# Be Your Own Teacher: Improve the Performance of Convolutional Neural Networks via Self Distillation

Linfeng Zhang[1] Jiebo Song[3] Anni Gao[3] Jingwei Chen[4] Chenglong Bao[2*] Kaisheng Ma[1*]
[1]Institute for Interdisciplinary Information Sciences, Tsinghua University
[2]Yau Mathematical Sciences Center, Tsinghua University
[3]Institute for Interdisciplinary Information Core Technology
[4]HiSilicon
{zhang-lf19, kaisheng, clbao}@mail.tsinghua.edu.cn
{songjb, gaoan}@iiisct.com, jean.chenjingwei@hisilicon.com

## Abstract

*Convolutional neural networks have been widely deployed in various application scenarios. In order to extend the applications' boundaries to some accuracy-crucial domains, researchers have been investigating approaches to boost accuracy through either deeper or wider network structures, which brings with them the exponential increment of the computational and storage cost, delaying the responding time.*

*In this paper, we propose a general training framework named self distillation, which notably enhances the performance (accuracy) of convolutional neural networks through shrinking the size of the network rather than aggrandizing it. Different from traditional knowledge distillation - a knowledge transformation methodology among networks, which forces student neural networks to approximate the softmax layer outputs of pre-trained teacher neural networks, the proposed self distillation framework distills knowledge within network itself. The networks are firstly divided into several sections. Then the knowledge in the deeper portion of the networks is squeezed into the shallow ones. Experiments further prove the generalization of the proposed self distillation framework: enhancement of accuracy at average level is 2.65%, varying from 0.61% in ResNeXt as minimum to 4.07% in VGG19 as maximum. In addition, it can also provide flexibility of depth-wise scalable inference on resource-limited edge devices. Our codes have been released on github[5].*

## 1. Introduction

With the help of convolutional neural networks, applications such as image classification [22, 34] ,object detection [28], and semantic segmentation [7, 40] are developing at an unprecedented speed nowadays. Yet, in some applications demanding intolerate errors such as automated driving and medical image analysis, prediction and analysis accuracy needs to be further improved, while at the same time, shorter response time is required. This leads to tremendous challenges on current convolutional neural networks. Traditional methods were focused on either performance improvement or reduction of computational resources (thus response time). On the one hand, for instance, ResNet 150 or even larger ResNet 1000 have been proposed to improve very limited performance margin but with massive computational penalty. On the other hand, with a pre-defined performance lost compared with best effort networks, various techniques have been proposed to reduce the computation and storage amount to match the limitations brought by hardware implementation. Such techniques include lightweight networks design [19, 16], pruning [12, 13] and quantization [5, 31]. Knowledge Distillation (KD) [15] was one of the available approaches, or even regarded as a trick, to achieve model compression.

As one of the popular compression approaches, knowledge distillation [15] is inspired by knowledge transfer from teachers to students. Its key strategy is to orientate compact student models to approximate over-parameterized teacher models. As a result, student models can gain significant performance boost which is sometimes even better than that of teacher's. By replacing the over-parameterized teacher model with a compact student model, high compression and rapid acceleration can be achieved. However, glories come with remaining problems. The first setback is low efficiency on knowledge transfer, which means student mod-

[5] https://github.com/ArchipLab-LinfengZhang/pytorch-self-distillation.

[6] This paper is supported by Institute for Interdisciplinary Information Core Technology, Beijing Academy of Artificial Intelligence and Zhongguancun Haihua Institute for Frontier Information Technology.

* Corresponding Authors.

