A Multi-Mode Modulator for Multi-Domain Few-Shot Classification Supplementary Material

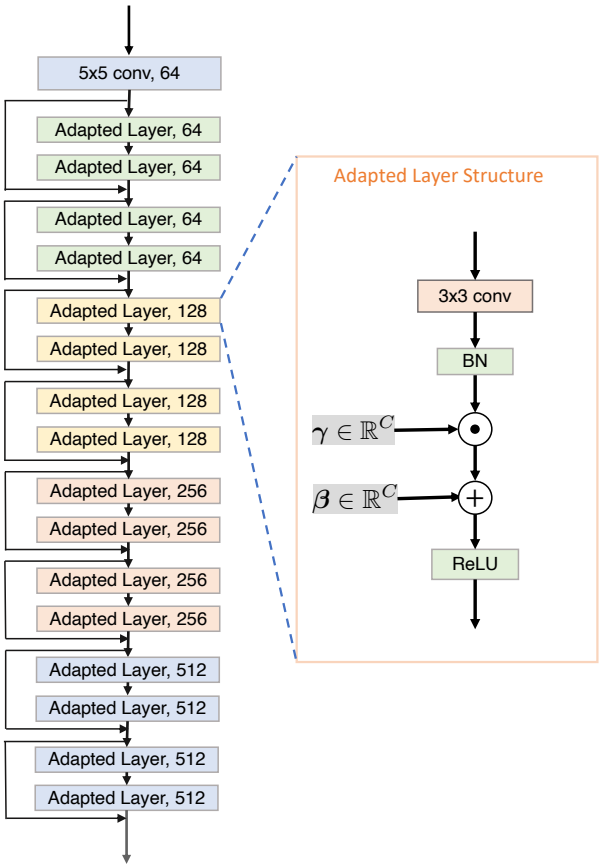

Figure 1. The ResNet18 structure and modulation applied on each convolutional layer.

1. Network Structure

1.1. ResNet18 structure and inserted modulation

The ResNet18 structure and adaptation applied on each convolutional layer is shown in Figure 1. The ResNet18 has a 5×5 convolutional pre-processing layer and 4 consecutive layer groups. Each layer group has 2 blocks with each block having 2 convolutional layers. The number of filters in the layer groups are $[64, 128, 256, 512]$. The detailed structure of each adapted layer is shown in Figure 1 (Right).

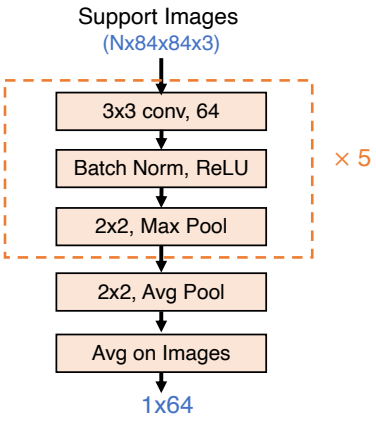

Figure 2. Detailed structure of the Lightweight Task Network.

1.2. Task Network

The structure of the task network is shown in Figure 2. The input are support images with resolution 84×84, the final output is a domain descriptor of dimension 1×64. Note that only five convolutional layers each with 64 filters contain learnable weights, which means that our task network is lightweight and efficient.

2. Classifier and Training Details

2.1. Classifier Details

In the main paper, we generally describe the Mahalanobis classifier in our method. Here, we show the details.

We first compute the adapted features for the support set, $\{\boldsymbol{z}_s\}_{s=1}^N = f_\theta(\{\boldsymbol{x}_s\}_{s=1}^N; \{\boldsymbol{\gamma}_\ell, \boldsymbol{\beta}_\ell\}_{\ell=1}^L)$. Then for each class, we compute the class mean $\boldsymbol{\mu}_k$ and regularized covariance matrix $\boldsymbol{Q}_k$ as

$$\boldsymbol{\mu}_k = \frac{1}{n_k} \sum_i \mathbb{I}[y_i = k] \boldsymbol{z}_i \,,$$

$$\boldsymbol{Q}_k = \lambda_k \boldsymbol{\Sigma}_k + (1 - \lambda_k) \boldsymbol{\Sigma} + \beta \boldsymbol{I}, \quad \lambda_k = \frac{n_k}{n_k + 1} \,.$$

