Data-Free Network Quantization With Adversarial Knowledge Distillation

Yoojin Choi[1], Jihwan Choi[2]*, Mostafa El-Khamy[1], Jungwon Lee[1]
[1]SoC R&D, Samsung Semiconductor Inc., San Diego, CA [2]DGIST, Korea
{yoojin.c,mostafa.e,jungwon2.lee}@samsung.com jhchoi@dgist.ac.kr

Abstract

Network quantization is an essential procedure in deep learning for development of efficient fixed-point inference models on mobile or edge platforms. However, as datasets grow larger and privacy regulations become stricter, data sharing for model compression gets more difficult and restricted. In this paper, we consider data-free network quantization with synthetic data. The synthetic data are generated from a generator, while no data are used in training the generator and in quantization. To this end, we propose data-free adversarial knowledge distillation, which minimizes the maximum distance between the outputs of the teacher and the (quantized) student for any adversarial samples from a generator. To generate adversarial samples similar to the original data, we additionally propose matching statistics from the batch normalization layers for generated data and the original data in the teacher. Furthermore, we show the gain of producing diverse adversarial samples by using multiple generators and multiple students. Our experiments show the state-of-the-art data-free model compression and quantization results for (wide) residual networks and MobileNet on SVHN, CIFAR-10, CIFAR-100, and Tiny-ImageNet datasets. The accuracy losses compared to using the original datasets are shown to be very minimal.

1. Introduction

Deep learning is now leading many performance breakthroughs in various computer vision tasks [1]. The state-of-the-art performance of deep learning came with over-parameterized deep neural networks, which enable extracting useful representations (features) of the data automatically for a target task, when trained on a very large dataset. The optimization framework of deep neural networks with stochastic gradient descent has become very fast and efficient recently with the backpropagation technique [2, Section 6.5], using hardware units specialized for matrix/tensor computations such as graphical processing units (GPUs). The benefit of over-parameterization is empirically shown to be the key factor of the great success of deep learning, but once we find a well-trained high-accuracy model, its deployment on various inference platforms faces different requirements and challenges [3, 4]. In particular, to deploy pre-trained models on resource-limited platforms such as mobile or edge devices, computational costs and memory requirements are the critical factors that need to be considered carefully for efficient inference. Hence, model compression, also called network compression, is an important procedure for development of efficient inference models.

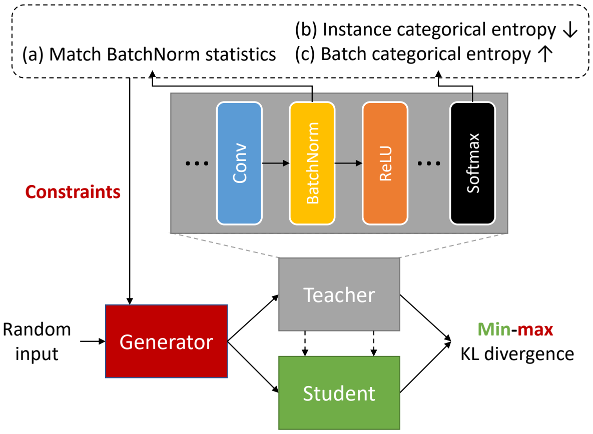

(b) Instance categorical entropy ↓
(a) Match BatchNorm statistics
(c) Batch categorical entropy ↑
Constraints
…
Conv
BatchNorm
ReLU
…
Softmax
Teacher
Random input
Generator
Student
Min-max
KL divergence

Figure 1: Data-free adversarial knowledge distillation. We minimize the maximum of the Kullback-Leibler (KL) divergence between the teacher and student outputs. In the maximization step for training the generator to produce adversarial images, the generator is constrained to produce synthetic images similar to the original data by matching the statistics from the batch normalization layers of the teacher.

Model compression includes various methods such as (1) weight pruning, (2) network quantization, and (3) distillation to a network with a more efficient architecture. Weight pruning and network quantization reduce the computational cost as well as the storage/memory size, without altering the network architecture. Weight pruning compresses a model by removing redundant weights completely from it, i.e., by setting them to be zero, so we can skip computation as well as memorization for the pruned weights [5–12]. Net-

*Work done when the author was with Samsung as a visiting scholar.

work quantization reduces the memory footprint for weights and activations by quantization and is usually followed by lossless source coding for compression [13–18]. Moreover, the convolutional and fully-connected layers can be implemented with low-precision fixed-point operations, e.g., 8-bit fixed-point operations, to lower latency and to increase power efficiency [19–25]. On the other hand, the network architecture can be modified to be simpler and easier to implement on a target platform. For example, the number of layers and/or the number of channels in each layer can be curtailed. Conventional spatial-domain convolution can be replaced with more efficient depth-wise separable convolution as in MobileNet [26].

Knowledge distillation (KD) is a well-known knowledge transfer framework to train a small "student" network under a guidance of a large pre-trained "teacher" model. The original idea from Hinton et al. in [27] utilizes the soft decision output of a well-trained classification model in order to help to train another small-size network. This original idea was further refined and advanced mostly (1) by introducing losses of matching the outputs from intermediate layers of the teacher and student [28–30], and (2) by using more sophisticate distance metrics, for example, mutual relations for multiple samples [31, 32].

One issue with existing model compression approaches (including KD) is that they are developed under a strong assumption that the original training data is accessible during the compression procedure. As datasets get larger, the distribution of datasets becomes more expensive and more difficult. Additionally, data privacy and security have emerged as one of primary concerns in deep learning. Consequently, regulations and compliance requirements around security and privacy complicate both data sharing by the original model trainer and data collection by the model compressor, for example, in the case of medical and bio-metric data. Thus, there is a strong need to compress a pre-trained model without access to the original or even alternative datasets.

There have been some attempts to address the problem of data sharing in model compression [33–36]. They aim to perform KD without the original datasets. The early attempts in [33, 34] circumvent this issue by assuming that some form of compressed and/or partial information on the original training data is provided instead, called meta-data, to protect the privacy and to reduce the size of the data to share. Given a pre-trained model with meta-data, for example, statistics of activation outputs (feature maps) at any intermediate layers, the input is inferred in a backward manner so it matches the statistics in the meta-data. On the other hand, in [35, 36], generators are introduced to produce synthetic samples for KD. Chen et al. [35] proposed training a generator by using the pre-trained teacher as a fixed discriminator. Micaelli et al. [36] used the mismatch between the teacher and the student as an adversarial loss for training a generator to produce adversarial examples for KD. The previous generator-based KD framework in [35] is rather heuristic, relying on ad-hoc losses. In [36], adversarial examples can be any images far different from the original data, which degrade the KD performance.

