Learning to Weight Samples for Dynamic Early-exiting Networks: Supplementary Materials

The organization of the supplementary material is as follows. We first introduce the detailed experiment setting in Sec. 1, including the datasets, model configuration, hyper-parameter selection and training details. Next, we provide the anytime prediction results of ImageNet in Sec. 2. Finally, the ablation studies of the *weight prediction network* (WPN) design are presented in Sec. 3.

1 Experimental Setup

1.1 Datasets

CIFAR-10 and CIFAR-100 [6] contain 10 and 100 classes of natural scene objects, respectively. Both datasets contain 50,000 training images and 10,000 test images. Each image has a resolution of 32×32 pixels. ImageNet [2] contains 1.2 million training images and 50,000 validation images in 1000 classes. Following [4], we use the basic data augmentation for CIFAR-10, CIFAR-100 and ImageNet. The long-tailed CIFAR dataset [1], unlike CIFAR and ImageNet, does not have a well-balance class distribution. It reduces the number of training samples in each class based on an exponential function $N_c' = N_c\mu^c, \mu \in (0, 1)$, where c is the class index, and N_c is the original training sample number.

1.2 Model Configuration

MSDNet. We follow the model configurations in [3]. For the CIFAR dataset, we use MSDNet with three different scales $(32 \times 32, 16 \times 16, 8 \times 8)$. The trained MSDNets have {5, 7} classifiers, where their depths are {15, 28}, respectively. The k^{th} classifier is attached to the $(\sum_{i=1}^{k} i)^{th}$ layer. On ImageNet, the MSDNet has four scales, with the k^{th} classifier attaching on the $(s \times k + 3)^{th}$ layer $(k = 1, ..., 5)$. The hyper-parameter s controls the total depth and the location of each classifier of the network, and is set as $4, 6$ and 7 to get models with different sizes.

RANet. We use the same model structure proposed in [9]. In the main paper, we conduct experiments on two RANets. They both have three scales but have three and four base features, respectively. Detailed configuration is as follows:

Model-C-1: The sizes of three base features are $32 \times 32, 16 \times 16, 8 \times 8$. Three sub-networks respectively corresponding to the base features have 6, 4, 2 convolutional blocks. We use the linear growth step mode to control the number of layers in each block, which means the number of layers in a block is added 2 to the previous one, and the base number of layers is 2. The channel numbers

in these base features are 16, 32, 64, which are input channels numbers for different sub-networks. The growth rates of the three sub-networks are 6, 12, 24. The Model-C-1 has 6 classifiers in total.

Model-C-2: The sizes of four base features are $32 \times 32, 16 \times 16, 16 \times 16, 8 \times 8$. These four sub-networks respectively corresponding to these base features have 8, 6, 4, 2 convolutional blocks. Linear growth step mode is used to control the depth of each block. The number of input channels and growth rates are 16, 32, 32, 64 and 6, 12, 12, 24, respectively. The Model-C-2 has 8 classifiers in total.

1.3 Hyper-parameter selection

For the budget controlling variable q, we hold out an extra validation set from the training data and tune q on it. We find 0.75 work well for CIFAR, and 0.5, 0.75 work well for ImageNet. The magnitude of perturbation δ (Sec. 3.2, weighted classification loss) is set as 0.8 on CIFAR and 0.3 on ImageNet.

1.4 Training details

We use the stochastic gradient descent (SGD) to train the backbone parameters $\mathbf{\Theta}_f$. The batch sizes are set to 1024 and 2048 for CIFAR and ImageNet, respectively. The learning rate for backbone parameters starts with 0.1×batch size/64 for CIFAR and 0.1×batch size/256 for ImageNet, decaying with a cosine shape. We use a momentum of 0.9 and a weight decay of 1×10^{-4} for both datasets. All the models (with or without our weighting mechanism) are trained for 300 epochs on CIFAR and 100 epochs on ImageNet.

Following [8], we adopt Adam [5] with an initial learning rate of 1×10^{-4} to optimize our WPN $\mathbf{\Theta}_g$. The interval I of updating WPN in Algorithm 1 is set to 1 and 100 for CIFAR and ImageNet, respectively.

In each iteration of our meta-learning algorithm, a mini-batch is chunked into two parts: one is used to train the backbone parameters $\mathbf{\Theta}_f$, while the other part is used for training the WPN parameters $\mathbf{\Theta}_g$. We then exchange the two parts to keep the total number of training iterations equal to the baseline strategy. For experiments of IMTA compatibility, we follow the pretrain-finetuning process in IMTA [7], and our meta-learning procedure is included in both phases.

