Knowledge Distillation via Route Constrained Optimization

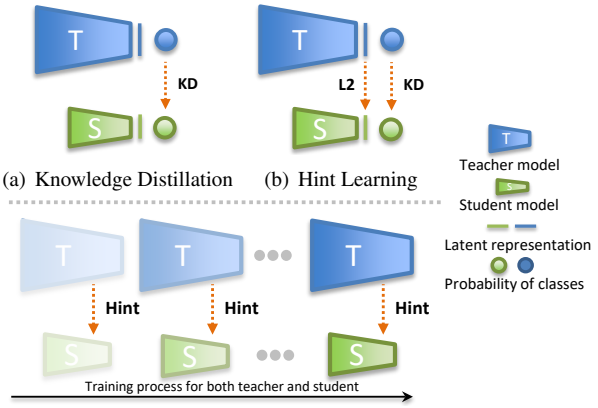
Xiao Jin[1*], Baoyun Peng[2*], Yichao Wu[1], Yu Liu[3], Jiaheng Liu[4],
Ding Liang[1], Junjie Yan[1], Xiaolin Hu[5]
[1] SenseTime Group Limited [2] National University of Defense Technology
[3] Chinese University of Hong Kong [4] Beihang University [5] Tsinghua University
jinxiaocuhk@gmail.com, pengbaoyun13@nudt.edu.cn, yuliu@ee.cuhk.edu.hk, liujiaheng@buaa.edu.cn
{wuyichao,liangding,yanjunjie}@sensetime.com, xlhu@mail.tsinghua.edu.cn

Abstract

Distillation-based learning boosts the performance of the miniaturized neural network based on the hypothesis that the representation of a teacher model can be used as structured and relatively weak supervision, and thus would be easily learned by a miniaturized model. However, we find that the representation of a converged heavy model is still a strong constraint for training a small student model, which leads to a higher lower bound of congruence loss. In this work, we consider the knowledge distillation from the perspective of curriculum learning by teacher's routing. Instead of supervising the student model with a converged teacher model, we supervised it with some anchor points selected from the route in parameter space that the teacher model passed by, as we called route constrained optimization (RCO). We experimentally demonstrate this simple operation greatly reduces the lower bound of congruence loss for knowledge distillation, hint and mimicking learning. On close-set classification tasks like CIFAR and ImageNet, RCO improves knowledge distillation by 2.14% and 1.5% respectively. For the sake of evaluating the generalization, we also test RCO on the open-set face recognition task MegaFace. RCO achieves 84.3% accuracy on one-to-million task with only 0.8 M parameters, which push the SOTA by a large margin.

1. Introduction

The performance of Convolutional Neural Network (CNN) can be significantly improved by the deeper and wider design of network structure. Whereas, it is hard to deploy these heavy networks on energetic consumption processor with limited memory. One way to deal with this situation is to make a trade-off between performance and speed by designing a miniaturized model to reduce the computational workload, at the cost of performance degradation. Thus, narrowing the performance gap between heavy model and miniaturized model becomes a research focus in recent years. Many methods were proposed to tackle this problem, such as model pruning [6, 15], quantization [12, 28] and knowledge transfer [10, 24].



(c) Route Constrained Hint Learning

Figure 1: Comparing to targeting only a converged teacher like KD and Hint-based learning, RCO narrows the performance gap by gradually mimicking the route sequence of teacher.

Among these approaches, knowledge distillation (KD) performs as an essential way to optimize a static model by mimicking the behavior (final predictions [10] or activations of hidden layers [24]) of a powerful teacher network, as shown in Figure 1(a) and Figure 1(b). Guided by this softened knowledge, a student network could pay more attention to extra supervision such as the probability correlation between classes rather than the one-hot label.

Previous methods only consider the final converged teacher model to teach small student network, which may result that the student stucks in approximating teacher's performance along with the increasing gap (in capacity) between teacher and student [22]. We observe that student

*Equal contribution.

supervised by teacher's early training stage has a much smaller performance gap with its teacher than that supervised by teacher's latter training stage. From the perspective of curriculum learning [1], a reasonable explanation beyond this observation is that teacher's knowledge is gradually becoming harder along with the training process. We claim that the intermediate states that teacher passed by are also valuable knowledge for easing the learning process and lowering the error bound of the student.

Based on this philosophy, we propose a new method called RCO which supervises student with the teacher's optimization route. Figure 1(c) shows the whole framework of RCO. Comparing to the single converged model, the route of teacher contains extra knowledge through providing an easy-to-hard learning sequence. By gradually mimicking such sequence, the student can learn more consistent with the teacher, therefore narrowing the performance gap. Besides, we analyze the impact of different learning sequence on performance and propose an efficient method based on a greedy strategy for generating sequence, which can be used to shorten the training paradigm meanwhile maintaining high performance.