In this paper, we propose an adversarial knowledge distillation framework, which minimizes the possible loss for a worst case (maximum loss) via adversarial learning, when the loss with the original training data is not accessible. The key difference from [36] lies in the fact that given any meta-data, we utilize them to constrain a generator in the adversarial learning framework. To avoid additional efforts to craft new meta-data to share, we use the statistics stored in batch normalization layers to constrain a generator to produce synthetic samples that mimic the original training data. Furthermore, we propose producing diverse synthetic samples by using multiple generators. We also empirically show that performing adversarial KD concurrently for multiple students yields better results. The proposed data-free adversarial KD framework is summarized in Figure 1.

For model compression, we perform experiments on two scenarios, (1) data-free KD and (2) data-free network quantization. The proposed scheme shows the state-of-the-art data-free KD performance on residual networks [37] and wide residual networks [38] for SVHN [39], CIFAR-10, CIFAR-100 [40], and Tiny-ImageNet[1], compared to the previous work [35, 36, 41]. Data-free network quantization (data-free quantization-aware training) has not been investigated before to the best of our knowledge. We use TensorFlow's quantization-aware training [24, 42] as the baseline scheme, and we evaluate the performance on residual networks, wide residual networks, and MobileNet trained on various datasets, when quantization-aware training is performed with the synthetic data generated from our data-free KD framework. The experimental results show marginal performance loss from the proposed data-free framework, compared to the case of using the original training datasets.

2. Related work

Data-free KD and quantization. Data-free KD attracts the interest with the need to compress pre-trained models for deployment on resource-limited mobile or edge platforms, while sharing original training data is often restricted due to privacy and license issues.

Some of early attempts to address this issue suggest using meta-data that are the statistics of intermediate features collected from a pre-trained model in [33, 34]. For example, the mean and variance of activation outputs for selected intermediate layers are proposed to be collected and assumed to be provided, instead of the original dataset. Given any meta-data, they find samples that help to train student net-

[1] `https://tiny-imagenet.herokuapp.com`

Table 1: Comparison of data-free KD and network quantization schemes based on (1) how they generate synthetic data and (2) whether they rely on meta-data or not.

Synthetic data	Meta-data	Data-free
Not used	N/A	[45]*
Inferred in the image domain	[33], [34]	[43], [41]*
Generated from generators	N/A	[35], [36], Ours*

* Used the statistics stored in batch normalization layers.

works by directly inferring them in the image domain such that they produce similar statistics as the meta-data when fed to the teacher. Recent approaches, however, aim to solve this problem without meta-data specifically designed for the data-free KD task. In [43], class similarities are computed from the weights of the last fully-connected layer, and they are used instead of meta-data. Very recently, it is proposed to use the statistics stored in batch normalization layers with no additional costs instead of crafting new meta-data [41].

On the other hand, some of the previous approaches introduce another network, called generator, that yields synthetic samples for training student networks [35, 36, 44]. They basically propose optimizing a generator so that the generator output produces high accuracy when fed to a pre-trained teacher. Adversarial learning was introduced to produce dynamic samples for which the teacher and the student poorly matched in their classification output and to perform KD on those adversarial samples [36].

To our knowledge, there are few works on data-free network quantization. Weight equalization and bias correction are proposed for data-free weight quantization in [45], but data-free activation quantization is not considered. Weight equalization is a procedure to transform a pre-trained model into a quantization-friendly model by re-distributing (equalizing) its weights across layers so they have smaller deviation in each layer and smaller quantization errors. The biases introduced in activations owing to weight quantization are calculated and corrected with no data but based on the statistics stored in batch normalization layers. We note that no synthetic data are produced in [45], and no data-free quantization-aware training is considered in [45]. We compare data-free KD and quantization schemes in Table 1.

Robust optimization. Robust optimization is a sub-field of optimization that addresses data uncertainty in optimization problems (e.g., see [46,47]). Under this framework, the objective and constraint functions are assumed to belong to certain sets, called "uncertainty sets." The goal is to make a decision that is feasible no matter what the constraints turn out to be, and optimal for the worst-case objective function. With no data provided, we formulate the problem of data-free KD into a robust optimization problem, while the uncertainty sets are decided based on the pre-trained teacher using the statistics at its batch normalization layers.

Adversarial attacks. Generating synthetic data that fool a pre-trained model is closely related to the problem of adversarial attacks (e.g., see [48]). Although their purpose is completely different from ours, the way of generating synthetic data (or adversarial samples) follows a similar procedure. In adversarial attacks, there are also two approaches, i.e., (1) generating adversarial images directly in the image domain [49–51] and (2) using generators to produce adversarial images [52–54].

Deep image prior. We also note that generator networks consisting of a series of convolutional layers can be used as a good regularizer that we can impose for image generation as prior [55]. Hence, we adopt generators, instead of adding any prior regularization [56] that is employed in [41] to obtain synthetic images without generators.

Generative adversarial networks (GANs). Adversarial learning is also well-known in GANs [57]. GANs are of great interest in deep learning for image synthesis problems. Mode collapse is one of well-known issues in GANs (e.g., see [58]). A straightforward but effective way to overcome mode collapse is to introduce multiple generators and/or multiple discriminators [59–62]. We also found that using multiple generators and/or multiple students (a student acts as a discriminator in our case) helps to produce diverse samples and avoid over-fitting in our data-free KD framework.

