Supplementary Material for Few-Shot Image Recognition by Predicting Parameters from Activations

Appendix

A. Network Architectures for MiniImageNet

In Section 2.4, we briefly present the network architectures we use to train on the MiniImageNet dataset to get the activation function $\mathbf{a}(\cdot)$. In total, we use two architectures for different purposes, and refer them as Ours-Simple and Ours-WRN, as shown in Table 3 of the paper.

Ours-Simple is used to farily compare with the previous state-of-the-art methods on the MiniImageNet dataset. The architecture for $\mathbf{a}(\cdot)$ is the same as that of Matching Network [4], and of the similar complexity with the other methods in comparison. Specifically, the following Table 1 shows the detailed configurations of Ours-Simple.

Group Name	Configuration
Conv1 Trans1	[3×3 Conv@64 + BN + ReLU] [2×2 Max Pooling]
Conv2 Trans2	[3×3 Conv@64 + BN + ReLU] [2×2 Max Pooling]
Conv3 Trans3	[3×3 Conv@64 + BN + ReLU] [2×2 Max Pooling]
Conv4 Trans4	[3×3 Conv@64 + BN + ReLU] [Global Avg Pooling]
FC	[Fully Connected Layer + SoftMax]

Table 1: The network architecture of Ours-Simple.

Ours-WRN aims to achieve better results on the MiniImageNet dataset by increasing the representation capacity of the activation function $\boldsymbol{a}(\cdot)$. We modify WRN-28-10 [6] which is originally designed for CIFAR dataset [3], to adapt to the different input size of the MiniImageNet dataset. The detailed configuration of the modified network architecture is presented in Table 2.

Group Name	Configuration
Conv1	$[3 \times 3, 80]$
Conv2	$\begin{bmatrix} 3 \times 3, 160 \\ 3 \times 3, 160 \end{bmatrix} \times 4$
Conv3	$\begin{bmatrix} 3 \times 3, 320 \\ 3 \times 3, 320 \end{bmatrix} \times 4$
Conv4	$\begin{bmatrix} 3 \times 3, 640 \\ 3 \times 3, 640 \end{bmatrix} \times 4$
Trans1 FC	[Global Avg Pooling] [Fully Connected Layer + SoftMax]

Table 2: The network architecture of Ours-WRN. Downsampling is performed by the first layers in groups Conv2, Conv3 and Conv4, each of which has 4 residual blocks.

B. Results for Low-Shot Setting

We compare our method with SGM [1], Matching Networks [4] and Model Regression [5] following the settings of [1]. The results of these methods are quoted from [1].

Previously in the few-shot setting, the number of examples per category is very limited, *e.g.* 1, 2, 3. In the low-shot setting [1], however, the number of examples per category can be up to 20, far greater than that in the few-shot setting we study in the paper. When we have about 20 images for each category, directly training a linear classifier would already be able to achieve a good accuracy. Fortunately, our proposed method can be easily adapt to this new setting. We made the following minor changes to the model proposed in the paper. First, in Eq. 2 where we minimize the classification loss based on the predicted parameters, we al-

Representation	Low-shot phase	n=1	2	5	10	20
RN10	Model Regression [5]	20.7	39.4	59.6	68.5	73.5
RN10-SGM [1]	With Generation [1]	32.8	46.4	61.7	69.7	73.8
RN10	Matching Network [4]	41.3	51.3	62.1	67.8	71.8
RN10	Ours	**48.5**	**61.0**	**69.8**	**74.2**	**75.6**
RN50-SGM [1]	With Generation [1]	45.1	58.8	72.7	79.1	82.6
RN50	Ours	**58.4**	**69.8**	**77.5**	**82.2**	**83.4**

Table 3: Top-5 accuracy on only novel classes.

Representation	Low-shot phase	n=1	2	5	10	20
RN10	Model Regression [5]	46.4	56.7	66.8	70.4	72.0
RN10-SGM [1]	With Generation [1]	54.3	62.1	71.3	75.8	78.1
RN10	Matching Network [4]	55.0	61.5	69.3	73.4	76.2
RN10	Ours	**62.4**	**70.1**	**75.5**	**78.2**	**78.9**
RN50-SGM [1]	With Generation [1]	63.6	71.5	80.0	83.3	85.2
RN50	Ours	**70.6**	**77.7**	**82.4**	**85.2**	**85.8**

Table 4: Top-5 accuracy on base and novel classes.

low both $\mathcal{D}_{\text{large}}$ and $\mathcal{D}_{\text{few}}$ to be sampled. This makes sure that the parameter predictor can smoothly adapt to the increased numbers of examples in $\mathcal{D}_{\text{few}}$. In our implementation, we uniformly sample images so that each category in both $\mathcal{D}_{\text{large}}$ and $\mathcal{D}_{\text{few}}$ has exactly one element in the training activation set and one element in the statistic set. Second, we use Adam optimizer [2] in training the parameter predictor. Consistent with the improvements over Matching Network [4] shown in MiniImageNet dataset, our method exhibits state-of-the-art performances on the settings of [1].

C. Sensitivity to p_{mean} on MiniImageNet

For the sensitivity study, we run 10 experiments for each different p_{mean} and get the box plots of the accuracies on MiniImageNet as in Fig. 1. The results show that our chosen $p_{\text{mean}} = 0.3$ for MiniImageNet has the maximum accuracy and a small variance.

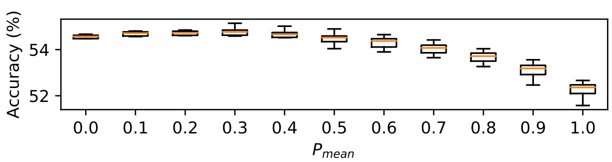

Figure 1: Sensitivity to p_{mean} on MiniImageNet.

References

[1] B. Hariharan and R. B. Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *ICCV*, 2017. 1, 2

[2] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 2

[3] A. Krizhevsky. Learning multiple layers of features from tiny images. In *Masters thesis, Department of Computer Science, University of Toronto*, 2009. 1

[4] O. Vinyals et al. Matching networks for one shot learning. In *NIPS*, 2016. 1, 2

[5] Y. Wang and M. Hebert. Learning to learn: Model regression networks for easy small sample learning. In *ECCV*, 2016. 1

[6] S. Zagoruyko and N. Komodakis. Wide residual networks. In *BMVC*, 2016. 1