Extensive experiments on CIFAR-100, ImageNet-1K and large scale face recognition show that RCO significantly outperforms knowledge distillation and other SOTA methods on all the three tasks. Moreover, our method can be combined with previous knowledge transfer methods and boost their performance. To sum up, our contribution could be summarized into three parts:

- We rethink the knowledge distillation model from the perspective of teacher's optimization path and get an significant observation that learning from the converged teacher model is not the optimal way.
- Based on the observation, we propose a novel method named RCO which utilizes the route in parameter space teacher network passed by as a constraint to bring a better optimization to student network.
- We demonstrate that the proposed RCO can be easily applied to both knowledge distillation and hint learning. Under the same data and computational cost, RCO outperforms KD by a large margin on CIFAR, ImageNet and a one-to-million face recognition benchmark Megaface.

2. Related Work

Neural Network Miniaturization. Many works study the problem of neural network miniaturization. They could be categorized into two methods: designing small network structure and improving the performance of small network via knowledge transfer. As for the former, many modifications on convolution were proposed since the original convolution took up too many computation resources. MobileNet [11] used depth-wise separable convolution to build block, ShuffleNet [31] used pointwise group convolution and channel shuffle. These methods could maintain a decent performance without adding too much computing burden at inference time. Besides, many studies [7, 23, 19, 9] focus on network pruning, which boosts the speed of inference through removing redundancy in a large CNN model. Han *et al*. [7] proposed to prune nonsignificant connections. Molchanov *et al*. [23] presented that they could prune filters with low importance, which were ranked by the impact on the loss. They approximated the change in the loss function with Taylor expansion. These methods typically need to tune the compression ratio of each layer manually. Most recently, Liu *et al*. [19] presented the network slimming framework. They constrained the scale parameters of each batch normalization [13] layer with sparsity penalty such that they could remove corresponding channels with lower scale parameters. He *et al*. [9] proposed to adopt reinforcement learning to exploit the design space of model compression. They benefited more from replacing manually tuning with automatical strategies.

As for the latter, the most two popular knowledge transfer methods are Knowledge Distillation [10] and FitNet [24]. We mainly consider these situations in this work.

Knowledge Distillation for Classification. Efficiently transferring knowledge from large teacher network to small student network is a traditional topic which has drawn more and more attention in recent years. Caruana *et al*. [2] advocated it for the first time. They claimed that knowledge of an ensemble of models could be transferred to the other single model. Then Hinton *et al*. [10] further claimed that knowledge distillation (KD) could transfer distilled knowledge to student network efficiently. By increasing the temperature, the logits (the inputs to the final softmax) contain richer information than one-hot labels. Afterward, [14] proposed to learn the curriculum from data by a network called MentorNet. [18] adopted a method to learn from noisy labels.

Learning Representation from Hint. Hint-based learning is often used for open-set classification such as face recognition and person Re-identification. FitNet [24] firstly introduced more supervision by exploiting intermediate-level feature maps from the hidden layers of teacher to guide training process of student. Afterward, Zagoruyko *et al*. [30] proposed the method to transfer attention maps from teacher to student. Yim *et al*. [29] defined the distilled knowledge from teacher network as the flow of the solution process (FSP), which is calculated by the inner product between feature maps from two selected layers.

Previous knowledge transfer methods only supervise student with converged teacher, thus fail to capture the knowledge during teacher's training process. Our work differs from existing approaches in that we supervise student with the knowledge transferred from teacher's training trajectory.

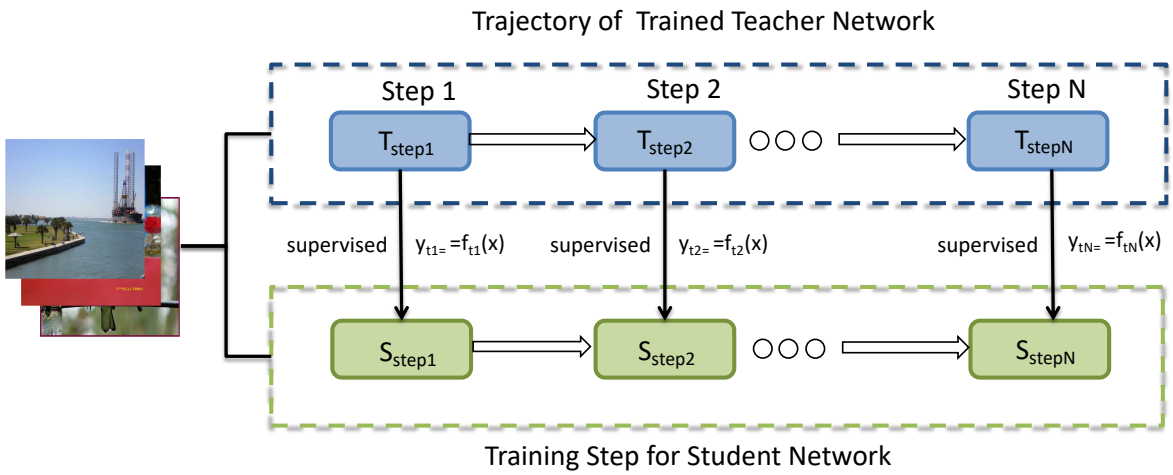


Figure 2: The overall framework of RCO. Previous knowledge transfer method only considers the converged teacher model. While RCO aims to supervise student with intermediate training state of teacher.