3. Data-free model compression

3.1. Knowledge distillation (KD)

Let $\mathbf{t}_\theta$ be a general non-linear neural network for classification, which is designed to yield a categorical probability distribution $P_\theta(y|\mathbf{x})$ for the label y of input $\mathbf{x}$ over the label set $\mathcal{C}$, i.e., $\mathbf{t}_\theta(\mathbf{x}) = [P_\theta(y|\mathbf{x})]_{y\in\mathcal{C}}$. Let $\mathbf{y}$ be the one-hot encoded ground-truth label y over the set $\mathcal{C}$ for input $\mathbf{x}$. The network $\mathbf{t}_\theta$ is pre-trained with a labeled dataset, called training dataset, of probability distribution $p(\mathbf{x}, \mathbf{y})$, as below:

$$\theta^* = \operatorname*{argmin}_\theta \mathbb{E}_{p(\mathbf{x},\mathbf{y})}[\mathcal{D}(\mathbf{y}, \mathbf{t}_\theta(\mathbf{x}))],$$

where $\mathbb{E}_{p(\mathbf{x},\mathbf{y})}$ is, in practice, an empirical expectation over the training dataset, and $\mathcal{D}$ stands for Kullback-Leibler (KL) divergence (e.g., see [63, Section 2.3]); note that the minimization of KL divergence is equivalent to the minimization of cross-entropy, given the distribution $p(\mathbf{x}, \mathbf{y})$.

Suppose that we want to train another neural network $\mathbf{s}_\phi$, called "student", possibly smaller and less complex than the pre-trained network $\mathbf{t}_{\theta^*}$, called "teacher." The student also produces its estimate of the categorical probability distribution for input $\mathbf{x}$ such that $\mathbf{s}_\phi(\mathbf{x}) = [Q_\phi(y|\mathbf{x})]_{y\in\mathcal{C}}$. Knowledge distillation [27] suggests to optimize the student by

$$\min_\phi \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\mathcal{D}(\mathbf{y}, \mathbf{s}_\phi(\mathbf{x})) + \lambda\mathcal{D}(\mathbf{t}_{\theta^*}(\mathbf{x}), \mathbf{s}_\phi(\mathbf{x}))\right], \quad (1)$$

where $\lambda \geq 0$; note that we omitted the temperature parameter for simplicity, which can be applied before softmax for $\mathbf{t}_{\theta^*}$ and $\mathbf{s}_\phi$ in the second KL divergence term of (1).

3.2. Data-free adversarial KD

As shown in (1), the original KD is developed under the assumption that a training dataset is given for the expectation over $p(\mathbf{x}, \mathbf{y})$. However, sharing a large dataset is expensive and sometimes not even possible due to privacy and security concerns. Hence, it is of interest to devise a method of KD in the situation where the training dataset is not accessible, but only a pre-trained teacher is given.

Robust optimization (e.g. see [46]) suggests minimizing the possible loss for a worst case scenario (maximum loss) with adversarial learning under data uncertainty, which is similar to the situation we encounter when we are not given a training dataset for optimization. To adopt the robust minimax optimization (also known as adversarial learning) in KD, we first introduce a generator network $\mathbf{g}_\psi$, which is used to produce synthetic adversarial data for the input to KD. Then, using the minimax approach, we propose data-free adversarial KD, which is given by

$$\min_\phi \max_\psi \{\mathbb{E}_{p(\mathbf{z})}[\mathcal{D}(\mathbf{t}_{\theta^*}(\mathbf{g}_\psi(\mathbf{z})), \mathbf{s}_\phi(\mathbf{g}_\psi(\mathbf{z})))] - \alpha L_\psi\}, \quad (2)$$

for $\alpha \geq 0$, where L_ψ is an additional loss that a pre-trained teacher can provide for the generator based on the generator output. We defer our proposed terms in L_ψ to Section 3.3.

Remark 1. Comparing (2) to the original KD in (1), we omit the first KL divergence term related to ground truth labels:

$$\min_\phi \mathbb{E}_{p(\mathbf{x})}[\mathcal{D}(\mathbf{t}_{\theta^*}(\mathbf{x}), \mathbf{s}_\phi(\mathbf{x}))]. \quad (3)$$

If we have a generator $\mathbf{g}_{\psi^*}$ optimized to mimic the training data exactly such that $p(\mathbf{x}) = \int p(\mathbf{z})\delta(\mathbf{x} - \mathbf{g}_{\psi^*}(\mathbf{z}))d\mathbf{z}$, then (3) reduces to

$$\min_\phi \mathbb{E}_{p(\mathbf{z})}[\mathcal{D}(\mathbf{t}_{\theta^*}(\mathbf{g}_{\psi^*}(\mathbf{z})), \mathbf{s}_\phi(\mathbf{g}_{\psi^*}(\mathbf{z})))].$$

However, we do not have access to the original training data and cannot find the optimal generator $\mathbf{g}_{\psi^*}$. Instead, we minimize the upper bound of $\mathbb{E}_{p(\mathbf{z})}[\mathcal{D}(\mathbf{t}_{\theta^*}, \mathbf{s}_\phi)]$ by solving the minimax problem in (2), while we give the generator some constraints with the auxiliary loss L_ψ for the generator to produce similar data as the original training data.

3.3. Generator constraints

We consider the following three auxiliary loss terms for the generator in the maximization step of (2) to make the generator produce "good" adversarial samples similar to the original data as much as possible based on the teacher.

(a) **Batch normalization statistics**. Batch normalization layers contain the mean and variance of layer inputs, which we can utilize as a proxy to confirm that the generator output is similar to the original training data. We propose using the KL divergence of two Gaussian distributions to match the mean and variance stored in batch normalization layers (which are obtained from the original data) and the empirical statistics obtained with the generator output.

(b) **Instance categorical entropy**. If the teacher is trained well enough for accurate classification, the generator output is of interest only when the categorical distribution output, i.e., softmax output, of the teacher yields small entropy (the probability for one category should be high); the entropy is minimized to zero if one category has probability 1. That is, we need small entropy for $\mathbf{t}_{\theta^*}(\mathbf{g}_\psi(\mathbf{z}))$ on each sampled $\mathbf{z}$.

(c) **Batch categorical entropy**. Assuming that each class appears in the dataset with similar probability, the categorical probability distribution averaged for any batch should tend to uniform distribution where the entropy is maximized to $\log_2 |\mathcal{C}|$. That is, we need high entropy for $\mathbb{E}_{p(\mathbf{z})}[\mathbf{t}_{\theta^*}(\mathbf{g}_\psi(\mathbf{z}))]$.

