this section, we will demonstrate the effectiveness of our proposed data-free knowledge distillation method and conduct massive ablation experiments to have an explicit understanding of each component in the proposed method.

4.1. Experiments on MNIST

We first implement experiments on the MNIST dataset, which is composed of 28×28 pixel images from 10 categories (from 0 to 9). The whole dataset consists of 60,000 training images and 10,000 testing images. For choosing hyper-parameters of the proposed methods, we take 10,000 images as a validation set from training images. Then, we train models on the full 60,000 images to obtain the ultimate network.

To make a fair comparison, we follow the setting in [16]. Two architectures are used for investigating the performance of proposed method, *i.e*. a convolution-based architecture and a network consists of fully-connect layers. For convolution models, we use LeNet-5 [12] as the teacher model and LeNet-5-HALF (a modified version with half the number of channels per layer) as the student model. For the second architecture, the teacher network consists of two hidden layers of 1,200 units (Hinton-784-1200-1200-10) [8] and student network consists of two hidden layers of

Table 2. Effectiveness of different components of the proposed data-free learning method.

One-hot loss		✓			✓	✓		✓
Information entropy loss			✓			✓	✓	✓
Feature maps activation loss				✓	✓		✓	✓
Top 1 accuracy	88.01%	78.77%	88.14%	15.95%	42.07%	97.25%	95.53%	**98.20%**

800 units (Hinton-784-800-800-10). The student networks have significantly fewer parameters than teacher networks. For our method, α and β in Fcn.7 are 0.1 and 5, respectively, and are tuned on the validation set. The generator was trained for 200 epochs using Adam. We use a deep convolutional generator[1] following [19] and add a batch normalization at the end of the generator to smooth the sample values.

Table 1 reports the results of different methods on the MNIST datasets. On LeNet-5 models, the teacher network achieves a 98.91% accuracy while the student network using the standard back-propagation achieves a 98.65% accuracy, respectively. Knowledge distillation improved the accuracy of student network to 98.91%. These methods use the original data to train the student network. We then train a student network exploiting the proposed method to evaluate the effectiveness of the synthetic data.

We first use the data randomly generated from normal distribution to training the student network. By utilizing the knowledge distillation, the student network achieves only an 88.01% accuracy. In addition, we further use another handwritten digits dataset, namely USPS [9], to conduct the same experiment for training the student network. Although images in two datasets have similar properties, the student network learned using USPS can only obtain a 94.56% accuracy on the MNIST dataset, which demonstrates that it is extremely hard to find an alternative to the original training dataset. To this end, Lopes *et al*. [16] using the "meta data", which is the activation record of original data, to reconstruct the dataset and achieved only a 92.47% accuracy. Noted that the upper bound of the accuracy of student network is 98.65%, which could be achieved only if we could find a dataset whose distribution is same as the original dataset (*i.e*. MNIST dataset). The proposed method utilizing generative adversarial networks achieved a 98.20% accuracy, which is much close to this upper bound. Also, the accuracy of student network using the proposed algorithm is superior to these using other data (normal distribution, USPS dataset and reconstructed dataset using "meta data"), which suggest that our method could imitate the distribution of training dataset better.

On the fully-connected models, the classification accuracies of teacher and student network are 98.39% and 98.11%, respectively. Knowledge Distillation brought the performance of student network by transferring information from teacher network to 98.39%. However, in the absence of training data, the result became unacceptable. Randomly generated noise only achieves 87.58% accuracy and "meta data" [16] achieves a higher accuracy of 91.24%. Using USPS dataset as alternatives achieves an accuracy of 93.99%. The proposed method results in the highest performance of 97.91% among all methods without the original data, which demonstrates the effectiveness of the generator.

[1] https://github.com/eriklindernoren/PyTorch-GAN/blob/master/implementations/dcgan/dcgan.py

4.2. Ablation Experiments

In the above sections, we have tested and verified the effectiveness of the proposed generative method for student network learning without training data. However, there are a number of components, *i.e*. three terms in Eq. 7, when optimizing the generator. We further conduct the ablation experiments for an explicit understanding and analysis.

