Few Shot Network Compression via Cross Distillation

Haoli Bai,[1] **Jiaxiang Wu,**[2] **Irwin King,**[1] **Michael Lyu**[1]
[1]The Chinese University of Hong Kong, [2]Tencent AI Lab
{hlbai, king, lyu}@cse.cuhk.edu.hk, jonathanwu@tencent.com

Abstract

Model compression has been widely adopted to obtain light-weighted deep neural networks. Most prevalent methods, however, require fine-tuning with sufficient training data to ensure accuracy, which could be challenged by privacy and security issues. As a compromise between privacy and performance, in this paper we investigate few shot network compression: given few samples per class, how can we effectively compress the network with negligible performance drop? The core challenge of few shot network compression lies in high estimation errors from the original network during inference, since the compressed network can easily over-fits on the few training instances. The estimation errors could propagate and accumulate layer-wisely and finally deteriorate the network output. To address the problem, we propose cross distillation, a novel layer-wise knowledge distillation approach. By interweaving hidden layers of teacher and student network, layer-wisely accumulated estimation errors can be effectively reduced. The proposed method offers a general framework compatible with prevalent network compression techniques such as pruning. Extensive experiments n benchmark datasets demonstrate that cross distillation can significantly improve the student network's accuracy when only a few training instances are available.

Introduction

Deep neural networks (DNNs) have achieved remarkable success in a wide range of applications, however, they suffer from substantial computation and energy cost. In order to obtain light-weighted DNNs, network compression techniques have been widely developed in recent years, including network pruning (He, Zhang, and Sun 2017; Luo, Wu, and Lin 2017; Wen et al. 2019), quantization (Han, Mao, and Dally 2016; Wu et al. 2016; 2018; Li et al. 2020) and knowledge distillation (Hinton, Vinyals, and Dean 2015; Romero et al. 2014).

Despite the success of previous efforts, a majority of them rely on the whole training data to reboot the compressed models, which could suffer from security and privacy issues. For instance, to provide a general service of network compression, the reliance on the training data may result in data leakage for customers.

To take care of security issues in network compression, some recent works (Chen et al. 2019; Bhardwaj, Suda, and Marculescu 2019; Lopes, Fenu, and Starner 2017) motivate from knowledge distillation (Hinton, Vinyals, and Dean 2015; Romero et al. 2014), and propose data-free fine-tuning by constructing pseudo inputs from the pre-trained teacher network. However, these methods highly rely on the quality of the pseudo inputs and are therefore limited to small-scale problems.

In order to obtain scalable network compression algorithms, a compromise between privacy and performance is to compress the network with few shot training instances, e.g., 1-shot for one training instances per class. Prevalent works (Li et al. 2018; Chen, Wang, and Pan 2019b) along this line extend knowledge distillation by minimizing layer-wise estimation errors (e.g., Euclidean distances) between the teacher and student network. The success of these approaches largely comes from the layer-wise supervision from the teacher network. Nevertheless, a key challenge in few shot network compression is rarely investigated in previous efforts: as there are few shot training samples available, the student network tend to over-fit on the training set and consequently suffer from high estimation errors from the teacher network during inference. Moreover, the estimation errors could propagate and accumulate layer-wisely (Dong, Chen, and Pan 2017) and finally deteriorate the student network.

To deal with the above challenge, we proceed along with few shot network compression and propose cross distillation, a novel layer-wise knowledge distillation approach. Cross distillation can effectively reduce the layer-wisely accumulated errors in the few shot setting, leading to a more powerful and generalizable student network. Specifically, to correct the errors accumulated in previous layers of the student network, we direct the teacher's hidden layers to the student network, which is called correction. Meanwhile, to make the teacher aware of the errors accumulated on the student network, we reverse the strategy by directing the student's hidden layers to the teacher network. With error-aware supervision from the teacher, the student can better

mimic the teacher's behavior, which is called imitation. The correction and imitation compensate each other, and to find a proper trade-off, we propose to take convex combinations between either loss functions of the two procedures, or hidden layers of the two networks. To better understand the proposed method, we also give some theoretical analysis on how convex combination of the two loss functions manipulates the layer-wisely propagated errors, and why cross distillation is capable of improving the student network. Our proposed method provides a universal framework to assist prevalent network compression techniques such as pruning (He, Zhang, and Sun 2017).

Extensive experiments and ablation studies are conducted on popular network architectures and benchmark datasets, and the results demonstrate that our proposed method can effectively reduce the estimation errors and improve the compressed model in the few shot setting, outperforming a number of competitive baselines.

Related Work

While most previous efforts on network compression rely on abundant training data for fine-tuning the compressed network, there is a recent trend on investigating security and privacy issues for network compression. These methods can be generally categorized into data-free methods and few-shot methods.

To perform data-free network compression, a simple way is to directly apply quantization (Banner et al. 2018) or low-rank factorization (Zhang et al. 2015; Ye et al. 2018) on network parameters, which usually degrade the network significantly when the compression rate is high. Recent efforts motivate from knowledge distillation (Hinton, Vinyals, and Dean 2015; Romero et al. 2014), which constructs pseudo inputs from the pre-trained teacher network based on its parameters (Nayak et al. 2019), feature map statistics (Lopes, Fenu, and Starner 2017; Bhardwaj, Suda, and Marculescu 2019), or an independently trained generative model (Chen et al. 2019) to simulate the distribution of the original training set. However, the generation of high-quality pseudo inputs could be challenging and expensive, especially on large-scale problems.

The other line of research considers network compression with few-shot training samples, which is a compromise between privacy and performance. To fully take advantage of the training data, a number of existing works (He, Zhang, and Sun 2017; Luo, Wu, and Lin 2017; Li et al. 2018; Chen, Wang, and Pan 2019b) extend knowledge distillation by layer-wisely minimizing the Euclidean distances between the teacher network and the student network. The layer-wise training is usually data-efficient as the student network receives layer-wise supervision from the teacher and there are fewer parameters to optimize comparing to back-propagation training of the entire student network (Romero et al. 2014). Aside from layer-wise regression, recently data from different but related domains are also utilized as auxiliary information to assist the pruning on the target domain (Chen, Wang, and Pan 2019a). Unlike data free compression techniques, few shot network compression can significantly improve the performance of the compressed network with only limited training instances, which is potentially helpful for large-scale real-world problems.

Our proposed cross distillation proceeds along the line of few shot network compression. As an extension of previous layer-wise regression methods, we pay extra attention to the reduction of estimation errors during inference, which are usually large as a result of over-fitting on few shot training instances. We remark that similar ideas of cross connection between two networks are also previously explored in multi-task learning (Gupta, Hoffman, and Malik 2016) to obtain mutual representations from different tasks. Our work differs in both the problem setting as well as the optimization method to obtain a compact and powerful compressed network.