Let $\mu(l, c)$ and $\sigma^2(l, c)$ be the mean and the variance stored in batch normalization layer l for channel c, which is learned from the original training data. Let $\hat{\mu}_\psi(l, c)$ and $\hat{\sigma}^2_\psi(l, c)$ be the corresponding mean and variance computed for the synthetic samples from the generator $\mathbf{g}_\psi$. The auxiliary loss L_ψ for the generator in (2) is given by

$$\begin{aligned} L_\psi &= \sum_{l,c} \mathcal{D}_\mathcal{N}((\hat{\mu}_\psi(l, c), \hat{\sigma}^2_\psi(l, c)), (\mu(l, c), \sigma^2(l, c))) \\ &+ \mathbb{E}_{p(\mathbf{z})}[H(\mathbf{t}_{\theta^*}(\mathbf{g}_\psi(\mathbf{z})))] - H(\mathbb{E}_{p(\mathbf{z})}[\mathbf{t}_{\theta^*}(\mathbf{g}_\psi(\mathbf{z}))]), \end{aligned} \quad (4)$$

where H denotes entropy (e.g., see [63, Section 2.1]), and $\mathcal{D}_\mathcal{N}((\hat{\mu}, \hat{\sigma}^2), (\mu, \sigma^2))$ is the KL divergence of two Gaussian distributions, which can be represented as

$$\mathcal{D}_\mathcal{N}((\hat{\mu}, \hat{\sigma}^2), (\mu, \sigma^2)) = \frac{(\hat{\mu} - \mu)^2 + \hat{\sigma}^2}{2\sigma^2} - \log \frac{\hat{\sigma}}{\sigma} - \frac{1}{2}. \quad (5)$$

Remark 2. If $\alpha = 0$ in (2), the proposed scheme reduces to the adversarial belief matching presented in [36]. Adding the auxiliary loss L_ψ, we constrain the generator so it produces synthetic images that yield similar statistics in the teacher as the original data, which helps the minimax optimization avoid any adversarial samples that are very different from the original data and leads to better distillation performance (basically we reduce the loss due to fitting the model for "bad" examples not close to the original dataset). For (b) and (c), we found that similar entropy loss terms are already proposed in [35]. Batch normalization statistics are used in [41, 45]. Yin et al. [41] find synthetic samples directly in the image domain with no generators by optimizing an input batch such that it produces similar batch normalization statistics in a pre-trained model. In contrast, we utilize batch normalization statistics to constrain generators. Furthermore, to match the mean and variance, the squared L2 distance is used in [41], while we propose using the KL divergence of two Gaussian distributions, which is a distance measure normalized by scale (i.e., standard deviation σ in (5)). In [45], batch normalization statistics are used to calculate any quantization biases for correction. No synthetic images are produced in [45].

3.4. Multiple generators and multiple students

Using mixture of generators has been proposed to avoid the mode collapse issue and to yield diverse samples that cover the whole support of a target dataset [62]. Similarly we propose training multiple generators in our data-free KD framework to increase the diversity of generated samples. Moreover, using multiple discriminators has been also proposed to reduce the mode collapse problem in GANs [59]. A similar idea can be adopted in our framework, since we utilize the KL divergence of the student and teacher outputs as the discriminator output. The average KL divergence between the teacher and the students are maximized in minimax optimization. Intuitively, taking average not only reduces the noise in minimax optimization using stochastic gradient descent, but also steers a generator to produce better adversarial samples that are poorly matched to every student in average. The final objective with multiple generators and multiple students is given by

$$\min_{\phi_i, 1\le i\le S} \max_{\psi_j, 1\le j\le G} \sum_{j=1}^{G} \left(\frac{1}{S} \sum_{i=1}^{S} \mathcal{D}_{\phi_i,\psi_j} - \alpha L_{\psi_j} \right),$$

$$\mathcal{D}_{\phi_i,\psi_j} \triangleq \mathbb{E}_{p(\mathbf{z})}[\mathcal{D}(\mathbf{t}_{\theta^*}(\mathbf{g}_{\psi_j}(\mathbf{z})), \mathbf{s}_{\phi_i}(\mathbf{g}_{\psi_j}(\mathbf{z})))],$$

where $\mathbf{s}_{\phi_i}$ is the i-th student and $\mathbf{g}_{\psi_j}$ is the j-th generator for $1 \le i \le S$ and $1 \le j \le G$.

3.5. Implementation

We summarize the proposed data-free adversarial KD scheme in Algorithm 1. Let $\mathbf{z}_1^B$ be the random input batch of size B to generators, and let $\mathcal{D}_{\phi_i,\psi_j}(\mathbf{z}_1^B)$ and $L_{\psi_j}(\mathbf{z}_1^B)$ be the losses computed and averaged over batch $\mathbf{z}_1^B$. We suggest "warm-up" training of generators, optionally, before the main adversarial KD. In the warm-up stage, we train generators only to minimize the auxiliary loss L_ψ so its output matches batch normalization statistics and entropy constraints when fed to the teacher. This pre-training procedure reduces generation of unreliable samples in the early steps of data-free KD. Furthermore, updating students more frequently than generators reduces the chances of falling into any local maximum in the minimax optimization. In the minimization step, one can additionally match intermediate layer outputs as proposed in [28–30]. Finally, data-free network quantization is implemented by letting the student be a quantized version of the teacher (see Section 4.2).

