Supplementary Material for
Few Sample Knowledge Distillation for Efficient Network Compression

Tianhong Li[1] Jianguo Li[2] Zhuang Liu[3] Changshui Zhang[4]
[1]MIT [2]Intel Labs [3]UC Berkeley [4]Dept. Automation, Tsinghua University
tianhong@mit.edu, jianguoli@intel.com, zhuangl@berkeley.edu, zsc@tsinghua.edu.cn

A: FSKD with different # BCD iterations

In our FSKD algorithm, we can apply the block-coordinate descent for several iterations. However, we do not observe noticeable gains for the iteration number $T > 1$ over $T = 1$ as shown in Figure 1, so that we set $T = 1$ in all our following experiments. This may be due to the reason that in each iteration, the sub-problem is a linear optimization problem so that we can find exact minimization, which is consistent with the finding by [3].

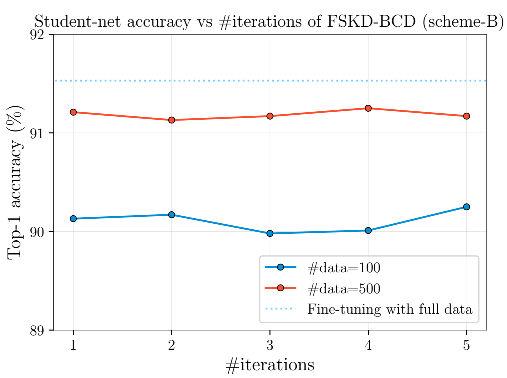


Figure 1: Accuracy vs #iterations of FSKD on CIFAR-10. Student-net is Prune-B by **filter pruning**.

B: FSKD-BCD vs. FSKD-SGD

	Acc. (%)	#Samples	Time (sec)
VGG-16	92.66	50000	
Prune-B + FSKD-BCD	90.17	100	3.7
Prune-B + FSKD-SGD	89.41	100	18.4
Prune-B + FSKD-BCD	91.21	500	19.3
Prune-B + FSKD-SGD	90.76	500	50.5

Table 1: Performance comparison between FSKD-BCD and FSKD-SGD by student-nets from **filter pruning** of VGG-16 with pruning scheme B on CIFAR-10.

In this section, we compared two FSKD optimization algorithms: FSKD-BCD uses the BCD algorithm on block-level and FSKD-SGD optimizes the loss all together with the standard SGD algorithm. We valuate both methods on VGG-16 models trained on CIFAR-10 and compressed using filter pruning (prune-B). As shown in Table 1, FSKD-BCD achieves better accuracy than FSKD-SGD while significantly improves time efficiency.

C: Proof of Theorem 1

Proof. When $\mathbf{W}$ is a point-wise convolution with tensor $\mathbf{W} \in \mathbb{R}^{n_o \times n_i \times 1 \times 1}$, both $\mathbf{W}$ and $\mathbf{Q}$ are degraded into matrix form. It is obvious that when condition $c1 \sim c3$ satisfied, the theorem holds with $\mathbf{W}' = \mathbf{Q} * \mathbf{W}$ in this case, where $*$ indicates matrix multiplication.

When $\mathbf{W}$ is a regular convolution with tensor $\mathbf{W} \in \mathbb{R}^{n_o \times n_i \times k \times k}$, the proof is non-trivial. Fortunately, recent work on network decoupling [1] presents an important theoretic result as the basis of our derivation.

Lemma 1. *Regular convolution can be exactly expanded to a sum of several depth-wise separable convolutions. Formally,* $\forall\, \mathbf{W} \in \mathbb{R}^{n_o \times n_i \times k \times k}$, $\exists\, \{\mathbf{P}_k, \mathbf{D}_k\}_{k=1}^{K}$, *where* $\mathbf{P}_k \in \mathbb{R}^{n_o \times n_i \times 1 \times 1}$, $\mathbf{D}_k \in \mathbb{R}^{1 \times n_i \times k \times k}$,

