Self-Guidance: Improve Deep Neural Network Generalization via Knowledge Distillation

Zhenzhu Zheng
University of Delaware
zzzheng@udel.edu

Xi Peng
University of Delaware
xipeng@udel.edu

Abstract

We present Self-Guidance, a simple way to train deep neural networks via knowledge distillation. The basic idea is to train sub-network to match the prediction of the full network, so-called "Self-Guidance". Under the "teacher-student" framework, we construct both teacher and student within the same target network. Student network is the sub-networks that randomly skip some portions of the full network. The teacher network is the full network, can be considered as the ensemble of all possible student networks. The training process is performed in a closed-loop: (1) Forward prediction contains two passes that generate student and teacher predictions. (2) Backward distillation allows knowledge transfer from the teacher back to students. Comprehensive evaluations show that our approach improves the generalization ability of deep neural networks to a significant margin. The results prove our superior performance in both image classification on CIFAR10, CIFAR100, and facial expression recognition on FER-2013 and RAF.

1. Introduction

Deep neural networks have achieved great success in computer vision tasks such as image classification [27, 59, 56], object detection [44, 39, 4]. But deep neural networks are often over-parameterized, which makes it not suitable for deployment, and easily suffering from over-fitting. To address this issue, one popular paradigm is Knowledge Distillation (KD), aiming at training *small* and *generalizable* models. The general idea is to transfer knowledge from a teacher (large) model to a student (small) model, where the student is trained to match the output of the teacher [21, 40, 58]. However, classical knowledge distillation [21] relies on a pre-trained teacher, which might not always be available in practice. To solve this problem, *online* [41, 63, 65, 5] and *self*-distillation [13, 54, 62, 57] methods are proposed using different strategies. *Self*-distillation approaches [13, 54, 62, 57] typically take advantage of the model generations during the training trajectory [13, 54] or the intermediate flow within the network [62]. However, many approaches on this line come with a complex workflow or architecture design.

Online distillation [63, 65, 5], on the other hand, intends to build a strong teacher role by a group of (student) peers, which are typically constructed via a multi-branch architecture. However, the multi-branch architecture design has drawbacks: First, the number of branches (students) would be limited subject to the available storage. This is a *storage-heavy* consumption approach for training. Second, and more importantly, due to the limited number of branches, the model would not have sufficient power to cover a large degree of uncertainty/variety in the solution space.

We aim for a storage-efficient training scheme while maintaining competitive performance. To create student models without sourcing a multi-branch architecture, we propose to generate student (peers) within the same network, which allow us to achieve a sufficient amount of student diversity, while without introducing any extra model parameters.

The basic idea is simple: Let the teacher be the full network, while the students be the sampled sub-networks. Both the teacher and students share weights since they are inside the same network. The teacher can be considered as the *implicit* ensemble of all students. The analogy is that the students and teacher are *Oneness*, where students are the smaller individual and together form a more powerful larger collection. Individual (student) absorbs knowledge from the collection (teacher), and the teacher grows out from students.

The student network is sampled by randomly skipping some portions of the full network during the forward pass. In this case, there can be exponentially many student networks to be generated. By exploiting the dynamic architecture within the network, a certain degree of diversity can be achieved. This is different from approaches based on multi-branch [65, 5], where student diversity is limited to the static branching structure. To gain better performance, however, they require extra components such as gating or attention.

The whole training process is performed in a closed-loop: *forward prediction* and *backward distillation*. The **forward prediction** contains two passes: (1) one pass goes through the full network to generate the teacher prediction; (2) another pass goes through a *randomly sampled* sub-network to generate the student prediction. The **backward distillation** aims to transfer knowledge from the teacher to all students, which is the teacher itself. The whole process can be considered as seamlessly incorporating distillation as a regularization into the training procedure. An overview is shown in Fig. 1.

2. Related Works

2.1. Knowledge Distillation

Knowledge Distillation (KD) originated from [3], popularized by [21], now become a hot research topic [16, 48] applied in many areas [51, 52, 64, 12]. The key problem is how to transfer the knowledge from a large teacher model to a small student model. It contains two major components: **knowledge** and **distillation scheme**.

Knowledge. Depending on what information that the student model try to mimic from the teacher model, KD methods can be broadly categorized into three categories [16]:

(1) **Response-based** knowledge refers to the final prediction of the teacher model. It is simple yet effective, and has been widely used in different tasks [7, 61, 31] and applications [43, 22]. The most popular form is also known as soft target [21, 1], which can be considered as label smoothing or regularization [25, 32, 11]. Our approach belongs to this category. (2) **Feature-based** knowledge is an extension of the response-based, which considered both the output of the last layer and the output of intermediate layers [40, 58, 24, 20, 37, 8, 49, 10, 19]. (3) **Relation-based** knowledge further explores the relationships between different layers [55, 60, 28, 35, 9, 30, 6] or data samples [30, 33, 34, 36, 46, 38].

Distillation Schemes. The distillation schemes can be directly divided into three main categories: **offline distillation**, **online distillation** and **self-distillation**.

