Supplementary Material: Cross-domain Few-shot Learning with Task-specific Adapters

Wei-Hong Li, Xialei Liu*, and Hakan Bilen
VICO Group, University of Edinburgh, United Kingdom
github.com/VICO-UoE/URL

1. Dataset

Meta-Dataset [22] is a few-shot classification benchmark that initially consists of ten datasets: ILSVRC_2012 [20] (ImageNet), Omniglot [11], FGVC-Aircraft [17] (Aircraft), CUB-200-2011 [23] (Birds), Describable Textures [4] (DTD), QuickDraw [9], FGVCx Fungi [3] (Fungi), VGG Flower [18] (Flower), Traffic Signs [8] and MSCOCO [14] then further expands with MNIST [12], CIFAR-10 [10] and CIFAR-100 [10]. We follow the standard procedure in [22] and consider both the 'Training on all datasets' (multi-domain learning) and 'Training on ImageNet only' (single-domain learning) settings. In 'Training on all datasets' setting, we follow the standard procedure and use the first eight datasets for meta-training, in which each dataset is further divided into train, validation and test set with disjoint classes. While the evaluation within these datasets is used to measure the generalization ability in the seen domains, the remaining five datasets are reserved as unseen domains in meta-test for measuring the cross-domain generalization ability. In 'Training on ImageNet only' setting, we follow the standard procedure and only use train split of ImageNet for meta-training. The evaluation of models is in the test split of ImageNet and the rest 12 datasets which are reserved as unseen domains in meta-test. As in [22], we evaluate our method on 600 randomly sampled tasks for each dataset with varying number of ways and shots, and report average accuracy and 95% confidence score in all experiments.

2. Implementation details

In this section, we explain the details of task-agnostic (feature extractor) learning and then task-specific (adapter) learning.

2.1. Task-agnostic learning

Here we consider learning the parameters of the feature extractor from either multiple or single domains.

*Xialei Liu is the corresponding author.

Multi-domain learning. When we learn the feature extractor from multiple domains, we consider two cases. In the first case, which we call vanilla multiple domain learning (or MDL), we design a deep network where we share all the layers across all domains and have domain-specific classifiers. This setting corresponds to Eq (1) in the main text. Second we consider a variant of MDL, URL [13] which also involves learning a single network with shared and domain-specific layers as such, however, it is learned by distilling information from multiple domain-specific networks as described [13]. In these two settings, as in [2, 6, 13], we build MDL and URL on the ResNet-18 [7] backbone and use 84×84 image size.

For optimization of both MDL and URL, we follow the same protocol in [13], use SGD optimizer and cosine annealing with a weight decay of 7×10^{-4} for learning 240,000 iterations. The learning rate is 0.03 and the annealing frequency is 48,000. As in [13], the batch size for ImageNet is 64×7 and is 64 for the other 7 datasets. We refer readers to [13] for more details.

Single domain learning (SDL). We also evaluate our method on a feature extractor that is learned on single domain which we call SDL. Here we evaluate our method on two backbones, ResNet-18 (SDL-ResNet-18) and ResNet34 (SDL-ResNet-34).

Backbone	learning rate	batch size	annealing freq.	max. iter.
SDL-ResNet-18	3×10^{-2}	64	48,000	480,000
SDL-ResNet-34	3×10^{-2}	128	48,000	480,000

Table 1. Training hyper-parameters of single domain learning.

SDL-ResNet-18. Following [6, 13, 22], we train a ResNet-18 on the train split of ImageNet and use 84×84 image size, which is denoted as SDL-ResNet-18. For optimization, we follow the training protocol in [6, 13]. Specifically, we use SGD optimizer and cosine annealing for all experiments with a momentum of 0.9 and a weight decay of 7×10^{-4}. Some other hyperparameters are shown in Tab. 1 as in [6, 13]. To regularize training, we also use the exact same data augmentations as in [6, 13], *e.g.* random crops and random color augmentations.

Test Dataset	ImageNet	Omniglot	Aircraft	Birds	Textures	Quick Draw	Fungi	VGG Flower	Traffic Sign	MSCOCO	MNIST	CIFAR-10	CIFAR-100
MDL	53.4 ± 1.1	93.8 ± 0.4	86.6 ± 0.5	78.6 ± 0.8	71.4 ± 0.7	81.5 ± 0.6	61.9 ± 1.0	88.7 ± 0.6	51.0 ± 1.0	49.7 ± 1.1	94.4 ± 0.3	66.7 ± 0.8	53.6 ± 1.0
Ours (MDL)	$\mathbf{55.6 \pm 1.0}$	$\mathbf{94.3 \pm 0.4}$	$\mathbf{86.7 \pm 0.5}$	$\mathbf{79.4 \pm 0.8}$	$\mathbf{73.2 \pm 0.8}$	$\mathbf{81.7 \pm 0.6}$	$\mathbf{64.0 \pm 0.9}$	$\mathbf{90.9 \pm 0.5}$	$\mathbf{81.1 \pm 0.9}$	$\mathbf{51.4 \pm 1.1}$	$\mathbf{96.9 \pm 0.3}$	$\mathbf{78.5 \pm 0.8}$	$\mathbf{64.3 \pm 1.1}$
URL [13]	58.8 ± 1.1	94.5 ± 0.4	89.4 ± 0.4	80.7 ± 0.8	77.2 ± 0.7	$\mathbf{82.5 \pm 0.6}$	$\mathbf{68.1 \pm 0.9}$	92.0 ± 0.5	63.3 ± 1.2	57.3 ± 1.0	94.7 ± 0.4	74.2 ± 0.8	63.6 ± 1.0
Ours (URL)	$\mathbf{59.5 \pm 1.0}$	$\mathbf{94.9 \pm 0.4}$	$\mathbf{89.9 \pm 0.4}$	$\mathbf{81.1 \pm 0.8}$	$\mathbf{77.5 \pm 0.7}$	81.7 ± 0.6	66.3 ± 0.9	$\mathbf{92.2 \pm 0.5}$	$\mathbf{82.8 \pm 1.0}$	$\mathbf{57.6 \pm 1.0}$	$\mathbf{96.7 \pm 0.4}$	$\mathbf{82.9 \pm 0.7}$	$\mathbf{70.4 \pm 1.0}$
SDL-ResNet-18	55.8 ± 1.0	67.4 ± 1.2	49.5 ± 0.9	71.2 ± 0.9	73.0 ± 0.6	53.9 ± 1.0	41.6 ± 1.0	87.0 ± 0.6	47.4 ± 1.1	53.5 ± 1.0	78.1 ± 0.7	67.3 ± 0.8	56.6 ± 0.9
Ours (SDL-ResNet-18)	$\mathbf{59.5 \pm 1.1}$	$\mathbf{78.2 \pm 1.2}$	$\mathbf{72.2 \pm 1.0}$	$\mathbf{74.9 \pm 0.9}$	$\mathbf{77.3 \pm 0.7}$	$\mathbf{67.6 \pm 0.9}$	$\mathbf{44.7 \pm 1.0}$	$\mathbf{90.9 \pm 0.6}$	$\mathbf{82.5 \pm 0.8}$	$\mathbf{59.0 \pm 1.0}$	$\mathbf{93.9 \pm 0.6}$	$\mathbf{82.1 \pm 0.7}$	$\mathbf{70.7 \pm 0.9}$
SDL-ResNet-34	62.2 ± 1.1	72.8 ± 1.1	62.9 ± 0.9	79.6 ± 0.8	75.6 ± 0.6	64.5 ± 0.8	47.4 ± 1.1	90.4 ± 0.6	54.8 ± 1.0	56.1 ± 1.0	79.3 ± 0.6	83.0 ± 0.6	74.8 ± 0.8
Ours (SDL-ResNet-34)	$\mathbf{63.7 \pm 1.0}$	$\mathbf{82.6 \pm 1.1}$	$\mathbf{80.1 \pm 1.0}$	$\mathbf{83.4 \pm 0.8}$	$\mathbf{79.6 \pm 0.7}$	$\mathbf{71.0 \pm 0.8}$	$\mathbf{51.4 \pm 1.2}$	$\mathbf{94.0 \pm 0.5}$	$\mathbf{81.7 \pm 0.9}$	$\mathbf{61.7 \pm 0.9}$	$\mathbf{94.6 \pm 0.5}$	$\mathbf{86.0 \pm 0.6}$	$\mathbf{78.3 \pm 0.8}$

Table 2. Results of attaching residual adapters to different baselines. 'SDL-ResNet-18' is the single domain model with ResNet-18 backbone pretrained on ImageNet. 'SDL-ResNet-34' is the single domain model with ResNet-34 backbone pretrained on ImageNet. 'MDL' is a vanilla Multi-Domain Learning (MDL) model trained on eight seen datasets jointly.