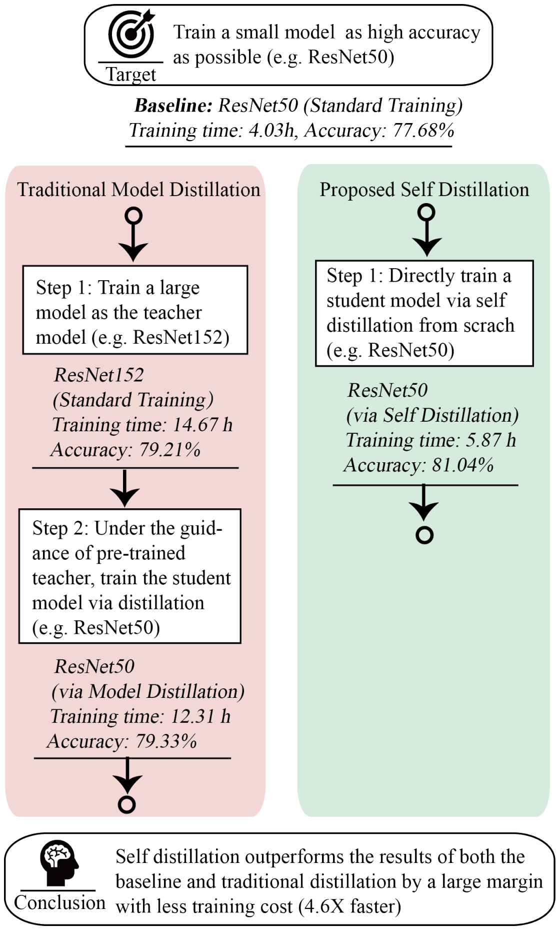

Target
Train a small model as high accuracy as possible (e.g. ResNet50)
Baseline: ResNet50 (Standard Training)
Training time: 4.03h, Accuracy: 77.68%
Traditional Model Distillation
Step 1: Train a large model as the teacher model (e.g. ResNet152)
ResNet152
(Standard Training)
Training time: 14.67 h
Accuracy: 79.21%
Step 2: Under the guidance of pre-trained teacher, train the student model via distillation (e.g. ResNet50)
ResNet50
(via Model Distillation)
Training time: 12.31 h
Accuracy: 79.33%
Proposed Self Distillation
Step 1: Directly train a student model via self distillation from scrach (e.g. ResNet50)
ResNet50
(via Self Distillation)
Training time: 5.87 h
Accuracy: 81.04%
Conclusion
Self distillation outperforms the results of both the baseline and traditional distillation by a large margin with less training cost (4.6X faster)

Figure 1. Comparison of training complexity, training time, and accuracy between traditional distillation and proposed self distillation (reported on CIFAR100).

els scarcely exploit all knowledge from teacher models. A distinguished student model which outperforms its teacher model remains rare. Another barrier is how to design and train proper teacher models. The existing distillation frameworks require substantial efforts and experiments to find the best architecture of teacher models, which takes a relatively long time.

As shown in Figure 1, in order to train a compact model to achieve as high accuracy as possible and to overcome the drawbacks of traditional distillation, we propose a novel self distillation framework. Instead of implementing two steps in traditional distillation, that is first, to train a large teacher model, and second, to distill the knowledge from it to the student model, we propose a one-step self distillation framework whose training points directly at the student model. The proposed self distillation not only requires less training time (from 26.98 hours to 5.87 hours on CIFAR100, a 4.6X time training shorten time), but also can accomplish much higher accuracy (from 79.33% in traditional distilllaitn to 81.04% on ResNet50).

In summary, we make the following principle contributions in this paper:

- Self distillation improves the performance of convolutional neural networks by a large margin at no expense of response time. 2.65% accuracy boost is obtained on average, varying from 0.61% in ResNeXt as minimum to 4.07% in VGG19 as maximum.

- Self distillation provides a single neural network executable at different depth, permitting adaptive accuracy-efficiency trade-offs on resource-limited edge devices.

- Experiments for five kinds of convolutional neural networks on two kinds of datasets are conducted to prove the generalization of this technique.

The rest of this paper is organized as follows. Section 2 introduces the related work of self distillation. Section 3 demonstrates the formulation and detail of self distillation. Section 4 shows the experiments results on five kinds of convolutional networks and two kinds of datasets. Section 5 explains the reason why self distillation works. Finally, a conclusion is brought forth in section 6.

## 2. Related Work

**Knowledge distillation:** knowledge distillation is one of the most popular techniques used in model compression [4, 15]. A large quantity of approaches have been proposed to reinforce the efficiency of student models' learning capability. Romero *et al.* firstly put forward FitNet in which the concept of hint learning was proposed, aiming at reducing the distance between feature maps of students and teachers [32]. Agoruyko *et al.* [42] considered this issue from the perspective of attention mechanism, attempting to align the features of attention regions. Furthermore, some researchers extended knowledge distillation to generative adversarial problem [33, 27].

In other domains, knowledge distillation also shows its potential. Furlanello *et al.* interactively absorbed the distillated student models into the teacher model group, through which the better generalization ability on test data is obtained [9]. Bagherinezhad *et al.* applied knowledge distillation to data argumentation, increasing the numerical value of labels to a higher entropy [2]. Papernot *et al.* regarded knowledge distillation as a tool to defend adversarial attack [30], and Gupta *et al.*, using the same methods, transferred the knowledge among data in different modals [10].

As shown above, in general, teacher models and student models work in their own ways respectively, and knowledge transfer flows among different models. In contrast, student

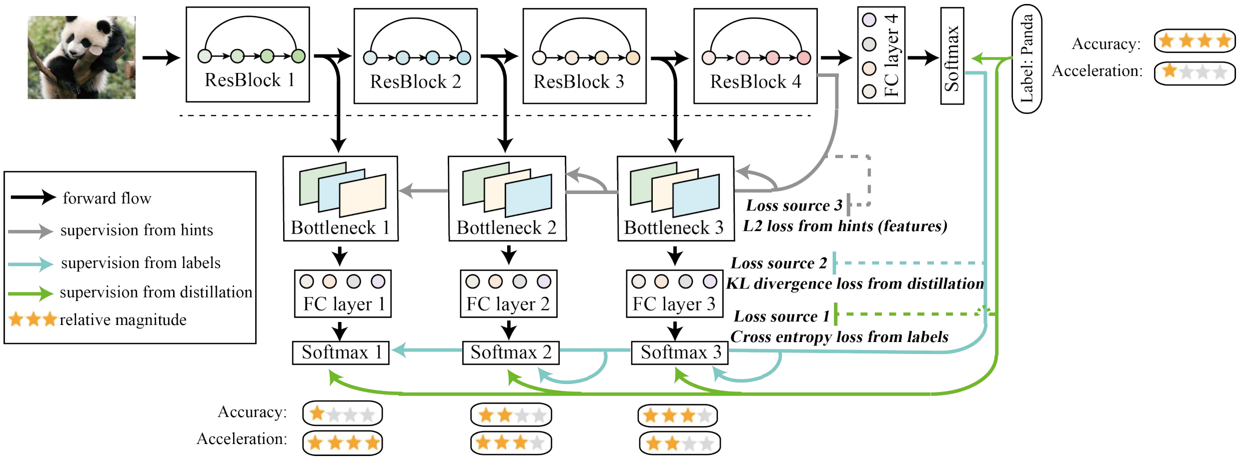


Figure 2. This figure shows the details of a ResNet equipped with proposed self distillation. (i) A ResNet has been divided into four sections according to their depth. (ii) Additional bottleneck and fully connected layers are set after each section, which constitutes multiple classifiers. (iii) All of the classifiers can be utilized independently, with different accuracy and response time. (iv) Each classifier is trained under three kinds of supervision as depicted. (v) Parts under the dash line can be removed in inference.

