Data-Free Knowledge Distillation via Feature Exchange and Activation Region Constraint
– *Supplementary Material* –

In the supplementary material, we provide additional details of our methods and additional analysis of the experimental results. We also discuss the limitations of our method and the potential negative societal impact.

A. Additional Method Details

A.1. Computation of Multi-scale Spatial Activation Region Consistency Constraint

Index	ResNet	Wide ResNet
L0	Conv. + BatchNorm + ReLU (+ Max pool)	Conv.
L1	Stage 1	Stage 1
L2	Stage 2	Stage 2
L3	Stage 3	Stage 3
L4	Stage 4	
L5		BatchNorm + ReLU
L6	Average pool	Average pool
L7	Fully-connected layer	Fully-connected layer

Table A. Simplified representation of the ResNet and Wide ResNet architectures.

Multi-scale spatial activation region consistency (mSARC) constraint is used for knowledge distillation between the student (S) and teacher (T) network in our method:

$$\mathcal{L}_{mSARC} = \mathrm{E}_{\hat{x}\sim p(\hat{x})} \sum_{k=1}^{t} ||CAM_{i_k}(S,\hat{x}) - CAM_{j_k}(T,\hat{x})||_2^2, \tag{A1}$$

where $k = 1, 2, ..., t$, i_k and j_k denote the layer index, $CAM_{i_k}(S,\hat{x})$ denotes the class activation maps (CAMs) [8] of S at layer i_k for $\hat{x}$, and $CAM_{j_k}(T,\hat{x})$ denotes the CAMs of T at layer j_k for $\hat{x}$. So we need to compute the CAMs for both teacher and student networks.

We use two different calculation methods to calculate CAMs of two networks for features from the last layer before the average pooling layer and from shallower layers, respectively. Details of these methods are provided as follows:

(i) For features from the last layer before the average pooling layer (L4 of ResNet and L5 of Wide ResNet in Table A) in S or T with the size of (nc, h, w) (nc is the number of its channel, $h \times w$ is the size of its feature map), we can compute its CAM for class c following [8]:

$$\text{CAM}^c = \sum_p w_p^c \times f_p, \tag{A2}$$

where f_p denotes the feature map of channel p, and w_p^c denotes the weight corresponding to class c for unit p (i.e., the p-th channel of the result of the global average pooling $\frac{1}{Z}\sum_{x,y} f_p(x,y)$) of the last fully-connected layer, which indicates the importance of f_p for class c.

In our implementation, we copy the weights of the last fully-connected layer to a convolutional layer which contains K convolutional kernels of size $1 \times 1 \times nc$. In which, K equals to the number of classes in the dataset, and nc is the number of channels of features before the last pooling layer. A feature of shape (K, h, w) can be obtained after feeding the feature to such a convolutional layer, where the i-th channel of the obtained feature is the CAM for class i. We use the obtained features of the student network and the teacher network to calculate the Eq. A1.

(ii) For features from shallower layers (L1, L2, and L3 in Table A) in S or T, the above method cannot be applied directly because it is designed only for the features from that last layer before the average pooling layer. Inspired by gradient-weighted class activation mapping (Grad-CAM) [7], we compute CAM of the shallower features for class c by

$$\text{CAM}^c = \sum_p \alpha_p^c \times f_p, \tag{A3}$$

where f_p denotes the feature map of channel p, and α_p^c dentoes the importance weights of f_p for class c. α_p^c is obtained by computing the gradient of the one-hot score for class c, y^c (before the softmax), with respect to feature maps f_p, i.e., $\frac{\partial y^c}{\partial f_p}$.

ResNet and VGG contain four convolutional stages of different feature map sizes. Wide ResNet contains three convolutional stages of different feature map sizes. The features used for calculating the CAMs of different networks are as follows: For ResNet and VGG architecture networks, we use the features obtained by their middle four convolutional stages (L1, L2, L3, and L4 of ResNet in Table A). For Wide ResNet architecture networks, we use the features obtained by their middle three convolutional stages and the last layer before the average pooling layer (L1, L2, L3, and L5 of Wide ResNet in Table A).

A.2. More Implementation Details

Teacher Network. The pre-trained teacher networks of ResNet-34, WRN-40-2, and VGG-11 used for CIFAR-10 and CIFAR-100 [4] are from [2]. The pre-trained teacher networks of ResNet-34 used for Tiny-ImageNet [5], Imagenette[1] and ImageNet100[2] are trained by ourselves using image-label pairs in its training set.