The ablation experiment is also conducted on the MNIST dataset. We used the LeNet-5 as a teacher network and LeNet-5-HALF as a student network. The training settings are same as those in Section 4.1. Table 2 reports the results of various design components. Using randomly generated samples, *i.e*. the generator G is not trained, the student network achieves an 88.01% accuracy. However, by utilizing one-hot loss and feature map activation loss or one of them, the generated samples are unbalanced, which results in the poor performance of the student networks. Only introducing information entropy loss, the student network achieves an 88.14% accuracy since the samples do not contain enough useful information. When combining $\mathcal{L}_{oh}$ or $\mathcal{L}_a$ with $\mathcal{L}_{ie}$, the student network achieves higher performance of 97.25% and 95.53%, respectively. Moreover, the accuracy of student network is 98.20% when using all these loss functions, which achieves the best performance. It is worth noticing that the combination of one-hot loss and information entropy is essential for training the generator, which is also utilized in some previous works [25, 10].

The ablation experiments suggest that each component of the loss function of G is meaningful. By applying the proposed method, G can generate balanced samples from different classes with a similar distribution as that in the original dataset, which is effective for the training of the student network.

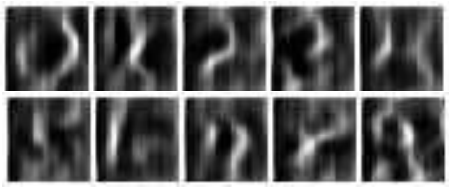
Table 3. Classification result on the CIFAR dataset.

Algorithm	Required data	FLOPS	#params	CIFAR-10	CIFAR-100
Teacher	Original data	∼1.16G	∼21M	95.58%	77.84%
Standard back-propagation	Original data	∼557M	∼11M	93.92%	76.53%
Knowledge Distillation [8]	Original data	∼557M	∼11M	94.34%	76.87%
Normal distribution	No data	∼557M	∼11M	14.89%	1.44%
Alternative data	Similar data	∼557M	∼11M	90.65%	69.88%
Data-Free Learning (DAFL)	No data	∼557M	∼11M	92.22%	74.47%

4.3. Visualization Results

After investigating the effectiveness of the proposed method, we further conduct visualization experiments on the MNIST dataset. There are 10 categories of handwritten digits from 0 to 9 in the MNIST dataset. The settings are same as that in Section 4.1.

(a) Averaged images on the MNIST dataset.

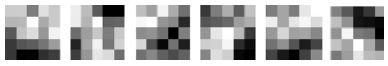


(b) Averaged images on the generated dataset.

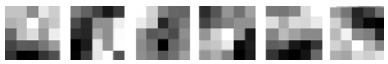
Figure 2. Visualization of averaged image in each category (from 0 to 9) on the MNIST dataset.

Figure 2 shows the visualization results of averaged images. Noted that the generated images are unlabeled, their classes are defined by the prediction of the teacher network. By exploiting the information of the given network as much as possible, we design loss function for the generator. Figure 2 (b) shows the mean of images of each class. Although no real image is provided, the generated images have similar patterns with the training images, which indicates that the generator can somehow learn the data distribution.

Filter visualization. Moreover, we visualize the filters of the LeNet-5 teacher network and student network in Figure 3. Though the student network is trained without real-world data, filters of the student network learned by the proposed method (see Figure 3 (b)) are still similar to those of the teacher network (see Figure 3 (a)). The visualization experiments further demonstrate that the generator can produce images that have similar patterns as the original images, and by utilizing generated samples, the student network could acquire valuable knowledge from the teacher network.



(a) Teacher filters.



(b) Student filters.

Figure 3. Visualization of filters in the first convolutional layer learned on the MNIST dataset. The top line shows filters trained using the original training dataset, and the bottom line shows filters obtained using samples generated by the proposed method.