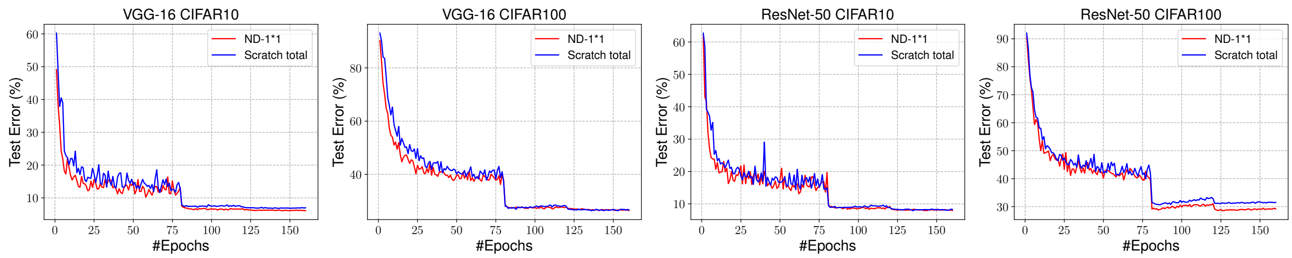
$$
\begin{aligned}
s.t.\ \ &(a) K \leq k^2; \\
&(b) \mathbf{W} = \sum\nolimits_{k=1}^{K} \mathbf{P}_k \circ \mathbf{D}_k,
\end{aligned}
\tag{1}
$$

where $\circ$ *is the compound operation, which means performing* $\mathbf{D}_k$ *before* $\mathbf{P}_k$.

Please refer to [1] for the details of proof for this Lemma. When $\mathbf{W}$ is applied to an input patch $\mathbf{x} \in \mathbb{R}^{n_i \times k \times k}$, we obtain a response vector $\mathbf{y} \in \mathbb{R}^{n_o}$ as

$$
\mathbf{y} = \mathbf{W} \otimes \mathbf{x}, \tag{2}
$$

where $y_o = \sum_{i=1}^{n_i} W_{o,i} \otimes x_i, o \in [n_o], i \in [n_i]$, and $\otimes$ here means convolution operation. $W_{o,i} = \mathbf{W}[o, i, :, :]$ is a tensor slice along the i-th input and o-th output channels, $x_i = \mathbf{x}[i, :, :]$ is a tensor slice along the i-th channel of 3D tensor $\mathbf{x}$.



((a)) ((b)) ((c)) ((d))

Figure 2: Test-accuracy at different epochs (a)VGG-16 on CIFAR-10, (b) VGG-16 on CIFAR-100, (c)ResNet-50 on CIFAR-10, (d)ResNet-50 on CIFAR-100. "scratch-total" is the 1st setting, while "ND-1*1"is the 2nd setting.

When point-wise convolution $\mathbf{Q}$ is added after $\mathbf{Q}$ without non-linear activation between them, we have

$$\mathbf{y}' = \mathbf{Q} \circ (\mathbf{W} \otimes \mathbf{x}). \tag{3}$$

With Lemma-1, we have

$$\mathbf{y}' = (\mathbf{Q} \circ \sum_{k=1}^{K} \mathbf{P}_k \circ \mathbf{D}_k) \otimes \mathbf{x} = (\sum_{k=1}^{K} (\mathbf{Q} * \mathbf{P}_k) \circ \mathbf{D}_k) \otimes \mathbf{x} \tag{4}$$

As both $\mathbf{Q}$ and $\mathbf{P}_k$ are degraded into matrix form, denoting $\mathbf{P}_k^{'} = \mathbf{Q} * \mathbf{P}_k$ and $\mathbf{W}' = \sum_{k=1}^{K} \mathbf{P}_k^{'} \circ \mathbf{D}_k$, we have $\mathbf{y}' = \mathbf{W}' \circ \mathbf{x}$. This proves the case when $\mathbf{W}$ is a regular convolution. □

D: Algorithm for iterative pruning and FSKD

Algorithm-1 describes the iteratively pruning and FSKD procedure to achieve extremely compression rate based on [2, 4, 5].

Algorithm 1 Iteratively pruning and FSKD Algorithm

input Teacher-net t, input data $\{X_i\}_{i=1}^{N}$,
prune-ratio-list $\{r_k\}_{k=1}^{K}$, number of iterations T
$s_{max} = \varnothing$
for $t = 1 : T$ **do**
  $q_{max} = 0$
  for $k = 1 : K$ **do**
    Prune s with ratio r_k to obtain student-net t
    Run FSKD with s, t and $\{X_i\}_{i=1}^{N}$, output s'
    Evaluation s' on validation set to obtain score q_k
    if $q_k > q_{mqx}$ **then**
      $q_{max} = q_k$
      $s_{max} = s'$
    end if
  end for
  Update teacher $t = s_{max}$
end for
output final student-net s_{max}.

E: Training only PW conv-layer is enough

People may challenge that learning 1×1-conv may loss representation power and ask why the added 1×1 convolution works so well with such few samples. According to the network decoupling theory (Lemma-1), any regular conv-layer could be decomposed into a sum of depthwise separable blocks, where each depthwise separable block consists of a depthwise (DW) convolution (for spatial correlation modeling) followed by a pointwise (PW) convolution (for cross-channel correlation modeling). The added 1×1 conv-layer is absorbed/merged into the previous PW layer finally. The decoupling yields that the number of parameters in PW-layer occupies most (>=80%) parameters of the whole network. We argue that learning only 1×1-conv is still very powerful, and make a **bold hypothesis** that PW conv-layer is more critical for performance than DW conv-layer. To verify this hypothesis, we conduct experiments on VGG16 and ResNet50 on CIFAR-10 and CIFAR-100 under below different settings.