Channel-wise Feature Exchange. The detailed architecture of our generative network G is given in Table B. We save the features before the first two upsampling layers and the last convolution layer (marked with *) into the feature pool when optimizing the generative network G, and use these saved features to perform channel-wise feature exchange (CFE). We perform CFE on the sampled features $F_i^1, F_i^2, ..., F_i^n$ and then feed the channel-wise feature exchanged features to layers of G ranging from $i+1$ to m, i.e., $G_{i+1,...,m}$ with a probability $p = 0.7$, where i is the layer index. The features F_i sampled from the feature pool are directly fed to $G_{i+1,...,m}$ with probability $(1-p)$.

Structure	Intput size	Output size
Linear	(256)	$(8HW)$
Reshape	$(8HW)$	$(128, \frac{H}{4}, \frac{W}{4})$
BatchNorm*	$(128, \frac{H}{4}, \frac{W}{4})$	$(128, \frac{H}{4}, \frac{W}{4})$
Upsampling (2)	$(128, \frac{H}{4}, \frac{W}{4})$	$(128, \frac{H}{2}, \frac{W}{2})$
Conv. $(k = 3, s = 1, p = 1)$ + BatchNorm + LeakyReLU* (0.2)	$(128, \frac{H}{2}, \frac{W}{2})$	$(128, \frac{H}{2}, \frac{W}{2})$
Upsampling	$(128, \frac{H}{2}, \frac{W}{2})$	$(128, H, W)$
Conv. $(k = 3, s = 1, p = 1)$ + BatchNorm + LeakyReLU* (0.2)	$(128, H, W)$	$(64, H, W)$
Conv. $(k = 3, s = 1, p = 1)$ + Sigmoid	$(64, H, W)$	$(3, H, W)$

Table B. The detailed architecture of the generative network G in our proposed method. H and W are the height and width of synthetic images.

[1] https://github.com/fastai/imagenette

[2] https://www.kaggle.com/datasets/ambityga/imagenet100

Training Details. The following are the training details of our method by default. We train our networks for max_epoch epochs in total. In each epoch, we first train the generative network and then train the student network. In detail, we first sample a mini-batch of noises and then optimize the generative network for $max\_g\_iterations$ iterations. After that, we store the features corresponding to these noises in a feature pool. Then we randomly sample features from the feature pool to generate synthetic training images to train the student network for $max\_kd\_iterations$ iterations. We set $max\_g\_iterations = 200$, $max\_kd\_iterations = 2000$, and $max\_epoch = 200$.

The generative network is trained by an Adam optimizer [3] with $\{\beta_1 = 0.5, \beta_2 = 0.999\}$ and with a learning rate starting from 0.1 and then gradually decreasing to 0 by cosine annealing scheduler [6]. The student network is trained by a SGD optimizer with $\{lr = 0.1, weight\_decay = 1e-4, momentum = 0.9\}$. The mini-batch size for optimizing the generative network and the student network is 256.

We set $\lambda_{cls} = 0.5$ and $\lambda_{BN} = 1$ in Eq. 2, $\lambda_{KL} = 900$ and $\lambda_{mSARC} = 1$ in Eq. 3, and $\tau = 30$ when computing $\mathcal{L}_{KL}$:

$$\mathcal{L}_{KL} = \mathrm{E}_{x,y\sim p(x,y)} KL(S(x;\theta_S)/\tau || T(x;\theta_T)/\tau) \tag{A4}$$

B. Further Experimental Analysis

B.1. Impact of mSARC on CAMs

Method	(a)		(b)	
	MAE	MSE	MAE	MSE
SpaceshipNet	11.58	696.59	10.78	576.87
SpaceshipNet w/o $\mathcal{L}_{mSARC}$	72.53	12778.76	73.02	12893.86

Table C. Difference of CAMs between the teacher network T and the student networks S trained by our method with and without using $\mathcal{L}_{mSARC}$ for (a) the complete 10,000 images in the CIFAR-10 test set and (b) a subset of 9,474 images in the CIFAR-10 test set that can be correctly classified into their ground-truth category by both student networks trained by our method with and without using $\mathcal{L}_{mSARC}$.

In our main manuscript, we report the impact of multi-scale spatial activation region consistency (mSARC) constraint on CAM on CIFAR-100. Here we report the results on CIFAR-10. We compute the CAM difference between the teacher and student network trained with and without $\mathcal{L}_{mSARC}$ for the 10,000 images in the CIFAR-10 test set. The results are shown in Table C (a). As can be observed, the CAM difference between the teacher network and the student network drastically declines after using $\mathcal{L}_{mSARC}$. We also compute the CAM differences for the 9,474 images in the CIFAR-10 test set that can be correctly classified to their ground-truth category by both the student networks trained by our method and our method without using $\mathcal{L}_{mSARC}$. The result are shown in Table C (b). The CAM difference still drastically declines after using $\mathcal{L}_{mSARC}$. These results positively indicate that mSARC can effectively reduce the CAM difference between the student and teacher network. In other words, it promotes the student network to learn discriminative cues from the same spatial region with the teacher network.