Algorithm 1 Data-free adversarial knowledge distillation.

```
Generator update interval: $m \ge 1$
Warm-up training for generators (optional)
for $n$ : 1 to $N_{\text{warm-up}}$ do
    for $j$ : 1 to $G$ do
        $\mathbf{z}_1^B \leftarrow [\mathcal{N}(0, I)]_1^B$
        $\psi_j \leftarrow \psi_j - \eta \nabla_{\psi_j} L_{\psi_j}(\mathbf{z}_1^B)$
    end for
end for
Adversarial knowledge distillation
for $n$ : 1 to $N$ do
    Maximization
    if $n \equiv 0 \bmod m$ then
        for $j$ : 1 to $G$ do
            $\mathbf{z}_1^B \leftarrow [\mathcal{N}(0, I)]_1^B$
            for $i$ : 1 to $S$ do
                $\mathcal{D}_{\phi_i,\psi_j}(\mathbf{z}_1^B) \leftarrow \mathcal{D}(\mathbf{t}_{\theta^*}(\mathbf{g}_{\psi_j}(\mathbf{z}_1^B)), \mathbf{s}_{\phi_i}(\mathbf{g}_{\psi_j}(\mathbf{z}_1^B)))$
            end for
            $\psi_j \leftarrow \psi_j + \eta \nabla_{\psi_j}(\frac{1}{S}\sum_{i=1}^{S} \mathcal{D}_{\phi_i,\psi_j}(\mathbf{z}_1^B) - \alpha L_{\psi_j}(\mathbf{z}_1^B))$
        end for
    end if
    Minimization
    $b \leftarrow \lfloor B/G \rfloor$
    for $j$ : 1 to $G$ do
        $\mathbf{z}_1^b \leftarrow [\mathcal{N}(0, I)]_1^b$
        $\mathbf{x}_1^{bj} \leftarrow$ concatenate$(\mathbf{x}_1^{b(j-1)}, \mathbf{g}_{\psi_j}(\mathbf{z}_1^b))$
    end for
    for $i$ : 1 to $S$ do
        $\phi_i \leftarrow \phi_i - \eta \nabla_{\phi_i} \mathcal{D}(\mathbf{t}_{\theta^*}(\mathbf{x}_1^{bG}), \mathbf{s}_{\phi_i}(\mathbf{x}_1^{bG}))$
    end for
end for
```

4. Experiments

We evaluate the proposed data-free adversarial KD algorithm on two model compression tasks: (1) data-free KD to smaller networks and (2) data-free network quantization.

Generator architecture. Let `conv3-`k denote a convolutional layer with k 3×3 filters and stride 1×1. Let `fc-`k be a fully-connected layer with k units. Let `upsampling` be a 2×2 nearest-neighbor upsampling layer. The generator input $\mathbf{z}$ is of size 512 and is sampled from the standard normal distribution. Given that the image size of the original data is `(W,H,3)`, we build a generator as below:

```
fc-8WH, reshape-(W/8,H/8,512)
upsampling, conv3-256, batchnorm, ReLU
upsampling, conv3-128, batchnorm, ReLU
upsampling, conv3-64, batchnorm, ReLU
conv3-3, tanh, batchnorm
```

Training. For training generators in maximization, we use Adam optimizer [64] with momentum 0.5 and learning rate 10^{-3}. On the other hand, for training students in minimization, we use Nesterov accelerated gradient [65] with momentum 0.9 and learning rate 0.1. The learning rates are annealed by cosine decaying [66]. We adopt the vanilla KD for data-free KD from WRN40-2 to WRN16-1 on CIFAR-10. We use 50 epochs in the warm-up stage and 200 epochs for the main adversarial KD, where each epoch consists of 400 batches of batch size 256. In the other cases, we adopt

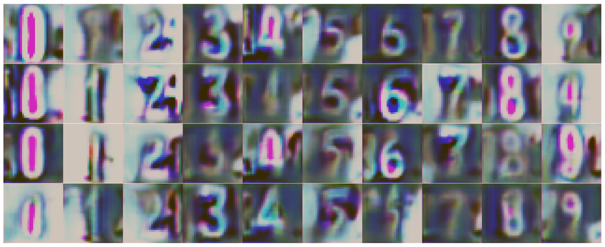


Figure 2: Example synthetic images generated in data-free KD from WRN40-2 to WRN16-1 for SVHN. Just for better presentation, we classify the synthetic images using the teacher and show 4 samples from 0 to 9 in each column.

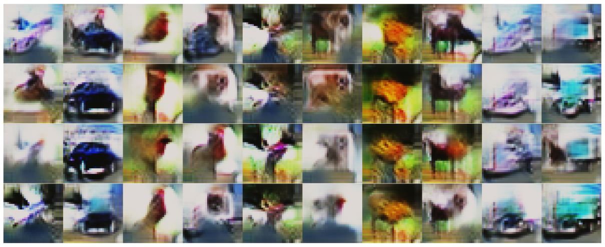


Figure 3: Example synthetic images generated in data-free KD from WRN40-2 to WRN16-1 for CIFAR-10. Similar to Figure 2, we classify the synthetic images using the teacher and show 4 samples for each class of CIFAR-10 (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck) in each column.

variational information distillation (VID) [30] to match intermediate layer outputs, where we reduce the number of batches per epoch to 200; VID is one of the state-of-the-art KD variants, and it yields better student accuracy with faster convergence. For the weighting factor α in (2), we perform experiments on $\alpha \in \{10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$ and choose the best results. The generator update interval m is set to be 10 for wide residual networks and 1 for the others. Except the results in Table 3, we use one generator and one student in our data-free KD, i.e., $G = S = 1$ in Algorithm 1.

4.1. Data-free model compression

We evaluate the performance of the proposed data-free model compression scheme on SVHN, CIFAR-10, CIFAR-100, and Tiny-ImageNet datasets for KD of residual networks (ResNets) and wide residual networks (WRNs). We summarize the main results in Table 2. We compare our scheme to the previous data-free KD methods in [35,36,41] and show that we achieve the state-of-the-art data-free KD performance in all evaluation cases. We also obtain the student accuracy when students are trained with the original datasets from scratch and by using variational information distillation (VID) in [30]. Table 2 shows that the accuracy losses of our data-free KD method are marginal, compared to the cases of using the original datasets.

Example synthetic images. We show example synthetic images obtained from generators trained with teachers pre-trained for SVHN, CIFAR-10, and CIFAR-100 datasets, respectively, in Figure 2, Figure 3, and Figure 4. The figures show that the generators regularized with pre-trained teachers produce samples that are similar to the original datasets.

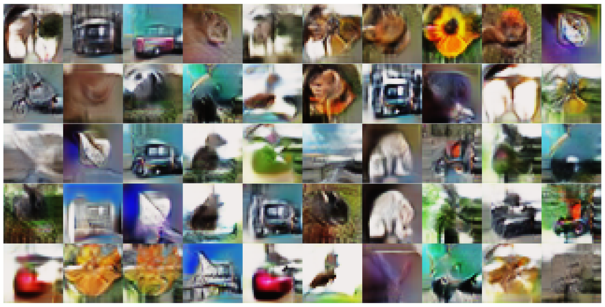


Figure 4: Example synthetic images generated in data-free KD from ResNet-34 to ResNet-18 for CIFAR-100.