Test Dataset	classifier	Aux-Net or Ad	serial or parallel	M or CW	β	#params	ImageNet	Omniglot	Aircraft	Birds	Textures	Quick Draw	Fungi	VGG Flower	Traffic Sign	MSCOCO	MNIST	CIFAR-10	CIFAR-100
NCC	NCC	-	-	-	✗	-	57.0 ± 1.1	94.4 ± 0.4	88.0 ± 0.5	80.3 ± 0.7	74.6 ± 0.7	81.8 ± 0.6	66.2 ± 0.9	91.5 ± 0.5	49.8 ± 1.1	54.1 ± 1.0	91.1 ± 0.4	70.6 ± 0.7	59.1 ± 1.0
MD	MD	-	-	-	✗	-	53.9 ± 1.0	93.8 ± 0.5	87.6 ± 0.5	78.3 ± 0.7	73.7 ± 0.7	80.9 ± 0.7	57.7 ± 0.9	89.7 ± 0.6	62.2 ± 1.1	48.5 ± 1.0	95.1 ± 0.4	68.9 ± 0.8	60.0 ± 0.9
LR	LR	-	-	-	✗	-	56.0 ± 1.1	93.7 ± 0.5	88.3 ± 0.6	79.7 ± 0.8	74.7 ± 0.7	80.0 ± 0.7	62.1 ± 0.8	91.1 ± 0.5	59.7 ± 1.1	51.2 ± 1.1	93.5 ± 0.5	73.1 ± 0.8	60.1 ± 1.1
SVM	SVM	-	-	-	✗	-	54.5 ± 1.1	94.3 ± 0.5	87.7 ± 0.5	78.1 ± 0.8	73.8 ± 0.8	80.0 ± 0.6	58.5 ± 0.9	91.4 ± 0.6	65.7 ± 1.2	50.5 ± 1.0	95.4 ± 0.4	72.0 ± 0.8	60.5 ± 1.1
Softmax	Softmax	-	-	-	✗	-	42.2 ± 1.0	85.3 ± 0.7	71.9 ± 0.8	59.6 ± 1.0	62.0 ± 0.8	61.2 ± 1.0	37.3 ± 0.9	66.7 ± 1.0	51.4 ± 1.1	48.2 ± 1.1	93.5 ± 0.5	70.4 ± 0.8	59.3 ± 1.0
KNN	KNN	-	-	-	✗	-	48.1 ± 1.1	94.1 ± 0.4	84.5 ± 0.6	70.7 ± 0.8	65.9 ± 0.8	74.8 ± 0.7	53.5 ± 0.9	86.0 ± 0.6	56.9 ± 1.2	44.7 ± 1.1	91.4 ± 0.5	60.3 ± 0.8	49.4 ± 1.0
PA	NCC	-	-	-	✓	-	58.8 ± 1.1	94.5 ± 0.4	89.4 ± 0.4	80.7 ± 0.8	77.2 ± 0.7	$\mathbf{82.5 \pm 0.6}$	$\mathbf{68.1 \pm 0.9}$	92.0 ± 0.5	63.3 ± 1.1	57.3 ± 1.0	94.7 ± 0.4	74.2 ± 0.8	63.5 ± 1.0
PA	Softmax	-	-	-	✓	-	53.4 ± 1.2	92.7 ± 0.5	85.7 ± 0.6	76.1 ± 0.9	73.9 ± 0.8	76.5 ± 0.8	51.1 ± 0.9	86.9 ± 0.7	52.5 ± 1.1	48.2 ± 1.1	94.3 ± 0.4	69.7 ± 0.8	60.4 ± 1.0
Finetune	NCC	-	-	-	✗	-	55.9 ± 1.2	94.0 ± 0.5	87.3 ± 0.6	77.8 ± 0.9	76.8 ± 0.8	75.3 ± 0.9	57.6 ± 1.1	91.5 ± 0.6	$\mathbf{86.1 \pm 0.9}$	53.1 ± 1.2	$\mathbf{96.8 \pm 0.4}$	80.9 ± 0.8	65.9 ± 1.1
Finetune	Softmax	-	-	-	✗	-	48.4 ± 1.2	92.2 ± 0.6	81.6 ± 0.9	70.3 ± 1.3	72.0 ± 0.9	73.5 ± 1.0	44.2 ± 1.1	90.3 ± 0.7	65.5 ± 1.4	41.0 ± 1.3	96.3 ± 0.4	71.6 ± 1.0	53.8 ± 1.4
Aux-S-CW	NCC	Aux-Net	serial	CW	✗	-	54.6 ± 1.1	93.5 ± 0.5	86.6 ± 0.5	78.6 ± 0.8	71.5 ± 0.7	79.3 ± 0.6	66.0 ± 0.9	87.6 ± 0.6	43.3 ± 0.9	49.1 ± 1.0	87.9 ± 0.5	62.8 ± 0.8	51.5 ± 1.0
Aux-R-CW	NCC	Aux-Net	residual	CW	✗	-	56.1 ± 1.1	94.2 ± 0.4	88.4 ± 0.5	80.6 ± 0.7	74.9 ± 0.6	82.0 ± 0.6	66.4 ± 0.9	91.6 ± 0.5	48.5 ± 1.0	53.5 ± 1.0	90.8 ± 0.5	70.2 ± 0.8	59.7 ± 1.0
Aux-S-CW	MD	Aux-Net	serial	CW	✗	-	55.1 ± 1.1	93.8 ± 0.5	86.8 ± 0.5	77.4 ± 0.8	73.2 ± 0.8	79.9 ± 0.7	57.4 ± 0.9	88.1 ± 0.7	58.4 ± 1.1	50.1 ± 1.1	92.7 ± 0.5	66.5 ± 0.8	55.7 ± 1.1
Aux-R-CW	MD	Aux-Net	residual	CW	✗	-	54.8 ± 1.1	93.8 ± 0.5	87.4 ± 0.5	78.2 ± 0.7	73.4 ± 0.7	81.1 ± 0.7	58.8 ± 0.9	90.1 ± 0.5	63.6 ± 1.2	48.5 ± 1.1	94.8 ± 0.4	69.6 ± 0.8	60.6 ± 0.9
Ad-S-CW	NCC	Ad	serial	CW	✗	0.06%	56.8 ± 1.1	94.8 ± 0.4	89.3 ± 0.5	80.7 ± 0.7	74.5 ± 0.7	81.6 ± 0.6	65.8 ± 0.9	91.3 ± 0.5	73.9 ± 1.1	53.6 ± 1.1	95.7 ± 0.4	78.4 ± 0.7	64.3 ± 1.0
Ad-R-CW	NCC	Ad	residual	CW	✗	1.57%	57.6 ± 1.1	94.7 ± 0.4	89.0 ± 0.4	81.2 ± 0.8	75.2 ± 0.7	81.5 ± 0.6	65.4 ± 0.8	91.8 ± 0.5	79.2 ± 1.1	54.7 ± 1.1	96.4 ± 0.4	79.5 ± 0.8	67.4 ± 1.0
Ad-S-M	NCC	Ad	serial	M	✗	12.50%	56.2 ± 1.1	94.4 ± 0.4	89.1 ± 0.5	80.6 ± 0.7	75.8 ± 0.7	81.6 ± 0.6	67.1 ± 0.9	92.1 ± 0.4	67.6 ± 1.2	54.8 ± 1.1	95.9 ± 0.4	78.9 ± 0.7	66.6 ± 1.1
Ad-R-M	NCC	Ad	residual	M	✗	10.93%	57.3 ± 1.1	94.9 ± 0.4	88.9 ± 0.5	81.0 ± 0.7	76.7 ± 0.7	80.6 ± 0.6	65.4 ± 0.9	91.4 ± 0.5	82.6 ± 1.0	55.0 ± 1.1	96.6 ± 0.4	82.1 ± 0.7	66.4 ± 1.1
Ad-R-CW-PA	NCC	Ad	residual	CW	✓	3.91%	58.6 ± 1.1	94.5 ± 0.4	$\mathbf{90.0 \pm 0.4}$	80.5 ± 0.8	$\mathbf{77.6 \pm 0.7}$	81.9 ± 0.6	67.0 ± 0.9	92.2 ± 0.5	80.2 ± 0.9	57.2 ± 1.0	96.1 ± 0.4	81.5 ± 0.8	$\mathbf{71.4 \pm 0.9}$
Ad-R-M-PA	NCC	Ad	residual	M	✓	13.27%	$\mathbf{59.5 \pm 1.0}$	$\mathbf{94.9 \pm 0.4}$	89.9 ± 0.4	$\mathbf{81.1 \pm 0.8}$	77.5 ± 0.7	81.7 ± 0.6	66.3 ± 0.9	$\mathbf{92.2 \pm 0.5}$	82.8 ± 1.0	$\mathbf{57.6 \pm 1.0}$	96.7 ± 0.4	$\mathbf{82.9 \pm 0.7}$	70.4 ± 1.0