While **offline** distillation requires a pre-trained teacher model, online and self-distillations aims to fulfill the absence of the teacher role from different aspects. Typically, **self**-distillation approaches take advantage of generation in the training trajectory [13, 54], the information flow within the network [62] or class information [57]. However, many approaches on this line come with a complex workflow or architecture design.

Online distillation [41, 63, 65, 5, 17, 50] allows both the teacher and student(s) study together from each other. The basic idea is to simultaneously training a group of student models by learning from peers predictions as an effective substitute for the static pre-trained Teacher. However, there are drawbacks. First, online ensemble KD simply aggregate students logits to form an ensemble teacher restrains the diversity of student peers, thus limiting the effectiveness of online learning learning. Second, existing approaches adopt a multi-branch architecture leading to storage-heavy consumption and also not flexible for ensemble in a more versatile or dynamic way. Our approach falls into this category. Different from traditional online distillation methods, we intends to generate **diversity** within the network instead of any auxiliary branches, leading to a *storage-efficient* solution.

Concurrent with our work, Mean Teacher [45] also construct the teacher model without extra parameters by using the average model weights of the training epochs. The difference is that [45] focus on the semi-supervised learning, while ours belongs to supervised learning and explore on the architecture aspect.

2.2. Structure Regularization

Structure regularization is one category of regularization methods which imposes constraints on the network weights and structure to reduce over-fitting. Dropout [42] randomly drops some connections during training to prevent units from co-adapting. Many following works share the idea of Dropout by randomly dropping different portion of network, such as DropConnect [47], StochDepth [23], Shake-Shake [14] and ShakeDrop [53]. For example, StochDepth [23] randomly drops a subset of layers during training. The final network can be viewed as an ensemble of many shallow networks.

2.3. Implicit Ensemble

An alternative to traditional ensembles, so-called “implicit” ensembles have high efficiency during both training and testing. From the *architecture* perspective, Dropout [42], DropConnect [47] and Stochastic Depth [23] can be considered as sampling sub-networks at different levels. Dropout [42] creates an ensemble out of a single model by “dropping” random sets of hidden nodes during each mini-batch. DropConnect [47] and Stochastic Depth [23] can be considered as specific cases of Dropout operating on the edge and layer level, respectively. In this work, we take advantage of implicit ensemble to generate student networks. This is different from one-shot architecture search [2] where the sub-network weights are dynamically generated, which requires a more complicated process.

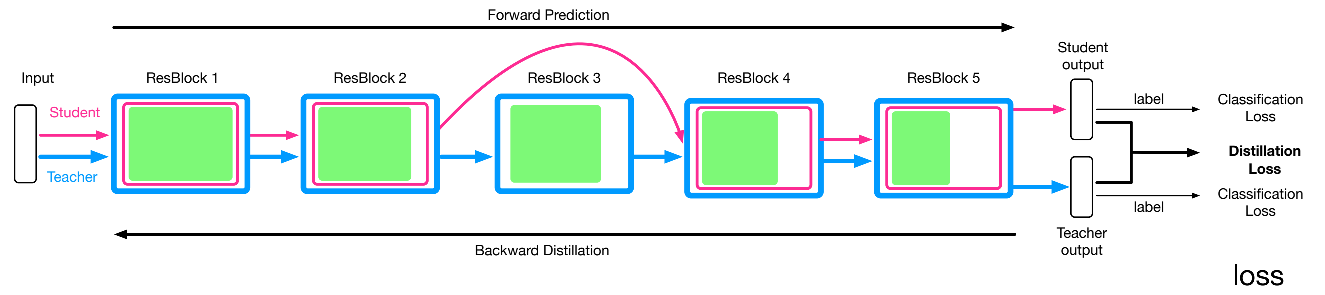


Figure 1: Overview. Teacher: full network. Student: sampled sub-network for each mini-batch. Other components are not shown for a simple illustration. Red: Student forward pass. Blue: Teacher forward pass. Green regions indicate the survival probability for the Residual Block (ResBlock), the larger the higher sample ("survival") probability is. Best view in color.

3. Method

3.1. Preliminary on Knowledge Distillation

Knowledge distillation [3, 21] is an effective technique to transfer knowledge from a strong teacher network to a target student network. The training task can be generalized as the following formulation:

$$\hat{\theta_S} = \arg\min_{\theta_S} \sum_{i}^{N} \mathcal{L}_{\text{ce}}(x_i, \theta_S, y_i) + \mathcal{L}_{\text{distill}}(x_i, \theta_S, \theta_T), \quad (1)$$

where x_i is an image, y_i is the ground-truth label, θ_S is the student network parameter, and θ_T is the teacher network parameter. The loss $\mathcal{L}_{\text{ce}}$ is the matching error between the network prediction and the ground-truth label. For classification, it is normally a cross-entropy loss. The loss of $\mathcal{L}_{\text{distill}}$ is the mimic error of the student towards the teacher. For example in [21], the teacher signal comes from the softmax prediction, and the loss is measured by the *Kullback-Leibler* divergence.