Ablation study. For ablation study, we evaluate the proposed data-free KD scheme with and without each term of the auxiliary loss L_ψ for the generator in (4), and the results are summarized in Figure 5. The bar graph shows that the major contribution comes from (a), which is to match batch normalization statistics (see Section 3.3). In Figure 6, we present the impact of the weighting factor α in (2) on KD performance. Moreover, to visually show the impact of α on the generation of synthetic data, we collect synthetic images for $\alpha = 10$ and $\alpha = 0.1$ and show them at different epochs in Figure 7. The figures show that smaller α yields more diverse adversarial images, since the generator is constrained less. As α gets larger, the generated images collapse to one mode for each class, which leads to over-fitting.

Multiple generators and multiple students. We show the gain of using multiple generators and/or multiple students in Table 3. We compare the cases of using two generators and/or two students. For the second generator, we replace one middle convolutional layer with a residual block. For KD to two students, we use identical students with different initialization. Table 3 shows that increasing the number of generators and/or the number of students results in better student accuracy in data-free KD.

4.2. Data-free network quantization

In this subsection, we present the experimental results of the proposed data-free adversarial KD scheme on network quantization. For the baseline quantization scheme, we use TensorFlow's quantization framework. In particular, we implement our data-free KD scheme in the quantization-aware training framework [24,42] of TensorFlow[2].

TensorFlow's quantization-aware training performs per-layer asymmetric quantization of weights and activations.

[2] https://github.com/tensorflow/tensorflow/tree/r1.15/tensorflow/contrib/quantize

Table 2: Comparison of the proposed data-free adversarial KD scheme to the previous works.

Original dataset	Teacher (# params)	Student (# params)	Teacher accuracy (%)	Student accuracy (%)					
				Data-free KD methods				Training from scratch*	VID [30]*
				Ours	[36]	[35]	[41]		
SVHN	WRN40-2 (2.2M)	WRN16-1 (0.2M)	98.04	**96.48**	94.06	N/A	N/A	97.67	97.60
CIFAR-10	WRN40-2 (2.2M)	WRN16-1 (0.2M)	94.77	**86.14**	83.69	N/A	N/A	90.97	91.78
		WRN40-1 (0.6M)		**91.69**	86.60	N/A	N/A	93.35	93.67
		WRN16-2 (0.7M)		**92.01**	89.71	N/A	N/A	93.72	94.06
	VGG-11 (9.2M)	ResNet-18 (11.2M)	92.37	**90.84**	N/A	N/A	90.36	94.56	91.47
	ResNet-34 (21.3M)	ResNet-18 (11.2M)	95.11	**94.61**	N/A	92.22	93.26	94.56	94.90
CIFAR-100	ResNet-34 (21.3M)	ResNet-18 (11.2M)	78.34	**77.01**	N/A	74.47	N/A	77.32	77.77
Tiny-ImageNet	ResNet-34 (21.4M)	ResNet-18 (11.3M)	66.34	**63.73**	N/A	N/A	N/A	64.87	66.01

* Used the original datasets.

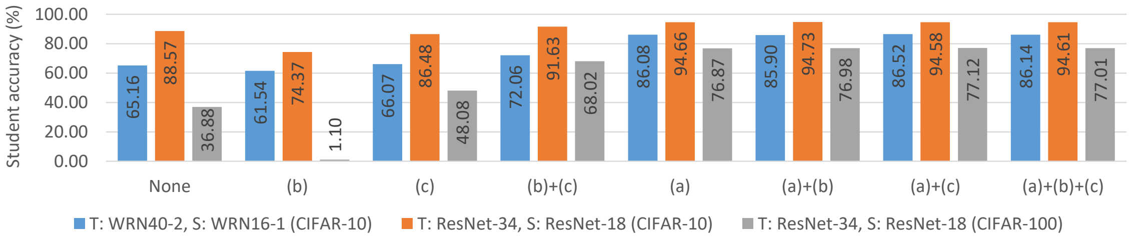
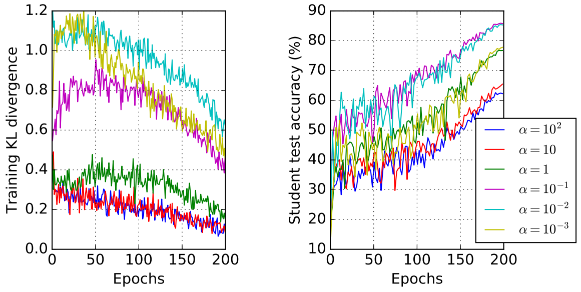


Figure 5: Ablation study on the three terms in the auxiliary loss L_ψ of (4), i.e., (a) batch normalization statistics, (b) instance categorical entropy, and (c) batch categorical entropy (see Section 3.3).



(a) Training KL divergence (b) Student test accuracy

Figure 6: Training KL divegence and student test accuracy of data-free KD for different values of α in (2). The student over-fits to the generator output when the weighting factor α is too large ($\alpha \in \{10, 10^2\}$).

Table 3: Comparison of the student accuracy (%) when using multiple generators and/or multiple students in our data-free KD from WRN40-2 to WRN16-1 on CIFAR-10.