Table 3. Comparisons to methods that learn classifiers and model adaptation methods during meta-test stage based on URL model. NCC, MD, LR, SVM, Softmax, KNN denote nearest centroid classifier, Mahalanobis distance, logistic regression, support vector machines, softmax classifier and k-nearest neighbors classifier respectively. PA indicates pre-classifier alignment. 'Aux-Net or Ad' indicates using Auxiliary Network to predict α or attaching adapter α directly. 'M or CW' means using matrix multiplication or channel-wise scaling adapters. 'S' and 'R' denote serial adapter and residual adapter, respectively. 'β' indicates using the pre-classifier adaptation. Mean accuracy, 95% confidence interval are reported. The first eight datasets are seen during training and the last five datasets are unseen and used for test only.

SDL-ResNet-34. We also apply our method to the single domain learning model with ResNet-34 backbone learned on ImageNet only as in [5]. We follow [5] and use higher-resolution (224×224) images for meta-training and meta-testing. For optimization, we follow the training protocol as in [6, 13]. Specifically, we use SGD optimizer and cosine annealing with a momentum of 0.9, a weight decay of 1×10^{-4} with a batch size of 128. Other hyperparameters are the same as in SDL-ResNet-18 and are shown in Tab. 1. To regularize training, we also use the exact same data augmentations as in [6, 13], *e.g.* random crops and random color augmentations with an additional stage that randomly downsamples and upsamples images as in [5].

2.2. Task-specific learning

Attaching and learning adapters. For the optimization of the adaptation parameters α which is attached directly and learned on support set and the pre-classifier adaptation β, we follow the optimization strategy in [13], initialize β as an identity matrix and optimize both α and β for 40 iterations using Adadelta [24] as optimizer. The learning rate of β is 0.1 for first eight datasets and 1 for the last five datasets as in [13] and we set the learning rate of α as half of the learning rate of β, *i.e.* 0.05 for the first eight datasets and 0.5 for the last five datasets. Note that, we learn α and β on a per-task basis using the task's support set during meta-test. That is, α and β are not re-used across the test tasks drawn from $\mathcal{D}_t$.

Predicting r_α. In case of modulating α with the auxiliary network, we follow the auxiliary training protocols in [2]. We train for 10K episodes to optimize the task encoder using Adam with a learning rate of 1×10^{-5} on eight training domains in meta-train. We validate every 5K iterations to save the best model for test.

3. More results

3.1. Our method with different feature extractors

Tab. 2 shows the results of our method (the proposed residual adapters in matrix form) when incorporated to different feature extractors, single domain model with ResNet-18 backbone (SDL-ResNet-18) pre-trained on ImageNet, single domain model with ResNet-34 (SDL-ResNet-34) pre-trained on ImageNet, vanilla multi-domain learning (MDL)

Test Dataset	Varying-Way Five-Shot					Five-Way One-Shot				
	Simple CNAPS [2]	SUR [6]	URT [15]	URL [13]	Ours	Simple CNAPS [2]	SUR [6]	URT [15]	URL [13]	Ours
ImageNet	47.2 ± 1.0	46.7 ± 1.0	48.6 ± 1.0	$\mathbf{49.4 \pm 1.0}$	48.3 ± 1.0	42.6 ± 0.9	40.7 ± 1.0	47.4 ± 1.0	$\mathbf{49.6 \pm 1.1}$	48.0 ± 1.0
Omniglot	95.1 ± 0.3	95.8 ± 0.3	96.0 ± 0.3	96.0 ± 0.3	$\mathbf{96.8 \pm 0.3}$	93.1 ± 0.5	93.0 ± 0.7	95.6 ± 0.5	95.8 ± 0.5	$\mathbf{96.3 \pm 0.4}$
Aircraft	74.6 ± 0.6	82.1 ± 0.6	81.2 ± 0.6	84.8 ± 0.5	$\mathbf{85.5 \pm 0.5}$	65.8 ± 0.9	67.1 ± 1.4	77.9 ± 0.9	$\mathbf{79.6 \pm 0.9}$	$\mathbf{79.6 \pm 0.9}$
Birds	69.6 ± 0.7	62.8 ± 0.9	71.2 ± 0.7	76.0 ± 0.6	$\mathbf{76.6 \pm 0.6}$	67.9 ± 0.9	59.2 ± 1.0	70.9 ± 0.9	$\mathbf{74.9 \pm 0.9}$	74.5 ± 0.9
Textures	57.5 ± 0.7	60.2 ± 0.7	65.2 ± 0.7	$\mathbf{69.1 \pm 0.6}$	68.3 ± 0.7	42.2 ± 0.8	42.5 ± 0.8	49.4 ± 0.9	53.6 ± 0.9	$\mathbf{54.5 \pm 0.9}$
Quick Draw	70.9 ± 0.6	79.0 ± 0.5	$\mathbf{79.2 \pm 0.5}$	78.2 ± 0.5	77.9 ± 0.6	70.5 ± 0.9	$\mathbf{79.8 \pm 0.9}$	79.6 ± 0.9	79.0 ± 0.8	79.3 ± 0.9
Fungi	50.3 ± 1.0	66.5 ± 0.8	66.9 ± 0.9	70.0 ± 0.8	$\mathbf{70.4 \pm 0.8}$	58.3 ± 1.1	64.8 ± 1.1	71.0 ± 1.0	75.2 ± 1.0	$\mathbf{75.3 \pm 1.0}$
VGG Flower	86.5 ± 0.4	76.9 ± 0.6	82.4 ± 0.5	89.3 ± 0.4	$\mathbf{89.5 \pm 0.4}$	79.9 ± 0.7	65.0 ± 1.0	72.7 ± 0.0	79.9 ± 0.8	$\mathbf{80.3 \pm 0.8}$
Traffic Sign	55.2 ± 0.8	44.9 ± 0.9	45.1 ± 0.9	57.5 ± 0.8	$\mathbf{72.3 \pm 0.6}$	55.3 ± 0.9	44.6 ± 0.9	52.7 ± 0.9	$\mathbf{57.9 \pm 0.9}$	57.2 ± 1.0
MSCOCO	49.2 ± 0.8	48.1 ± 0.9	52.3 ± 0.9	$\mathbf{56.1 \pm 0.8}$	56.0 ± 0.8	48.8 ± 0.9	47.8 ± 1.1	56.9 ± 1.1	59.2 ± 1.0	$\mathbf{59.9 \pm 1.0}$
MNIST	88.9 ± 0.4	90.1 ± 0.4	86.5 ± 0.5	89.7 ± 0.4	$\mathbf{92.5 \pm 0.4}$	$\mathbf{80.1 \pm 0.9}$	77.1 ± 0.9	75.6 ± 0.9	78.7 ± 0.9	$\mathbf{80.1 \pm 0.9}$
CIFAR-10	66.1 ± 0.7	50.3 ± 1.0	61.4 ± 0.7	66.0 ± 0.7	$\mathbf{72.0 \pm 0.7}$	50.3 ± 0.9	35.8 ± 0.8	47.3 ± 0.9	54.7 ± 0.9	$\mathbf{55.8 \pm 0.9}$
CIFAR-100	53.8 ± 0.9	46.4 ± 0.9	52.5 ± 0.9	57.0 ± 0.9	$\mathbf{64.1 \pm 0.8}$	53.8 ± 0.9	42.9 ± 1.0	54.9 ± 1.1	61.8 ± 1.0	$\mathbf{63.7 \pm 1.0}$
Average Seen	69.0	71.2	73.8	76.6	**76.7**	65.0	64.0	70.6	73.4	**73.5**
Average Unseen	62.6	56.0	59.6	65.2	**71.4**	57.7	49.6	57.5	62.4	**63.4**
Average All	66.5	65.4	68.3	72.2	**74.6**	62.2	58.5	65.5	69.2	**69.6**
Average Rank	4.1	3.9	3.4	2.1	**1.5**	3.8	4.5	3.3	**1.7**	**1.7**

Table 4. Results of Varying-Way Five-Shot and Five-Way One-Shot scenarios. Mean accuracy, 95% confidence interval are reported.