3.2. Preliminary on Structure Regularization

Structure regularization is one category of regularization methods that imposes constraints on the network weights and structure to reduce over-fitting. Dropout [42] randomly drops some connections during training to prevent units from co-adapting. Many following works share the idea of Dropout by randomly dropping network layers or branches.

While sub-network generation strategy is not the focus of our work, in this paper, we adopt StochDepth [23] which randomly dropping network layers as a means to generate sub-network as student.

In particular, during training StochDepth [23] sample sub-networks via randomly dropping entire ResBlocks and bypassing their transformations through skip connections. In Residual networks [18], a ResBlock output is as follows:

$$H_l = \text{ReLU}(f_l(H_{l-1}) + H_{l-1}) \quad (2)$$

where H_l denotes the output of the l^{th} layer, $f_l(\cdot)$ represents a typical convolutional transformation from layer $l-1$ to l. Here we assume a ReLU activation function.

Let $b_l \in \{0, 1\}$ denote a Bernoulli random variable indicating whether the l^{th} ResBlock is active ($b_l = 1$) or not ($b_l = 0$). The survival probability of the l^{th} ResBlock is denoted as $p_l = Pr(b_l = 1)$.

Based on Eq. 2, the update rule of ResBlock for sub-networks is re-formulated as

$$H_l^{\text{Train}} = \text{ReLU}(b_l f_l(H_{l-1}^{\text{Train}}) + H_{l-1}^{\text{Train}}) \quad (3)$$

That being said, if $b_l = 1$, Eq. 3 functions as a ResBlock; if $b_l = 0$, the ResBlock reduces to a skip connection.

During testing, all blocks are activated, the update rule becomes the combination of all possible networks with each layer weighted by its survival probability:

$$H_l^{\text{Test}} = \text{ReLU}(p_l f_l(H_{l-1}^{\text{Test}}) + H_{l-1}^{\text{Test}}) \quad (4)$$

3.3. Self-Guidance

The basic idea is to train sub-network to match the prediction of the full network, so-called "Self-Guidance". Under the "teacher-student" framework, we construct both teacher and student within the same target network. Student network is the sub-networks that randomly skip some portions of the full network. The teacher network is the full network, can be considered as the ensemble of all possible student networks.

Student-Teacher Predictions. To generate student and teacher prediction, we borrow the idea of Dropout and its variants. Specifically, we consider the full network (teacher) as the implicit ensemble of all possible sub-networks (students). In particular, we take advantage of the training (Eq. 3) and testing (Eq. 4) update rules to generate student and teacher predictions in the same network, respectively.

Thus, our approach contains two forward passes: (1) one pass goes through a *randomly sampled* sub-network to generate student prediction (Eq. 3); (2) another pass goes through the full network to generate student prediction (Eq. 4);

Distillation. We denote by T and S the teacher and student networks respectively. Let p_{Net}, $Net = \{T, S\}$ be the prediction output and z^i_{Net} be the K class logits where $i = 1, ..., K$, which is followed by the softmax function $p_{Net} = s(z^i_{Net}) = \frac{exp(z^i_{Net}/\tau)}{\sum_j exp(z^j_{Net}/\tau)}$ with temperature τ ($\tau = 1$ for Cross Entropy loss). The distillation loss L_{distil} is the *Kullback-Leibler* divergence which measure the discrepancy between the student and teacher prediction using soft label ($\tau > 1$).

Overall Loss Function. Our training paradigm is formulated as:

$$\hat{\theta_T} = \arg\min_{\theta_T} \sum_i^N \mathcal{L}^S_{\text{ce}}(x_i, \theta_S, y_i) \quad + \\ \mathcal{L}^T_{\text{ce}}(x_i, \theta_T, y_i) + \lambda * \tau^2 * \mathcal{L}_{\text{distill}}(x_i, \theta_S, \theta_T), \qquad (5)$$

where $\mathcal{L}_{ce}$ is cross-entropy loss, $\mathcal{L}_{distill} = KL(p^S, p^T)$ with $\tau > 1$ indicating soft label, and λ is the trade-off parameter to scale the losses. Following [65], we multiply the distillation loss term by a factor τ^2 to ensure that the relative contributions of ground-truth and teacher prediction remain roughly unchanged.

Training and Deployment. Thus, without sourcing to a multi-branch architecture, our method is simple: we use two forward passes to generate S and T predictions, respectively, update the full-network based on Eq. 5. Note that this leads to storage-efficient approach. Once the model is trained, we can simply use the teacher prediction for deployment. And only one forward pass is needed as the model testing normally does. Summary is in Algorithm 1.

4. Experiments

4.1. Image Classification

Datasets. We used two multi-class categorization benchmark datasets in our evaluations. (1) CIFAR10 [26]: A natural images dataset that contains 50,000/10,000 training/test samples drawn from 10 object classes (in total 60,000 images). Each class has 6,000 images sized at 32×32 pixels. (2) CIFAR100 [26]: A similar dataset as CIFAR10 that also contains 50,000/10,000 training/test images but covering 100 fine-grained classes. Each class has 600 images.