and teacher models in our proposed self distillation method come from the same convolutional neural networks.

**Adaptive Computation:** Some researchers incline to selectively skip several computation procedures to remove redundancy. Their work can be witnessed from three different angles: layers, channels and images.

*Skipping some layers in neural networks.* Huang *et al.* proposed random layer-wise dropout in training [18]. Some researchers extended this idea to inference. Wang *et al.* and Wu *et al.* further extended the layer-wise dropout from training to inference by introducing additional controller modules or gating functions based on the current input [37, 36]. Another extension of the layer-wise dropout solution is to design early-exiting prediction branches to reduce the average execution depth in inference [17, 1, 35, 23].

*Skipping some channels in neural networks.* Yu *et al.* proposed switchable batch normalization to dynamically adjust the channels in inference [39].

*Skipping less important pixels of the current input images.* Inspired by the intuition that neural networks should focus on critical details of input data [3], reinforcement learning and deep learning algorithms are utilized to identify the importance of pixels in the input images before they are feed into convolutional neural networks [29, 8].

**Deep Supervision:** Deep supervision is based on the observation that classifiers trained on highly discriminating features can improve the performance in inference [24]. In order to address the vanishing gradient problem, additional supervision is added to train the hidden layers directly. For instance, significant performance gain has been observed in tasks like image classification [24], objection detection [26, 25, 28], and medical images segmentation [40, 7].

The multi-classifier architecture adopted in the proposed self distillation framework is similar to deeply supervised net [24]. The main difference in self distillation is that shallow classifiers are trained via distillation instead of only labels, which leads to an obvious higher accuracy supported by experiments results.

## 3. Self Distillation

In this section, we put forward self distillation techniques as depicted in Figure 2. We construct the self distillation framework in the following ways of thinking: To begin with, the target convolutional neural network is divided into several shallow sections according to its depth and original structure. For example, ResNet50 is divided into 4 sections according to ResBlocks. Secondly, a classifier, combined with a bottleneck [14] layer and a fully connected layer which are only utilized in training and can be removed in inference, is set after each shallow section. The main consideration of adding the bottleneck layer is to mitigate the impacts between each shallow classifier, and to add L2 loss from hints. While in training period, all the shallow sections with corresponding classifiers are trained as student models via distillation from the deepest section, which can be conceptually regarded as the teacher model.

In order to improve the performance of the student models, three kinds of losses are introduced during training processes:

- Loss Source 1: Cross entropy loss from labels to not only the deepest classifier, but also all the shallow classifiers. It is computed with the labels from the training dataset and the outputs of each classifer's softmax layer. In this way, the knowledge hidden in the dataset is introduced directly from labels to all the classifiers.

| Neural Networks | Baseline | Classifier 1/4 | Classifier 2/4 | Classifier3/4 | Classifier 4/4 | Ensemble |
|---|---|---|---|---|---|---|
| VGG19(BN) | 64.47 | 63.59 | 67.04 | 68.03 | 67.73 | 68.54 |
| ResNet18 | 77.09 | 67.85 | 74.57 | 78.23 | 78.64 | 79.67 |
| ResNet50 | 77.68 | 68.23 | 74.21 | 75.23 | 80.56 | 81.04 |
| ResNet101 | 77.98 | 69.45 | 77.29 | 81.17 | 81.23 | 82.03 |
| ResNet152 | 79.21 | 68.84 | 78.72 | 81.43 | 81.61 | 82.29 |
| ResNeXt29-8 | 81.29 | 71.15 | 79.00 | 81.48 | 81.51 | 81.90 |
| WideResNet20-8 | 79.76 | 68.85 | 78.15 | 80.98 | 80.92 | 81.38 |
| WideResNet44-8 | 79.93 | 72.54 | 81.15 | 81.96 | 82.09 | 82.61 |
| WideResNet28-12 | 80.07 | 71.21 | 80.86 | 81.58 | 81.59 | 82.09 |
| PyramidNet101-240 | 81.12 | 69.23 | 78.15 | 80.98 | 82.30 | 83.51 |

Table 1. Experiments results of accuracy (%) on CIFAR100 (the number marked in red is lower than its baseline).

| Neural Networks | Baseline | Classifier 1/4 | Classifier 2/4 | Classifier 3/4 | Classifier 4/4 | Ensemble |
|---|---|---|---|---|---|---|
| VGG19(BN) | 70.35 | 42.53 | 55.85 | 71.07 | 72.45 | 73.03 |
| ResNet18 | 68.12 | 41.26 | 51.94 | 62.29 | 69.84 | 68.93 |
| ResNet50 | 73.56 | 43.95 | 58.47 | 72.84 | 75.24 | 74.73 |

Table 2. Experiments results of top-1 accuracy (%) on ImageNet (the number marked in red is lower than its baseline).