Test Dataset	CNAPS [19]	Simple CNAPS [2]	TransductiveCNAPS [1]	SUR [6]	URT [15]	FLUTE [21]	tri-M [16]	URL [13]	Ours
ImageNet	50.8 ± 1.1	56.5 ± 1.1	57.9 ± 1.1	54.5 ± 1.1	55.0 ± 1.1	51.8 ± 1.1	$\mathbf{58.6 \pm 1.0}$	57.5 ± 1.1	57.4 ± 1.1
Omniglot	91.7 ± 0.5	91.9 ± 0.6	94.3 ± 0.4	93.0 ± 0.5	93.3 ± 0.5	93.2 ± 0.5	92.0 ± 0.6	94.5 ± 0.4	$\mathbf{95.0 \pm 0.4}$
Aircraft	83.7 ± 0.6	83.8 ± 0.6	84.7 ± 0.5	84.3 ± 0.5	84.5 ± 0.6	87.2 ± 0.5	82.8 ± 0.7	88.6 ± 0.5	$\mathbf{89.3 \pm 0.4}$
Birds	73.6 ± 0.9	76.1 ± 0.9	78.8 ± 0.7	70.4 ± 1.1	75.8 ± 0.8	79.2 ± 0.8	75.3 ± 0.8	80.5 ± 0.7	$\mathbf{81.4 \pm 0.7}$
Textures	59.5 ± 0.7	70.0 ± 0.8	66.2 ± 0.8	70.5 ± 0.7	70.6 ± 0.7	68.8 ± 0.8	71.2 ± 0.8	76.2 ± 0.7	$\mathbf{76.7 \pm 0.7}$
Quick Draw	74.7 ± 0.8	78.3 ± 0.7	77.9 ± 0.6	81.6 ± 0.6	$\mathbf{82.1 \pm 0.6}$	79.5 ± 0.7	77.3 ± 0.7	81.9 ± 0.6	82.0 ± 0.6
Fungi	50.2 ± 1.1	49.1 ± 1.2	48.9 ± 1.2	65.0 ± 1.0	63.7 ± 1.0	58.1 ± 1.1	48.5 ± 1.0	$\mathbf{68.8 \pm 0.9}$	67.4 ± 1.0
VGG Flower	88.9 ± 0.5	91.3 ± 0.6	$\mathbf{92.3 \pm 0.4}$	82.2 ± 0.8	88.3 ± 0.6	91.6 ± 0.6	90.5 ± 0.5	92.1 ± 0.5	92.2 ± 0.5
Traffic Sign	56.5 ± 1.1	59.2 ± 1.0	59.7 ± 1.1	49.8 ± 1.1	50.1 ± 1.1	58.4 ± 1.1	63.0 ± 1.0	63.3 ± 1.2	$\mathbf{83.5 \pm 0.9}$
MSCOCO	39.4 ± 1.0	42.4 ± 1.1	42.5 ± 1.1	49.4 ± 1.1	48.9 ± 1.1	50.0 ± 1.0	52.8 ± 1.1	54.0 ± 1.0	$\mathbf{55.8 \pm 1.1}$
MNIST	-	94.3 ± 0.4	94.7 ± 0.3	94.9 ± 0.4	90.5 ± 0.4	95.6 ± 0.5	96.2 ± 0.3	94.5 ± 0.5	$\mathbf{96.7 \pm 0.4}$
CIFAR-10	-	72.0 ± 0.8	73.6 ± 0.7	64.2 ± 0.9	65.1 ± 0.8	78.6 ± 0.7	75.4 ± 0.8	71.9 ± 0.7	$\mathbf{80.6 \pm 0.8}$
CIFAR-100	-	60.9 ± 1.1	61.8 ± 1.0	57.1 ± 1.1	57.2 ± 1.0	67.1 ± 1.0	62.0 ± 1.0	62.6 ± 1.0	$\mathbf{69.6 \pm 1.0}$
Average Seen	71.6	74.6	75.1	75.2	76.7	76.2	74.5	80.0	80.2
Average Unseen	-	65.8	66.5	63.1	62.4	69.9	69.9	69.3	77.2
Average All	-	71.2	71.8	70.5	71.2	73.8	72.7	75.9	79.0
Average Rank	-	6.3	4.9	5.8	5.7	4.3	4.8	2.7	1.5

Table 5. Comparison state-of-the-art methods on Meta-Dataset (using a multi-domain feature extractor of [13]). Mean accuracy, 95% confidence interval are reported. The first eight datasets are seen during training and the last five datasets are unseen and used for test only.

and URL [13]. We see that attaching and learning residual adapters can significantly improve the performance on all domains over SDL-ResNet-18, SDL-ResNet-34 and MDL and obtain better performance on most domains over URL (11 out of 13 domains). This strongly indicates that our method can efficiently adapt the model for unseen categories and domains with few support samples while being agnostic to the feature extractor with different backbone and resolution of images.

3.2. Task-specific parameterizations

In Tab. 3, we report additional 95% confidence interval of each dataset to the main paper for the comparison of different r_α choices based on the URL model. The first eight datasets are seen during training and the last five datasets are unseen and used for test only. We can see that the confidence intervals for different methods have marginal differences.

3.3. Varying-way 5-shot and 5-way-1-shot

In the main paper, we only report the average accuracy of Varying-Way Five-Shot and Five-Way One-Shot scenarios due to limited space, and detailed results are depicted in Tab. 4. In the table, we report the Mean accuracy, 95% confidence interval of each dataset. The first eight datasets are seen during training and the last five datasets are unseen and used for test only. URT and URL are two strong baselines surpassing both Simple CNAPS and SUR, while Ours outperforms them on most datasets, especially on unseen domains.

3.4. Results evaluated with updated evaluation protocol.

As the code from Meta-dataset has been updated, we evaluate all methods with the updated evaluation protocol

Test Dataset	ImageNet	Omniglot	Aircraft	Birds	Textures	Quick Draw	Fungi	VGG Flower	Traffic Sign	MSCOCO	MNIST	CIFAR-10	CIFAR-100
10 iterations	55.5 ± 1.1	93.9 ± 0.5	86.4 ± 0.5	78.6 ± 0.7	73.3 ± 0.7	81.9 ± 0.6	63.1 ± 0.9	90.3 ± 0.5	77.6 ± 1.0	50.6 ± 1.1	96.9 ± 0.3	77.0 ± 0.8	62.6 ± 1.1
20 iterations	56.2 ± 1.1	94.7 ± 0.4	86.3 ± 0.5	78.3 ± 0.8	73.9 ± 0.7	81.6 ± 0.6	63.4 ± 0.9	90.1 ± 0.6	79.4 ± 1.0	52.8 ± 1.1	97.2 ± 0.3	78.6 ± 0.8	65.9 ± 1.1
40 iterations	55.6 ± 1.0	94.3 ± 0.4	86.7 ± 0.5	79.4 ± 0.8	73.2 ± 0.8	81.7 ± 0.6	64.0 ± 0.9	90.9 ± 0.5	81.1 ± 0.9	51.4 ± 1.1	96.9 ± 0.3	78.5 ± 0.8	64.3 ± 1.1
60 iterations	55.9 ± 1.1	95.1 ± 0.4	85.9 ± 0.6	77.5 ± 0.8	74.7 ± 0.7	80.9 ± 0.6	62.1 ± 0.9	90.7 ± 0.6	82.2 ± 0.9	52.2 ± 1.1	97.0 ± 0.4	78.4 ± 0.8	64.4 ± 1.1

Table 6. Sensitivity of performance to number of iterations based on MDL model.