4.4. Experiments on CIFAR

To further evaluate the effectiveness of our method, we conduct experiments on the CIFAR dataset. We used a ResNet-34 as the teacher network and ResNet-18 as the student network[2], which is complex and advanced for further investigating the effectiveness of the proposed method. These networks are optimized using Nesterov Accelerated Gradient (NAG) and the weight decay and the momentum are set as 5×10^{-4} and 0.9, respectively. We train the networks for 200 epochs and the initial learning rate is set as 0.1 and divided by 10 at 80 and 120 epochs, respectively. Random flipping, random crop and zero padding are used for data augmentation as suggested in [7]. G and the student networks of the proposed method are trained for 2,000 epochs and the other settings are same as those in MNIST experiments.

Table 3 reports the classification results on the CIFAR-10 and CIFAR-100 datasets. The teacher network achieves a 95.58% accuracy in CIFAR-10. The student network using knowledge distillation achieves a 94.34% accuracy, which is slightly higher than that of standard BP (93.92%).

We then explore to optimize the student network without true data. Since the CIFAR dataset is more complex than MNIST, it is impossible to optimize a student network using randomly generated data which follows the normal distribution. Therefore, we then regard the MNIST dataset without labels as an alternative data to train the student network using the knowledge distillation. The student network only achieves a 28.29% accuracy on the CIFAR-10 dataset. Moreover, we train the student network using the CIFAR-

[2]https://github.com/kuangliu/pytorch-cifar

100 dataset, which has considerable overlaps with the original CIFAR-10 dataset, but this network only achieves a 90.65% accuracy, which is obviously lower than that of the teacher model. In contrast, the student network trained utilizing the proposed method achieved a 92.22% accuracy with only synthetic data.

Besides CIFAR-10, we further verify the capability of the proposed method on the CIFAR-100 dataset, which has 100 categories and 600 images per class. Therefore, the dimensionality of the input random vectors for the generator in our method is increased to 1,000. The accuracy of the teacher network is 77.84% and that of the student network is only 76.53%, respectively. Using normal distribution data, MNIST, and CIFAR-10 to train the student network cannot obtain promising results, as shown in Table 3. In contrast, the student network learned by exploiting the proposed method obtained a 74.47% accuracy without any real-world training data.

4.5. Experiments on CelebA

Besides the CIFAR dataset, we conduct our experiments on the CelebA dataset, which contains 202,599 face images of pixel 224×224. To evaluate our approach fairly, we used AlexNet [11] to classify the most balanced attribute in CelebA [14] following the settings in [16]. The student network is AlexNet-Half, which number of filters is half of AlexNet. The original teacher network has about 57M parameters while the student network has only about 40M parameters. The networks is optimized for 100 epochs using Adam with a learning rate of 10^{-4}. We use an alternative model of DCGAN [19] to generate color images of 224×224. The hyper-parameters of the proposed method are same as those in MNIST and CIFAR experiments and G.

Table 4 reported the classification results of student networks on the CelebA dataset by exploiting the proposed method and state-of-the-art learning methods. The teacher network achieves an 81.59% accuracy and the student network using the standard BP achieves an 80.82% accuracy, respectively. Lopes *et al*. [16] achieves only a 77.56% accuracy rate using the "meta data". The accuracy of the student network trained using the proposed method is 80.03%, which is comparable with that of the teacher network.

Table 4. Classification result on the CelebA dataset.

Algorithm	FLOPS	Accuracy
Teacher	∼711M	81.59%
Standard back-propagation	∼222M	80.82%
Knowledge Distillation [8]	∼222M	81.35%
Meta data [16]	∼222M	77.56%
Data-Free Learning (DAFL)	∼222M	80.03%

4.6. Extended Experiments

Massive experiments are conducted on several benchmarks to verify the performance of the DAFL method for learning student networks using generated images. Wherein, architectures of used student networks are more portable than those of teacher networks. To investigate the difference between original training images and generated images, we use these generated images to train networks of the same architectures as those of teacher networks using the proposed methods. The results are reported in Table 5.