The differences of CAMs are computed as follows: We first calculate the CAMs of both the teacher and student networks using [8]. Next, we resize the obtained CAM tensors to the size of the input image and rescale their values to the range of [0,255]. Finally, we calculate the mean absolute error (MAE) and mean sqaure error (MSE) between the resized and rescaled CAMs of the teacher and student networks.

B.2. Impact of Positions of the Swapped Channels

In this section, we examine the impact of the positions of the swapped channels on the performance of our method. In previous experiments, we randomly swapping 50% channels from F_i^a when using channel-wise feature exchange (CFE) for features F_i^a and features F_i^b. Here, we evaluate the performance of swapping channels of different positions. Specifically, we equally divide the channels of feature into four groups (group 0, group 1, group 2, and group 3), group i contains the channels from positions ranging from $(i-1) \times \frac{nc}{4}$ to $(i \times \frac{nc}{4} - 1)$. We swap different channel groups in feature F_i^a for CFE. The results are shown in Table D. The results show that there is no significant difference in the impact of swapping different channel groups when performing CFE on the test accuracy. Comparing these results with ours by randomly swapping 50% of channels demonstrates that it is required to swap channels of random positions when performing CFE to obtain synthetic training images with better diversity.

Channels from F_i^a				Channels from F_i^b				Test accuracy
group 0 (0-25%)	group 1 (25-50%)	group 2 (50-75%)	group 3 (75-100%)	group 0 (0-25%)	group 1 (25-50%)	group 2 (50-75%)	group 3 (75-100%)	
✓					✓	✓	✓	77.25%
	✓			✓		✓	✓	77.14%
		✓		✓	✓		✓	77.17%
			✓	✓	✓	✓		76.85%
✓	✓					✓	✓	77.17%
		✓	✓	✓	✓			77.15%
	✓	✓		✓			✓	77.16%
✓			✓		✓	✓		77.12%
Randomly swap 50% of channels								77.41%

Table D. The influence of swapping different positions of channels when performing CFE.

B.3. Impact of Number of Optimized Images when Optimizing the Generative Network

Our method optimizes G using a mini-batch of noises z for each epoch. After optimizing G using $\mathcal{L}_G$, G can synthesize a mini-batch of images $\hat{x} = G(z)$ that can be classified to a category by the teacher network T and satisfied the BN constraint of $\mathcal{L}_{BN}$, we term these images as **optimized** images. The features of optimized images of each epoch are stored for distillation. In this section, we examine the impact of the total number of optimized images when optimizing the generative network G in the training process. We conduct experiments on CIFAR-10, and the results are shown in Table E. To better compare, we also report the results of using the same number of real images from the CIFAR-10 training set instead of synthetic images for distillation. We can see that the test accuracies gradually increase when we use more optimized images for distillation. When we use features of only 50 images, the test accuracy achieves 91.23%, surpassing the result of using 50 real images (24.27%) by a large margin. This suggests that our method can improve the diversity of the synthetic images for training the student network.

Methods	Test accuracy (%)						
	Teacher	50 (0.1%)	250 (0.5%)	500 (1%)	1000 (2%)	2500 (5%)	51200 (102.4%)
SpaceshipNet	95.70	91.23	93.84	94.50	94.92	95.21	95.39
Distillation using real images	95.70	24.27	36.48	42.80	61.78	80.73	

Table E. The influence of number of optimized images when optimizing the generative network G on CIFAR-10. The percentages in parentheses denote the ratio of the number of optimized images to the number of images in the CIFAR-10 training set.

C. Limitations and Potential Negative Societal Impact

In this paper, we do not directly evaluate our method on a dataset of the size of ImageNet [1] due to the limitation of computational resources. Instead, we evaluate our method on the image resolution of ImageNet by conducting experiments on Imagenette and ImageNet100.

The datasets we use are all public datasets and contain no people, so it is unlikely to raise IRB or copyright issues.

This paper may involve some negative social implications. For example, the techniques in this paper may be applied to migrate knowledge from unlicensed models, thereby threatening the copyright of model owners without access to the original data. This approach makes it difficult for model owners to protect the parameters of their models deployed at terminal devices and trace the misappropriation. However, this work suggests that models may be at risk of being misappropriated, thus promoting research on model copyright protection. We believe that the positive impact of this work outweighs its negative impact.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009.

[2] Gongfan Fang, Yifan Bao, Jie Song, Xinchao Wang, Donglin Xie, Chengchao Shen, and Mingli Song. Mosaicking to distill: Knowledge distillation from out-of-domain data. *NeurIPS*, 34, 2021.

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[4] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

[5] Ya Le and Xuan Yang. Tiny imagenet visual recongnition challenge. *Technical Report*, 2015.

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

[7] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, pages 618–626, 2017.

[8] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, pages 2921–2929, 2016.