Test Dataset	ImageNet	Omniglot	Aircraft	Birds	Textures	Quick Draw	Fungi	VGG Flower	Traffic Sign	MSCOCO	MNIST	CIFAR-10	CIFAR-100
10 iterations	58.4 ± 1.1	94.8 ± 0.4	89.9 ± 0.4	81.3 ± 0.7	76.6 ± 0.7	81.8 ± 0.6	68.4 ± 0.9	92.5 ± 0.5	76.5 ± 1.1	55.6 ± 1.1	96.4 ± 0.4	79.0 ± 0.7	66.9 ± 1.0
20 iterations	58.2 ± 1.1	94.8 ± 0.4	89.9 ± 0.4	81.1 ± 0.7	77.5 ± 0.8	81.9 ± 0.6	68.0 ± 0.9	92.4 ± 0.5	81.8 ± 1.0	57.8 ± 1.1	96.7 ± 0.4	81.7 ± 0.8	69.1 ± 0.9
40 iterations	59.5 ± 1.0	94.9 ± 0.4	89.9 ± 0.4	81.1 ± 0.8	77.5 ± 0.7	81.7 ± 0.6	66.3 ± 0.9	92.2 ± 0.5	82.8 ± 1.0	57.6 ± 1.0	96.7 ± 0.4	82.9 ± 0.7	70.4 ± 1.0
60 iterations	58.7 ± 1.1	94.9 ± 0.4	89.5 ± 0.5	80.8 ± 0.7	77.4 ± 0.8	81.8 ± 0.6	66.2 ± 0.9	92.5 ± 0.5	83.7 ± 0.9	56.9 ± 1.0	96.6 ± 0.3	82.0 ± 0.8	72.0 ± 0.9

Table 7. Sensitivity of performance to number of iterations based on URL model.

from the Meta-dataset [1] and report the results [2] in Tab. 5. As shown in Tab. 5, the update does not affect much on the results and our method rank 1.5 in average and the state-of-the-art method URL rank 2.7. Our method outperforms other methods on most domains (9 out of 13), especially obtaining significant improvement on 5 unseen datasets than the second best method, *i.e*. Average Unseen (+7.9). More specifically, our method obtains significant better results than the second best approach (URL) on Traffic Sign (+20.2), CIFAR-10 (+8.7), and CIFAR-100 (+7.0).

3.5. Ablation study

Here, we conduct ablation study of our method with the URL model, unless stated otherwise.

Sensitivity analysis for number of iterations. In our method, we optimize the attached parameters (α, β) with 40 iterations. Figure 1 and Figure 2 report the results with 10, 20, 40, 60 iterations and indicates that our method (solid green) converges to a stable solution after 20 iterations and achieves better average performance on all domains than the baseline URL (dash green). The mean accuracy with 95% confidence interval are reported in Tabs. 6 and 7

Influence of α and β. We evaluate different components of our method and report the results in Tab. 8. The results show that both residual adapters α and the linear transformation β help adapt features to unseen classes while residual adapters significantly improve the performance on unseen domains. The best results are achieved by using both α and β.

Initialization analysis for adapters. Here, we investigate using different initialization strategies for adapters: i) Identity initialization: in this work we initialize each residual adapter as an identity matrix scaled by a scalar δ and we set $\delta = 1e-4$; ii) randomly initialization: alternatively, we can randomly initialize each residual adapter. The results of different initialization are summarized in Fig. 3. We can see that our methods with different initialization strategies obtain similar results, which indicates that our method works also with randomly initialization and again verifies the stability of our method. Detailed results of each datasets are shown in Tab. 9.

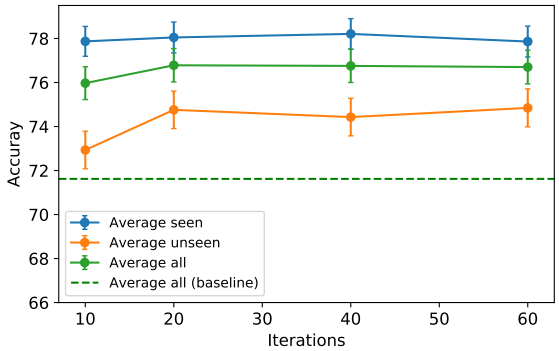

Figure 1. Sensitivity of performance to number of iterations based on MDL model.

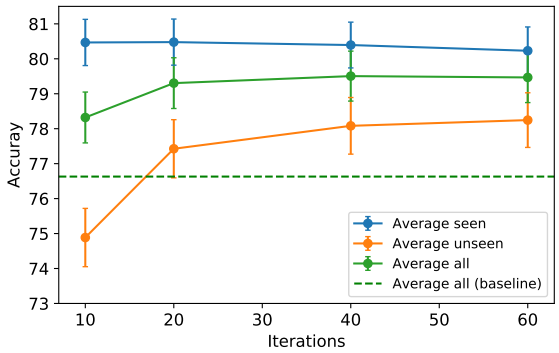

Figure 2. Sensitivity of performance to number of iterations based on URL model.

Layer analysis for adapters. Here we investigate whether it is sufficient to attach the adapters only to the later layers. We evaluate this on ResNet18 which is composed of four

[1] As mentioned in https://github.com/google-research/meta-dataset/issues/54, we also set the shuffle_buffer_size as 1000 to evaluate all methods and report the results in Tab. 5. This change does not affect much on the results as the datasets we used were shuffled using the latest data convert code from Meta-Dataset.

[2] The results of Simple CNAPS [2] and Transductive CNAPS [1] are reproduced by the authors and reported at https://github.com/peymanbateni/simple-cnaps. Results of FLUTE [21] and tri-M [16] are from their papers. We reproduce the results of SUR [6] and URT [15] with the updated evaluation protocol for fair comparison.

Test Dataset	ImageNet	Omniglot	Aircraft	Birds	Textures	Quick Draw	Fungi	VGG Flower	Traffic Sign	MSCOCO	MNIST	CIFAR-10	CIFAR-100
Ours w/o α & β	57.0 ± 1.1	94.4 ± 0.4	88.0 ± 0.5	80.3 ± 0.7	74.6 ± 0.7	81.8 ± 0.6	66.2 ± 0.9	91.5 ± 0.5	49.8 ± 1.1	54.1 ± 1.0	91.1 ± 0.4	70.6 ± 0.7	59.1 ± 1.0
Ours w/o β	57.3 ± 1.1	$\mathbf{94.9 \pm 0.4}$	88.9 ± 0.5	81.0 ± 0.7	76.7 ± 0.7	80.6 ± 0.6	65.4 ± 0.9	91.4 ± 0.5	82.6 ± 1.0	55.0 ± 1.1	96.6 ± 0.4	82.1 ± 0.7	66.4 ± 1.1
Ours w/o α	58.8 ± 1.1	94.5 ± 0.4	89.4 ± 0.4	80.7 ± 0.8	77.2 ± 0.7	$\mathbf{82.5 \pm 0.6}$	$\mathbf{68.1 \pm 0.9}$	92.0 ± 0.5	63.3 ± 1.2	57.3 ± 1.0	94.7 ± 0.4	74.2 ± 0.8	63.6 ± 1.0
Ours	$\mathbf{59.5 \pm 1.0}$	$\mathbf{94.9 \pm 0.4}$	$\mathbf{89.9 \pm 0.4}$	$\mathbf{81.1 \pm 0.8}$	$\mathbf{77.5 \pm 0.7}$	81.7 ± 0.6	66.3 ± 0.9	$\mathbf{92.2 \pm 0.5}$	$\mathbf{82.8 \pm 1.0}$	$\mathbf{57.6 \pm 1.0}$	$\mathbf{96.7 \pm 0.4}$	$\mathbf{82.9 \pm 0.7}$	$\mathbf{70.4 \pm 1.0}$

Table 8. Effect of each component. We build our method on the URL model and 'Ours w/o α & β' means we remove both residual adapters α and the pre-classifier adaptation layer β in our method.