3. Route Constrained Optimization

3.1. Teacher-Student Learning Mechanism

For better illustration, we refer teacher network as ϕ_t with parameters W_t, and student network as ϕ_s with parameters W_s. $P_t = softmax(z_t)$ and $P_s = softmax(z_s)$ represent output predictions of teacher and student respectly, z_t and z_s for logits of teacher and student. The idea beyond KD is to let the student mimic the behavior of teacher by minimizing the cross-entropy loss and Kullback–Leibler divergence between predictions of teacher and student as follows:

$$L_{KD} = H(P_s, y) + \lambda KL(P_s^\tau, P_t^\tau), \tag{1}$$

where τ is a relaxation hyperparameter (referred as Temperature in [10]) for softening the output of teacher network, and λ is a hyper-parameter for balancing cross-entropy and KL divergence loss. In several works [26, 17] the KL divergence is replaced by euclidean distance,

$$L_{\text{mimic}} = \frac{1}{n}\sum_{i=1}^{n} \|\boldsymbol{f}_s - \boldsymbol{f}_t\|_2^2, \tag{2}$$

f represents the feature representations.

3.2. Difficulty in Optimizing Student

In common, the teacher invokes a larger and deeper network for arriving at a lower local minimum and achieving higher performance. It is hard for a smaller and shallower student to mimic such a large teacher due to the huge gap (in capacity) between teacher and student[22]. Usually, the network is trained to minimize the objective function by using stochastic gradient descent. Due to the high non-convex of the loss function, there are many local minima in the training process of deep neural networks. When the network converges to a certain local minimum, its training loss will converge to a certain (or similar) value regardless of different initializations.

Network	Epoch	10	40	120	240
ResNet-50	T top-1(%)	53.07	56.53	77	79.52
	T loss	1.680	1.199	0.067	0.009
MobileNetV2	S top-1(%)	51.21	57.62	66.05	68.71
	S loss	0.511	1.189	3.758	4.218

Table 1: The performance of student network trained with different epochs from teacher's training trajectory on CIFAR-100 dataset. "T" and "S" stand for teacher and student separately. "loss" represents training loss.

Could we reach a better local-minimal than this condition? We consider changing the optimization objective. More specifically, the student is trained by mimicking less deterministic target first, then moving forward to deterministic one, hoping in this way the student has a smaller gap with the teacher. To validate this, we use different intermediate states of teacher to supervise student, and use the training loss and top-1 accuracy to evaluate the difference between target teacher and converged student. MobileNetV2 [25] is adopted as student and ResNet-50 [8] as teacher. The teacher network is trained by cross-entropy loss, and the student network is trained by KD loss. We select checkpoints of teacher at 10th epoch, 40th epoch, 120th epoch, and 240th epoch as the target to train student network separately. The checkpoint at 240th epoch is the final converged model, and checkpoint at 10th epoch is least deterministic in the analysis.

Table 1 summarizes the results. It can be observed from the table that the student guided by the less deterministic target has lower training loss, and more convergent target brings larger gap in performance. In other words, the more

convergent teacher means a harder target for the student to approach.

Inspired by curriculum learning [1] that the local minima can be promoted by the easy-to-hard learning process, we take the sequence of teacher's intermediate states as the curriculum to help the student reach a better local minimum.

3.3. RCO

For better illustration, we refer the intermediate training states (checkpoints) used to form the learning sequence as **anchor points**. Suppose there are n anchor points on the teacher's trajectory. The overall framework of RCO is shown in Figure 2.

Without loss of generality, let $\boldsymbol{C} = C_1, C_2, ..., C_n$ represent the anchor points set, and the correpsonding outputs are $\phi_t(x; W_{C_1}), \phi_t(x; W_{C_2}), ..., \phi_t(x; W_{C_n})$. The training process for student is started from random initialization. Then we train the student step-by-step to mimic the anchor point on teacher's trajectory until finishing training with the last anchor point. At i_{th} step, the learning target of student is switched to the output $\phi_t(x; W_{C_i})$ of i_{th} anchor point. The optimization goal of i_{th} step is as follows:

$$\begin{aligned} L_{KD}(W_s, W_{C_i}) =& H(\phi_s(x; W_s), y) + \\ & \lambda H(\phi_s(x; W_s), \phi_t(x; W_{C_i})), \end{aligned} \tag{3}$$

where $i \in \{1, 2, ..., n\}$. The parameter $\boldsymbol{W_s}$ is optimized by learning to these anchor points sequentially. Algorithm 1 describes the details of the whole training paradigm.

Algorithm 1 Route Constrained Optimization

Require: anchor points set from pre-trained teacher network: $C_1, C_2, ..., C_n$, student network with parameter W_i
$i = 1$
Randomly initialize W_i
while $i \leq n$ **do**
  Initialize teacher network with C_i anchor, get W_{C_i}
  if $i > 1$ **then**
    Initialize W_i with W_{i-1}
  end if
  update the W_i by optimizing $L_{KD}(W_i, W_{C_i})$
  $i = i + 1$
end while
get W_n as the final weights of student.