- Loss Source 2: KL (Kullback-Leibler) divergence loss under teacher's guidance. The KL divergence is computed using softmax outputs between students and teachers, and introduced to the softmax layer of each shallow classifier. By introducing KL divergence, the self distillation framework affects the teacher's networks, the deepest one, to each shallow classifier.

- Loss Source 3: L2 loss from hints. It can be obtained through computation of the L2 loss between features maps of the deepest classifier and each shallow classifier. By means of L2 loss, the inexplicit knowledge in feature maps is introduced to each shallow classifier's bottleneck layer, which induces all the classifiers' feature maps in their bottleneck layers to fit the feature maps of the deepest classifier.

For that all the newly added layers (parts under the dash line in Figure 2) are only applied during training, they exert no influence during inference. Adding these parts during inference provides another option for dynamic inference for energy constrained edge devices.

### 3.1. Formulation

Given $N$ samples $X = \{x_i\}_{i=1}^{N}$ from $M$ classes, we denote the corresponding label set as $Y = \{y_i\}_{i=1}^{M}$, $y_i \in \{1, 2, ..., M\}$. Classifiers (the proposed self distillation has multiple classifiers within a whole network) in the neural network are denoted as $\Theta = \{\theta_{i/C}\}_{i=1}^{C}$, where $C$ denotes the number of classifiers in convolutional neural networks. A softmax layer is set after each classifier.

$$q_i^c = \frac{\exp\left(z_i^c/T\right)}{\Sigma_j^c \exp\left(z_j^c/T\right)} \tag{1}$$

Here $z$ is the output after fully connected layers. $q_i^c \in \mathbb{R}^{\mathbb{M}}$ is the $i_{th}$ class probability of classifier $\theta_{c/C}$. $T$, which is normally set to 1, indicates the temperature of distillation [15]. A larger T makes the probability distribution softer.

### 3.2. Training Methods

In self distillation, the supervision of each classifier $\theta_{i/C}$ except for the deepest classifier comes from three sources. Two hyper-parameters $\alpha$ and $\lambda$ are used to balance them.

$$(1-\alpha) \cdot CrossEntropy(q^i, y) \tag{2}$$

The first source is the cross entropy loss computed with $q^i$ and labels $Y$. Note that $q^i$ denotes the softmax layer's output of classifier $\theta_{i/C}$ .

$$\alpha \cdot KL(q^i, q^C) \tag{3}$$

The second source is the Kullback-Leibler divergence between $q^i$ and $q^C$. We aim to make shallow classifiers approximate the deepest classifier, which indicates the supervision from distillation. Note that $q^C$ means the softmax layer's output of the deepest classifier.

$$\lambda \cdot \|F_i - F_C\|_2^2 \tag{4}$$

The last supervision is from the hint of the deepest classifier. A hint is defined as the output of teacher models hidden layers, whose aim is to guide the student models'

| Teacher Model | Student Model | Baseline | KD [15] | FitNet [32] | AT [42] | DML [43] | Our approach |
|---|---|---|---|---|---|---|---|
| ResNet152 | ResNet18 | 77.09 | 77.79 | 78.21 | 78.54 | 77.54 | 78.64 |
| ResNet152 | ResNet50 | 77.68 | 79.33 | 80.13 | 79.35 | 78.31 | 80.56 |
| WideResNet44-8 | WideResNet20-8 | 79.76 | 79.80 | 80.48 | 80.65 | 79.91 | 80.92 |
| WideResNet44-8 | WideResNet28-12 | 80.07 | 80.95 | 80.53 | 81.46 | 80.43 | 81.58 |

Table 3. Accuracy (%) comparison with traditional distillation on CIFAR100.

| Neural Networks | Method | Classifier 1/4 | Classifier 2/4 | Classifier3/4 | Classifier 4/4 | Ensemble |
|---|---|---|---|---|---|---|
| ResNet18 | DSN | 67.23 | 73.80 | 77.75 | 78.38 | 79.27 |
| | Our approach | 67.85 | 74.57 | 78.23 | 78.64 | 79.67 |
| ResNet50 | DSN | 67.87 | 73.80 | 74.54 | 80.27 | 80.67 |
| | Our approach | 68.23 | 74.21 | 75.23 | 80.56 | 81.04 |
| ResNet101 | DSN | 68.17 | 75.43 | 80.98 | 81.01 | 81.72 |
| | Our approach | 69.45 | 77.29 | 81.17 | 81.23 | 82.03 |
| ResNet152 | DSN | 67.60 | 77.04 | 81.06 | 81.35 | 81.83 |
| | Our approach | 68.84 | 78.72 | 81.43 | 81.61 | 82.29 |

Table 4. Accuracy (%) comparison with deeply supervised net [24] on CIFAR100.

learning [32]. It works by decreasing the distance between feature maps in shallow classifiers and in the deepest classifier. However, because the feature maps in different depth have different sizes, extra layers should be added to align them. Instead of using a convolutional layer [32], we use a bottleneck architecture which shows positive effects on model's performance. Note that $F_i$ and $F_C$ denote features in the classifier $\theta_i$ and features in the deepest classifier $\theta_C$ respectively.