Algorithm 1: Self-Guidance

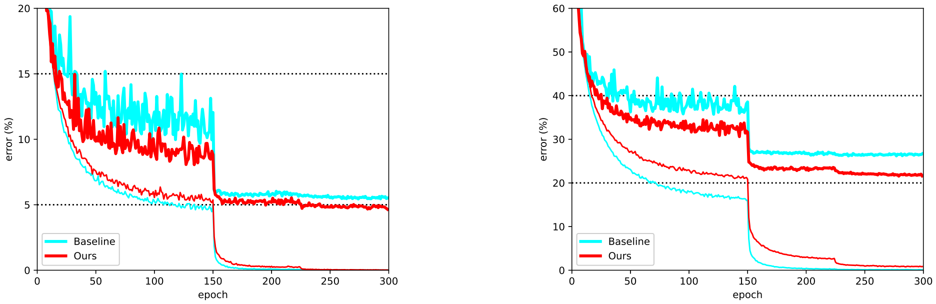
```
Input: Labelled training data $(x, y)$; Training epoch number $M$;
Output: Trained model $\theta_T$ (teacher);
   /* Training                                  */
Initialization: $i = 1$; Randomly initialize $\theta_T$;
   Assign survival probability $p_l$ to each Residual Block.
while $i \leq M$ do
    for each mini-batch do
        Randomly sample a sub-network (student) $\theta_S$;
        Compute student prediction. Eq. 3;
        Compute teacher prediction. Eq. 4;
        Compute soft labels of $S$ and $T$;
        Update $\theta_T$ by SGD algorithm. Eq. 5.
    end
end
/* Testing                                       */
Deployment: Use $\theta_T$ and teacher prediction. Eq. 4.
```

Method	CIFAR10	CIFAR100	Params
ResNet-32 [18]	6.34	30.14	0.47M
ResNet-32 + **Ours**	**5.73** (+0.61)	**27.65** (+2.49)	0.47M
ResNet-50 [18]	6.07	28.37	0.76M
ResNet-50 + **Ours**	**5.28** (+0.79)	**25.33** (+3.04)	0.76M
ResNet-110 [18]	5.43	26.18	1.15M
ResNet-110 + **Ours**	**4.65** (+0.78)	**21.60** (+4.59)	1.15M

Table 1: Image Classification Results on CIFAR10/100. Metric: Top-1 error rate (%).

Setup. For all datasets, we adopted the same experimental settings as for making fair comparisons [23, 65]. We used the SGD with Nesterov momentum and set the momentum to 0.9 with weight decay 1e-4. Batch size is 128, training epoch is 300. We deployed a standard learning rate schedule that drops from 0.1 to 0.01 at 50 % training and to 0.001 at 75%. Following [21], we set τ = 3 in all the experiments. Cross-validation of hyper-parameters[1] may give better performance but at the cost of extra model tuning. Trade-off parameter λ is set to be 0.25. We adopted the common top-1 classification error rate.

[1] For student network generation, we follow the same hyper-parameter setting as [23].



(a) CIFAR10

(b) CIFAR100

Figure 2: Training and testing error for ResNet-110 on CIFAR10 and CIFAR100, repectively. Thin lines indicate training error, **bold** lines indicate **test** error.

Method	CIFAR10	CIFAR100	Params
baseline [18]	6.34	30.14	0.47M
DML [63]	6.22	28.47	1.4M
ONE [65]	5.89	**27.30**	1.18M
Ours	**5.73**	27.65	**0.47M**

Table 2: Comparison with online distillation methods. Image classification error rates (Top-1, %) on CIFAR10 and CIFAR100. Target Network: ResNet-32 [18]. **Bold**: best result. Underline: second best.

Results. Table 1 compares top-1 error rate performances of varying-capacity state-of-the-art network models trained by the standard procedure and our approach on CIFAR10/100. We observe that all different target networks benefit from our training algorithm achieving significant performance gains. This suggests a general superiority of our approach for online knowledge distillation even without introducing extra parameters to the target model.

Table 2 shows the comparison with state-of-the-art online distillation methods on ResNet-32. Our approach provides competitive performance while maintaining storage-efficient. On CIFAR10, ours achieves a 5.73% error rate, which surpasses other online distillation methods. On CIFAR100, we maintain a 27.65% error rate being similar to ONE as 27.30% while using only about 40% model parameters of its amount.

Fig. 2a and 2b show the training and test errors for ResNet-110 on CIFAR10/100 respectively. We have two observations: (1) With our approach, the gap between training and test error is reduced significantly. This suggests that KL divergence as the distillation loss function as regularization effectively. It demonstrates that our approach as a training algorithm can prevent overfitting effectively. (2) For standard training, almost no progress is made after 225 epochs. With our approach, both training and test error continue to reduce. This suggests that our approach facilitates the ease of optimization and can achieve better local minimums.

