

# —Supplementary Material— Make Baseline Model Stronger: Embedded Knowledge Distillation in Weight-Sharing Based Ensemble Network

Shuchang Lyu  
lyushuchang@buaa.edu.cn

Qi Zhao  
zhaoqi@buaa.edu.cn

Yujing Ma  
zy1902407@buaa.edu.cn

Lijiang Chen  
chenlijiang@buaa.edu.cn

Department of Electronics and  
Information Engineering, Beihang  
University  
Beijing, China

Supplementary material provides details that could not be included in the paper submission due to page limitations. Sec. 1 provides details on the principle of hyper-parameter adjusting. Sec. 2 describes a more general branch points setting strategy, which shows the flexibility of EKD-FWSNet. Sec. 3 shows the comparison and analysis of computation cost during training. Sec. 4 shows the visualization and analysis, which can prove the interpretability of our method.

## 1 Hyper-Parameter Adjusting Principle

In this paper, we propose simple yet efficient optimizing method. Only two scalars are served as hyper-parameters, where  $w$  and  $\alpha$  are respectively applied to adjust the proportions of  $KL$  distillation loss and  $MSE$  ensemble-attention loss. For different tasks, we adopt different hyper-parameter adjusting strategies to make model training work well. Tab. 1 shows the configuration of these two scalars on different tasks. The principle of adjusting the hyper-parameter is listed as follows. 1) We tend to use higher value of  $w$  for more complex task (CIFAR100, tiny-ImageNet). Because the pattern of class probabilities is more complex when the number of categories is large. 2) We set smaller  $\alpha$  when applying  $MSE$  loss on large-scale feature maps to avoid abnormal values caused by large loss. 3) When more branches are added, we may decrease the factor of cross-entropy loss to avoid exploding gradients. Fortunately, it rarely happens in our proposed models. 4) For ResNet [1] series and EfficientNet [2] series, we use a fix set of hyper-parameter shown in Tab.1, which shows the robustness and less training sensitivity of our proposed EKD-FWSNet.

Models	Datasets		
	CIFAR-10	CIFAR-100	tiny-ImageNet
ResNet-20/32/44/56	(15, 1)	(60, 1)	-
ResNet-18/34	-	(100, 1)	(100, 0.1)
EfficientNet-b0/b2/b4	-	(100, 1)	(100, 0.1)

Table 1: The configuration of hyper-parameter on different datasets. We denote the two hyper-parameter in format of  $(w, \alpha)$

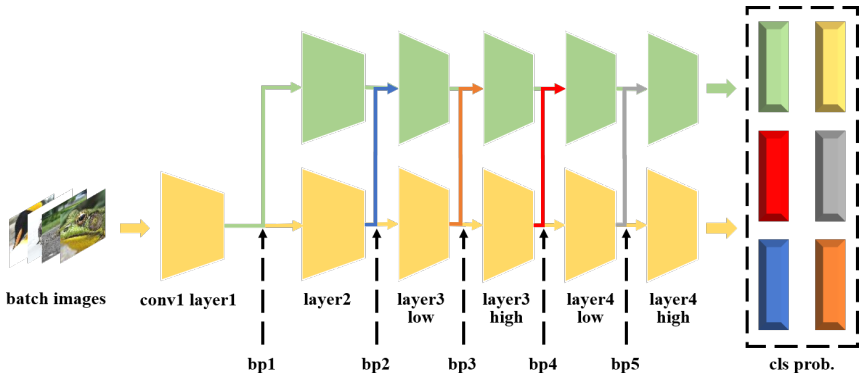


Figure 1: An architecture of six-branch EKD-FWSNet. The main branch is ResNet18/34, we set five branch points, where “bp3” and “bp5” are respectively set inside “layer3” and “layer4” of main branch.

## 2 Block-Wise Branch Point Setting in EKD-FWSNet

As mentioned in paper, only setting branch points between layers is not flexible. In this section, we introduce block-wise (i.e. inside layers) branch point setting. Designing in this way, EKD-FWSNet will have more forward paths. Fig. 1 shows a six-forward paths (six-branch) EKD-FWSNet respectively using ResNet18/34 as main branches. We show the classification accuracy in Tab. 2.

Even though six-branch EKD-FWSNet has little superiority compared to the result of four-forward paths EKD-FWSNet in Tab.3 of main paper, the block-wise branch point setting strategy makes EKD-FWSNet more flexible. Compared to previous works [14, 15], EKD-FWSNet can flexibly extend to an end-to-end ensemble network with larger number of branches using little trainable parameters.

Models		Branch-num	Err-rate
ResNet-18@layer1	EKD-FWSNet	4	20.49
ResNet-18@layer1	EKD-FWSNet	6	20.63
ResNet-34@layer1	EKD-FWSNet	4	19.94
ResNet-34@layer1	EKD-FWSNet	6	19.91

Table 2: Classification results of six-forward paths EKD-FWSNet. We use ResNet-18/34 as baseline network and evaluate on CIFAR-100. In EKD-FWSNet (Fig.1), we set five split points, two of which are block-wise branch points. The first split point is at layer1.

### 3 Comparison and Analysis of Computation Cost in Training Phase

As Mentioned in main paper, some recent novel works [4, 7, 8] show the power of “student-classmate” KD based ensemble network (Fig.1 **middle** of main paper). However, with the increase of classmate branches, the training burden increase obviously, which will complicate training process. Comparing with those methods, our proposed method is more compact and flexible. To intuitively show the computation cost in training phase, we compare the training FLOPs in Tab. 3.