Here, $\mathbb{I}[y_i = k]$ is the indicator function and $n_k = \sum_i \mathbb{I}[y_i = k]$ is the number of samples in class k of the support set $\mathcal{S}$. λ_k is a ratio to balance the task covariance $\boldsymbol{\Sigma}$

and the class covariance $\boldsymbol{\Sigma}_k$:

$$\boldsymbol{\Sigma} = \frac{1}{n}\sum_i (\boldsymbol{z}_i - \boldsymbol{\mu})(\boldsymbol{z}_i - \boldsymbol{\mu})^T ,$$

$$\boldsymbol{\Sigma}_k = \frac{1}{n_k}\sum_i \mathbb{I}[y_i = k](\boldsymbol{z}_i - \boldsymbol{\mu}_k)(\boldsymbol{z}_i - \boldsymbol{\mu}_k)^T ,$$

where $\boldsymbol{\mu} = \frac{1}{n}\sum_i \boldsymbol{z}_i$ is the task-mean. Given a query feature $\boldsymbol{z}_q = f_\theta(\boldsymbol{x}_q; \{\boldsymbol{\gamma}_\ell, \boldsymbol{\beta}_\ell\}_{\ell=1}^L)$, the class probability is constructed as

$$p(y_q = k|\boldsymbol{x}_q) \propto \exp(-(\boldsymbol{z}_q - \boldsymbol{\mu}_k)^T \boldsymbol{Q}_k^{-1}(\boldsymbol{z}_q - \boldsymbol{\mu}_k)) .$$

2.2. Loss Function

To train the network, we need to maximize the log-likelihood of query examples, *i.e.*, minimize the classification loss:

$$L_{\text{classification}} = \sum_{D\sim D_{\text{tr}}} \sum_{(\mathcal{S},\mathcal{Q})\sim D} \sum_{(\boldsymbol{x}_q, y_q)\sim Q} -\log p(y_q|\boldsymbol{x}_q)) . \tag{1}$$

This is consistent with the problem definition in Section 3 of the main paper. At first, a dataset D is randomly selected from the training datasets $D_{\text{tr}} = \{D_1, D_2, \ldots, D_n\}$. Then, the task $(\mathcal{S}, \mathcal{Q})$ is sampled from dataset D. Finally, $(\boldsymbol{x}_q, y_q)$ are sampled from the query set $\mathcal{Q}$ to compute the classification loss as the negative log-likelihood of all query examples.

With the domain classification loss (Eq. 3 in Section 4.2.2 of the main paper) added, the final loss is

$$L = L_{\text{classification}} + L_{\text{domain}} . \tag{2}$$

3. More Experiments

3.1. The effectiveness of hard-gating

To study the effectiveness of the hard-gating mechanism for parameter selection, we implement a model variant that removes the domain classification loss (*i.e.*, in Eq. 3 of main paper, $\lambda = 0$ or equivalently $L_{\text{domain}} = 0$) and employs a soft-gating mechanism. This variant is named *Spec(soft)*. As shown in Table 1, the soft-gating model *Spec(soft)* performs much worse than hard-gating model *Spec(hard)* with the same number of learnable parameters. This verifies the importance of hard-gating with the domain classification loss to perform the domain-level supervision.

3.2. Parameter fusion

In the main paper, after we obtained the domain-specific parameters $(\boldsymbol{\gamma}^g, \boldsymbol{\beta}^g)$ with selection and got the domain-cooperative parameters $(\boldsymbol{\gamma}^a, \boldsymbol{\beta}^a)$ with attention, we formulate the parameter fusion as follows

$$\boldsymbol{\gamma} = \boldsymbol{\alpha}\boldsymbol{\gamma}^g + (1 - \boldsymbol{\alpha})\boldsymbol{\gamma}^a \tag{3}$$

$$\boldsymbol{\beta} = \boldsymbol{\alpha}\boldsymbol{\beta}^g + (1 - \boldsymbol{\alpha})\boldsymbol{\beta}^a . \tag{4}$$

Table 1. Effect of the hard-gating mechanism

Dataset	*Spec(soft)*	*Spec(hard)*
ImageNet	55.2	**55.6**
Omniglot	83.8	**88.2**
Aircraft	77.0	**82.1**
Birds	72.1	**73.3**
Textures	**72.4**	68.2
QuickDraw	71.3	**75.1**
Fungi	41.8	**48.4**
VGGFlower	**89.5**	85.9
TrafficSigns	73.2	**74.4**
MSCOCO	49.2	**53.2**
MNIST	93.3	**94.8**
CIFAR10	70.9	**73.0**
CIFAR100	56.0	**61.9**
In-Domain Avg	70.4	**72.1**
Out-of-Domain Avg	68.5	**71.5**
Overall Avg	69.7	**71.9**
Learnable Parameters	0.22M	0.22M

Table 2. Comparisons of different fusion strategies.