(1) We train the network from random initialization with 160 epochs with learning-rate decay 1/10 at 80, 120 epochs from 0.01 to 0.0001.

(2) We start from a random initialized network (VGG16 or ResNet50), and do full rank decoupling ($K = k^2$ in Eq. 1) so that channels in DW layers are orthogonal, and PW layers are still fully random. Note that Lemma-1 ensures the network before and after decoupling are equivalent (i.e., able to transfer back and force from each other). We keep all the DW-layers fixed (with orthogonal basis from random data), and train only the PW layers with 160 epochs. We denote this scheme as ND-1*1.

Model	CIFAR-10(%)	CIFAR-100(%)
VGG-16	93.00	73.35
VGG-16 (ND-1*1)	93.91	73.61
ResNet-50	92.64	69.93
ResNet-50 (ND-1*1)	93.51	70.83

Table 2: Results by two schemes (1) full training (2) only training pointwise conv-layers (ND-1*1).

Note that except the setting explicitly described, all the other configurations (including training epochs, hyper-parameters, hardware platform, etc) are kept the same on both experimental cases. Table 2 lists the experimental results on these two cases on both datasets with two different network structures. It is obvious that the 2nd case (ND-1*1) clearly outperforms the 1st case. Figure 2 further illustrates the test accuracy at different training epochs, which clear shows that the 2nd case (ND-1*1) converges faster and better than the 1st case. This experiment verifies our hypothesis that when keeping DW channels orthogonal, training only the pointwise (1×1) conv-layer is accurate enough, or even better than training all the parameters together.

F: Filter Visualization

In this section, we try to answer why FSKD works so well that it can provide almost the same results as that of fine-tuning with full training set. We conduct experiments based on VGG-13 on CIFAR-10. For a given VGG-13 network, We first decouple a conv-layer to obtain one DW conv-layer and one PW conv-layer, as is done in network decoupling [1]. Then we visualize the PW conv-layer of the decoupled layer. For simplicity, we only visualize the PW conv-layer of the first decoupled layer. We do the visualization on three VGG-13 network with different parameters:

(1) Initialize the VGG-13 network with the MSRA initialization (Figure 3 left).

(2) Run SGD based fine-tuning on 500 samples for VGG-13 with random initialization until convergence (Figure 3 middle).

(3) Run FSKD on 500 samples for VGG-13 with SGD based initialization (Figure 3 right). The teacher network is also a VGG-13 trained on full CIFAR-10 training set.

It clearly shows that the PW conv-layer before fine-tuning is fairly random on the value range, the one after fine-tuning is less random, while the one after FSKD further starts to show some regular patterns, which demonstrates that FSKD can distill the knowledge from the teacher-net to student-net effectively with few samples.

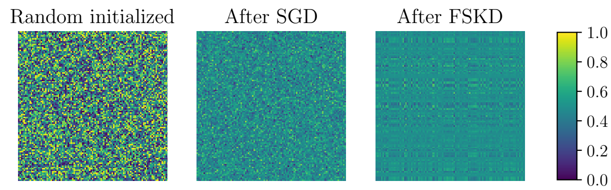


Figure 3: Decouple VGG-13 into DW conv-layer and PW conv-layers, and show one PW conv-layer with random initialization (left), after SGD based fine-tuning (middle), and after FSKD (right). Note values of the PW tensor are scaled into the range (0,1.0) by the min/max values of the tensor for better visualization.

References

[1] Jianbo Guo, Yuxi Li, Weiyao Lin, Yurong Chen, and Jianguo Li. Network decoupling: From regular to depthwise separable convolutions. In *BMVC*, 2018. 1, 3

[2] Song Han, Huizi Mao, Bill Dally, et al. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *NIPS*, 2016. 2

[3] Mingyi Hong, Xiangfeng Wang, Meisam Razaviyayn, and Zhi-Quan Luo. Iteration complexity analysis of block coordinate descent methods. *Mathematical Programming*, 163(1-2), 2017. 1

[4] Hao Li, Asim Kadav, Durdanovic I, et al. Pruning filters for efficient convnets. *ICLR*, 2017. 2

[5] Zhuang Liu, Jianguo Li, Zhiqiang Shen, et al. Learning efficient convolutional networks through network slimming. In *ICCV*, 2017. 2