Test Dataset	ImageNet	Omniglot	Aircraft	Birds	Textures	Quick Draw	Fungi	VGG Flower	Traffic Sign	MSCOCO	MNIST	CIFAR-10	CIFAR-100
Ours(SDL-ResNet-18)-I	59.5 ± 1.1	78.2 ± 1.2	72.2 ± 1.0	74.9 ± 0.9	77.3 ± 0.7	67.6 ± 0.9	44.7 ± 1.0	90.9 ± 0.6	82.5 ± 0.8	59.0 ± 1.0	93.9 ± 0.6	82.1 ± 0.7	70.7 ± 0.9
Ours(SDL-ResNet-18)-R	58.2 ± 1.0	78.4 ± 1.2	71.1 ± 1.1	74.4 ± 1.0	77.1 ± 0.7	67.2 ± 1.0	45.9 ± 1.0	90.7 ± 0.6	81.9 ± 1.0	57.7 ± 1.1	94.1 ± 0.5	81.9 ± 0.7	70.5 ± 0.9
Ours(MDL)-I	55.6 ± 1.0	94.3 ± 0.4	86.7 ± 0.5	79.4 ± 0.8	73.2 ± 0.8	81.7 ± 0.6	64.0 ± 0.9	90.9 ± 0.5	81.1 ± 0.9	51.4 ± 1.1	96.9 ± 0.3	78.5 ± 0.8	64.3 ± 1.1
Ours(MDL)-R	56.0 ± 1.1	94.1 ± 0.4	87.1 ± 0.5	79.7 ± 0.8	74.0 ± 0.7	82.0 ± 0.6	62.6 ± 0.9	90.6 ± 0.6	80.9 ± 0.9	51.7 ± 1.1	96.9 ± 0.4	77.7 ± 0.9	65.8 ± 1.1
Ours(URL)-I	59.5 ± 1.0	94.9 ± 0.4	89.9 ± 0.4	81.1 ± 0.8	77.5 ± 0.7	81.7 ± 0.6	66.3 ± 0.9	92.2 ± 0.5	82.8 ± 1.0	57.6 ± 1.0	96.7 ± 0.4	82.9 ± 0.7	70.4 ± 1.0
Ours(URL)-R	58.8 ± 1.1	94.9 ± 0.4	90.5 ± 0.4	81.8 ± 0.6	77.7 ± 0.7	82.3 ± 0.6	66.8 ± 0.9	92.6 ± 0.5	83.7 ± 0.8	57.7 ± 1.1	96.9 ± 0.4	82.5 ± 0.7	72.0 ± 0.9

Table 9. Initialization analysis of adapters. 'Ours(URL)-I' indicates our method using URL as the pretrained model and initializing residual adapters as identity matrix (scaled by $\delta = 0.0001$) while 'Ours(URL)-R' means our method initialize residual adapters randomly.

Test Dataset	ImageNet	Omniglot	Aircraft	Birds	Textures	Quick Draw	Fungi	VGG Flower	Traffic Sign	MSCOCO	MNIST	CIFAR-10	CIFAR-100
Ours (block4)	59.0 ± 1.1	95.0 ± 0.4	90.0 ± 0.4	80.6 ± 0.8	77.8 ± 0.7	82.3 ± 0.6	68.2 ± 0.9	91.8 ± 0.6	70.6 ± 1.1	57.1 ± 1.1	95.9 ± 0.4	77.2 ± 0.8	65.9 ± 1.0
Ours (block3,4)	60.4 ± 1.1	94.7 ± 0.4	90.0 ± 0.5	80.4 ± 0.7	77.8 ± 0.7	82.2 ± 0.6	67.2 ± 0.8	92.5 ± 0.5	77.2 ± 1.0	57.9 ± 1.0	96.7 ± 0.3	78.8 ± 0.9	68.6 ± 0.9
Ours (block2,3,4)	59.6 ± 1.1	94.9 ± 0.4	89.9 ± 0.5	81.0 ± 0.8	78.2 ± 0.7	82.4 ± 0.6	67.6 ± 0.9	92.3 ± 0.5	81.5 ± 1.0	57.9 ± 1.0	96.6 ± 0.4	81.5 ± 0.8	70.6 ± 1.0
Ours (block-all)	59.5 ± 1.0	94.9 ± 0.4	89.9 ± 0.4	81.1 ± 0.8	77.5 ± 0.7	81.7 ± 0.6	66.3 ± 0.9	92.2 ± 0.5	82.8 ± 1.0	57.6 ± 1.0	96.7 ± 0.4	82.9 ± 0.7	70.4 ± 1.0

Table 10. Block (layer) analysis for adapters based on URL model.

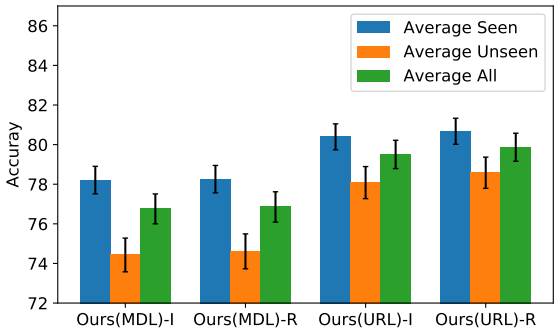

Figure 3. Initialization analysis for adapters. '-I' indicates identity initialization and '-R' is randomly initialization.

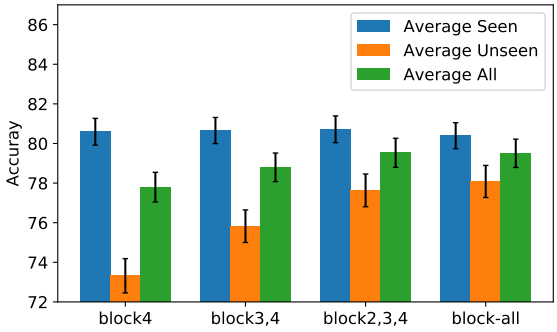

Figure 4. Block (layer) analysis for adapters.

blocks and attach the adapters to only later blocks (block4, block3,4, block2,3,4 and block-all. Figure 4 shows that applying our adapters to only the last block (block4) obtains around 78% average accuracy on all domains which outperforms the URL. With attaching residual adapters to more layers, the performance on unseen domains is improved significantly while the one on seen domains remains stable. The mean accuracy with 95% confidence interval for layer analysis are shown in Tab. 10.

Decomposing residual adapters. Here we investigate whether one can reduce the number of parameters in the adapters while retaining its performance by using matrix decomposition. As in deep neural network, the adapters in earlier layers are relatively small, we then decompose the adapters in the last two blocks only where the adapter dimensionality goes up to 512×512. Figure 5 shows that our method can achieve good performance with less parameters by decomposing large residual adapters, (*e.g.* when $N = 32$ where the number of additional parameters equal to around 4% vs 13%, the performance is still comparable to the original form of residual adapters, *i.e.* N=0). Results of each datasets in Tab. 11, also show that, by decomposing large residual adapters, the performance of our method is still comparable to the original form of residual adapters (*i.e.* Ours) with less parameters.

The similar conclusion can be drawn from results (shown in Fig. 6) of our method using decomposed residual adapters in all layers. When N increases, *i.e.*, smaller residual adapters, the average accuracy on all domains is still comparable to the original form of residual adapters (*i.e.* N=0) with less parameters though the average accuracy on unseen domains drops slightly. From the results depicted in Tab. 12, we can see that when N increases, the performance of most domains are still comparable to the original form of residual adapters (*i.e.* Ours) while the performance on Traffic Sign drops slightly as the adapters in earlier layers are small and when N is larger the decomposed residual adapters might be

Test Dataset	ImageNet	Omniglot	Aircraft	Birds	Textures	Quick Draw	Fungi	VGG Flower	Traffic Sign	MSCOCO	MNIST	CIFAR-10	CIFAR-100
Ours	59.5 ± 1.0	94.9 ± 0.4	89.9 ± 0.4	81.1 ± 0.8	77.5 ± 0.7	81.7 ± 0.6	66.3 ± 0.9	92.2 ± 0.5	82.8 ± 1.0	57.6 ± 1.0	96.7 ± 0.4	82.9 ± 0.7	70.4 ± 1.0
Ours(N=2)	58.9 ± 1.1	95.2 ± 0.4	89.7 ± 0.5	80.9 ± 0.7	76.7 ± 0.7	81.4 ± 0.6	67.7 ± 0.9	92.2 ± 0.5	82.4 ± 1.0	57.1 ± 1.0	96.5 ± 0.4	82.4 ± 0.7	70.3 ± 1.0
Ours(N=4)	58.7 ± 1.1	94.9 ± 0.4	89.7 ± 0.5	80.3 ± 0.7	77.0 ± 0.7	82.5 ± 0.6	67.2 ± 0.9	92.5 ± 0.5	82.6 ± 1.0	57.5 ± 1.1	96.5 ± 0.4	82.5 ± 0.7	70.8 ± 0.9
Ours(N=8)	59.1 ± 1.1	95.0 ± 0.4	89.8 ± 0.5	80.2 ± 0.8	77.2 ± 0.7	82.1 ± 0.6	67.0 ± 0.9	92.2 ± 0.5	82.5 ± 1.0	57.2 ± 1.1	96.8 ± 0.4	82.6 ± 0.7	71.8 ± 0.9
Ours(N=16)	58.2 ± 1.1	94.7 ± 0.4	90.1 ± 0.4	80.3 ± 0.8	76.9 ± 0.7	81.7 ± 0.6	67.6 ± 0.9	92.0 ± 0.5	81.8 ± 1.0	58.1 ± 1.1	96.4 ± 0.4	81.8 ± 0.7	71.1 ± 0.9
Ours(N=32)	59.2 ± 1.1	94.8 ± 0.4	89.6 ± 0.5	80.0 ± 0.8	77.3 ± 0.6	82.4 ± 0.6	67.2 ± 0.9	92.1 ± 0.5	82.1 ± 1.0	57.1 ± 1.0	96.7 ± 0.3	81.6 ± 0.8	71.1 ± 0.9

Table 11. Results of using decomposed RA on layer3,4.