Method	FER-2013	RAF	Params
ResNet-32 [18]	32.85	18.55	0.47M
ResNet-32 + **Ours**	**31.33**	**17.80**	0.47M
	(+1.52)	(+0.75)	
ResNet-50 [18]	31.83	17.67	0.76M
ResNet-50 + **Ours**	**30.38**	**16.43**	0.76M
	(+1.45)	(+1.24)	
ResNet-110 [18]	32.35	19.62	1.15M
ResNet-110 + **Ours**	**30.48**	**16.53**	1.15M
	(+1.87)	(+3.09)	

Table 3: Facial Expression Recognition Results. error rates (Top-1, %) on FER-2013 and RAF.

4.2. Facial Expression Recognition

Datasets. We used two benchmarks facial expression datasets with 7 human facial expressions. (1) FER-2013 [15]: It consists of 28,709 gray-scale images for training and 3,589 for testing. (2) RAF [29] is a real-world facial expression recognition dataset, which contains 12,271 RGB images for training and 3,068 for testing. Here we use the basic 7 expression categories.

Setup. Similar to CIFAR, we resize images to 32x32. For FER-2013, image channels are duplicated to make them RGB images. All training settings and hyperparameters are the same as the image classification task. (See Sec. 4.1.)

Results. (1) **Improve generalization ability.** Tab. 3 shows the classification results on FER-2013 and RAF on

Method	FER-2013	RAF	Params
baseline [18]	32.19	18.55	0.47M
DML [63]	31.91	18.48	1.4M
ONE [65]	31.90	17.85	1.18M
OKDDip [5]	31.40	**17.42**	1.53M
Ours	**31.33**	17.80	**0.47M**

Table 4: Comparison with online distillation methods. Facial Expression Recognition error rates (Top-1, %) on FER-2013 and RAF. Target Network: ResNet-32 [18]. **Bold**: best result. Underline: second best.

ResNets as the target networks with a variety of depths. It shows that our approach consistently improves the performances to a significant margin on depth 32, 50, and 110. We observe that the peak performance comes from depth 50, and degrade when depth increases to 110. With our approach, we are able to train ResNet-110 with 3.09% improvement, indicating a strong ability to prevent overfitting. (2) **Storage-efficiency.** Tab. 4 show the comparison with state-of-the-art online distillation methods. It shows that our approach reaches on-par or even better performances without introducing extra parameters, while other methods typically use 2.5 to 3.25 times parameters, due to the multi-branch architecture. This indicates our approach is storage efficiency while maintaining competitive performance.

4.3. Ablation Study

Effect of trade-off parameter λ. We try different values for the trade-off parameter λ as 0, 0.1, 0.25, 0.5, 1 and 3 using ResNet-50 on CIFAR10, and ResNet-110 on CIFAR100 respectively. Results are summarized in Table 5. Training and test errors are shown in Fig. 3 and 4. Due to the highly similar behaviors (overlapped curves) between $\lambda = 0.1$ and $\lambda = 0.25$, we do not plot $\lambda = 0.1$ for better visual clarity.

Even with different architectures and datasets, we find that the trade-off parameter $\lambda = 0.25$ achieves the best results in both scenarios. This suggests the balance between the classification and distillation loss can be generic and can easily generalize across architecture or datasets. We observe that λ between 0.1 and 0.4 should work well generally. This suggests our hyper-parameter λ is not sensitive, and works well in a decent range of values.

Model Component Analysis. Table 6 shows the benefits of individual components of our approach on CIFAR100 using ResNet-110 as target network. We have these observations: (1) **W/O Online Distillation** by setting trade-off $\lambda = 0$, our approach can be considered as Stochastic Depth [23] but with a small difference: for each backpropagation, Stochastic Depth update the weights of the sub-network, while our approach updates weights of the full network. It shows a 1.77 % performance drop from the full method. (2) **W/O Backward full network** yields a degraded performance with a large deviation. This indicates knowledge transfer to all students is important. This is because all student networks are shared weights, updating the full network leads to a stronger student in the next forward pass. An alternative explanation is that the teacher can be considered as all students together. Updating weights for the full network will lead to a stronger teacher in the next iteration. This suggests our approach achieves the efficacy of knowledge transfer between the teacher and student in an online manner.

Feature Visualizations. Fig. 5 shows the t-SNE visualization of student/teacher features on the penultimate layer during the training procedure, using ResNet-50 as target network train on CIFAR100. Specifically, student A and B can be considered as having the same capacity, both have 20 ResBlocks being active (skipped 4 ResBlocks).

We have the following observations:

(1) More concentrated clusters are observed in the feature distributions from the teacher, indicating the teacher network generates more *discriminating* features than the students. This matches our expectation that the implicit ensemble (teacher) outperforms significantly than its single constitute component (student), ensuring the students can distill knowledge from the teacher which is much stronger.

(2) Student networks have large *variations*. All three student networks generate diverse feature distributions throughout the training process. The same observation can be found between Student A and B, even they have the same capacity. This indicates a large amount of *diversity* among the students, which is an important foundation to build a strong ensemble.

(3) Features become more *separable* as the training proceeds. This observation holds for both student and teacher networks, which suggests that training proceeds effectively.