3.4. Rationale for RCO

From the perspective of curriculum learning, the easy-to-hard learning sequence can help the model get a better local minimum [1]. RCO is similar to curriculum learning but different in that it provides an easy-to-hard labels sequence on teacher's trajectory.

Let $\mathcal{Y}_i$ be the output of i_{th} anchor point. The outputs of whole anchor points constrcut the space $\Omega = \{\mathcal{Y}_i | i = 1, 2, ..., n\}$. The results shown in Table 1 premises that the intermediate states on teacher's trajectory construct an easy-to-hard sequence, e.g. $\mathcal{Y}_i$ is easier to mimic than $\mathcal{Y}_{i+1}$ while $\mathcal{Y}_n$, the converged model, is the hardest objective for a small student.

Let the $\mathcal{X}$ be the training data. The training data and output of i_{th} anchor pair $(\mathcal{X}, \mathcal{Y}_i)$ provide a lesson. Then the curriculum sequence can be formulated as follows:

$$\{(\mathcal{X}, \mathcal{Y}_i) | i = 1, ..., n\}. \tag{4}$$

Without loss of generality, let $\mathcal{L}_\lambda(\mathcal{X}; \theta)$ represents a single-parameter family of cost functions such that L_1 can be easily optimized, while L_N is the criterion that we actually wish. During the sequential training of RCO, increasing λ means adding the hardness of learning through switching anchor points. Let $\mathcal{D}$ represent the hardness metric for a learning target. As shown in Section 3.2, more convergence of anchor means more hardness of learning target,

$$\mathcal{D}(\phi(\mathcal{X}, W_{C_i}) < \mathcal{D}(\phi(\mathcal{X}, W_{C_{i+1}}) \quad \forall i > 0. \tag{5}$$

In curriculum learning [1] the sequence of learning is generated by spliting the $\mathcal{X}$ in to several "lessons" with different hardness depending on a predefined criterion. While RCO can be seen as a more flexible approach, which gradually changes the hardness of target labels $\mathcal{Y}$. Both curriculum learning and RCO work by easy-to-hard learning to move θ gradually into the basin of attraction of a dominant (if not global) minimum [1].

3.5. Strategy for Selecting Anchor Points

Equal Epoch Interval Strategy. Typically, the teacher network could produce tremendous checkpoints during the training process. To find the optimal learning sequence, one can search it with brute force. However, given n possible intermediate states, there are 2^n possible sequences, which is impractical to implement. A straightforward strategy is supervising the student by every state (epoch/iteration) on teacher's trajectory. However, mimicking every state is dispensable and time-consuming since adjacent training states are very close to each other. Given limited time, a more efficient way is to sample epochs with equal epoch interval (**EEI**), e.g. select one for every four epochs.

Although efficient in time, EEI is a quite simple ad-hoc method that ignores the hardness between different anchor points, and it would lead to an improper curriculum sequence. The desirable property of the curriculum sequence should be efficient to quickly learn and smooth in hardness to better bridge the gap between teacher and student.

Greedy Search Strategy. To delve into optimization route of student when learning to teacher, we count the KL

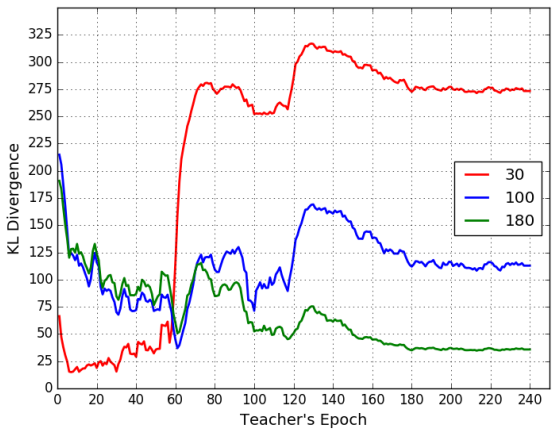


Figure 3: The curve of KL divergence loss between student supervised by teacher's different epochs (30th, 100th, 180th) and teacher's all 240 epochs on validation set.

divergence between outputs of student and different target states of teacher on validation set, which consists of 10k examples random sampled from training set. The teacher is trained by dropping learning rate at 60th,120th,180th epochs. We choose 30th, 100th, and 180th epochs as target states to supervise the student respectively. Figure 3 shows the KL divergences curve between student and intermediate states of teacher. From the figure we can observe that the student supervised by teacher's 30th epoch is very close with teacher's 30th epoch, but has large gap with teacher's latter epochs, especially after teacher dropping learning rate. The same observations can be found from student supervised by teacher's other epochs.

Table 1 and Figure 3 give us two insights: a particular student has the ability to learn harder target limited in a certain range; supervised by a better teacher would promote the ability of student.

