A. Additional details

We have followed the experimental choices from [39] for FEMNIST, CIFAR-10-C, TinyImageNet-C and [19] for iWildCam, unless we specify otherwise. This includes the splits of data into training, validation and test sets as well as the splits of domains into meta-training, meta-validation and meta-test sets of domains. For iWildCam we use the OOD splits from [19].

A.1. Models

Feature extractor and classifier:

- FEMNIST: CNN with three convolutional layers, hidden dimension of 128, batch normalization, ReLU activation, kernel of 5, padding of 2. Classifier consists of two fully-connected layers with 200 hidden units and ReLU activation in between. The input shape of images is 28×28.
- CIFAR-C: Same architecture as for FEMNIST, but with three input channels instead of 1 (colour images). The input shape of images is 32×32.
- TinyImageNet-C: ImageNet pre-trained ResNet50 [13]. The classifier consists of one fully connected layer. The input shape of images is 64×64.
- iWildCam: ImageNet pre-trained ResNet50 [13]. The classifier consists of one fully connected layer. The input shape of images is 112×112.

Adaptation-specific components:

- CML: Context network is used to transform the support examples – three convolutional layers, 64 hidden units, kernel size of 5, padding of 2, with batch normalization and ReLU activation. The output of the network has the same shape as input. To create the context we average the context network outputs across the support examples and use the same context for all query examples in the task.
- CXDA: All key details are explained in the main text, and we provide more detailed explanations here. The size of the fully-connected layers depends on the flattened shape of the features – only one cross-attention layer is used. After multiplying attention weights with projected values ($\boldsymbol{Av}$), we transform the output further using projection matrix $\boldsymbol{W}$, similarly as [4]. However, we do not use a further MLP model that would contain multiple layers and non-linearity, so we also follow [4] in this aspect. The output of the cross-attention module has the same shape as input. As part of CXDA, batch normalization statistics of the feature extractor are updated too using the support set.

A.2. Training

Dataset-specific training details:

- FEMNIST: SGD with learning rate of 10^{-4}, momentum of 0.9 and weight decay of 10^{-4}, trained for 200 epochs, with validation set evaluated every 10 epochs, and early stopping based on accuracy.
- CIFAR-C: SGD with learning rate of 10^{-2}, momentum of 0.9 and weight decay of 10^{-4}, trained for 100 epochs, with validation set evaluated every 10 epochs, and early stopping based on accuracy.
- TinyImageNet-C: SGD with learning rate of 10^{-2}, momentum of 0.9 and weight decay of 10^{-4}, trained for 50 epochs, with validation set evaluated every 5 epochs, and early stopping based on accuracy.
- iWildCam: Adam with learning rate of 3×10^{-5}, no weight decay, trained for 50 epochs, with validation set evaluated every 5 epochs, and early stopping based on macro F1 score.

In all cases we use cross-entropy loss, and the cross-attention parameters are optimized in the same way as the main model. In each iteration we use a task that has 5 domains with 20 support examples for each sampled domain, and there are 20 query examples from one selected domain from the set of current domains.

Details about fine-tuning (FT) of the pre-trained ERM model: we perform fine-tuning by taking 10x smaller learning rate compared to the ones used during training and then performing 10 steps on the task's support data. We consider two losses: 1) entropy minimization (EM) and 2) information maximization (IM) of the support set example predictions during the fine-tuning. We reset the model to the pre-trained state for each test task.

A.3. Latent and continual domain adaptation

SF-OCDA When using SF-OCDA, pre-training is the same as for ERM, but the adaptation on evaluation tasks is specialized. As part of SF-OCDA [40] repurposed into our setup, we perform 10 update steps with 10x smaller learning rate, similar to our other back-propagation based baselines. Following [40], we use cross-entropy loss between the pseudo-labels and the predicted labels, which are predicted for clean support data or augmented support data with 70% and 30% probability respectively. The augmentations are more advanced and include color jitter, random affine transformation, Gaussian blur, random horizontal flip and Gaussian noise.

CoTTA Similar to other back-propagation based approaches, only the adaptation to evaluation tasks is unique and uses 10 update steps with 10x smaller learning rate than used during pre-training. During adaptation we directly follow [36] and use the support examples for adaptation. The key idea of the method is to use weight-averaged and augmentation-averaged predictions in order to reduce error accumulation. Additionally a small random part of neurons is restored to the pre-trained weights in each iteration, which helps prevent catastrophic forgetting.

SLA SLA [7] modifies the architecture of the main model to include gates and corrections (adapters) to handle latent domains. The adapters are trained alongside the main model in the pre-training stage. Since already two adapters are shown to perform well in [7], we also use two of them. Support examples of evaluation tasks are not used for adaptation because the model with trained adapters is directly used to make predictions on the query examples, with the adaptation done by using the adapters.

B. Qualitative analysis

We provide qualitative analysis of our approach in Figure 5. The analysis shows that similar images from the same location are given the largest weights, but also relevant images from other locations are given larger weights.

C. Runtime analysis

We provide a more detailed runtime analysis in Table 3. Table 3 (left) shows that the time per task (combination of adaptation and inference) of CXDA is very similar to feed-forward baselines and significantly faster than the time of fine-tuning (back-propagation) baselines. Table 3 (right) shows that the episodic pre-training takes longer for the smaller datasets, but the difference is small for large datasets and models. However, the pre-training time is not of particular interest to us because of the focus on fast adaptation to new tasks during deployment. Overall back-propagation-based approaches have significantly larger inference times than the feed-forward ones, showing the need for specialized feed-forward adaptation methods. We additionally note the total runtime of fine-tuning methods is closer to the feed-forward approaches because their training is done in a standard way. All experiments within the same benchmark used the same GPU and number of CPUs.