To sum up, the loss function of the whole neural networks consists of the loss function of each classifier, which can be written as:

$$\begin{aligned} loss &= \sum_i^C loss_i \\ &= \sum_i^C \Big( (1-\alpha) \cdot CrossEntropy(q^i, y) \\ &+ \alpha \cdot KL(q^i, q^C) + \lambda \cdot ||F_i - F_C||_2^2 \Big) \end{aligned} \tag{5}$$

Note that $\lambda$ and $\alpha$ for the deepest classifier are zero, which means the deepest classifier's supervision just comes from labels.

## 4. Experiments

We evaluate self distillation on five convolutional neural networks (ResNet [14], WideResNet [41], Pyramid ResNet [11], ResNeXt [38], VGG [34]) and two datasets (CIFAR100 [21], ImageNet [6]). Learning rate decay, $l_2$ regularizer and simple data argumentation are used during the training process. All the experiments are implemented by PyTorch on GPU devices.

### 4.1. Benchmark Datasets

**CIFAR100:** CIFAR100 dataset [21] consists of tiny (32x32 pixels) RGB images. It has 100 classes and contains 50K images in training set and 10K images in testing set. Kernel sizes and strides of neural networks are adjusted to fit the size of tiny images.

**ImageNet:** ImageNet2012 classification dataset [6] is composed of 1000 classes according to WordNet. Each class is depicted by thousands of images. We resize them into 256x256 pixels RGB images. Note that reported accuracy of ImageNet is computed on the validation set.

### 4.2. Compared with Standard Training

Results of experiments on CIFAR100 and ImageNet are displayed in Table 1 and Table 2 respectively. An ensemble result is obtained by simply adding the weighted outputs of the softmax layer in each classifier. It is observed that (i) all the neural networks benefit significantly from self distillation, with an increment of 2.65% in CIFAR100 and 2.02% in ImageNet on average. (ii) The deeper the neural networks are, the more improvement on performance they acquire, for example, an increment of 4.05% in ResNet101 and 2.58% in ResNet18. (iii) Generally speaking, naive ensemble works effectively on CIFAR100 yet with less and sometimes negative influence on ImageNet, which may be caused by the larger accuracy drop in shallow classifiers, compared with that on CIFAR100. (iv) Classifiers' depth plays a more crucial part in ImageNet, indicating there is less redundancy in neural networks for a complex task.

### 4.3. Compared with Distillation

Table 3 compares results of self distillation with that of five traditional distillation methods on CIFAR100 dataset.

| Neural Networks | Attribute | Baseline | Classifier 1/4 | Classifier 2/4 | Classifier3/4 | Classifier 4/4 | Ensemble |
|---|---|---|---|---|---|---|---|
| ResNet18 | Accuracy | 77.09 | 67.23 | 73.80 | 77.75 | 78.38 | 79.27 |
| | Acceleration | 1.00X | 3.11X | 1.87X | 1.30X | 1.00X | 0.93X |
| ResNet50 | Accuracy | 77.68 | 67.87 | 73.80 | 74.54 | 80.27 | 80.67 |
| | Acceleration | 1.00X | 4.64X | 2.20X | 1.23X | 1.00X | 0.93X |
| ResNet101 | Accuracy | 77.98 | 68.17 | 75.43 | 80.98 | 81.01 | 81.72 |
| | Acceleration | 1.00X | 9.00X | 4.27X | 1.11X | 1.00X | 0.96X |
| ResNet152 | Accuracy | 79.21 | 68.84 | 78.22 | 81.43 | 81.61 | 82.29 |
| | Acceleration | 1.00X | 13.36X | 4.29X | 1.07X | 1.00X | 0.98X |

Table 5. Acceleration and accuracy (%) for ResNet on CIFAR100.

Here we focus on the accuracy boost of each method when the student models have the same computation and storage amount. From Table 3, we make the following observations: (i) All the performance of distillation methods outperforms the directly trained student networks. (ii) Although self distillation doesn't have an extra teacher, it still outperforms most of the rest distillation methods.

One significant advantage of self distillation framework is that it doesn't need an extra teacher. In contrast, traditional distillation needs to design and train an over-parameterized teacher model at first. Designing a high quality teacher model needs tremendous experiments to find the best depth and architecture. In addition, training an over-parameterized teacher model takes much longer time. These problems can be directly avoided in self distillation, where both teachers and students models are sub-sections of itself. As depicted in Figure 1, 4.6X acceleration in training time can be achieved by self distillation compared with other distillation methods.

### 4.4. Compared with Deeply Supervised Net

The main difference between deeply supervised net and self distillation is that self distillation trains shallow classifiers from the deepest classifier's distillation instead of labels. The advantages can be seen in experiments, as shown in Table 4, which compares the accuracy of each classifier in ResNet trained by deep supervision or self distillation on CIFAR100. The observations can be summarized as follows: (i) Self distillation outperforms deep supervision in every classifier. (ii) Shallow classifiers benefit more from self distillation.

The reasons for the phenomena are easy to understand. In self distillation, (i) extra bottleneck layers are added to detect classifier-specific features, avoiding conflicts between shallow and deep classifiers. (ii) Distillation method has been employed in training the shallow classifiers instead of labels to boost the performance. (iii) Better shallow classifiers can obtain more discriminating features, which enhances the deeper classifiers performance in return.