Inspired by these insights, we propose a greedy search strategy (**GS**) to find efficient and hardness-smooth curriculum sequence. The goal of greedy strategy is to find the one which is on the boundary of range that student can learn. To find those boundary anchor points, a metric is introduced as follows:

$$\begin{aligned} r_{ij} &= \frac{\mathcal{H}_j - \mathcal{H}_i}{\mathcal{H}_i}, i, j \in \{i+1, i+2, ..., N\}, \\ \mathcal{H}_i &= \mathcal{H}(\phi_s(\mathcal{X}^{'}, W_s), \phi_t(\mathcal{X}^{'}, W_{t_i}), \\ \mathcal{H}_j &= \mathcal{H}(\phi_s(\mathcal{X}^{'}, W_s), \phi_t(\mathcal{X}^{'}, W_{t_j}), \end{aligned} \tag{6}$$

where $\mathcal{H}$ is the KL divergence, and $\mathcal{X}^{'}$ is the validation set. r_{ij} evaluates the hardness of j_{th} epochs of teacher for a student guided by i_{th} epochs. Then we refer a hyper-parameter δ as the threshold which indicates the learning ability of student. When $r_{ij} > \delta$, it means that j_{th} epoch is hard for student trained by i_{th} to learn, and $r_{ij} < \delta$ means inverse. Based on the above philosophy, we give the complete **GS** strategy in Algorithm 2.

It seems the anchor points that near the point of tuning learning rate are more important than other anchor points. Intuitively, according to Algorithm 2, the optimal learning sequence must contain at least one anchor point from different learning rate stage. Since this section mainly focuses on the strategy for anchor points selection, we provide the empirical value of $\delta = 0.8$ for MobileNetV2 to achieve a better balance between performance and training cost. Note that although our experiments are based on SGD, GS is also applicable for other optimization methods like SGDR[20], since prerequisites are still true.

Algorithm 2 Greedy Search

Require: Student network with parameter W_s after mimicking former i_{th} anchor point C_i, where $i \in \{1, 2, ..., N\}$, relaxation factor δ.
compute KL divergence $\mathcal{H}_i$
$j = i + 1$
while $j < N$ **do**
  compute $\mathcal{H}_j$ on validation set
  compute $r_{ij} = \frac{\mathcal{H}_j - \mathcal{H}_i}{\mathcal{H}_i}$
  if $r_{ij} > \delta$ **then**
    Return j-1;
  end if
  $j = j + 1$
end while
Return N;

4. Experiments

Common Settings. The backbone network for teacher in all experiments is ResNet-50. For the student structure, instead of using smaller ResNet, we use more compact MobileNetV2 as well as its variants with different FLOPs, since MobileNetV2 has proven to be highly effective in keeping high accuracy while maintaining low FLOPs in many tasks. Expansion ratio and width multiplier are two tunable parameters to control the complexity of the MobileNetV2. We make default configuration by setting expansion ratio to 6 and width multiplier to 0.5. The relaxation is 5 in KD loss. Note that all these experiments are based on GS that usually produces about 4 anchor points.

4.1. Experiment on CIFAR-100

The CIFAR-100 dataset contains 50 000 images in training set and 10 000 images in validation set with size 32×32. In this experiment, for the teacher network we set initial learning rate to 0.05 and divide it by 10 at 150th, 180th, 210th epochs and we train for 240 epochs. We set weight decay to 5e-4, batch size to 64 and use SGD with momen-

tum. For the student network, the setting is almost identical with teacher's except that the initial learning rate is 0.01.

We compare the top-1 accuracy of CIFAR-100 dataset and show the result in Table 2. From the result we can find that our method improves about 2.1% on top-1 compared with KD.

Although the base student network is small and fast, it is common that some specific cases or applications require the model to be smaller and faster. To further investigate the effectiveness of the proposed method, we conduct extensive experiments by applying RCO to a series of MobileNetV2 with different width multipliers and expansion ratios. We set expansion ratio to 4, 6, 8, 10 and width multiplier to 0.35, 0.5, 0.75, 1.0 separately, which forms totally 16 different combinations. The FLOPs of these models are shown at Table 3. We rank the model according to the width multiplier and draw the result on Figure 4. From the figure we can make the following observations: (i) The proposed method exhibits consistently superiority in all settings. (ii) The student network with smaller capacity(e.g. MobileNetV2 with T=4, Width=0.35) generally gains more improvement from RCO. (iii) Although the model with expansion ratio set to 10 and width multiplier set to 0.35 has larger FLOPs than the model with expansion ratio set to 4 and width multiplier set to 0.5, the former setting shows performance reduction among all three methods. It indicates that parameterizing expansion ratio with 10 and width multiplier to 0.35 largely limits representation power.

Method	Network	MFlops	top-1	Loss
T-Softmax	ResNet-50	2.6k	79.34	-
S-Softmax	MobileNetV2	13.5	61.88	-
S-KD	MobileNetV2	13.5	68.71	1.59
S-RCO	MobileNetV2	13.5	**70.85**	1.45

Table 2: Results on CIFAR-100

Expansion ratio	Width multiplier			
	0.35	0.5	0.75	1.0
4	5.4	9.8	19.3	32.1
6	7.3	13.5	27.2	45.6
8	9.1	17.1	35	59.1
10	11	20.8	42.8	72.6

Table 3: Complexity (MFLOPs) for MobileNetV2 with different settings

4.2. Experiment on ImageNet

The ImageNet dataset contains 1000 classes of images with various sizes. It is the most popular dataset in classification task. In this experiment, for the training of teacher network, we set initial learning rate to 0.4 and drop by 0.1