Methods

Our goal is to obtain a compact student network $\mathcal{F}^S$ from the over-parameterized teacher network $\mathcal{F}^T$. Given few shot training instances $\{\mathbf{x}_n, \mathbf{y}_n\}_{n=1}^N$, we denote their corresponding l-th convolutional feature map of the teacher network $\mathcal{F}^T$ as $\mathbf{h}_l^T = \sigma(\mathbf{W}_l^T * \mathbf{h}_{l-1}^T) \in R^{N\times c_i\times k\times k}$, where $\sigma(\cdot)$ is the activation function, $*$ is the convolutional operation, $\mathbf{W}_l^T \in R^{c_o\times c_i\times k\times k}$ is the 4-D convolutional kernel, and N, c_i, c_o and k are the number of training size, input channels, output channels and the kernel size respectively. Batch normalization layers are omitted as they can be readily fused into convolutional. Similar notations hold for $\mathcal{F}_l^S$. In the following, we drop the layer index l.

Unlike standard knowledge distillation approaches, here we adopt layer-wise knowledge distillation which can take layer-wise supervision from the teacher network. As is shown in Figure 1(a), with previous layers being fixed, layer-wise distillation aims to find the optimal $\mathbf{W}_*^S$ that minimizes the Euclidean distance between $\mathbf{h}^T$ and $\mathbf{h}^S$, i.e.,

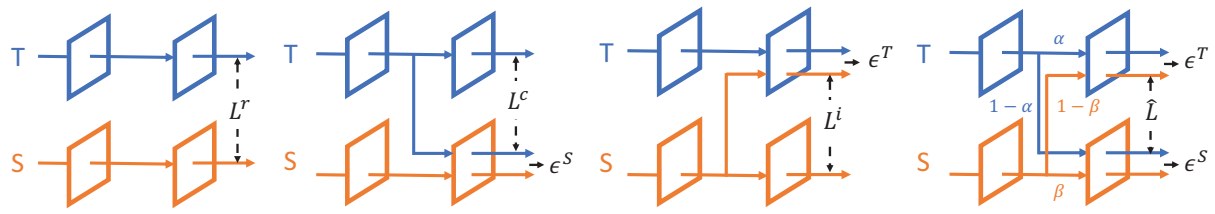
$$\mathbf{W}_*^S = \arg\min_{\mathbf{W}^S} \frac{1}{N}\mathcal{L}^r(\mathbf{W}^S) + \lambda\mathcal{R}(\mathbf{W}^S), \qquad (1)$$

where $\mathcal{L}^r(\mathbf{W}^S) = \|\sigma(\mathbf{W}^T * \mathbf{h}^T) - \sigma(\mathbf{W}^S * \mathbf{h}^S)\|_F^2$ is the called estimation error, and $\mathcal{R}(\mathbf{W}^S)$ is some regularization tuned by λ. Despite that one can obtain a decent compact network by Equation 1 with abundant training data (He, Zhang, and Sun 2017; Luo, Wu, and Lin 2017), when there are only few shot training instances, the student network $\mathcal{F}^S$ tends to suffer from high estimation errors on the test set as a result of over-fitting. Moreover, the errors propagate and enlarge layer-wisely (Dong, Chen, and Pan 2017), and finally lead to a large performance drop on $\mathcal{F}^S$.

Cross Distillation

To address the above issue, we propose cross distillation, a novel layer-wise distillation method targeting at few shot network compression. Since the estimation errors are accumulated on the student network $\mathcal{F}^S$ and $\mathbf{h}^T$ are taken as the target during layer-wise distillation, we direct $\mathbf{h}^T$ to $\mathcal{F}^S$ in substitution of $\mathbf{h}^S$ to reduce the historically accumulated errors, as is shown in Figure 1(b). We now replace $\mathcal{L}^r$ in Equation 1 by the correction loss defined as

$$\mathcal{L}^c(\mathbf{W}^S) = \|\sigma(\mathbf{W}^T * \mathbf{h}^T) - \sigma(\mathbf{W}^S * \mathbf{h}^T)\|_F^2. \qquad (2)$$



(a) Layer-wise distillation (b) Correction (c) Imitation (d) Soft cross distillation

Figure 1: The four categories of layerwise distillation. a) is the traditional pattern; b) guides the teacher to student in order to compensate estimation errors on the student; c) guides the student to the teacher to make it aware of the errors on the student; d) offers a soft connection to balance b) and c) .

In the forward pass of $\mathcal{F}^S$, however, directing $\mathbf{h}^T$ to $\mathcal{F}^S$ results in inconsistency $\epsilon^S = \|\sigma(\mathbf{W}^S * \mathbf{h}^T) - \sigma(\mathbf{W}^S * \mathbf{h}^S)\|_F^2$ because $\mathcal{F}^S$ takes $\mathbf{h}^T$ from $\mathcal{F}^T$ in the training while it is expected to behave along during inference. Therefore, minimizing the regularized $\mathcal{L}^c$ could lead to a biasedly-optimized student net.

In order to maintain the consistency during forward pass for $\mathcal{F}^S$ and simultaneously make the teacher aware of the accumulated errors on the student net, we can inverse the strategy by guiding $\mathbf{h}^S$ to $\mathcal{F}^T$, as is shown in Figure 1(c). We call this process as imitation, since the student network tries to mimic the behavior of the teacher network given its current estimations. Similarly we can substitute $\mathcal{L}^r$ in Equation 1 with the imitation loss defined as

$$\mathcal{L}^i(\mathbf{W}^S) = \|\sigma(\mathbf{W}^T * \mathbf{h}^S) - \sigma(\mathbf{W}^S * \mathbf{h}^S)\|_F^2. \quad (3)$$

Despite the teacher network now can provide error-aware supervised signal, such connection brings inconsistency on the teacher network, i.e., $\epsilon^T = \|\sigma(\mathbf{W}^T * \mathbf{h}^S) - \sigma(\mathbf{W}^T * \mathbf{h}^T)\|_F^2$. As a result of ϵ^T, the errors in $\mathbf{h}^S$ is be enlarged by $\mathbf{W}^T$ during layer-wise propagation, leading to deviated supervision for $\mathcal{F}^S$ that deteriorates the distillation.