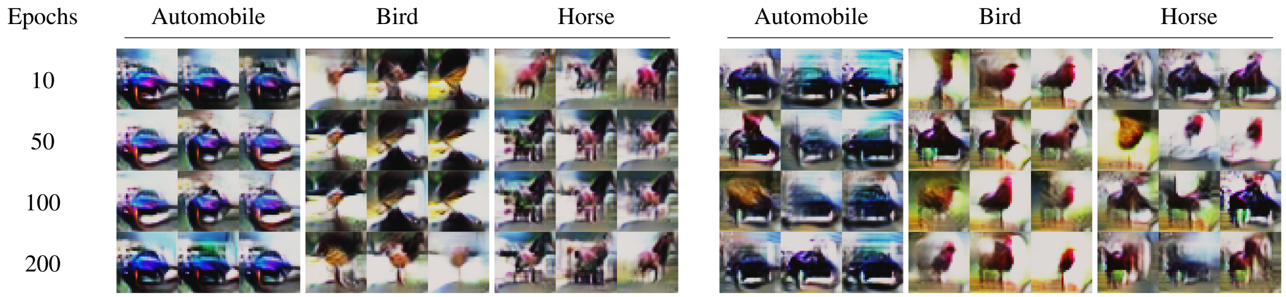
# students (S) \ # generators (G)	1	2
1	86.14	86.67
2	86.44	**87.04**

For quantization only, no data are needed for weight quantization, but quantization of activations requires representative data, which are used to collect the range (the minimum and the maximum) of activations and to determine the quantization bin size based on the range. In our data-free quantization, we use synthetic data from a generator as the representative data. To this end, we train a generator with no adversarial loss as in the warm-up stage of Algorithm 1 (see DF-Q in Table 4). For our data-free quantization-aware training, we utilize the proposed adversarial KD on top of Tensorflow's quantization-aware framework, where a quantized network is set as the student and a pre-trained floating-point model is given as the teacher, which is denoted by DF-QAT-KD in Table 4.

We follow the training hyperparameters as described in Section 4.1, while we set the initial learning rate for KD to be 10^{-3}. We use 200 epochs for the warm-up stage and 50 epochs for quantization-aware training with data-free KD. We adopt the vanilla KD with no intermediate layer output matching terms. We summarize the results in Table 4.

For comparison, we evaluate three conventional data-dependent quantization schemes using the original training datasets, i.e., quantization only (Q), quantization-aware training (QAT), and quantization-aware training with KD (QAT-KD). As presented in Table 4, our data-free quantization shows very marginal accuracy losses less than 2% for 4-bit/8-bit weight and 8-bit activation quantization in all the evaluated cases, compared to using the original datasets.

Finally, we compare our data-free quantization to using alternative datasets. We consider two cases (1) when a sim-



(a) $\alpha = 10$

(b) $\alpha = 0.1$

Figure 7: Example synthetic images generated for CIFAR-10 automobile, bird, and horse classes in different training epochs. We compare two cases with $\alpha = 10$ and $\alpha = 0.1$ to show the impact of the weighting factor α in (2) on the generator output.

Table 4: Results of network quantization with the proposed data-free adversarial KD scheme. For our data-free quantization, we show the results for data-free quantization only (DF-Q) and data-free quantization-aware training with data-free KD (DF-QAT-KD). For conventional data-dependent quantization [24], we show the results for quantization only (Q), quantization-aware training (QAT), and quantization-aware training with KD (QAT-KD).

Original dataset	Pre-trained model (accuracy %)	Quantization bit-width for weights / activations	Quantized model accuracy (%)				
			Ours (data-free)		Data-dependent [24]*		
			DF-Q	DF-QAT-KD	Q	QAT	QAT-KD
SVHN	WRN16-1 (97.67)	8 / 8	97.67	97.74	97.70	97.71	97.78
		4 / 8	91.92	97.53	93.83	97.66	97.70
CIFAR-10	WRN16-1 (90.97)	8 / 8	90.51	90.90	90.95	91.21	91.16
		4 / 8	86.29	88.91	86.74	90.92	90.71
	WRN40-2 (94.77)	8 / 8	94.47	94.76	94.75	94.91	95.02
		4 / 8	93.14	94.22	93.56	94.73	94.42
CIFAR-100	ResNet-18 (77.32)	8 / 8	76.68	77.30	77.43	77.84	77.73
		4 / 8	71.02	75.15	69.63	75.52	75.62
Tiny-ImageNet	MobileNet v1 (64.34)	8 / 8	51.76	63.11	54.48	61.94	64.53

* Used the original datasets.

Table 5: Impact of using different datasets for 4-bit weight and 8-bit activation quantization.

Dataset used in KD	Quantized model accuracy (%) before / after fine-tuning with KD		
	WRN16-1 (SVHN)	WRN40-2 (CIFAR-10)	ResNet-18 (CIFAR-100)
SVHN	93.83 / 97.70	71.89 / 92.08	13.41 / 65.07
CIFAR-10	93.50 / 97.24	93.56 / 94.42	67.50 / 75.62
CIFAR-100	94.11 / 97.26	92.18 / 94.10	69.63 / 75.62
Ours (data-free)	91.92 / 97.53	93.14 / 94.22	71.02 / 75.15

ilar dataset is used (e.g., CIFAR-100 instead of CIFAR-10) and (2) when a mismatched dataset is used (e.g., SVHN instead of CIFAR-10). The results in Table 5 show that using a mismatched dataset degrades the performance considerably. Using a similar dataset achieves comparable performance to our data-free scheme, which shows small accuracy losses less than 0.5% compared to using the original datasets. We note that even alternative data, which are safe from privacy and regulatory concerns, are hard to collect in usual cases.

5. Conclusion

In this paper, we proposed data-free adversarial KD for network quantization and compression. No original data are used in the proposed framework, while we train a generator to produce synthetic data adversarial to KD. In particular, we propose matching batch normalization statistics in the teacher to additionally constrain the generator to produce samples similar to the original training data. We used the proposed data-free KD scheme for compression of various models trained on SVHN, CIFAR-10, CIFAR-100, and Tiny-ImageNet datasets. In our experiments, we achieved the state-of-the-art data-free KD performance over the existing data-free KD schemes. For network quantization, we obtained quantized models that achieve comparable accuracy to the models quantized and fine-tuned with the original training datasets. The proposed framework shows great potential to keep data privacy in model compression.

References

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[3] Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, and Joel S Emer. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, 105(12):2295–2329, 2017.

[4] Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. Model compression and acceleration for deep neural networks: The principles, progress, and challenges. *IEEE Signal Processing Magazine*, 35(1):126–136, 2018.

[5] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, pages 1135–1143, 2015.

[6] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2074–2082, 2016.

[7] Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient DNNs. In *Advances In Neural Information Processing Systems*, pages 1379–1387, 2016.

[8] Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variational dropout sparsifies deep neural networks. In *International Conference on Machine Learning*, pages 2498–2507, 2017.