5. Conclusions

We proposed Self-Guidance, a simple training scheme that improves deep neural network generalization. It conducts ensembles that pertain to a large degree of variety in an on-the-fly manner without introducing extra model parameters. It naturally integrates the properties of structure regularization and knowledge distillation, which leads to a generic training strategy with both higher performance and memory efficiency.

References

[1] Lei Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *NeurIPS*, 2014.

Method (depth=50)	CIFAR10	Method (depth=110)	CIFAR100
ResNet-50 [18]	6.06	ResNet-110 [18]	25.33
ResNet-50+SD [23]	6.05	ResNet-110+SD [23]	22.61
ResNet-50+Ours λ=0	5.31	ResNet-110+Ours λ=0	22.63
ResNet-50+Ours λ=0.1	5.29	ResNet-110+Ours λ=0.1	21.25
ResNet-50+Ours λ=0.25	**5.28**	ResNet-110+Ours λ=0.25	**21.08**
ResNet-50+Ours λ=0.5	5.62	ResNet-110+Ours λ=0.5	21.97
ResNet-50+Ours λ=1	5.78	ResNet-110+Ours λ=1	22.40
ResNet-50+Ours λ=3	90.00 (loss blow up)	ResNet-110+Ours λ=3	23.51

Table 5: Effect of trade-off parameter λ. Evaluations on CIFAR10/100 using ResNet-50 and ResNet-110 as target network, respectively. Metric: Top-1 Error rate (%). **Bold**: the best result, Underline: second best result.

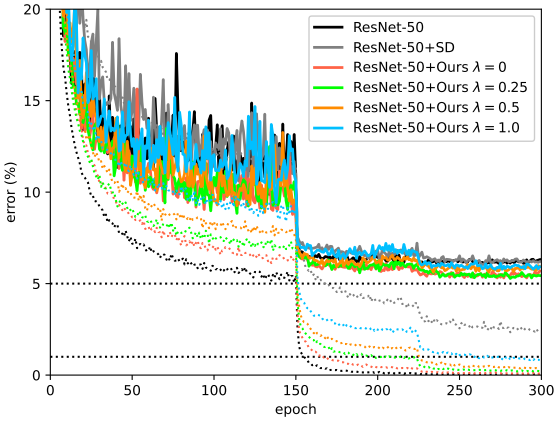


Figure 3: Training and Test error for our approach with various values for the trade-off parameter. Dataset: CIFAR10; Target network: ResNet-50. Dash line indicates training errors, **solid** lines indicate test errors. *Trade-off $\lambda = 3$ is not shown due to loss blow up.*

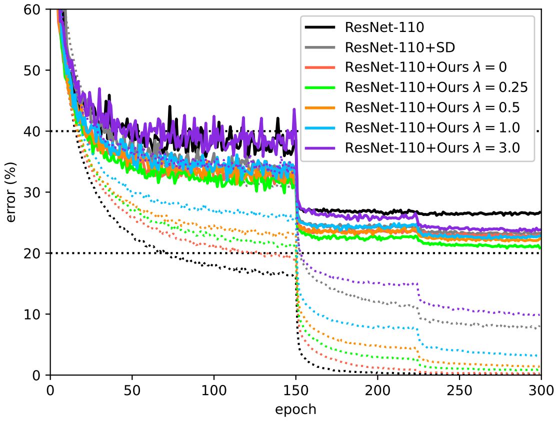


Figure 4: Training and Test error for our approach with various values for the trade-off parameter. Dataset: CIFAR100; Target network: ResNet-110. Dash line indicates training errors, **solid** lines indicate test errors.

Configuration	Error (%)
Baseline [18]	25.33
Stochastic Depth [23]	22.61
W/O Online Distillation ($\lambda = 0$)	22.63
W/O Backward Full network	23.15
Full	**21.60**

Table 6: Model component analysis of ResNet-110 as target network on CIFAR100.

[2] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Smash: one-shot model architecture search through hypernetworks. *arXiv preprint arXiv:1708.05344*, 2017.

[3] Cristian Bucilua, 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, 2006.