Consequently, the correction loss $\mathcal{L}^c$ and the imitation loss $\mathcal{L}^i$ compensate each other, and it is necessary to find a proper balance between them. A natural choice is through convex combination tuned by μ, i.e.

$$\tilde{\mathcal{L}} = \mu\mathcal{L}^c + (1-\mu)\mathcal{L}^i, \quad \mu \in [0,1]. \quad (4)$$

Substituting $\mathcal{L}^r$ in Equation 1 with $\tilde{\mathcal{L}}$ yields the objective function for cross distillation.

Theoretical Analysis The inconsistency gaps ϵ^T and ϵ^S of cross distillation make it still unclear how the proposed method manipulates the propagation of estimation errors, and why minimizing the regularized $\tilde{\mathcal{L}}$ is on the right direction to improve the student net $\mathcal{F}^S$. To theoretically justify cross distillation, we follow (Friedlander and Tseng 2007) to substitute $\mathcal{L}^r$ with $\tilde{\mathcal{L}}$, and equivalently reformulate the unconstrained problem in Equation 1 to the constrained optimization problem as

$$\min_{\mathbf{W}^S \in \mathcal{C}} \tilde{\mathcal{L}}, \quad \mathcal{C} = \{\mathbf{W}^S | \mathcal{R}(\mathbf{W}^S) \leq \epsilon(\lambda)\}, \quad (5)$$

where $\mathcal{C}$ is a compact set determined by the regularization $\mathcal{R}(\mathbf{W}^S)$ and λ. With Equation 5, we can now bound the gap of cross entropy between $\mathcal{F}^T$ and $\mathcal{F}^S$ for classification[1] with the following theorem.

Theorem 1. *Suppose both $\mathcal{F}^T$ and $\mathcal{F}^S$ are L-layer convolutional neural networks followed by the un-pruned softmax fully-connected layer. If the activation functions $\sigma(\cdot)$ are Lipchitz-continuous such as* $\mathrm{ReLU}()$*, the gap of softmax cross entropy $\mathcal{L}^{ce}$ between the network logits $\mathbf{o}^T = \mathcal{F}^T(\mathbf{x})$ and $\mathbf{o}^S = \mathcal{F}^S(\mathbf{x})$ can be bounded by*

$$|\mathcal{L}^{ce}(\mathbf{o}^T;\mathbf{y}) - \mathcal{L}^{ce}(\mathbf{o}^S;\mathbf{y})| \leq C\tilde{\mathcal{L}}_L + \sum_{l=1}^{L-1}\prod_{k=l}^{L} C_k'(\mu)\tilde{\mathcal{L}}_l, \quad (6)$$

where C and $C'(\mu)$ are constants and $C'(\mu)$ is linear in μ.

Theorem 1 shows that 1) the gap of cross entropy between the student network $\mathcal{F}^S$ and teacher network $\mathcal{F}^T$ is upper bounded by $\tilde{\mathcal{L}}$, and therefore layer-wise minimization of the constrained optimization problem in Equation 5 could decrease the gap of cross entropy and finally improve $\mathcal{F}^S$. 2) The tightness of the upper bound is controlled by the trade-off hyper-parameter μ, which is a L-th order polynomial. A proper choice of μ may lead to a tighter bound that could better decrease the cross entropy gap. We leave the proof of Theorem 1 in the Appendix.

Soft Cross Distillation Although minimizing $\tilde{\mathcal{L}}$ is theoretically supported, the computation of $\tilde{\mathcal{L}}$ involves two loss terms with four convolutions to compute per batch of data, which doubles the training time. Here we propose another variant to balance $\mathcal{L}^c$ and $\mathcal{L}^i$ by empirically soften the hard connection of $\mathbf{h}^S$ and $\mathbf{h}^T$, as is shown in Figure 1(d). We define feature maps $\hat{\mathbf{h}}^T$ and $\hat{\mathbf{h}}^S$ after cross connection as the convex combination of $\mathbf{h}^T$ and $\mathbf{h}^S$, i.e.,

$$\begin{bmatrix}\hat{\mathbf{h}}^T \\ \hat{\mathbf{h}}^S\end{bmatrix} = \begin{bmatrix}\alpha & 1-\alpha \\ 1-\beta & \beta\end{bmatrix}\begin{bmatrix}\mathbf{h}^T \\ \mathbf{h}^S\end{bmatrix}, \quad (7)$$

where $\alpha, \beta \in [0,1]$ are the hyper-parameters that adjust how many percentages are used for cross connection, and therefore the magnitude of inconsistencies ϵ^T and ϵ^S can be well

[1]For regression problems, similar theorem can be established as well.

controlled. The convex combination ensures the norm of input to be nearly identical after cross connection (assuming $\|\mathbf{h}^T\| \approx \|\mathbf{h}^S\|$), and therefore parameter magnitude stays unchanged. We define the loss of soft cross distillation as

$$\hat{\mathcal{L}}(\mathbf{W}^S) = \|\sigma(\mathbf{W}^T * \hat{\mathbf{h}}^T) - \sigma(\mathbf{W}^S * \hat{\mathbf{h}}^S)\|_F^2, \quad (8)$$

which can substitute the estimation error $\mathcal{L}^r$ in Equation 1 as an alternative way for cross distillation.

Combined with Network Pruning

Cross distillation can be readily combined with a set of popular network compression techniques such as pruning or quantization, by taking different regularization $\mathcal{R}(\mathbf{W}^S)$ in Equation 1. Here we take pruning as an illustration example. For non-structured pruning, we choose $\mathcal{R} = \|\mathbf{W}^S\|_1 = \sum_{i,j,h,w} |W^S_{ijhw}|$; and for structured pruning such as channel pruning, we choose $\mathcal{R}(\mathbf{W}^S) = \|\mathbf{W}^S\|_{2,1} = \sum_i \|\mathbf{W}^S_i\|_2$, where $\mathbf{W}^S_i \in R^{c_i \times k \times k}$.