### 4.5. Scalable Depth for Adapting Inference

Recently, a popular solution to accelerate convolutional neural networks is to design a scalable network, which means the depth or width of neural networks can change dynamically according to application requirements. For example, in the scenarios where response time is more important than accuracy, some layers or channels could be abandoned at runtime for acceleration [39].

With a sharing backbone network, adaptive accuracy-acceleration tradeoff in inference becomes possible on resource-limited edge devices, which means that different depth classifiers can be automatically employed in applications according to dynamic accuracy demands in real word. As can be observed in Table 5 that (i) three in four neural networks outperform their baselines by classifier 3/4, with an acceleration ratio of 1.2X on average. 3.16X acceleration ratio can be achieved with an accuracy loss at 3.3% with classifier 2/4. (ii) Ensemble of the deepest three classifiers can bring 0.67% accuracy improvement on average level with only 0.05% penalty for computation, owing to that different classifiers share one backbone network.

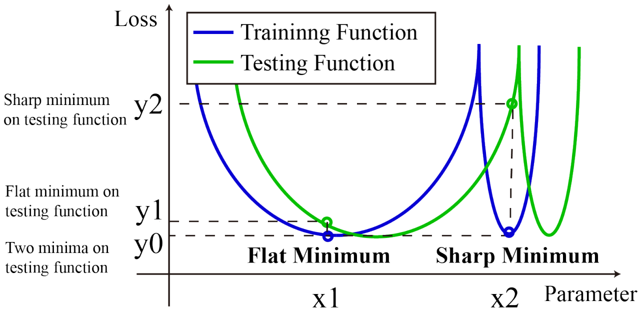
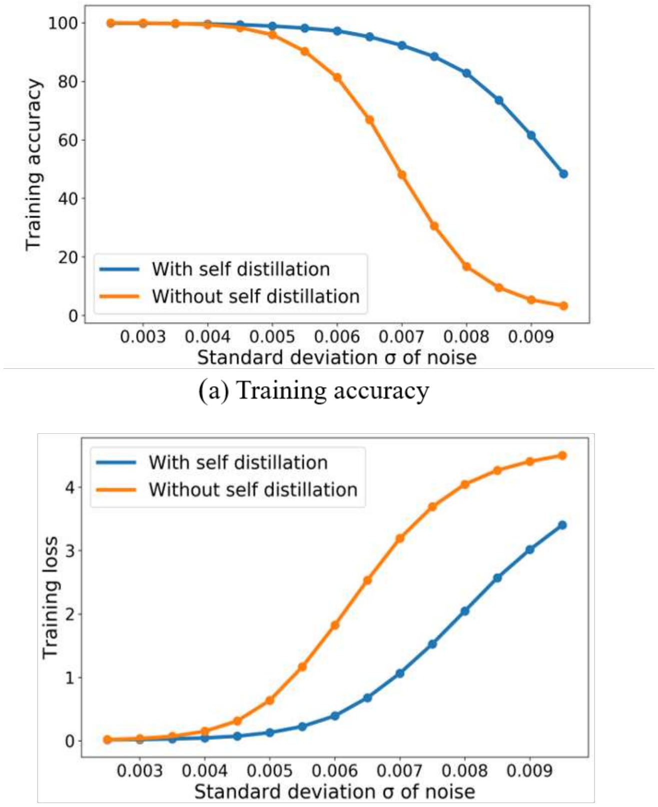

Loss
Traininng Function
Testing Function
Sharp minimum on testing function
y2
Flat minimum on testing function
y1
Two minima on testing function
y0
Flat Minimum
Sharp Minimum
x1
x2
Parameter

Figure 3. An intuitive explanation of the difference between flat and sharp minima [20].

## 5. Discussion and Future Works

In this section, we discuss the possible explanations of notable performance improvement brought by self distillation from perspectives of flat minima, vanishing gradients, and discriminating features, which will be followed by the section of future works for further improvement.



(a) Training accuracy

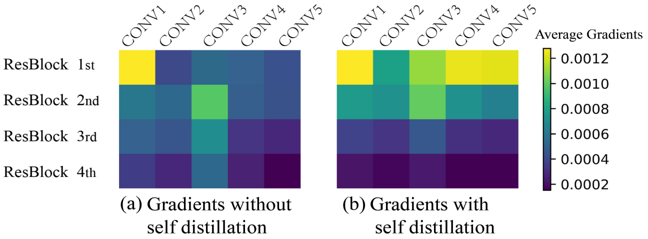
(b) Training loss

Figure 4. Comparison of training accuracy and loss with increasing Gaussian noise: models trained with self distillation are more tolerant to noise - flat minima.

**Self distillation can help models converge to flat minima which features in generalization inherently.** It is universally acknowledged that although shallow neural networks (e.g. AlexNet) can also achieve almost zero loss on the training set, their performance on test set or in practical applications is far behind over-parameterized neural networks (e.g. ResNet) [20]. Keskar *et al.* proposed explanations that over-parameters models may converge easier to the flat minima, while shallow neural networks are more likely to be caught in the sharp minima, which is sensitive to the bias of data [20]. Figure 3 gives an intuitive explanation of the difference between flat and sharp minima. The X axis represents the parameters of models in one dimension. The Y axis is the value of loss function. The two curves denote the loss curves on training set and test set. Both two minima (x1 for flat mimima and x2 for sharp minima) can achieve extremely small loss on the training set (y0). Unfortunately, the training set and the test set are not independently and identically distributed. While in the test, x1 and x2 are still utilized to find the minima y1 and y2 in the testing curve, which causes severe bias in the sharp mimina curve (y2 - y0 is much larger than y1 - y0).