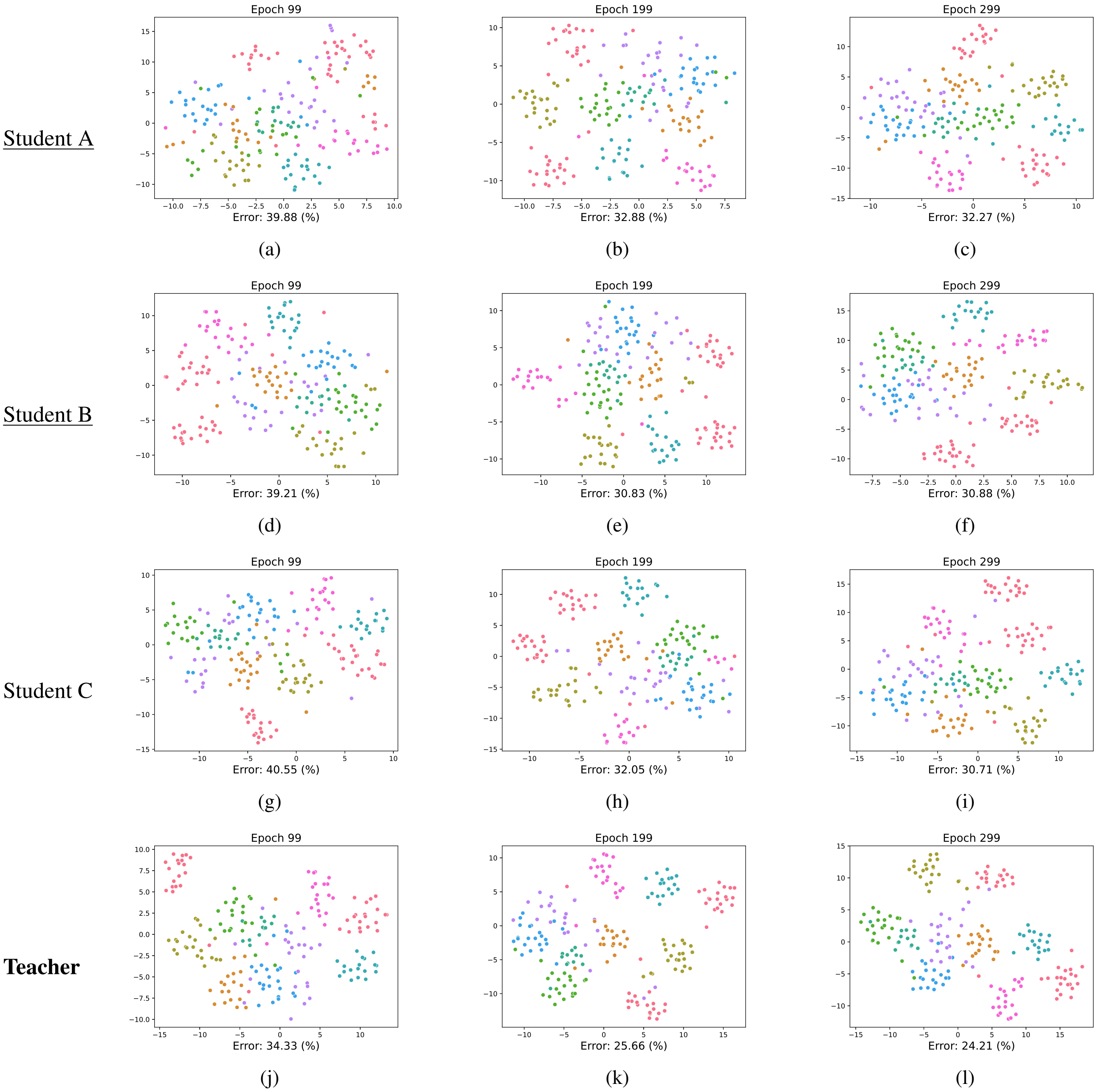
[4] Qi Cai, Yingwei Pan, Yu Wang, Jingen Liu, Ting Yao, and Tao Mei. Learning a unified sample weighting network for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14173–14182, 2020.

[5] Defang Chen, Jian-Ping Mei, Can Wang, Yan Feng, and Chun Chen. Online knowledge distillation with diverse peers. In *AAAI*, 2020.

[6] Defang Chen, Jian-Ping Mei, Yuan Zhang, Can Wang, Zhe Wang, Yan Feng, and Chun Chen. Cross-layer distillation with semantic calibration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7028–7036, 2021.

[7] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. In *Advances in neural information processing systems*, 2017.

[8] Hanting Chen, Yunhe Wang, Chang Xu, Chao Xu, and Dacheng Tao. Learning student networks via feature embed-



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Figure 5: t-SNE visualization of student/teacher features during the training process. Student A: 111111111110101010111111; Student B: 111111111110111110111001; Student C: 111111111001111111011010. '1' indicates the ResBlock at the corresponding layer is active, and '0' indicates inactive. Student A and B can be considered as having the same capacity, both have 20 ResBlocks being active (skipped 4 ResBlocks).

ding. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):25–35, 2020.

[9] Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Darkrank: Accelerating deep metric learning via cross sample similarities transfer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[10] Inseop Chung, SeongUk Park, Jangho Kim, and Nojun Kwak. Feature-map-level online adversarial knowledge distillation. In *International Conference on Machine Learning*, pages 2006–2015. PMLR, 2020.

[11] Qianggang Ding, Sifan Wu, Hao Sun, Jiadong Guo, and Shu-Tao Xia. Adaptive regularization of labels. *arXiv preprint*

arXiv:1908.05474, 2019.

[12] Zhiyuan Fang, Jianfeng Wang, Lijuan Wang, Lei Zhang, Yezhou Yang, and Zicheng Liu. Seed: Self-supervised distillation for visual representation. In *ICLR*, 2021.

[13] Tommaso Furlanello, Zachary C Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. In *PMLR*, 2018.

[14] Xavier Gastaldi. Shake-shake regularization. *arXiv preprint arXiv:1705.07485*, 2017.

[15] Ian J Goodfellow, Dumitru Erhan, Pierre Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, et al. Challenges in representation learning: A report on three machine learning contests. In *International conference on neural information processing*, pages 117–124. Springer, 2013.