To solve Equation 8 regularized by the above penalties, we can adopt the proximal gradient method (Parikh, Boyd, and others 2014), i.e., iteratively update $\mathbf{W}^S$ by:

$$\mathbf{W}^S_{t+1} = \mathrm{Prox}_{\lambda\mathcal{R}}(\mathbf{W}^S_t - \eta\nabla\tilde{\mathcal{L}}(\mathbf{W}^S_t)), \quad (9)$$

where $\mathrm{Prox}_{\lambda\mathcal{R}}(u) = \arg\min_x \frac{1}{2}\|x-u\|_F^2 + \mathcal{R}(x)$ is the proximal operator for $\mathcal{R}$. When $\mathcal{R}$ is chosen as $\|\cdot\|_1$, the proximal mapping can be expressed as the soft-threshold determined by λ, i.e.,

$$\mathrm{Prox}_{\lambda\|\cdot\|_1}(W^S_{ijhw}) = \begin{cases} W^S_{ijhw} - \lambda & W^S_{ijhw} > \lambda \\ 0 & |W^S_{ijhw}| \le \lambda \\ W^S_{ijhw} + \lambda & W^S_{ijhw} < -\lambda \end{cases}. \quad (10)$$

For structured pruning, since $\mathcal{R} = \|\mathbf{W}^S\|_{2,1}$ is separable w.r.t. $\mathbf{W}^S_i$, the proximal mapping for $\mathrm{Prox}_{\lambda\|\cdot\|_{2,1}}(\mathbf{W}^S_i)$ can be computed as

$$\mathrm{Prox}_{\lambda\|\cdot\|_2}(\mathbf{W}^S_i) = \max(1 - \frac{\lambda}{\|\mathbf{W}^S_i\|_2}, 0) \cdot \mathbf{W}^S_i, \quad (11)$$

and the solution to Equation 9 can be obtained group-wisely from Equation 11.

As suggested by past works (Zhu and Gupta 2017; He, Zhang, and Sun 2017), we linearly increase λ to smoothly prune the student network, which empirically gives better results. Given the maximum number of training steps T and the target sparsity ratio r assigned by users, we update λ by $\lambda_t = r + (1-r) * t/T$. An overall workflow of our proposed method is given in Algorithm 1.

Finally, we remark that our method works for network quantization as well. By taking $\mathcal{R}(\mathbf{W}^S)$ as the penalty to quantization points, our method can be combined with Straight Through Estimator (STE) (Bengio, Léonard, and Courville 2013) or ProxQuant (Bai, Wang, and Liberty 2019). See Appendix for details.

Algorithm 1 Cross distillation

Input:
The pre-trained teacher model $\mathcal{F}^T$;
Training samples $\{\mathbf{x}_n, \mathbf{y}_n\}_{n=1}^N$;
Target sparsity ratio r;
Output:
The compact student model $\mathcal{F}^S$

for $l = 1, ...L$ **do**
  for $t = 1, ...T$ **do**
    Forward pass $\{\mathbf{x}_n\}_{n=1}^N$ to obtain $\mathbf{h}^T_{l-1}$ and $\mathbf{h}^S_{l-1}$;
    Calculate the loss in Equation 4 or 8;
    Update $\mathbf{W}^S_t$ with SGD/Adam optimizer;
    Obtain $\mathbf{W}^S_{t+1}$ with $\mathrm{Prox}_{\lambda\mathcal{R}}$ in Equation 10 or 11;
    Increase the pruning threshold λ_t linearly;
  end for
end for

Experiments

We conduct a series of experiments to verify the effectiveness of cross distillation. We take structured and unstructured pruning for demonstration, both of which are popular approaches to reduce computational FLOPs and sizes of neural networks. To better understand the proposed method, we also provide further analysis on how cross distillation help reduce the estimation error against varying size of the training set. Due to limited space, we only present main results, while additional experiments and detailed implementations can be found in the Appendix. Our implementation in PyTorch is available at https://github.com/haolibai/Cross-Distillation.git.

Setup

Throughout the experiment, we use VGG (Simonyan and Zisserman 2014) and ResNet (He et al. 2016) as base networks, and evaluations are performed on CIFAR-10 and ImageNet-ILSVRC12. As we consider the setting of few shot image classification, we randomly select K-shot instances per class from the training set. All experiments are averaged over five runs with different random seeds, and results of means and standard deviations are reported [2]

Baselines For structured pruning, we compare our proposed methods against a number of baselines: 1) L1-norm pruning (Li et al. 2016), a data-free approach; 2) Back-propagation (BP) based fine-tuning on L1-norm pruned models; 3) FitNet (Romero et al. 2014) and 4) FSKD (Li et al. 2018), both of which are knowledge distillation methods; 5) ThiNet (Luo, Wu, and Lin 2017) and 6) Channel Pruning (CP) (He, Zhang, and Sun 2017), both of which are layer-wise regression based channel pruning methods. For unstructured pruning, we modify 1) to element-wise L1-norm based pruning (Zhu and Gupta 2017). Besides, 4) FSKD, 5) ThiNet and 6) CP are removed since they are only applicable in channel pruning.

For our proposed method, we compare to three variants for ablation study: Ours-NC (no cross distillation) by solving Equation 1, Ours by solving Equation 4 and Ours-S (soft cross distillation) by solving Equation 8. For Ours, we choose $\mu = 0.6$ for VGG networks and $\mu = 0.9$ for

[2]Note that for each run, we fix the random seed and remove all the randomness such as data augmentation and data shuffling.

Table 1: The top-1 accuracy (%) of structured pruning with VGG-16 on CIFAR-10 with different training sizes. We use VGG-50% as the pruning scheme, and the original accuracy of the original model is 93.51%.

Methods	1	2	3	5	10	50
L1-norm	$14.36_{\pm 0.00}$	$14.36_{\pm 0.00}$	$14.36_{\pm 0.00}$	$14.36_{\pm 0.00}$	$14.36_{\pm 0.00}$	$14.36_{\pm 0.00}$
BP	$49.24_{\pm 1.76}$	$49.32_{\pm 1.88}$	$51.39_{\pm 1.53}$	$55.73_{\pm 1.19}$	$57.48_{\pm 0.91}$	$64.69_{\pm 0.43}$
FSKD	$47.91_{\pm 1.82}$	$55.44_{\pm 1.71}$	$61.76_{\pm 1.39}$	$65.69_{\pm 1.08}$	$72.20_{\pm 0.74}$	$75.46_{\pm 0.49}$
FitNet	$48.51_{\pm 2.51}$	$71.51_{\pm 2.03}$	$76.22_{\pm 1.95}$	$81.10_{\pm 1.13}$	$85.40_{\pm 1.02}$	$88.46_{\pm 0.76}$
ThiNet	$58.06_{\pm 1.71}$	$72.07_{\pm 1.68}$	$75.37_{\pm 1.59}$	$78.03_{\pm 1.24}$	$81.15_{\pm 0.85}$	$86.12_{\pm 0.45}$
CP	$66.03_{\pm 1.56}$	$75.23_{\pm 1.49}$	$77.98_{\pm 1.47}$	$81.53_{\pm 1.29}$	$83.59_{\pm 0.78}$	$87.27_{\pm 0.27}$
Ours-NC	$65.57_{\pm 1.61}$	$75.44_{\pm 1.69}$	$78.40_{\pm 1.53}$	$81.20_{\pm 1.13}$	$84.07_{\pm 0.83}$	$87.67_{\pm 0.29}$
Ours	$\mathbf{69.25_{\pm 1.39}}$	$\mathbf{80.65_{\pm 1.47}}$	$\mathbf{82.08_{\pm 1.41}}$	$\mathbf{84.91_{\pm 0.98}}$	$\mathbf{86.61_{\pm 0.71}}$	$87.64_{\pm 0.24}$
Ours-S	$68.53_{\pm 1.59}$	$76.83_{\pm 1.43}$	$80.16_{\pm 1.32}$	$84.28_{\pm 1.19}$	$86.30_{\pm 0.79}$	$\mathbf{88.65_{\pm 0.33}}$