Inspired by the work of Zhang *et al.* [43], we conduct the following experiments to show that the proposed self distillation framework can converge to a flat minimun. Two 18-layer ResNets have been trained on CIFAR100 dataset firstly, one with self distillation and the other one not. Then Gaussian noise is added to the parameters of the two models and then their entropy loss and predicted accuracy on the training set are obtained and plotted in Figure 4. As can be seen in Figure 4(a), the training set accuracy in the model trained with self distillation maintains at a very high level with noise level, presented as standard deviation of the Gaussian noises, keeping increasing, while the training accuracy in the model without self distillation drops severely, as shown in Figure 4(a). Same observations and conclusions can be obtained in Figure 4(b) with training loss as the metric. Based on the above observations, we conclude that the models trained with self distillation are more flat. According to the conclusion sourced from Figure 3, the model trained with self distillation is more robust to perturbation of parameters. Note that the 4/4 classifier is used in self distillation ResNet for a fair comparison. To sum up, the model trained without self distillation is much more sensitive to the Gaussian noise. These experiments results support our view that self distillation helps models find flat minima, permitting better generalization performance.

**Self distillation prevents models from vanishing gradient problem.** Due to vanishing gradient problem, very deep neural networks are hard to train, although they show better generalization performance. In self distillation, the supervision on the neural networks is injected into different depth. It inherits the ability of DSN [24] to address the vanishing gradient problem to some extent. Since the work of Lee *et al.* [24] has given the justification mathematically, we conduct the following experiments to support it.

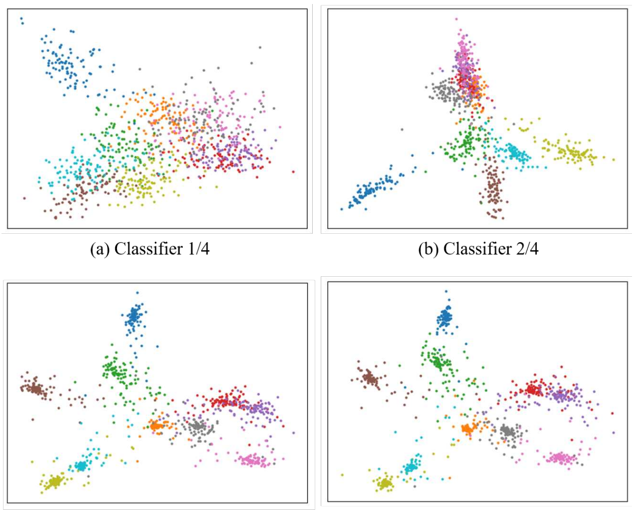


(a) Gradients without self distillation

(b) Gradients with self distillation

Figure 5. Statistics of layer-wised gradients.

Two 18-layer ResNets are trained, one of them equipped with self distillation and the other one not. We compute the mean magnitude of gradients in each convolutional layer as shown in Figure 5. It is observed that the magnitude of gradients of the model with self distillation (Figure 5(a)) is larger than the one without self distillation (Figure 5(b)), especially in the first and second ResBlocks.



(a) Classifier 1/4

(b) Classifier 2/4

(c) Classifier 3/4

(d) Classifier 4/4

Figure 6. PCA (principal component analysis) visualization of feature distribution in four classifiers.

**More discriminating features are extracted with deeper classifiers in self distillation.** Since there are multiple classifiers existing in self distillation, features of each classifier can be computed and analyzed to demonstrate their discriminating principle. As depicted in Figure 6, experiments on WideResNet trained on CIFAR100 are conducted to compare features of different classifiers.

Figure 6 visualizes the distances of features in different classifiers. To begin with, it is obvious that the deeper the classifier is, the more concentrated clusters are observed. In addition, the changes of the distances in shallow classifiers, as shown in Figure 6(a,b), are more severe than that in deep classifiers, as demonstrated in Figure 6(c,d).

| Classifier | SSE* | SSB** | SSE/SSB | Accuracy |
|---|---|---|---|---|
| Classifier1/4 | 20.85 | 1.08 | 19.21 | 71.21 |
| Classifier2/4 | 8.69 | 1.15 | 7.54 | 80.86 |
| Classifier3/4 | 11.42 | 1.87 | 6.08 | 81.58 |
| Classifier4/4 | 11.74 | 2.05 | 5.73 | 81.59 |

*SSE: Sum of squares due to error.

**SSB: Sum of squares between groups.

Table 6. Measurement of sort separability and accuracy (%) for each classifier on WideResNet28-12.

Table 6 further summarizes the sort separability for each classifier. SSE stands for sum of squares due to error, and SSB is short for sum of squares between groups. The smaller the SSE is, the denser the clusters are. Also, the clusters become more discriminating with the SSB growing. Here we use SSE/SSB to evaluate the distinct capability of the models. The smaller it is, the more clear the classifier is. It can be seen in Table 6 that the SSE/SSB decreases as classifier goes deeper. In summary, the more discriminating feature maps in the classifier, the higher accuracy the model achieves.