[16] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

[17] Qiushan Guo, Xinjiang Wang, Yichao Wu, Zhipeng Yu, Ding Liang, Xiaolin Hu, and Ping Luo. Online knowledge distillation via collaborative learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11020–11029, 2020.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[19] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1921–1930, 2019.

[20] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3779–3787, 2019.

[21] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NIPS Workshop*, 2015.

[22] Hanzhe Hu, Shuai Bai, Aoxue Li, Jinshi Cui, and Liwei Wang. Dense relation distillation with context-aware aggregation for few-shot object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10185–10194, 2021.

[23] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, 2016.

[24] Jangho Kim, SeongUk Park, and Nojun Kwak. Paraphrasing complex network: Network compression via factor transfer. *arXiv preprint arXiv:1802.04977*, 2018.

[25] Seung Wook Kim and Hyo-Eun Kim. Transferring knowledge to smaller network with class-distance loss. In *ICLRW*, 2017.

[26] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[27] 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.

[28] Seunghyun Lee and Byung Cheol Song. Graph-based knowledge distillation by multi-head attention network. In *BMVC*, 2019.

[29] Shan Li, Weihong Deng, and JunPing Du. Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 2584–2593. IEEE, 2017.

[30] Yufan Liu, Jiajiong Cao, Bing Li, Chunfeng Yuan, Weiming Hu, Yangxi Li, and Yunqiang Duan. Knowledge distillation via instance relationship graph. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7096–7104, 2019.

[31] Zhong Meng, Jinyu Li, Yong Zhao, and Yifan Gong. Conditional teacher-student learning. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6445–6449. IEEE, 2019.

[32] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When does label smoothing help? In *NeurIPS*, 2019.

[33] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *CVPR*, pages 3967–3976, 2019.

[34] Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *ECCV*, pages 268–284, 2018.

[35] Nikolaos Passalis, Maria Tzelepi, and Anastasios Tefas. Heterogeneous knowledge distillation using information flow modeling. In *CVPR*, pages 2339–2348, 2020.

[36] Nikolaos Passalis, Maria Tzelepi, and Anastasios Tefas. Probabilistic knowledge transfer for lightweight deep representation learning. *IEEE Transactions on Neural Networks and Learning Systems*, 32(5):2030–2039, 2020.

[37] Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. Alp-kd: Attention-based layer projection for knowledge distillation. *arXiv preprint arXiv:2012.14022*, 2020.

[38] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5007–5016, 2019.

[39] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[40] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.

[41] Guocong Song and Wei Chai. Collaborative learning for deep neural networks. In *NeurIPS*, 2018.

[42] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *PMLR*, 2014.

[43] Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*, 2020.

[44] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep neural networks for object detection. 2013.

[45] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, pages 1195–1204, 2017.

[46] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1365–1374, 2019.

[47] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. Regularization of neural networks using dropconnect. In *ICML*, 2013.

[48] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[49] Xiaobo Wang, Tianyu Fu, Shengcai Liao, Shuo Wang, Zhen Lei, and Tao Mei. Exclusivity-consistency regularized knowledge distillation for face recognition. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16*, pages 325–342. Springer, 2020.

[50] Guile Wu and Shaogang Gong. Peer collaborative learning for online knowledge distillation. In *AAAI*, 2021.

[51] Yibo Hu Xiang Wu, Ran He and Zhenan Sun. Learning an evolutionary embedding via massive knowledge distillation. *International journal of Computer Vision*, 2020.

[52] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge distillation meets self-supervision. In *ECCV*, pages 588–604. Springer, 2020.

[53] Yoshihiro Yamada, Masakazu Iwamura, Takuya Akiba, and Koichi Kise. Shakedrop regularization for deep residual learning. *IEEE Access*, 7:186126–186136, 2019.

[54] Chenglin Yang, Lingxi Xie, Chi Su, and Alan L Yuille. Snapshot distillation: Teacher-student optimization in one generation. In *CVPR*, 2019.

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

[56] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[57] Sukmin Yun, Jongjin Park, Kimin Lee, and Jinwoo Shin. Regularizing class-wise predictions via self-knowledge distillation. In *CVPR*, 2020.

[58] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017.

[59] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[60] Chenrui Zhang and Yuxin Peng. Better and faster: knowledge transfer from multiple self-supervised learning tasks via graph distillation for video classification. *arXiv preprint arXiv:1804.10069*, 2018.

[61] Feng Zhang, Xiatian Zhu, and Mao Ye. Fast human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3517–3526, 2019.

[62] Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *CVPR*, 2019.

[63] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *CVPR*, 2018.

[64] Long Zhao, Xi Peng, Yuxiao Chen, Mubbasir Kapadia, and Dimitris N Metaxas. Knowledge as priors: Cross-modal knowledge generalization for datasets without superior knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6528–6537, 2020.

[65] Xiatian Zhu, Shaogang Gong, et al. Knowledge distillation by on-the-fly native ensemble. In *NeurIPS*, 2018.