Table 2: The top-5 accuracy (%) of structured pruning with ResNet-34 on ILSVRC-12 with different training sizes. The first three columns use 50, 100 and 500 randomly sampled training instances, while the last three columns use $K = 1, 2, 3$ samples per class. We use Res-50% as the pruning scheme, and the top-1 and top-5 accuracies of the original model are 73.32% and 91.40%.

Methods	50	100	500	1	2	3
L1-norm	$72.94_{\pm 0.00}$	$72.94_{\pm 0.00}$	$72.94_{\pm 0.00}$	$72.94_{\pm 0.00}$	$72.94_{\pm 0.00}$	$72.94_{\pm 0.00}$
BP	$83.18_{\pm 1.86}$	$84.32_{\pm 1.29}$	$85.34_{\pm 0.89}$	$85.76_{\pm 0.73}$	$86.05_{\pm 0.51}$	$86.29_{\pm 0.56}$
FSKD	$82.53_{\pm 1.52}$	$84.58_{\pm 1.13}$	$86.67_{\pm 0.78}$	$87.08_{\pm 0.76}$	$87.23_{\pm 0.52}$	$87.20_{\pm 0.43}$
FitNet	$86.86_{\pm 1.81}$	$87.12_{\pm 1.63}$	$87.73_{\pm 0.96}$	$87.66_{\pm 0.84}$	$88.61_{\pm 0.76}$	$\mathbf{89.32_{\pm 0.78}}$
ThiNet	$85.67_{\pm 1.57}$	$85.54_{\pm 1.39}$	$86.97_{\pm 0.89}$	$87.42_{\pm 0.76}$	$87.52_{\pm 0.68}$	$87.53_{\pm 0.50}$
CP	$86.34_{\pm 1.24}$	$86.38_{\pm 1.37}$	$87.41_{\pm 0.80}$	$88.03_{\pm 0.66}$	$87.98_{\pm 0.49}$	$88.21_{\pm 0.37}$
Ours-NC	$86.51_{\pm 1.71}$	$86.61_{\pm 1.20}$	$87.92_{\pm 0.75}$	$87.98_{\pm 0.60}$	$88.63_{\pm 0.49}$	$88.82_{\pm 0.38}$
Ours	$86.95_{\pm 1.59}$	$87.60_{\pm 1.13}$	$88.34_{\pm 0.69}$	$88.17_{\pm 0.73}$	$88.57_{\pm 0.40}$	$88.59_{\pm 0.41}$
Ours-S	$\mathbf{87.42_{\pm 1.69}}$	$\mathbf{87.73_{\pm 1.17}}$	$\mathbf{88.60_{\pm 0.82}}$	$\mathbf{88.40_{\pm 0.61}}$	$\mathbf{88.84_{\pm 0.48}}$	$88.87_{\pm 0.35}$

ResNets. For Ours-S, we set $(\alpha, \beta) = (0.9, 0.3)$ on VGG networks and $(0.9, 0.5)$ on ResNets. Sensitivity analysis on these hyper-parameters are presented later. Details on parameter settings and baseline implementations are in the Appendix.

Pruning schemes The structured pruning schemes are similar to those used in (Li et al. 2016; 2018). For the VGG-16 network, we denote the three pruning schemes in (Li et al. 2018) in the ascending order of sparsity as VGG-A, VGG-B and VGG-C respectively. We further prune 50% channels layer-wisely and denote the resulting scheme as VGG-50%. For ResNet-34, we remove $r\%$ channels in the middle layer of the first three residual blocks with some sensitive layers skipped (e.g., layer 2, 8, 14, 16). The last residual block is kept untouched. The resulting structured pruning schemes are denoted as Res-$r\%$. Besides, we further remove 50% channels for the last block to reduce more FLOPs when $r = 70\%$, denoted as Res-70%+. The reduction of model sizes and computational FLOPs for structured pruned models are shown in the Appendix.

In terms of unstructured pruning, we follow a similar pattern in (Zhu and Gupta 2017) by removing $r = \{50\%, 70\%, 90\%, 95\%\}$ parameters for both the VGG network and ResNet, and each layer is treated equally.

Results

Structured Pruning We evaluate structured pruning with VGG-16 on CIFAR-10 and ResNet-34 on ILSVRC-12. Table 1 and 2 shows the results with different number of training instances when the pruning schemes are fixed. It can be observed that both Ours and Ours-S generally outperform the rest baselines on both networks, whereas Ours enjoys a larger advantage on VGG-16 while Ours-S is superior on ResNet-34. Meanwhile, as the training size decreases, cross distillation brings more advantages comparing to the rest baselines, indicating that the layer-wise regression can benefit more from cross distillation when the student network over-fits more seriously on fewer training samples.

In the next, we fix the training size and change the pruning schemes. We keep $K = 5$ on CIFAR-10 and $K = 1$ on ILSVRC-12, and the results are listed in Table 3 and Table 4 respectively. Again on both datasets our proposed cross distillation performs consistently better comparing to the rest approaches. Besides, the gain from cross distillation becomes larger as the sparsity of the student network increases (e.g., VGG-C and ResNet-70%+). We suspect that networks with sparser structures tend to suffer more from higher estimation errors, which poses more necessity for cross distillation to reduce the errors.

Table 3: The top-1 accuracy (%) of different structured pruning schemes with VGG-16 on CIFAR-10. 10 samples per class are used.