**Future Works** *Automatic adjustment of newly introduced hyper-parameters.* To balance the loss of cross entropy, KL divergence, and hint loss, two hyper-parameters $\lambda$ and $\alpha$ are introduced as shown in Equation 5. Through the experiments, we find out that these two hyper-parameters have impacts on the performance. Due to limited computation resources, we have not done a through investigation. In the near future, automatic adjustment of the two hyper-parameters can be explored using learning rate decay like or momenta inspired algorithms.

*Is the flat minimum found by self distillation ideal?* Another unexplored domain is that we find a phenomenon during training that after the convergence of self distillation, continuing training of the deepest classifiers using conventional training method can further boost the performance from 0.3% to 0.7%, which are not included in all the Tables in the paper. Despite that shallow classifiers help find the flat minimum, at the final stage of the training, they also prevent the deepest classifier from convergence. Alternately switching between multiple training methods might further help the convergence.

## 6. Conclusion

We have proposed a novel training technique called self distillation and shown its advantage by comparing it with deeply supervised net and the previous distillation methods. This technique abandons the extra teacher model required in previous distillation methods and provides an adaptive depth architecture for time-accuracy tradeoffs at runtime. We also have explored the principle behind self distillation from the perspective of flat minima, gradients and discriminating feature.

Self distillation is more of a training technique to boost model performance rather than a method to compress or accelerate models. Although most of the previous research focuses on knowledge transfer among different models, we believe that knowledge transfer approaches inside one model like self distillation are also very promising.

## References

[1] Manuel Amthor, Erik Rodner, and Joachim Denzler. Impatient dnns-deep neural networks with dynamic time budgets. In *British Machine Vision Conference*, 2016.

[2] Hessam Bagherinezhad, Maxwell Horton, Mohammad Rastegari, and Ali Farhadi. Label refinery: Improving imagenet classification through label progression. In *European Conference on Computer Vision*, 2018.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2015.

[4] Cristian Bucilu, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541. ACM, 2006.

[5] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in neural information processing systems*, pages 3123–3131, 2015.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition*, pages 248–255. Ieee, 2009.

[7] Qi Dou, Hao Chen, Yueming Jin, Lequan Yu, Jing Qin, and Pheng-Ann Heng. 3d deeply supervised network for automatic liver segmentation from ct volumes. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 149–157. Springer, 2016.

[8] Jianlong Fu, Heliang Zheng, and Tao Mei. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, page 3, 2017.

[9] Tommaso Furlanello, Zachary C Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. In *International Machine Learning Conference*, 2018.

[10] Saurabh Gupta, Judy Hoffman, and Jitendra Malik. Cross modal distillation for supervision transfer. In *Computer Vision and Pattern Recognition*, pages 2827–2836, 2016.

[11] Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 6307–6315. IEEE, 2017.

[12] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. 2016.

[13] 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.

[14] 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.

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *Advances in neural information processing systems*, 2014.

[16] 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. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, 2017.

[17] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Q Weinberger. Multi-scale dense networks for resource efficient image classification. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2017.

[18] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016.

[19] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. In *International Conference on Learning Representations*, 2016.

[20] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017.

[21] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[23] Jason Kuen, Xiangfei Kong, Zhe Lin, Gang Wang, Jianxiong Yin, Simon See, and Yap-Peng Tan. Stochastic downsampling for cost-adjustable inference and improved regularization in convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7929–7938, 2018.

[24] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial Intelligence and Statistics*, pages 562–570, 2015.

[25] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 4, 2017.

[26] Tsung-Yi Lin, Priyal Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[27] Peiye Liu, Wu Liu, Huadong Ma, Tao Mei, and Mingoo Seok. Ktan: Knowledge transfer adversarial network. In *Association for the Advance of Artificial Intelligence*, 2019.

[28] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[29] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in neural information processing systems*, pages 2204–2212, 2014.

[30] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE*

*Symposium on Security and Privacy*, pages 582–597. IEEE, 2016.

[31] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, pages 525–542. Springer, 2016.

[32] 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.

[33] Zhiqiang Shen, Zhankui He, and Xiangyang Xue. Meal: Multi-model ensemble via adversarial learning. In *Association for the Advance of Artificial Intelligence*, 2019.

[34] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. 2015.

[35] Andreas Veit and Serge Belongie. Convolutional networks with adaptive inference graphs. In *European conference on computer vision*, 2018.

[36] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 409–424, 2018.

[37] Zuxuan Wu, Tushar Nagarajan, Abhishek Kumar, Steven Rennie, Larry S Davis, Kristen Grauman, and Rogerio Feris. Blockdrop: Dynamic inference paths in residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8817–8826, 2018.

[38] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 5987–5995. IEEE, 2017.

[39] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable neural networks. In *International Conference on Learning Representations*, 2019.

[40] Lequan Yu, Xin Yang, Hao Chen, Jing Qin, and Pheng-Ann Heng. Volumetric convnets with mixed residual connections for automated prostate segmentation from 3d mr images. In *Association for the Advance of Artificial Intelligence*, pages 66–72, 2017.

[41] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *British Machine Vision Conference*, 2016.

[42] 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.

[43] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328, 2018.