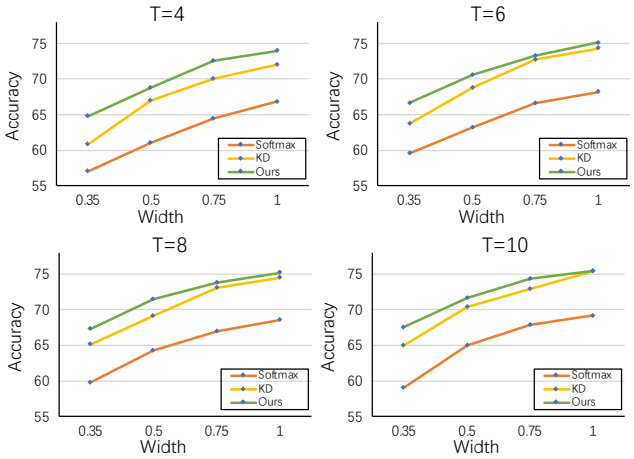


Figure 4: CIFAR-100 top-1 accuracy of MobileNetV2 with different settings. The "Width" in x-axis represents width multiplier and "T" in the title stands for the expansion ratio. The proposed method acquires more promotion with smaller student network.

at 15k, 30k and 45k iterations and we train for 50k iterations. We set weight decay to 5e-4, batch size to 3072 and use SGD with momentum. As for the student network, we set initial learning rate to 0.1 and drop by 0.1 at 45k, 75k and 100k iterations and we train for 130k iterations. We set weight decay to 5e-4, batch size to 3072 and use SGD with momentum. In order to keep training stable, we use warm-up suggested by [4] when training with large batch size. We compare the top-1 and top-5 accuracy of ImageNet dataset and show the result in Table 4. From the result we can find that our method improves about 1.5%/0.7% on top-1/top-5 compared with KD, which verifies that RCO is applicable to large-scale classification.

Method	Network	top-1	top-5
Teacher-Softmax	ResNet-50	75.49	92.48
Student-Softmax	MobileNetV2	64.2	85.4
Student-KD	MobileNetV2	66.75	87.3
Student-RCO	MobileNetV2	**68.21**	**88.04**

Table 4: Results on ImageNet

4.3. Experiment on Face Recognition

Unlike classification, the network in face recognition usually contains a feature layer implemented as a fully-connected layer to represent the projection of each identity. Empirical evidence [21] shows that mimicking the feature layer as the way used in FitNet [24] could bring more improvements for student network. We follow this setting in our baseline experiments.

We take two popular face recognition datasets MS-Celeb-1M [5] and IMDb-Face [27] as our training set and validate our method on MegaFace. The **MS-Celeb-1M**

is a large public face dataset which contains one million identities with different age, sex, skin, and nationality and is widely used in face recognition area. The **IMDb-Face** dataset contains about 1.7 million faces, 59k identities. All images are obtained from IMDb website. The **MegaFace** is one of the most popular benchmarks that could perform face recognition under up to 1 million distractors. This benchmark is evaluated through probe and gallery images from FaceScrub.

In this experiment, for the teacher network, we set initial learning rate to 0.1 and drop by 0.1 at 100k, 140k, 170k iterations and we train for 220k iterations. We set weight decay to 5e-4, batch size to 1024 and use SGD with momentum. We resize the input image to 224×224 without augmentation. We use ArcFace [3] to train the teacher network. As for the student network, we set initial learning rate to 0.05 and drop by 0.1 at 180k, 210k iterations and we train for 240k iterations. The rest settings are identical to the teacher.

We show our result on Table 5. From the table we can see that on this challenging face recognition task, RCO largely boosts the performance of MobileNetV2 [25] compared to original hint-based learning.

	top-1 @ distractor size					
Method	e^1	e^2	e^3	e^4	e^5	e^6
Teacher	99.78	99.67	99.38	98.86	97.70	94.83
Softmax	99.20	96.37	91.49	84.45	75.60	65.91
FitNet	99.62	98.80	96.83	93.53	88.28	81.02
RCO	**99.69**	**99.01**	**97.52**	**94.84**	**90.55**	**84.3**

Table 5: Results on MegaFace

4.4. Ablation Studies

Although RCO achieves decent result in previous experiments, the extra training time that it brings is not negligible. Even if we just construct the learning sequence with 4 anchor points, it still needs 4 times training epochs compared with KD or Softmax. Since training time plays an important role in either research or industrial, we consider using the same time as KD to verify the robustness of RCO. Note that we set expansion ratio to 4 and width multiplier to 0.35 for the backbone MobileNetV2 in this section.

Comparison under Limited Training Epochs. Previous experiments commonly need more training epochs than KD. Consider performing RCO with EEI strategy on CIFAR-100. Let M_{gap} be the epoch interval used in EEI. To get 4 anchor points, we can set M_{gap} to 60. Then the selected anchor points should be 60th, 120th, 180th, and 240th epoch. The student trained with the learning sequence needs 960 epochs in total since each anchor point is trained for 240 epochs to ensure convergence.

