A.Appendix

A.1 Algorithm

Algorithm 1 Vanilla Trainer

```
Require:
    Learning rate $\gamma > 0$,
    momentum coefficient $0 < \mu < 1$
    loss function $\mathcal{L}$
Initialize:
    $\boldsymbol{\theta}, \boldsymbol{v} \leftarrow 0$
while $\boldsymbol{\theta}$ not converged do
    $\boldsymbol{g} \leftarrow \nabla_{\boldsymbol{\theta}} \mathcal{L}(x, y, \boldsymbol{\phi}; \boldsymbol{\theta})$
    $\boldsymbol{v} \leftarrow \boldsymbol{g} + \mu \boldsymbol{v}$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma \boldsymbol{v}$
end while
```

Algorithm 2 Distillation-Oriented Trainer

```
Require:
    Learning rate $\gamma > 0$,
    momentum coefficient $0 < \mu < 1$,
    momentum difference $0 < \Delta < 1 - \mu$,
    loss functions $\mathcal{L}_{\mathrm{CE}}$, $\mathcal{L}_{\mathrm{KD}}$ and
    corresponding weights $\alpha, 1 - \alpha$.
Initialize: $\boldsymbol{\theta}, \boldsymbol{v}_{\mathrm{ce}} \leftarrow 0, \boldsymbol{v}_{\mathrm{kd}} \leftarrow 0$
while $\boldsymbol{\theta}$ not converged do
    $\boldsymbol{g}_{\mathrm{ce}} \leftarrow \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{CE}}(x, y; \boldsymbol{\theta})$
    $\boldsymbol{g}_{\mathrm{kd}} \leftarrow (1 - \alpha) \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{KD}}(x, \boldsymbol{\phi}; \boldsymbol{\theta})$
    $\boldsymbol{v}_{\mathrm{ce}} \leftarrow \boldsymbol{g}_{\mathrm{ce}} + (\mu - \Delta) \boldsymbol{v}_{\mathrm{ce}}$
    $\boldsymbol{v}_{\mathrm{kd}} \leftarrow \boldsymbol{g}_{\mathrm{kd}} + (\mu + \Delta) \boldsymbol{v}_{\mathrm{kd}}$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma (\boldsymbol{v}_{\mathrm{ce}} + \boldsymbol{v}_{\mathrm{kd}})$
end while
```

A.2 A toy experiment for better understanding

We conduct a series of *toy* experiments to intuitively illustrate the optimization behaviors of DOT. Concretely, we initialize a 2-d (trainable) tensor as the logits for a binary classification task. Then, we employ a loss function composed of two parts: (1) a cross-entropy loss (where the target class is 1), and (2) a distillation loss (where the teacher's prediction is a constant 0.7). We use a vanilla SGD and our proposed DOT to respectively optimize the loss function, and the prediction of the 2-d tensor is shown in Figure 8. It suggests that applying DOT makes the 2-d tensor more similar to the teacher's prediction (0.7 is the ideal output of a student if distillation loss is well optimized). What's more, DOT could search a wide range of the loss landscape (great fluctuations in Figure 8), which helps the model to get rid of sharp local minima. We hope this toy experiment could provide insights for an intuitive understanding of the working mechanism of DOT.

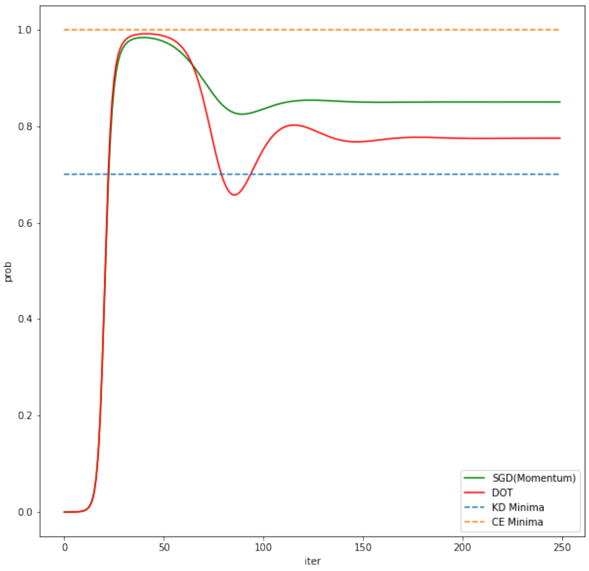

Figure 8: Toy experiments for analyzing optimization behaviors of SGD and DOT. It conveys that DOT could help the student network converge to minima satisfying the distillation loss well.