Methods	VGG-50%	VGG-A	VGG-B	VGG-C
L1-norm	$14.36_{\pm 0.00}$	$88.32_{\pm 0.00}$	$32.87_{\pm 0.00}$	$10.00_{\pm 0.00}$
BP	$55.73_{\pm 1.19}$	$93.10_{\pm 0.09}$	$87.17_{\pm 0.49}$	$62.45_{\pm 1.25}$
FSKD	$65.69_{\pm 1.08}$	$93.52_{\pm 0.23}$	$90.69_{\pm 0.12}$	$81.79_{\pm 1.01}$
FitNet	$85.40_{\pm 1.02}$	$93.50_{\pm 0.06}$	$92.42_{\pm 0.32}$	$84.65_{\pm 1.53}$
ThiNet	$81.15_{\pm 0.85}$	$93.61_{\pm 0.05}$	$92.20_{\pm 0.16}$	$79.19_{\pm 0.91}$
CP	$83.59_{\pm 0.78}$	$\mathbf{93.70_{\pm 0.04}}$	$92.29_{\pm 0.15}$	$80.82_{\pm 0.73}$
Ours-NC	$84.07_{\pm 0.83}$	$\mathbf{93.69_{\pm 0.07}}$	$92.35_{\pm 0.14}$	$83.90_{\pm 0.78}$
Ours	$\mathbf{86.61_{\pm 0.71}}$	$\mathbf{93.65_{\pm 0.08}}$	$\mathbf{92.60_{\pm 0.11}}$	$\mathbf{85.81_{\pm 0.80}}$
Ours-S	$86.30_{\pm 0.79}$	$\mathbf{93.70_{\pm 0.07}}$	$\mathbf{92.68_{\pm 0.13}}$	$85.10_{\pm 0.75}$

Table 4: The top-5 accuracy (%) of different structured pruning schemes with ResNet-34 on ILSVRC-12. 1 sample per class is used.

Methods	Res-30%	Res-50%	Res-70%	Res-70%+
L1-norm	$84.54_{\pm 0.00}$	$72.94_{\pm 0.00}$	$31.84_{\pm 0.00}$	$15.30_{\pm 0.00}$
BP	$88.66_{\pm 0.59}$	$85.76_{\pm 0.73}$	$80.04_{\pm 0.90}$	$63.25_{\pm 1.05}$
FSKD	$89.56_{\pm 0.52}$	$87.08_{\pm 0.76}$	$80.82_{\pm 0.62}$	$67.04_{\pm 0.56}$
FitNet	$88.56_{\pm 0.58}$	$87.66_{\pm 0.84}$	$\mathbf{82.72_{\pm 0.88}}$	$68.31_{\pm 0.81}$
ThiNet	$89.74_{\pm 0.65}$	$87.42_{\pm 0.76}$	$79.40_{\pm 0.66}$	$63.65_{\pm 0.78}$
CP	$89.65_{\pm 0.78}$	$88.03_{\pm 0.66}$	$81.13_{\pm 0.85}$	$68.18_{\pm 0.79}$
Ours-NC	$\mathbf{90.34_{\pm 0.53}}$	$87.98_{\pm 0.60}$	$82.11_{\pm 0.71}$	$69.03_{\pm 0.92}$
Ours	$90.08_{\pm 0.47}$	$88.17_{\pm 0.65}$	$\mathbf{82.71}_{\pm 0.76}$	$\mathbf{73.53}_{\pm 0.74}$
Ours-S	$\mathbf{90.32_{\pm 0.58}}$	$\mathbf{88.40_{\pm 0.61}}$	$82.65_{\pm 0.68}$	$69.47_{\pm 0.79}$

Unstructured Pruning For unstructured pruning, here we present results of the VGG-16 network on ILSVRC-12 dataset. Similar to structured pruning, we first fix the pruning scheme and vary the training size, and the results are given in Table 5. It can be observed that both Ours and Ours-S significantly outperform the rest methods, and the improvement is even larger comparing to structured pruning One reason could be the irregular sparsity of network parameters cab better compensate the layer-wisely accumulated errors on $\mathcal{F}^S$.

Similarly, we test our methods with different sparsities and hold the training size fixed as $K = 1$, and Table 6 shows the results. As the sparsity r increases, cross distillation brings more improvement, especially on VGG-95% with a nearly 10% and 14% increase of accuracy for Ours and Ours-S respectively.

Further Analysis

The Estimation Errors v.s. Inconsistency Cross distillation brings the inconsistencies ϵ^T, ϵ^S that could affect the reduction of estimation errors $\mathcal{L}^r$. To quantitatively investigate the effects, we compare ϵ^T, ϵ^S as well as $\mathcal{L}^r$ at different layers of the VGG-16 network on the test set of CIFAR-10. We take three student networks trained by the correction loss $\mathcal{L}^c$, the imitation loss $\mathcal{L}^i$ as well as soft distillation loss $\hat{\mathcal{L}}$ respectively. We choose unstructured pruning with VGG-90% and vary K between $\{1, 10\}$, and the results are shown in Figure 2(a), 2(b) and 2(c) respectively. Note that we have normalized the loss values by dividing the nonzero leftmost bar in each sub-figure.

It can be observed that the student net trained by $\mathcal{L}^c$ has a large ϵ^S with $\epsilon^T = 0$, and vice versa for that trained by $\mathcal{L}^i$. On the contrary, the student net trained by $\hat{\mathcal{L}}$ shows both lower ϵ^T and ϵ^S, and the estimation error $\mathcal{L}^r$ is properly reduced as well. The results indicate that by properly controlling the magnitude of inconsistencies ϵ^T and ϵ^S with soft connection, cross distillation can indeed reduce estimation errors $\mathcal{L}^r$ and improve the student network.

Generalization Ability One potential issue troubles us is the generalization of cross distillation, since the training of Ours and Ours-S is somehow biased comparing to Ours-NC that directly minimizes the estimation error $\mathcal{L}^r$. Since estimation errors $\mathcal{L}^r$ among feature maps and cross entropy $\mathcal{L}^{ce}$ of logits directly reflect the closeness between $\mathcal{F}^T$ and $\mathcal{F}^S$ during inference, we compare both results among student nets obtained by Ours-NC, Ours and Ours-S respectively. We again take unstructured pruning with VGG-90% on the test set of CIFAR-10, and the rest settings are kept unchanged. For ease of comparison, we similarly divide values of Ours and Ours-S by those obtained by Ours-NC. Ratios smaller than 1 indicate a more generalizable student net.

From Figure 3, we can find that while the ratios in shallower layers are above 1, they rapidly go down at deeper layers such as conv4.3 as well as the logits, which is consistent with Figure 3 that cross distillation tends to better benefit deeper layers. Moreover, although increasing K from 1 to 10 gives lower ratios of $\mathcal{L}^r$ at convolutional layers, the ratios of $\mathcal{L}^{ce}$ increases at the network logits, which lead to less improvement for classification when more training samples are available. The phenomenons are consistent with the results in Table 1, Table 2 and Table 5. In summary, cross distillation can indeed generalize well when $\mathcal{F}^T$ and $\mathcal{F}^S$ are properly mixed in the few shot setting.