Dataset	*Average*	*Weighted*	*Channel*
ImageNet	57.4	58.5	**58.6**
Omniglot	91.5	91.4	**92.0**
Aircraft	82.3	81.9	**82.8**
Birds	74.8	**75.4**	75.3
Textures	71.0	69.6	**71.2**
QuickDraw	77.7	**78.1**	77.3
Fungi	48.3	**48.5**	**48.5**
VGGFlower	90.3	**90.6**	90.5
TrafficSigns	77.9	77.3	**78.0**
MSCOCO	52.2	**53.0**	52.8
MNIST	95.4	95.2	**96.2**
CIFAR10	75.1	74.7	**75.4**
CIFAR100	61.9	61.7	**62.0**
In-Domain Avg	74.2	74.3	**74.5**
Out-of-Domain Avg	72.5	72.4	**72.9**
Overall Avg	73.5	73.5	**73.9**
Learnable Parameters	7.71M	7.72M	7.78M

Here, $\boldsymbol{\alpha} \in \mathbb{R}^{1\times C}$ is the channel-wise fusion ratio computed as $\boldsymbol{\alpha} = \text{sigmoid}(\boldsymbol{V}_{\mathcal{S}}\boldsymbol{W}^f + \boldsymbol{b}^f)$.

To investigate the effectiveness of this channel-wise fusion strategy, we implement two simple fusion strategies: *Average* and *Weighted*. For *Average*, $\alpha = 0.5$ is the naïve average fusion ratio. For *Weighted*, $\boldsymbol{\alpha} \in \mathbb{R}^{1\times 1}$ is computed as $\boldsymbol{\alpha} = \text{sigmoid}(\boldsymbol{V}_{\mathcal{S}}\boldsymbol{W}^{f_1} + \boldsymbol{b}^{f_1})$. From Table 2, we can see that the proposed channel-wise fusion achieved the highest performance among all three fusion strategies, showing the effectiveness of channel-wise fusion. Moreover, *Channel* only introduces 0.06M additional learnable parameters, which is negligible.

To further explore how this channel-wise fusion strategy influence the performance, we visualize the fusion ratio $\boldsymbol{\alpha}$ in the first layer group (64-dim). Figure 3 indicates that diverse fusion is applied for various datasets and dimensions. The yellow areas indicate high fusion ratios (*i.e.*, large $\boldsymbol{\alpha}$) on the *domain-specific* parameters, while dark areas indicate high fusion ratio (*i.e.*, large $1 - \boldsymbol{\alpha}$) on the *domain-cooperative* parameters. Omniglot, QuickDraw and MNIST

Table 3. Comparison to the state-of-the-art methods on META-DATASET. Due to the shuffling issue[2], Meta-Dataset updated the evaluation on TrafficSigns. Therefore, We report the updated accuracy of all methods on TrafficSigns (i.e. 63.0 ± 1.0 for *tri*-M) here.

Dataset	ProtoMAML [5]	AR-CNAPS [4]	TaskNorm [2]	SimpleCNAPS [1]	SUR-pf [3]	SUR [3]	*tri*-M (Ours)
ImageNet	47.9±1.1	52.3±1.0	50.6±1.1	**58.6±1.1**	56.4±1.2	56.3±1.1	**58.6±1.0**
Omniglot	82.9±0.9	88.4±0.7	90.7±0.6	91.7±0.6	88.5±0.8	**93.1±0.5**	**92.0±0.6**
Aircraft	74.2±0.8	80.5±0.6	83.8±0.6	82.4±0.7	79.5±0.8	**85.4±0.7**	82.8±0.7
Birds	70.0±1.0	72.2±0.9	74.6±0.8	**74.9±0.8**	**76.4±0.9**	71.4±1.0	**75.3±0.8**
Textures	67.9±0.8	58.3±0.7	62.1±0.7	67.8±0.8	**73.1±0.7**	71.5±0.8	71.2±0.8
QuickDraw	66.6±0.9	72.5±0.8	74.8±0.7	77.7±0.7	75.7±0.7	**81.3±0.6**	77.3±0.7
Fungi	42.0±1.1	47.4±1.0	48.7±1.0	46.9±1.0	48.2±0.9	**63.1±1.0**	48.5±1.0
VGGFlower	88.5±0.7	86.0±0.5	**89.6±0.6**	**90.7±0.5**	**90.6±0.5**	82.8 ±0.7	**90.5±0.5**
TrafficSigns	52.4±1.1	56.5±1.1	56.5±1.1	59.2±1.0	52.2±0.8	53.4±1.0	**63.0±1.0**
MSCOCO	41.3±1.0	42.6±1.1	43.4±1.0	46.2±1.1	**52.1±1.0**	**52.4±1.1**	**52.8±1.1**
MNIST	NA	92.7±0.4	92.3±0.4	93.9±0.4	93.2±0.4	94.3±0.4	**96.2±0.3**
CIFAR10	NA	61.5±0.7	69.3±0.8	**74.3±0.7**	66.4±0.8	66.8±0.9	**75.4±0.8**
CIFAR100	NA	50.1±1.0	54.6±1.1	**60.5±1.0**	57.1±1.0	56.6±1.0	**62.0±1.0**
In-Domain Avg	67.5	69.7	71.9	73.8	73.6	**75.6**	74.5
Out-of-Domain Avg	46.8	60.7	63.2	66.8	64.2	64.7	**69.9**
Overall Avg	63.4	66.2	68.5	71.1	70.0	71.4	**72.7**
Learnable Parameters	10.49M	13.4M	9.39M	8.60M	1.67M	79.45M	7.78M
Forward Pass	1	1	1	1	8	8	1