A.3 More pairs on CIFAR-100

We conduct more experiments on CIFAR-100 following CRD's protocol, and results are reported in Table 6 and 7 which verifies the universality of DOT.

teacher	student	KD	DOT
WRN-40-2	WRN-16-2	74.92	75.85
WRN-40-2	WRN-40-1	73.54	74.06
ResNet56	ResNet20	70.66	71.07
ResNet110	ResNet20	70.67	71.22
ResNet110	ResNet32	73.08	73.72
ResNet32×4	ResNet8×4	73.33	75.12
VGG13	VGG8	72.98	73.77

Table 6: Pairs of the same architecture.

teacher	student	KD	DOT
VGG13	MobileNetV2	67.37	68.21
ResNet50	MobileNetV2	67.35	68.36
ResNet50	VGG8	73.81	74.38
ResNet32×4	ShuffleNetV1	74.07	74.58
ResNet32×4	ShuffleNetV2	74.45	75.55
WRN-40-2	ShuffleNetV1	74.83	75.92

Table 7: Pairs of the different architectures.

A.4 Does longer training time help for better convergence?

In Section 3 of the manuscript, we visualize and analyze the loss curves and reveal a *trade-off* issue caused by introducing distillation loss. We further conduct the experiment for longer training epochs, *i.e.*, applying a smaller learning rate for extra epochs to study whether the trade-off could be alleviated. Concretely, we further train the network with both task and distillation losses for 60 epochs and decay the learning rate every 30 epochs. Results in Table 8 indicate that longer training still cannot significantly decrease the training task loss. The task loss after longer training is still around 0.38, while the task loss of the vanilla baseline is 0.2379. It indicates that the trade-off issue still remains, further supporting the existence of optimization conflict between task loss and distillation loss.

epoch	baseline	240	270	300
validation top-1	72.50	73.33	73.51	73.63
training task loss	**0.2379**	0.3844	0.3801	0.3818

Table 8: Results of training students with both task and distillation losses for longer epochs.

A.5 Why does DOT perform better on challenging datasets?

As shown in Table 3, DOT works better on challenging datasets, *e.g.*, DOT achieves +1∼2%, 3∼6% and 1∼2% performance gain on CIFAR-100, Tiny-ImageNet and ImageNet, respectively. We believe the reason is that the teacher could transfer more useful and valuable knowledge on the challenging tasks, and dominating the optimization with distillation loss could better leverage the knowledge, which means the upper bound of the performance gain for DOT is higher.

A.6 About tuning Δ

The only hyper-parameter introduced by our DOT is Δ. We notice that the values of Δ need adjustments on different datasets. However, the improvement is satisfactory without tuning Δ, *i.e.*, Δ *is not a sensitive hyper-parameter*. Concretely, the value of Δ for KD+DOT on CIFAR100 is set as 0.075, the for CRD+DOT is set as 0.05, as well as for DKD+DOT. The values of Δ for all methods on Tiny-ImageNet are set as 0.075. As for ImageNet, knowledge from teachers is more valuable and reliable, so we set Δ as 0.09 for both KD+DOT and DKD+DOT.

A.7 Implementation of DOT

It is worth mentioning that the implementation of DOT for feature-based methods is not the same as for logit-based methods (*e.g.*KD and DKD). The reason is as follows: The extra distillation loss of logit-based methods is a KL-Divergence applied on the student's logits and the teacher's logits, so there are no other extra parameters to optimize and all the parameters of the student network are involved. On the contrary, feature-based methods need extra modules and parameters as connectors between students' features and teachers' features. And the final fully-connected layer (the classifier) is not involved when computing the gradients of feature-distillation loss. In other words, different losses involve different network parameters in the feature-based distillation methods, so directly applying different momentums for the different losses will lead to a "gradient inconsistency" problem. To solve this problem, DOT only applies different momentums on the parameters involved by both task and distillation losses. For parameters involved by only one loss (*e.g.*, the final fully-connected layer), momentums are the same as the baseline.