Sensitivity Analysis Finally, we present sensitivity analysis for cross distillation. We perform grid search by varying $\mu \in [0, 1]$ for Ours and $(\alpha, \beta) \in [0, 1]^2$ for Ours-S at an interval of 0.1. We take VGG-16 for structured pruning and ResNet-56 for unstructured pruning on CIFAR-10 with $K = 5$, while ILSVRC-12 experiments adopt the same setting of μ and (α, β) found by these experiments. From Figure 4(a) and Figure 4(b), Ours consistently outperforms Ours-NC, where the best configurations appear at around $\mu = 0.6$ for VGG-16 and $\mu = 0.9$ for ResNet-56. Furthermore, we find that simply using the correction loss $\mu = 0.0$ or the imitation loss $\mu = 1.0$ also achieve reasonable results[3]. In terms of Ours-S in Figure 4(c) and 4(d) , we find that on left regions $\{(\alpha, \beta)|\alpha+\beta < 1\}$ $\mathcal{F}^T$ and $\mathcal{F}^S$ permute the input too much and thereon lead to significant drops of performance. For right regions $\{(\alpha, \beta)|\alpha + \beta > 1\}$, most configurations consistently outperform Ours-NC $(1.0, 1.0)$, and the peaks occur somewhere in the middle of the regions.

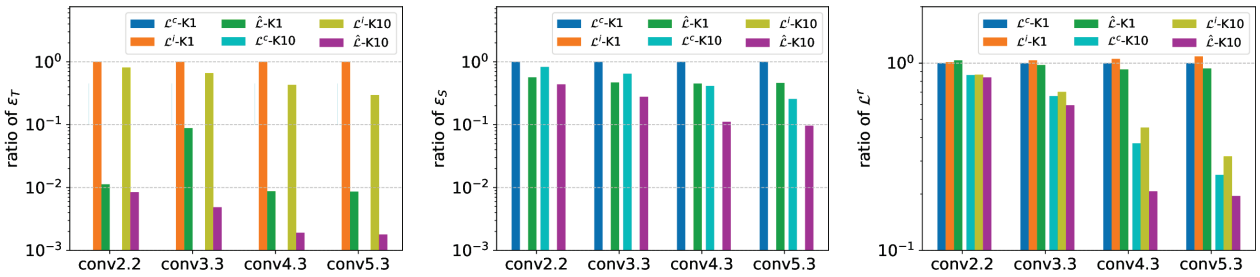
Conclusion

In this paper, we present cross distillation, a novel knowledge distillation approach for learning compact student net-

[3]The accuracies are 83.44% and 83.32% respectively on VGG-16, and 84.93% and 86.63% respectively on ResNet-56.

Table 5: The top-5 accuracy (%) of unstructured pruning with VGG-16 on ILSVRC-12 with different training sizes. The first three columns use 50, 100 and 500 randomly sampled training instances, while the last three columns use $K = 1, 2, 3$ samples per class. We use Res-90% as the pruning scheme, and the top-1 and top-5 accuracies of the original model are 73.72% and 91.51%.

Methods	50	100	500	1	2	3
L1-norm	$0.5_{\pm 0.00}$	$0.5_{\pm 0.00}$	$0.5_{\pm 0.00}$	$0.5_{\pm 0.00}$	$0.5_{\pm 0.00}$	$0.5_{\pm 0.00}$
BP	$42.87_{\pm 2.07}$	$48.78_{\pm 1.43}$	$65.47_{\pm 1.15}$	$71.25_{\pm 0.97}$	$74.85_{\pm 0.71}$	$76.04_{\pm 0.48}$
FitNet	$52.66_{\pm 2.93}$	$57.09_{\pm 2.14}$	$76.59_{\pm 1.45}$	$80.14_{\pm 1.23}$	$82.27_{\pm 0.70}$	$83.14_{\pm 0.51}$
Ours-NC	$78.73_{\pm 1.78}$	$83.29_{\pm 1.12}$	$85.04_{\pm 0.93}$	$85.36_{\pm 0.61}$	$85.21_{\pm 0.41}$	$85.49_{\pm 0.46}$
Ours	$\mathbf{83.81_{\pm 1.49}}$	$86.21_{\pm 1.09}$	$87.19_{\pm 0.96}$	$87.61_{\pm 0.82}$	$87.78_{\pm 0.45}$	$87.86_{\pm 0.39}$
Ours-S	$83.67_{\pm 1.52}$	$\mathbf{86.72_{\pm 1.23}}$	$\mathbf{87.82_{\pm 1.04}}$	$\mathbf{88.14_{\pm 0.74}}$	$\mathbf{88.23_{\pm 0.61}}$	$\mathbf{88.38_{\pm 0.43}}$



(a) ϵ^T (b) ϵ^S (c) $\mathcal{L}^r$

Figure 2: The comparisons among inconsistencies ϵ^T, ϵ^S as well as estimation errors $\mathcal{L}^r$ on the test set of CIFAR-10. The colors denote what kind of loss and values of K are adopted for training. Best viewed in color.

Table 6: The top-5 accuracy (%) of unstructured pruning with VGG-16 on ILSVRC-12 with different pruning schemes. 1 sample per class is adopted.

Methods	VGG-50%	VGG-70%	VGG-90%	VGG-95%
L1-norm	$89.21_{\pm 0.00}$	$66.91_{\pm 0.00}$	$0.5_{\pm 0.00}$	$0.50_{\pm 0.00}$
BP	$90.61_{\pm 0.20}$	$88.08_{\pm 0.19}$	$71.25_{\pm 0.97}$	$42.37_{\pm 1.59}$
FitNet	$88.36_{\pm 0.46}$	$86.76_{\pm 0.67}$	$80.14_{\pm 1.23}$	$59.08_{\pm 1.78}$
Ours-NC	$91.47_{\pm 0.12}$	$91.16_{\pm 0.10}$	$85.21_{\pm 0.41}$	$66.74_{\pm 1.36}$
Ours	$91.58_{\pm 0.06}$	$91.24_{\pm 0.14}$	$87.61_{\pm 0.49}$	$76.65_{\pm 1.23}$
Ours-S	$\mathbf{91.68_{\pm 0.09}}$	$\mathbf{91.54_{\pm 0.11}}$	$\mathbf{88.14_{\pm 0.61}}$	$\mathbf{80.64_{\pm 1.03}}$