[9] Christos Louizos, Karen Ullrich, and Max Welling. Bayesian compression for deep learning. In *Advances in Neural Information Processing Systems*, pages 3290–3300, 2017.

[10] Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through L_0 regularization. In *International Conference on Learning Representations*, 2018.

[11] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2018.

[12] Bin Dai, Chen Zhu, Baining Guo, and David Wipf. Compressing neural networks using the variational information bottleneck. In *International Conference on Machine Learning*, pages 1135–1144, 2018.

[13] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. In *International Conference on Learning Representations*, 2016.

[14] Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. Towards the limit of network quantization. In *International Conference on Learning Representations*, 2017.

[15] Karen Ullrich, Edward Meeds, and Max Welling. Soft weight-sharing for neural network compression. In *International Conference on Learning Representations*, 2017.

[16] Eunhyeok Park, Junwhan Ahn, and Sungjoo Yoo. Weighted-entropy-based quantization for deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7197–7205, 2017.

[17] Frederick Tung and Greg Mori. Deep neural network compression by in-parallel pruning-quantization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[18] Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. Universal deep neural network compression. *IEEE Journal of Selected Topics in Signal Processing*, 2020.

[19] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. XNOR-Net: Imagenet classification using binary convolutional neural networks. In *Proceedings of the European Conference on Computer Vision*, pages 525–542, 2016.

[20] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. DoReFa-Net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.

[21] Chenzhuo Zhu, Song Han, Huizi Mao, and William J Dally. Trained ternary quantization. In *International Conference on Learning Representations*, 2017.

[22] Zhaowei Cai, Xiaodong He, Jian Sun, and Nuno Vasconcelos. Deep learning with low precision by half-wave Gaussian quantization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5918–5926, 2017.

[23] Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. LQ-Nets: Learned quantization for highly accurate and compact deep neural networks. In *Proceedings of the European Conference on Computer Vision*, pages 365–382, 2018.

[24] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2704–2713, 2018.

[25] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. HAQ: Hardware-aware automated quantization with mixed precision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8612–8620, 2019.

[26] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[27] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[28] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. FitNets: Hints for thin deep nets. In *International Conference on Learning Representations*, 2015.

[29] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *International Conference on Learning Representations*, 2017.

[30] Sungsoo Ahn, Shell Xu Hu, Andreas Damianou, Neil D Lawrence, and Zhenwen Dai. Variational information distillation for knowledge transfer. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition, pages 9163–9171, 2019.

[31] Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. DarkRank: Accelerating deep metric learning via cross sample similarities transfer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[32] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3967–3976, 2019.

[33] Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. Data-free knowledge distillation for deep neural networks. In *NeurIPS Workshop on Learning with Limited Data*, 2017.

[34] Kartikeya Bhardwaj, Naveen Suda, and Radu Marculescu. Dream distillation: A data-independent model compression framework. In *ICML Joint Workshop on On-Device Machine Learning and Compact Deep Neural Network Representations (ODML-CDNNR)*, 2019.

[35] Hanting Chen, Yunhe Wang, Chang Xu, Zhaohui Yang, Chuanjian Liu, Boxin Shi, Chunjing Xu, Chao Xu, and Qi Tian. Data-free learning of student networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3514–3522, 2019.

[36] Paul Micaelli and Amos J Storkey. Zero-shot knowledge transfer via adversarial belief matching. In *Advances in Neural Information Processing Systems*, pages 9547–9557, 2019.

[37] 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.

[38] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference*, pages 87.1–87.12, 2016.

[39] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[40] Alex Krizhevsky. Learning multiple layers of features from tiny images. *Technical report, Univ. of Toronto*, 2009.

[41] Hongxu Yin, Pavlo Molchanov, Zhizhong Li, Jose M Alvarez, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via DeepInversion. *arXiv preprint arXiv:1912.08795*, 2019.

[42] Raghuraman Krishnamoorthi. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv preprint arXiv:1806.08342*, 2018.

[43] Gaurav Kumar Nayak, Konda Reddy Mopuri, Vaisakh Shaj, Venkatesh Babu Radhakrishnan, and Anirban Chakraborty. Zero-shot knowledge distillation in deep networks. In *International Conference on Machine Learning*, pages 4743–4751, 2019.

[44] Jaemin Yoo, Minyong Cho, Taebum Kim, and U Kang. Knowledge extraction with no observable data. In *Advances in Neural Information Processing Systems*, pages 2701–2710, 2019.

[45] Markus Nagel, Mart van Baalen, Tijmen Blankevoort, and Max Welling. Data-free quantization through weight equalization and bias correction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1325–1334, 2019.

[46] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust Optimization*, volume 28. Princeton University Press, 2009.

[47] Dimitris Bertsimas, David B Brown, and Constantine Caramanis. Theory and applications of robust optimization. *SIAM review*, 53(3):464–501, 2011.

[48] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018.

[49] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2014.

[50] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, pages 39–57, 2017.

[51] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[52] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4422–4431, 2018.

[53] Huaxia Wang and Chun-Nam Yu. A direct approach to robust deep learning using adversarial networks. In *International Conference on Learning Representations*, 2019.

[54] Yunseok Jang, Tianchen Zhao, Seunghoon Hong, and Honglak Lee. Adversarial defense via learning to generate diverse attacks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2740–2749, 2019.

[55] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9446–9454, 2018.

[56] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks, 2015. `https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html` [Online; accessed 18-April-2020].

[57] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[58] Ian Goodfellow. NIPS 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[59] Ishan Durugkar, Ian Gemp, and Sridhar Mahadevan. Generative multi-adversarial networks. In *International Conference on Learning Representations*, 2017.

[60] Tu Nguyen, Trung Le, Hung Vu, and Dinh Phung. Dual discriminator generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2670–2680, 2017.

[61] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (GANs). In *International Conference on Machine Learning*, pages 224–232, 2017.

[62] Quan Hoang, Tu Dinh Nguyen, Trung Le, and Dinh Phung. MGAN: Training generative adversarial nets with multiple generators. In *International Conference on Learning Representations*, 2018.

[63] Thomas M Cover and Joy A Thomas. *Elements of Information Theory*. John Wiley & Sons, 2012.

[64] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[65] Yurii Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$. In *Doklady AN USSR*, volume 269, pages 543–547, 1983.

[66] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.