It can be found in Table 5 that LeNet-5 and HintonNet on the MNIST dataset achieve a 98.91% accuracy and a 98.39% accuracy, respectively. In contrast, accuracies of student networks trained from scratch with same architectures are 98.47% and 98.08%, respectively, which are very close to those of teacher networks. In addition, student networks on the CIFAR-10 and the CIFAR-100 datasets also obtain similar results to those of teacher networks. These results demonstrate that the proposed method can effectively approximate the original training dataset by extracting information from teacher networks. If the network architectures are given, we can even replicate the teacher networks and achieve similar accuracies.

Table 5. Classification results on various datasets.

Dataset	Model	Accuracy	
		Teacher	Student
MNIST	LeNet-5 [12]	98.91%	98.47%
MNIST	HintonNet [8]	98.39%	98.08%
CIFAR-10	ResNet-34 [7]	95.58%	93.21%
CIFAR-100	ResNet-34 [7]	77.84%	75.32%
CelebA	AlexNet [11]	81.59%	80.56%

5. Conclusion

Conventional methods require the original training dataset for fine-tuning the compressed deep neural networks with an acceptable accuracy. However, the training set and detailed architecture information of the given deep network are routinely unavailable due to some privacy and transmission limitations. In this paper, we present a novel framework to train a generator for approximating the original dataset without the training data. Then, a portable networks can be learned effectively through the knowledge distillation scheme. Experiments on benchmark datasets demonstrate that the proposed method DAFL method is able to learn portable deep neural networks without any training data.

Acknowledgement

This work is supported by National Natural Science Foundation of China under Grant No. 61876007, 61872012 and Australian Research Council Project DE-180101438.

References

[1] Wenlin Chen, James T Wilson, Stephen Tyree, Kilian Q Weinberger, and Yixin Chen. Compressing neural networks with the hashing trick. In *ICML*, 2015. 1

[2] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*, 2016. 2

[3] Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *NIPS*, 2014. 1, 2

[4] Yinpeng Dong, Hang Su, Jun Zhu, and Fan Bao. Towards interpretable deep neural networks by leveraging adversarial examples. *arXiv preprint arXiv:1708.05493*, 2017. 4

[5] Yunchao Gong, Liu Liu, Ming Yang, and Lubomir Bourdev. Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*, 2014. 1, 2

[6] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015. 1, 2

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 3, 7, 8

[8] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1, 2, 3, 5, 7, 8

[9] Jonathan J. Hull. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994. 6

[10] Himalaya Jain, Joaquin Zepeda, Patrick Pérez, and Rémi Gribonval. Subic: A supervised, structured binary code for image search. In *ICCV*, pages 833–842, 2017. 6

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012. 1, 8

[12] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5, 8

[13] Quanquan Li, Shengying Jin, and Junjie Yan. Mimicking very efficient network for object detection. In *CVPR*, pages 7341–7349. IEEE, 2017. 2

[14] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, pages 3730–3738, 2015. 8

[15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015. 1

[16] Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. Data-free knowledge distillation for deep neural networks. *arXiv preprint arXiv:1710.07535*, 2017. 2, 3, 5, 6, 8

[17] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *CVPR*, pages 5188–5196, 2015. 5

[18] Augustus Odena. Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583*, 2016. 3

[19] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 3, 6, 8

[20] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, pages 525–542. Springer, 2016. 2

[21] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015. 1

[22] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, Dhruv Batra, et al. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, pages 618–626, 2017. 4

[23] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013. 5

[24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 1

[25] Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015. 6

[26] Suraj Srinivas and R Venkatesh Babu. Data-free parameter pruning for deep neural networks. *arXiv preprint arXiv:1507.06149*, 2015. 2, 3

[27] Yunhe Wang, Chang Xu, Shan You, Dacheng Tao, and Chao Xu. Cnnpack: packing convolutional neural networks in the frequency domain. In *NIPS*, pages 253–261, 2016. 2

[28] Zhaohui Yang, Yunhe Wang, Chuanjian Liu, Hanting Chen, Chunjing Xu, Boxin Shi, Chao Xu, and Chang Xu. Legonet: Efficient convolutional neural networks with lego filters. In *ICML*, pages 7005–7014, 2019. 2

[29] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *CVPR*, 2017. 2

[30] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833. Springer, 2014. 4

[31] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. In *CVPR*, pages 8827–8836, 2018. 4