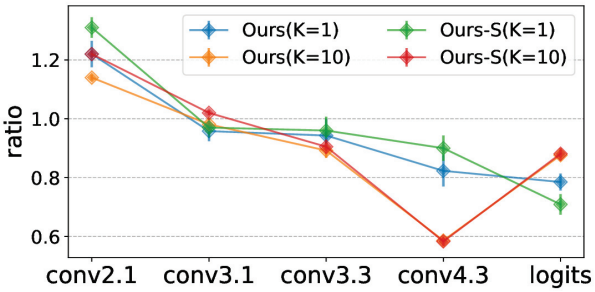
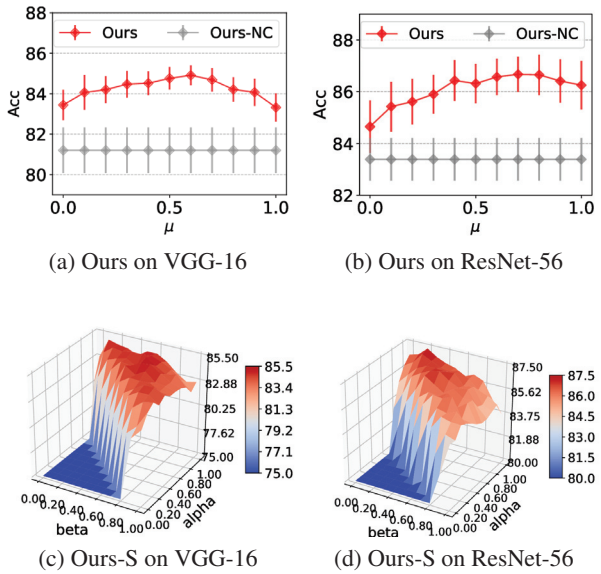


Figure 3: The estimation errors $\mathcal{L}^r$ of Ours and Ours-S, both of which are divided by Ours-NC. Best viewed in color.



(a) Ours on VGG-16 (b) Ours on ResNet-56

(c) Ours-S on VGG-16 (d) Ours-S on ResNet-56

Figure 4: Sensitivity analysis of $\mu \in [0, 1]$ for Ours and (α, β) on $[0, 1]^2$ for Ours-S.

work given limited number of training instances. By reducing estimation errors between the student network and

teacher network, cross distillation can bring a more powerful and generalizable student network. Extensive experiments on benchmark datasets demonstrate the superiority of our method against various competitive baselines.

Acknowledgement This work is supported by the Research Grant Coucil of the Hong Kong Special Administrative Region, China (No.CUHK 14208815 and No.CUHK 14210717 of the General Research Fund). We sincerely thank Xin Dong, Jiajin Li, Jiaxing Wang and Shilin He for helpful discussions, as well as the anonymous reviewers for insightful suggestions.

References

Bai, Y.; Wang, Y.-X.; and Liberty, E. 2019. Proxquant: Quantized neural networks via proximal operators. *ICLR*.

Banner, R.; Hubara, I.; Hoffer, E.; and Soudry, D. 2018. Scalable methods for 8-bit training of neural networks. In *NIPS*, 5145–5153.

Bengio, Y.; Léonard, N.; and Courville, A. C. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR* abs/1308.3432.

Bhardwaj, K.; Suda, N.; and Marculescu, R. 2019. Dream distillation: A data-independent model compression framework. *arXiv preprint arXiv:1905.07072*.

Chen, H.; Wang, Y.; Xu, C.; Yang, Z.; Liu, C.; Shi, B.; Xu, C.; Xu, C.; and Tian, Q. 2019. Dafl: Data-free learning of student networks. In *ICCV*.

Chen, S.; Wang, W.; and Pan, S. J. 2019a. Cooperative pruning in cross-domain deep neural network compression. In *IJCAI*.

Chen, S.; Wang, W.; and Pan, S. J. 2019b. Deep neural network quantization via layer-wise optimization using limited training data. In *AAAI*.

Dong, X.; Chen, S.; and Pan, S. 2017. Learning to prune deep neural networks via layer-wise optimal brain surgeon. In *NIPS*, 4857–4867.

Friedlander, M. P., and Tseng, P. 2007. Exact regularization of convex programs. *SIAM Journal on Optimization* 18(4):1326–1350.

Gupta, S.; Hoffman, J.; and Malik, J. 2016. Cross modal distillation for supervision transfer. In *CVPR*, 2827–2836.

Han, S.; Mao, H.; and Dally, W. J. 2016. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *ICLR*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

He, Y.; Zhang, X.; and Sun, J. 2017. Channel pruning for accelerating very deep neural networks. In *ICCV*.

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

Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; and Graf, H. P. 2016. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*.

Li, T.; Li, J.; Liu, Z.; and Zhang, C. 2018. Knowledge distillation from few samples. *arXiv preprint arXiv:1812.01839*.

Li, Y.; Dong, X.; Zhang, S.; Bai, H.; Yuanpeng, C.; and Wang, W. 2020. Rtn: Reparameterized ternary network. In *AAAI*.

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

Luo, J.-H.; Wu, J.; and Lin, W. 2017. Thinet: A filter level pruning method for deep neural network compression. In *ICCV*, 5058–5066.

Nayak, G. K.; Mopuri, K. R.; Shaj, V.; Babu, R. V.; and Chakraborty, A. 2019. Zero-shot knowledge distillation in deep networks. *arXiv preprint arXiv:1905.08114*.

Parikh, N.; Boyd, S.; et al. 2014. Proximal algorithms. *Foundations and Trends® in Optimization* 1(3):127–239.

Romero, A.; Ballas, N.; Kahou, S. E.; Chassang, A.; Gatta, C.; and Bengio, Y. 2014. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*.

Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Wen, L.; Zhang, X.; Bai, H.; and Xu, Z. 2019. Structured pruning of recurrent neural networks through neuron selection. *arXiv preprint arXiv:1906.06847*.

Wu, J.; Leng, C.; Wang, Y.; Hu, Q.; and Cheng, J. 2016. Quantized convolutional neural networks for mobile devices. In *CVPR*.

Wu, J.; Zhang, Y.; Bai, H.; Zhong, H.; Hou, J.; Liu, W.; and Huang, J. 2018. Pocketflow: An automated framework for compressing and accelerating deep neural networks. In *NIPS Workshop on CDNNRIA*.

Ye, J.; Wang, L.; Li, G.; Chen, D.; Zhe, S.; Chu, X.; and Xu, Z. 2018. Learning compact recurrent neural networks with block-term tensor decomposition. In *CVPR*, 9378–9387.

Zhang, X.; Zou, J.; He, K.; and Sun, J. 2015. Accelerating very deep convolutional networks for classification and detection. *IEEE TPAMI* 38(10):1943–1955.

Zhu, M., and Gupta, S. 2017. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*.