Test Dataset	ImageNet	Omniglot	Aircraft	Birds	Textures	Quick Draw	Fungi	VGG Flower	Traffic Sign	MSCOCO	MNIST	CIFAR-10	CIFAR-100
Ours	59.5 ± 1.0	94.9 ± 0.4	89.9 ± 0.4	81.1 ± 0.8	77.5 ± 0.7	81.7 ± 0.6	66.3 ± 0.9	92.2 ± 0.5	82.8 ± 1.0	57.6 ± 1.0	96.7 ± 0.4	82.9 ± 0.7	70.4 ± 1.0
Ours(N=2)	58.1 ± 1.1	94.8 ± 0.4	89.7 ± 0.5	80.2 ± 0.8	76.9 ± 0.7	82.1 ± 0.6	67.8 ± 0.9	92.0 ± 0.6	82.5 ± 0.9	56.9 ± 1.1	96.7 ± 0.3	82.0 ± 0.8	70.3 ± 1.0
Ours(N=4)	59.6 ± 1.1	94.8 ± 0.4	89.9 ± 0.5	80.3 ± 0.8	77.4 ± 0.7	82.6 ± 0.6	66.6 ± 0.9	92.9 ± 0.5	79.7 ± 1.1	57.6 ± 1.1	96.5 ± 0.4	80.9 ± 0.8	70.6 ± 1.0
Ours(N=8)	58.2 ± 1.1	94.6 ± 0.4	89.6 ± 0.5	81.2 ± 0.8	76.6 ± 0.7	82.7 ± 0.6	66.5 ± 0.9	92.3 ± 0.5	78.1 ± 1.1	57.3 ± 1.0	96.3 ± 0.3	81.0 ± 0.8	70.9 ± 0.9
Ours(N=16)	58.9 ± 1.1	94.6 ± 0.4	89.7 ± 0.5	80.1 ± 0.7	77.0 ± 0.7	82.1 ± 0.6	68.4 ± 0.9	91.9 ± 0.5	78.3 ± 1.0	57.8 ± 1.1	96.0 ± 0.4	82.0 ± 0.7	70.3 ± 1.0

Table 12. Results of using decomposed RA on all layers.

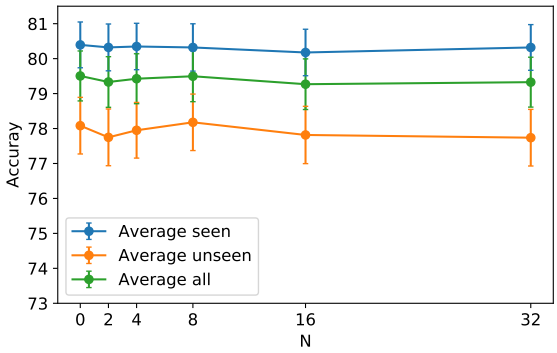

Figure 5. Decomposed residual adapters on block-3,4.

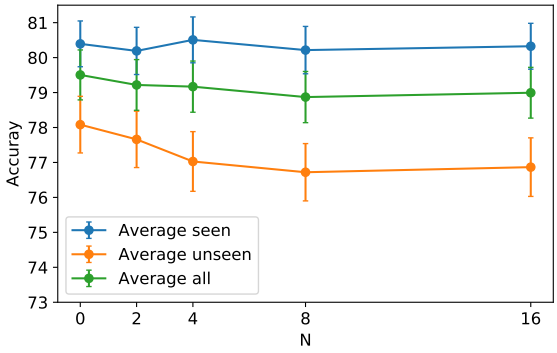

Figure 6. Decomposed residual adapters on all layers.

too small to tranform the features. In overall, our method can achieve good performance with less parameters by decomposing large residual adapters.

Training time. The training time (meta-train) of our method is equal to the one of URL (hence no additional cost), *i.e*. 48 hours in multi-domain setting, 6 hours for Resnet-18 and 33 hours for Resnet-34 in single-domain learning in one Nvidia V100 GPU. Whereas CTX meta-training requires 8 Nvidia V100 GPUs for 7 days and approximately 40 times more expensive than ours. During the meta-test stage, the model parameters are further trained using support set of each episode. Meta-test training cost is depicted in Tab. 12 for Meta-Dataset tasks. URL baseline only finetunes parameters of PA β. Finetune+NCC updates the entire backbone parameters. Ours learn RA and PA parameters. While URL is the fastest baseline, as it does not require backpropagating the error to early layers, ours is more efficient than finetuning all the backbone parameters.

Test Dataset	Image -Net	Omni -glot	Air- craft	Birds	Tex- tures	Quick Draw	Fungi	VGG Flower	Traffic Sign	MS- COCO	MNIST	CIFAR -10	CIFAR -100
URL	0.7	0.7	0.4	0.7	0.4	1.0	1.0	0.5	0.9	0.9	0.4	0.4	1.0
Finetune+NCC	7.7	2.5	7.4	7.0	5.8	9.3	8.7	6.6	9.1	9.0	6.5	6.7	9.3
Ours (URL+RA+PA)	7.2	2.4	6.1	6.8	4.8	8.9	7.4	5.2	8.8	8.3	6.0	6.2	8.6

Table 12. Computation cost (# second per task) during meta-test.

3.6. Qualitative results

We qualitatively analyze our method and compare it to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in Figs. 7 to 19 by illustrating the nearest neighbors in all test datasets given a query image as in [13]. It is clear that our method produces more correct neighbors than other methods. While other methods retrieve images with more similar colors, shapes and backgrounds, *e.g*. in Figs. 15, 16, 18 and 19, our method is able to retrieve semantically similar images. More specifically, as shown in Fig. 10, our method correctly produces neighbors of the bird in the query image while other methods pick images with similar appearances or similar background, *e.g*. images with twigs. In Fig. 15, other methods mainly retrieve the triangle sign while our method is able to retrieve the correct sign with illumination distortion. In Fig. 19, other methods including SUR, URT are distracted by the blue background but our method select the correct shark images. It again suggests that our method is able to quickly adapt the features for unseen few-shot tasks.

Figure 7. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in ImageNet. Green and red colors indicate correct and false predictions respectively.

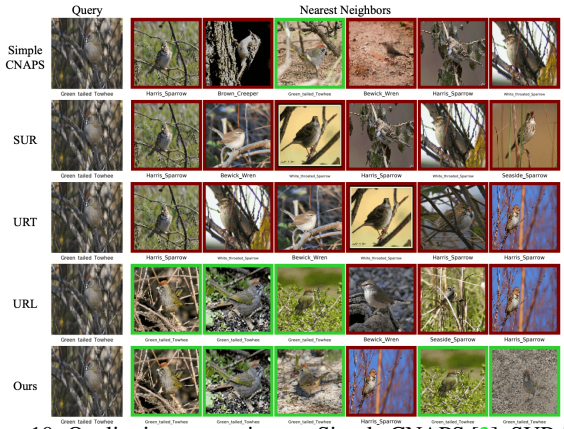

Figure 10. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in Birds. Green and red colors indicate correct and false predictions respectively.

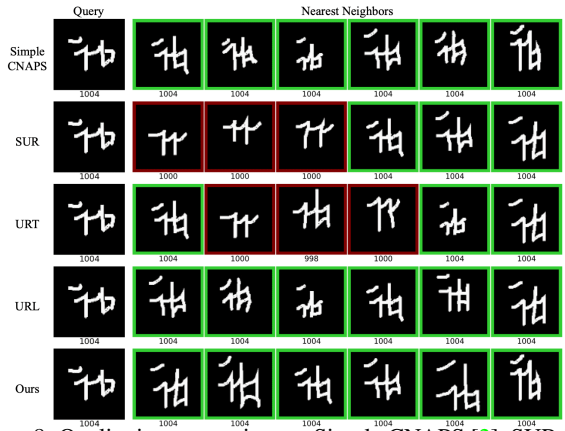

Figure 8. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in Omniglot. Green and red colors indicate correct and false predictions respectively.