Experiments on CIFAR-10, CIFAR-100 and long-tailed CIFAR are conducted on Nvidia Geforce RTX 2080 Ti GPUs, and the training for ImageNet models is conducted on Nvidia Tesla A100 GPUs.

2 Anytime prediction results of ImageNet

The ImageNet results in dynamic inference setting have been presented in the paper. In this section, the anytime prediction results of different sized MSDNet are shown in Tab. 1, Tab. 2, Tab. 3, respectively.

Table 1: **Anytime prediction results** of a MSDNet with $k = 4$ on ImageNet

Exit index		1	2	3	4	5
Params ($\times 10^6$)		4.24	8.77	13.07	16.75	23.96
Mul-Add ($\times 10^9$)		0.34	0.69	1.01	1.25	1.36
Accuracy	MSDNet	59.00	66.78	70.12	71.42	72.90
	Ours	**59.54** (↑0.54)	**67.22** (↑0.44)	**71.03** (↑0.91)	**72.33** (↑0.91)	**73.93** (↑1.03)

Table 2: **Anytime prediction results** of a MSDNet with $k = 6$ on ImageNet

Exit index		1	2	3	4	5
Params ($\times 10^6$)		4.24	10.78	17.84	24.58	38.68
Mul-Add ($\times 10^9$)		0.34	0.92	1.52	1.99	2.19
Accuracy	MSDNet	58.69	68.75	72.35	73.59	74.63
	Ours	**60.05** (↑1.36)	**69.13** (↑0.38)	**73.33** (↑0.98)	**75.19** (↑1.60)	**76.30** (↑1.67)

Table 3: **Anytime prediction results** of a MSDNet with $k = 7$ on ImageNet

Exit index		1	2	3	4	5
Params ($\times 10^6$)		4.24	11.89	20.58	29.16	47.54
Mul-Add ($\times 10^9$)		0.34	1.06	1.81	2.42	2.68
Accuracy	MSDNet	58.81	69.45	73.18	74.32	75.35
	Ours	**59.24** (↑0.43)	**69.65** (↑0.20)	**73.94** (↑0.76)	**75.66** (↑1.34)	**76.72** (↑1.37)

Table 4: Ablation study of the WPN size on CIFAR-100.

Exit index		1	2	3	4	5
Multi-Add ($\times 10^6$)		6.86	14.35	27.54	41.71	58.48
Top-1 Acc	Baseline	61.12	63.60	69.00	71.32	72.51
	d=1, w=10	62.06	65.28	70.19	71.47	72.77
	d=1, w=100	63.10	66.21	69.36	71.66	72.98
	d=1, w=500	62.61	65.71	69.14	72.12	73.36
	d=1, w=1000	63.06	66.07	68.60	72.10	73.25
	d=2, w=500	62.95	65.88	69.17	71.56	73.18
	d=4, w=500	63.13	66.09	69.41	71.96	73.06

3 More Ablation Studies

WPN design. For a given k-exit model, the input and output dimensions of the WPN is fixed as k. We conduct ablation studies of the WPN *depth* (the number of hidden layers, d) and *width* (the number of neurons of the hidden layer, w) on CIFAR-100. Note that our default setting in the paper is d=1 and w=500 (for a 5-exit model). The results in Table 4 demonstrate that the performance is relatively stable as long as the WPN is not too small. The gain mainly comes from the overall *meta-learning* paradigm and our novel meta *objective*.

References

1. Cui, Y., Jia, M., Lin, T.Y., Song, Y., Belongie, S.: Class-balanced loss based on effective number of samples. In: CVPR (2019)
2. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
3. Huang, G., Chen, D., Li, T., Wu, F., van der Maaten, L., Weinberger, K.: Multi-scale dense networks for resource efficient image classification. In: ICLR (2018)
4. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR (2017)
5. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)
6. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep. (2009)
7. Li, H., Zhang, H., Qi, X., Yang, R., Huang, G.: Improved techniques for training adaptive deep networks. In: ICCV (2019)
8. Shu, J., Xie, Q., Yi, L., Zhao, Q., Zhou, S., Xu, Z., Meng, D.: Meta-weight-net: Learning an explicit mapping for sample weighting. In: NeurIPS (2019)
9. Yang, L., Han, Y., Chen, X., Song, S., Dai, J., Huang, G.: Resolution Adaptive Networks for Efficient Inference. In: CVPR (2020)