To fairly compare the computation cost of our method with previous KD and ensemble learning based methods, we first calculate the training parameters and FLOPs during training phase of individual baseline model. Then we carefully reimplement the models [4, 7, 8] and calculate their training cost. As shown in Tab. 3, DML [8] uses a two-branch (two ResNet-32) structure. Therefore, the training parameters and the training FLOPs are approximately the two times of baseline model. If more branches are involved, parameters and FLOPs will multiply. OEM [7] combines several branches with different size. However, each branch is fed with high-resolution feature maps, which also costs a lot both on parameters and FLOPs. KD-ONE [4] first utilizes low-level layers as weight-sharing blocks and then constructs multiple branches using separate high-level layers. Comparing with DML and OEM, this design largely lowers the training FLOPs.

Compared to DML and OEM, EKD-FWSNet can train a baseline model better with less training parameters and FLOPs. Compared to three-branch KD-ONE, EKD-FWSNet uses less parameters but more FLOPs. Then, we increase the branch number and make further comparison. When adding more branches, KD-ONE will introduce more high-level layers. Since high-level layers contain large group of parameters, the training parameters and FLOPs of six-branch KD-ONE rapidly increase, especially training parameters. To construct a six-branch EKD-FWSNet, we adopt block-wise branch point setting strategy (Sec.2, Fig. 1). From Tab.3, it is clear that EKD-FWSNet has no training parameters increase. From three-branch to six-branch structure, EKD-FWSNet has less training FLOPs increase (2.73G vs 3.31G).

In summary, the huge superiority of EKD-FWSNet is low training parameter cost. Another huge superiority of EKD-FWSNet is flexibility. when more branches are constructed, no extra training parameters are involved and the computation cost increase in a slow way.

Methods	Params	FLOPs	Branch-num	Err-rate
ResNet-32 (baseline)	0.46M	3.40G	1	30.73
OEM [10]	0.95M	7.02G	5	29.03
DML [8]	0.93M	6.81G	2	28.90
KD-ONE [9]	1.17M	5.61G	3	26.61
KD-ONE [9]	2.22M	8.92G	6	-
EKD-FWSNet	<b>0.90M</b>	6.71G	3	<b>26.46</b>
EKD-FWSNet	<b>0.90M</b>	9.44G	6	<b>26.22</b>

Table 3: Comparison and analysis of computation cost in training phase. Parameters and FLOPs are calculated during training forward process with standard input size  $224 \times 224$ . We use ResNet-32 as baseline network and compare the classification performance on CIFAR-100. Additionally, we reimplement the models of OEM [10], DML [8] and KD-ONE [9] to calculate the computation cost.

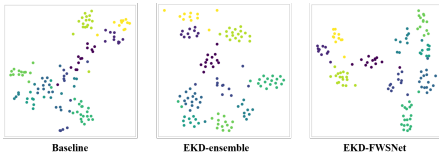


Figure 2: Visualization results of logits on CIFAR-10 with T-SNE. We select ResNet-20 as baseline network. “EKD-ensemble” indicates the ensemble teacher logits of EKD-FWSNet

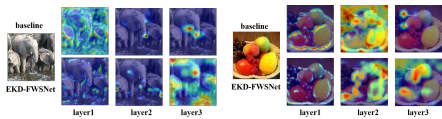


Figure 3: The attention map visualization using Grad-CAM method. We select ResNet-18 as baseline network. In the map, the warmer the color, the higher region attention.

## 4 Visualization and Analysis

**Class probability visualization with T-SNE.** To intuitively show the patterns of predictions, we apply T-SNE [9] on the final logits of networks. In Fig. 2, we compare the performance through the scatter plot generated by network training individually and training with EKD-FWSNet. In scatter plots of EKD-FWSNet, the clusters of each class are tighter (smaller intra-class distance) and distance between clusters are larger (larger inter-class distance). All in all, Fig. 2 shows the effectiveness and interpretability of our proposed EKD-FWSNet.

**Attention region visualization with Grad-CAM.** To show the performance on feature representation of our proposed EKD-FWSNet, we adopt the Grad-CAM [9] to visualize attention region of each layer’s feature map. As shown in Fig. 3, network training with EKD-FWSNet can obtain better attention on different layers. Specifically, EKD-FWSNet training networks can generate less-noise low-level feature maps and richer-information high-level feature maps.

## References

- [1] Zeyuan Allen-Zhu and Yuanzhi Li. Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. *CoRR*, abs/2012.09816, 2020.
- [2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
- [3] Xu Lan, Xiatian Zhu, and Shaogang Gong. Knowledge distillation by on-the-fly native ensemble. In *Advances in Neural Information Processing Systems*, pages 7528–7538, 2018.
- [4] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.
- [5] Selvaraju Ramprasaath R, Cogswell Michael, Das Abhishek, Vedantam Ramakrishna, Parikh Devi, and Batra Dhruv. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.
- [6] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning (ICML)*, volume 97, pages 6105–6114, 2019.
- [7] Devesh Walawalkar, Zhiqiang Shen, and Marios Savvides. Online ensemble model compression using knowledge distillation. In *European Conference on Computer Vision (ECCV)*, volume 12364, pages 18–35, 2020.
- [8] Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. Deep mutual learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 4320–4328, 2018.