We then speed up the EEI strategy from multi-stage to one stage (**one-stage EEI**), where we only train student for 240 epochs, by simply modifying the training paradigm as follows: the student is initially supervised by 60th epoch of teacher for the student's first 60 epochs, then supervised by teacher's 120 epoch for the next 60 epochs, and so on.

In one-stage EEI, it is natural to evaluate the impact of different number of anchor points. Let K be the size of training set. We start the M_{gap} from the smallest case, where M_{gap} is 1/(K /$BatchSize$) (It is 1.28E-3 in Table 6, which means student mimic teacher's every iteration). Then gradually increase M_{gap} to the largest case, where M_{gap} is the maximum epoch (240) and RCO degrades into KD.

From the perspective of optimization route, we find method in [33] could be regarded as a particular case of RCO when setting M_{gap} to the smallest value, and matching logits with KD loss instead of MSE loss. Besides, We also follow [32] to implement DML and make a comparison with KD. The result in Table 6 shows that RCO outperforms other methods in all settings. By properly selecting M_{gap} to 10, RCO gets 4.2% and 3.8% improvement on CIFAR-100 compared with KD and DML respectively.

Method	M_{gap}	Anchor Number	top-1
DML[32]	-	-	61.13
RL[33]	1.28E-3	187500	61.63
	1	240	62.74
	2	120	63.78
	4	60	64.21
RCO	**10**	**24**	**65.01**
	20	12	63.88
	60	4	64.5
KD	240	1	60.79

Table 6: Comparison of RCO based on One-stage EEI with other knowledge transfer methods under limited training epochs. It clearly shows RCO outperforms other methods by using same training epochs.

Comparison on Different Strategies. Since the strategy is the most crucial part of RCO, we make a comparison between these strategies. For practical considerations, we limit the training epoch to no more than four times the epochs of KD. We have chosen the following strategies to compare: **one-stage EEI, EEI-x, GS**, where the "x" in "EEI-x" represents the number of selected anchor points with EEI strategy. The results are shown in Table 7. From the result we make the following observations: 1) all strategies show great superiority to KD, 2) GS is the best strategy among them, thus should be used when training time is not a constraint.

4.5. Visualization

Visualization of Trajectory. In order to further analyze our method, we plot student's training trajectory using PCA

Strategy	one-stage	M_{gap}	Total Epoch	top-1
KD	✓	240	240	60.79
one-stage EEI	✓	10	240	**65.01**
EEI-2	✗	120	480	61.43
EEI-3	✗	80	720	63.34
EEI-4	✗	60	960	65.27
GS	✗	-	720	**65.41**

Table 7: Comparison of RCO based on different strategies on CIFAR-100. GS achieves the best result among all strategies.

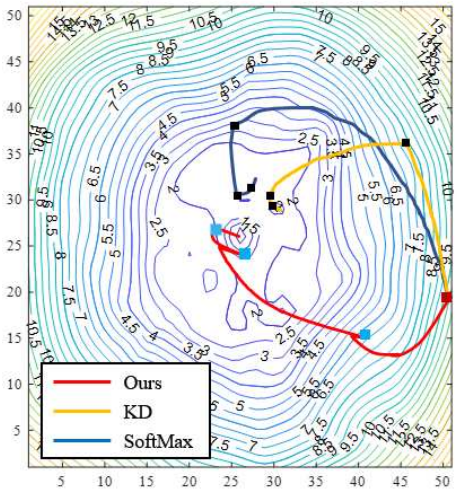


Figure 5: Visualization of student's training trajectories by PCA directions for MobileNetV2 on CIFAR-100 dataset. These student networks are trained by different methods (SoftMax, KD, and Ours) and initialized with the same random parameters. The red curve stands for student trained with RCO, the blue dot on the line represents the location guided by intermediate anchor point. The red curve arrives at the lowest local minimum among them.

directions suggested by Li *et al*. [16]. The process is as follows: Given n training epochs, let W_{m_i} denote model parameters at epoch i and the final estimate as W_{m_n}. Then we apply PCA to the matrix $[W_{m_0} - W_{m_n}; ...; W_{m_{n-1}} - W_{m_n}]$ and choose the most two principal directions.

In Figure 5 the training trajectory of MobileNetV2 on CIFAR-100 is plotted for student in three modes: 1) Softmax, 2) KD, 3) the proposed method (Ours). For a fair comparison, three students are initialized with the same parameters (marked as red dot) and are trained for 240 epochs each, where RCO uses one-stage EEI with three anchor points. For the curve of RCO, the blue dots on the line show the epochs where student is guided by intermediate anchor points. For KD or Softmax, epochs where the learning rate was decreased are shown as black dots.

The first anchor point keeps the student away from the direction suggested by Softmax or KD and arrives at an intermediate state. The state itself may not lie in a well-performed parameter space, but with the guidance of succeeding anchor points the student network eventually reaches to a deeper local minimum, which has adequately demonstrated the importance of optimization route from teacher network.