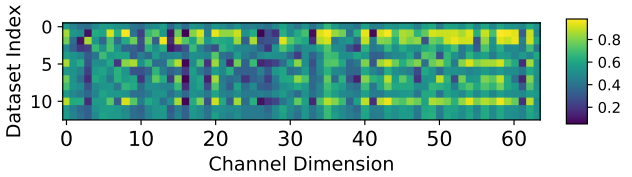

Figure 3. The channel-wise fusion ratio on all datasets. Y-axis represents [*ImageNet, Omniglot, Aircraft, Birds, Textures, QuickDraw, Fungi, VGGFlower, TrafficSings, MSCOCO, MNIST, CIFAR10, CIFAR100*] ranging from 0 to 12. X-axis represents channel dimension.

have high fusion ratios on the domain-specific parameters since their features are far from the backbone (pre-trained on ImageNet) and require specialized and separated feature modulation. On the contrary, ImageNet, CIFAR10, and CIFAR100 have high ratios on the domain-cooperative parameters, since these datasets are visually similar to the pre-trained backbone and try to explore more correlations among datasets.

3.3. Other Comparison Experiments

SUR [3] with Mahalanobis distance. In original SUR paper, the authors utilize an Euclidean distance for classification. However, recently SimpleCNAPs [1] shows that the second-order classifier (i.e. Mahalanobis distance) performs better than Euclidean distance. Therefore, we follow SimpleCNAPs to utilize the Mahalanobis classifier.

In this situation, one would wonder the performance of SUR with Mahalanobis distance. To answer this question, we equip the original SUR implementation with Mahalanobis classifier. Moreover, since the feature dimension of SUR is 4096[3], it requires computing a covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{4096\times4096}$ and its inverse, which is time-consuming and unstable to train. Thus, we implement a fast variant by only considering the diagonals of the covariance matrix. As shown in Table 4, SUR variants with both the full and diagonal Mahalanobis classifiers are much slower and less accurate than ours. We conjecture that especially under the few-shot setting, estimating the high-dimensional covariance matrix become infeasible.

Table 4. Comparisons of different methods with Mahalanobis classifier. "Diag" denotes the method only considering diagonals of the covariance matrix.

	Classifier	Dim	Overall Acc	Infer time
SUR	Mahalanobis	4096	68.92	67.2s/task
SUR	Diag Mahalanobis	4096	69.91	16.2s/task
SimpleCNAPs	Mahalanobis	512	72.20	0.44s/task
Ours	Mahalanobis	512	**73.90**	0.41s/task

3.4. Updated Results for Shuffled TrafficSigns

Due to the shuffling issue reported at the Meta-Dataset repo, Meta-Dataset recommends a new evaluation protocol for TrafficSigns with shuffling at test time. Here, we report the updated results for our method and the baselines in Table 3.

References

[1] Peyman Bateni, Raghav Goyal, Vaden Masrani, Frank Wood, and Leonid Sigal. Improved few-shot visual classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14493–14502, 2020. 3

[2] https://github.com/google-research/meta-dataset/issues/54

[3] SUR concatenates the features extracted from all domain networks.

[2] John Bronskill, Jonathan Gordon, James Requeima, Sebastian Nowozin, and Richard E Turner. Tasknorm: Rethinking batch normalization for meta-learning. *arXiv preprint arXiv:2003.03284*, 2020. 3

[3] Nikita Dvornik, Cordelia Schmid, and Julien Mairal. Selecting relevant features from a multi-domain representation for few-shot classification. In *European Conference on Computer Vision (ECCV)*, 2020. 3

[4] James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. In *Advances in Neural Information Processing Systems*, pages 7959–7970, 2019. 3

[5] Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, et al. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International Conference on Learning Representations*, 2019. 3