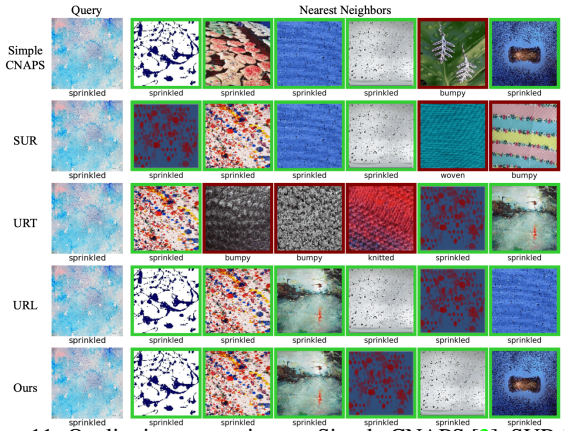

Figure 11. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in Textures. Green and red colors indicate correct and false predictions respectively.

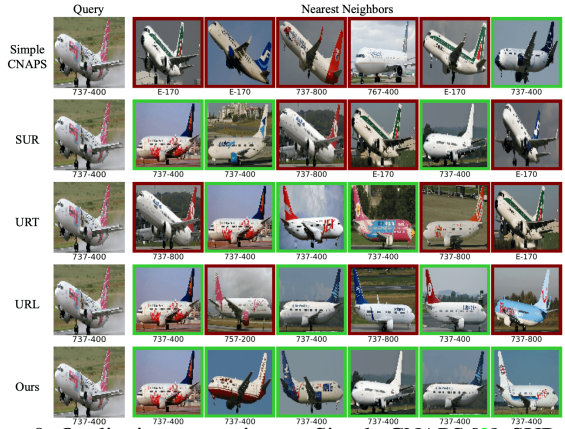

Figure 9. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in Aircraft. Green and red colors indicate correct and false predictions respectively.

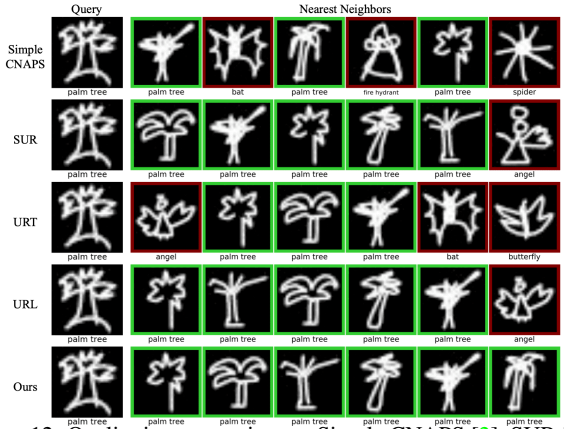

Figure 12. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in Quick Draw. Green and red colors indicate correct and false predictions respectively.

Figure 13. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in Fungi. Green and red colors indicate correct and false predictions respectively.

Figure 14. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in VGG Flower. Green and red colors indicate correct and false predictions respectively.

Figure 15. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in Traffic Sign. Green and red colors indicate correct and false predictions respectively.

Figure 16. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in MSCOCO. Green and red colors indicate correct and false predictions respectively.

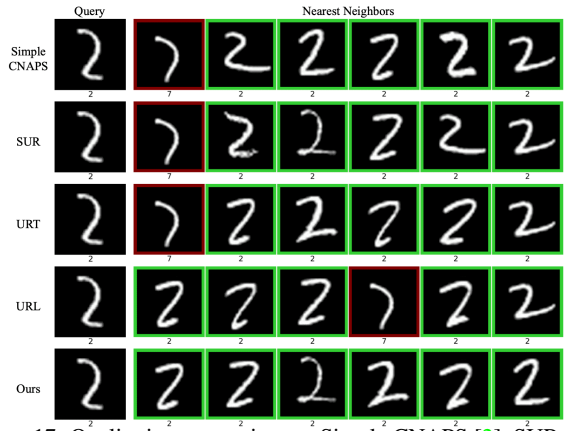

Figure 17. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in MNIST. Green and red colors indicate correct and false predictions respectively.

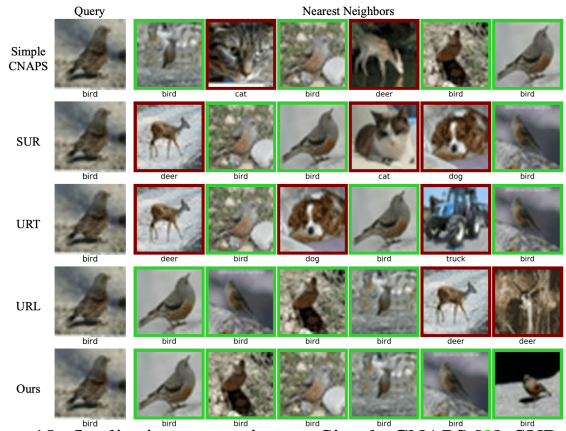

Figure 18. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in CIFAR-10. Green and red colors indicate correct and false predictions respectively.

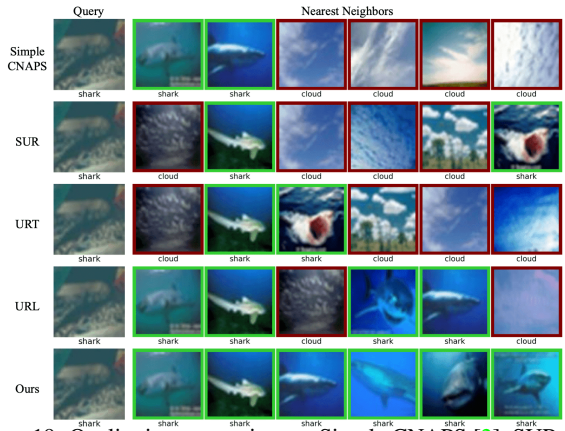

Figure 19. Qualitative comparison to Simple CNAPS [2], SUR [6], URT [15], and URL [13] in CIFAR-100. Green and red colors indicate correct and false predictions respectively.

References

[1] Peyman Bateni, Jarred Barber, Jan-Willem van de Meent, and Frank Wood. Enhancing few-shot image classification with unlabelled examples. *arXiv preprint arXiv:2006.12245*, 2020. 3, 4

[2] Peyman Bateni, Raghav Goyal, Vaden Masrani, Frank Wood, and Leonid Sigal. Improved few-shot visual classification. In *CVPR*, pages 14493–14502, 2020. 1, 2, 3, 4, 6, 7, 8, 9

[3] Schroeder Brigit and Cui Yin. Fgvcx fungi classification challenge. *online*, 2018. 1

[4] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *CVPR*, pages 3606–3613, 2014. 1

[5] Carl Doersch, Ankush Gupta, and Andrew Zisserman. Crosstransformers: spatially-aware few-shot transfer. In *NeurIPS*, 2020. 2

[6] Nikita Dvornik, Cordelia Schmid, and Julien Mairal. Selecting relevant features from a multi-domain representation for few-shot classification. In *ECCV*, pages 769–786, 2020. 1, 2, 3, 4, 6, 7, 8, 9

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 1

[8] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. Detection of traffic signs in real-world images: The german traffic sign detection benchmark. In *IJCNN*, pages 1–8. Ieee, 2013. 1

[9] Jonas Jongejan, Rowley Henry, Kawashima Takashi, Kim Jongmin, and Fox-Gieg Nick. The quick, draw! a.i. experiment. *online*, 2016. 1

[10] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Citeseer*, 2009. 1

[11] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015. 1

[12] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[13] Wei-Hong Li, Xialei Liu, and Hakan Bilen. Universal representation learning from multiple domains for few-shot classification. *ICCV*, 2021. 1, 2, 3, 6, 7, 8, 9

[14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 1

[15] Lu Liu, William Hamilton, Guodong Long, Jing Jiang, and Hugo Larochelle. A universal representation transformer layer for few-shot image classification. In *ICLR*, 2021. 3, 4, 6, 7, 8, 9

[16] Yanbin Liu, Juho Lee, Linchao Zhu, Ling Chen, Humphrey Shi, and Yi Yang. A multi-mode modulator for multi-domain few-shot classification. In *ICCV*, pages 8453–8462, 2021. 3, 4

[17] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 1

[18] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008. 1

[19] James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. In *NeurIPS*, 2019. 3

[20] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 1

[21] Eleni Triantafillou, Hugo Larochelle, Richard Zemel, and Vincent Dumoulin. Learning a universal template for few-shot dataset generalization. In *ICML*, 2021. 3, 4

[22] 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 *ICLR*, 2020. 1

[23] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. *California Institute of Technology*, 2011. 1

[24] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012. 2