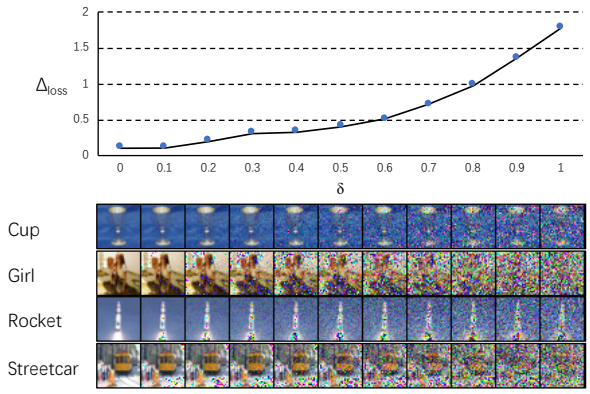


Figure 6: The curve of loss gap between KD loss and RCO loss along with Gaussian noise. δ represents the scale of Gaussian noise. Δ_{loss} represents the loss gap between KD loss to RCO loss. Larger Δ_{loss} means lower loss of RCO than KD. The bottom row shows part of noised images.

Visualization of Robustness to Noise. Besides the visualization of optimization trajectory, we also observe that the new local minimum has better generalization capacity and is more robust to random noise in input space. We consider bringing noise to the testing image. Firstly we calculate the standard deviation σ_{in} for each image and set the δ ranging from 0.0 to 1.0 by step 0.1. The noise is sampled from $\mathcal{N}(0, \sigma^2)$, where $\sigma = \sigma_{in} * \delta$. We choose some noised images and show them at the bottom row of Figure 6. The images are clear at first column, but as the δ increases, the images become illegible, especially for the last column. We ran this experiment on model trained both with KD and RCO and compared their loss. The loss gap from KD to RCO becomes more significant as the increasing of δ, which suggests that model trained with RCO is more robust to noise than KD. The result is on top of Figure 6.

5. Conclusion

We have proposed a simple but effective and generally applicable method to boost the performance of small student network. By constructing an easy-to-hard sequence of learning target, student network could achieve much higher performance compared with other knowledge transfer methods. Moreover, we offer two available strategies to construct the sequence of anchor points. For future work, we would like to explore the strategy to design the learning sequence automatically.

Acknowledgement This work is sponsored in part by the National Key R&D Program of China under Grant (No. 2018YFB2101100) and NSFC under Grant 61836014.

References

[1] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *International Conference on Machine Learning*, pages 41–48, 2009.

[2] Rich Caruana and Alexandru Niculescu-Mizil. Model compression. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 535–541, 2006.

[3] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. 2018.

[4] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[5] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: Challenge of recognizing one million celebrities in the real world. *Electronic Imaging*, 2016(11):1–6, 2016.

[6] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[7] 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.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[9] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li Jia Li, and Song Han. Amc: Automl for model compression and acceleration on mobile devices. 2018.

[10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. 2014.

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

[12] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, El Yaniv Ran, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *Journal of Machine Learning Research*, 18, 2016.

[13] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[14] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, 2018.

[15] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016.

[16] Hao Li, Zheng Xu, Gavin Taylor, and Tom Goldstein. Visualizing the loss landscape of neural nets. *arXiv preprint arXiv:1712.09913*, 2017.

[17] Quanquan Li, Shengying Jin, and Junjie Yan. Mimicking very efficient network for object detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7341–7349. IEEE, 2017.

[18] Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1910–1918, 2017.

[19] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2755–2763. IEEE, 2017.

[20] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. 2016. *arXiv preprint arXiv:1608.03983*.

[21] Ping Luo, Zhenyao Zhu, Ziwei Liu, Xiaogang Wang, Xiaoou Tang, Ping Luo, Zhenyao Zhu, Ziwei Liu, Xiaogang Wang, and Xiaoou Tang. Face model compression by distilling knowledge from neurons. In *AAAI Conference on Artificial Intelligence*, 2016.

[22] Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant: Bridging the gap between student and teacher. *arXiv preprint arXiv:1902.03393*, 2019.

[23] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. 2016.

[24] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[25] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[26] Gregor Urban, Krzysztof J. Geras, Samira Ebrahimi Kahou, Ozlem Aslan, Shengjie Wang, Rich Caruana, Abdelrahman Mohamed, Matthai Philipose, and Matt Richardson. Do deep convolutional nets really need to be deep and convolutional? *Nature*, 521, 2016.

[27] Fei Wang, Liren Chen, Cheng Li, Shiyao Huang, Yanjie Chen, Chen Qian, and Chen Change Loy. The devil of face recognition is in the noise. *arXiv preprint arXiv:1807.11649*, 2018.

[28] Jiaxiang Wu, Leng Cong, Yuhang Wang, Qinghao Hu, and Jian Cheng. Quantized convolutional neural networks for mobile devices. In *Computer Vision and Pattern Recognition*, pages 4820–4828, 2016.

[29] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2017.

[30] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. 2016.

[31] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. 2017.

[32] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. *arXiv preprint arXiv:1706.00384*, 6, 2017.

[33] Guorui Zhou, Ying Fan, Runpeng Cui, Weijie Bian, Xiaoqiang Zhu, and Kun Gai. Rocket launching: A universal and efficient framework for training well-performing light net. *stat*, 1050:16, 2017.