D. Additional analyses

The additional analyses use pre-trained models that were trained with tasks that have 100 support examples coming from 5 domains. We then deploy them to tasks that have 1) variable number of support domains or 2) variable support set sizes. Tables 4 and 5 show that the best performance is obtained when there are fewer domains and that CXDA can handle well also cases when there is a large number of domains. Tables 6 and 7 confirm CXDA is scalable and outperforms other approaches even if the support set size changes to more or fewer examples. Since the latent and continual domain adaptation methods have not shown to be promising in the main setup, we do not evaluate them as part of the additional analyses.

E. Patch-to-patch attention

We provide additional comparison of image-to-image and patch-to-patch (P2P) attention in Table 8. Compared to standard image-to-image CXDA, CXDA P2P performs slightly worse, likely because using whole images is simpler and provides useful regularization. In terms of time, the detailed comparison depends on the setup, but overall both need relatively similar time in practice.

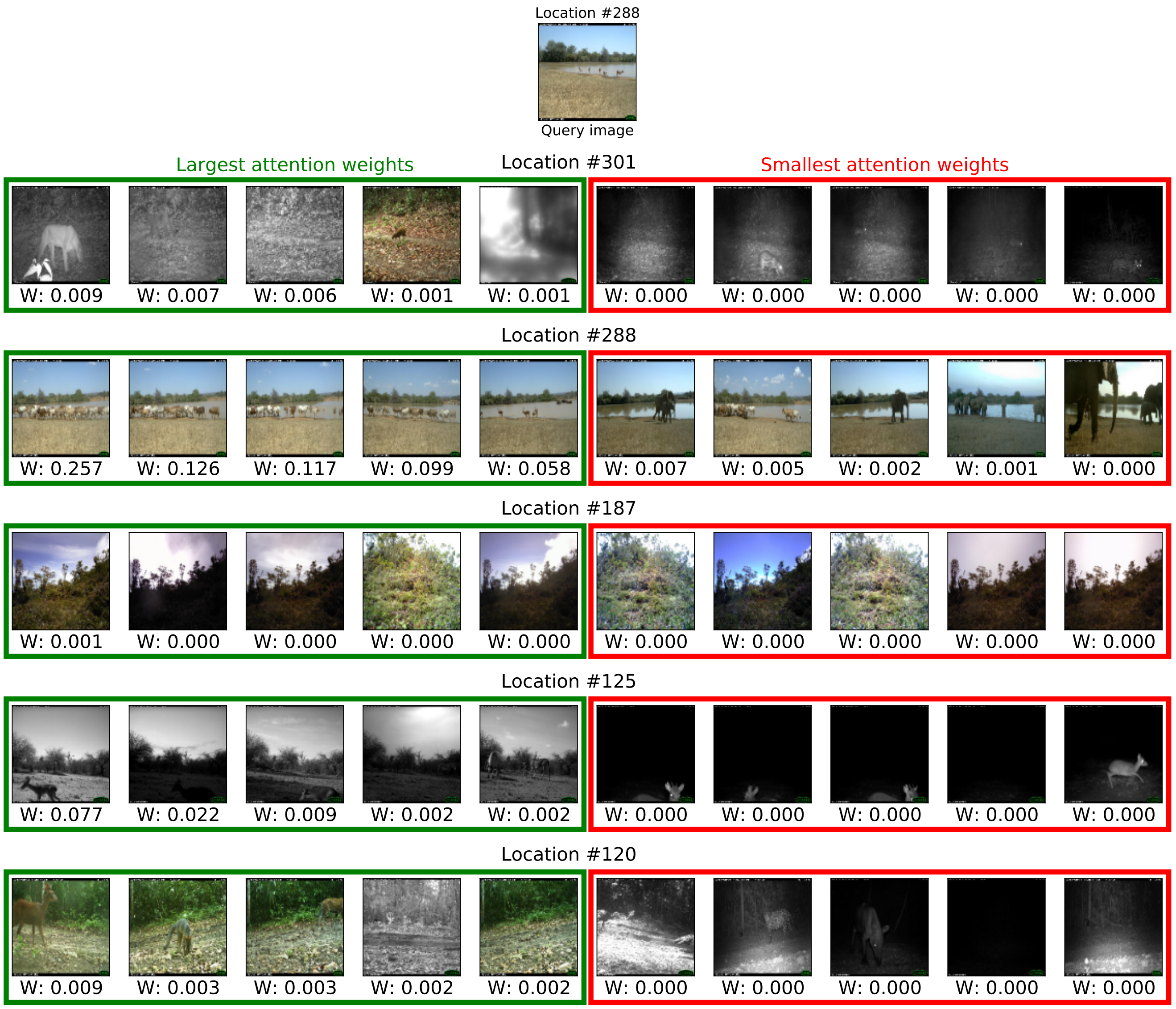

Figure 5. Analysis of attention weights for an example task in iWildCam, with a query image coming from location (camera trap) #288. We show the five support examples in each domain that have the largest and smallest attention weights. Similar images from the same location (#288) are given the largest weights, but also relevant images from other locations (e.g. #125) are given larger weights. The examples with the smallest attention weights visually do not seem relevant.

	Time for adaptation and inference on a task (ms/task)				Total runtime for pre-training and adaptation (minutes)			
Approach	FEMNIST	CIFAR-C	TinyImageNet-C	iWildCam	FEMNIST	CIFAR-C	TinyImageNet-C	iWildCam
ERM	12.0 ± 0.1	15.7 ± 0.3	52.8 ± 0.2	352.0 ± 2.8	38.3 ± 0.3	26.6 ± 0.2	94.1 ± 0.2	440.2 ± 5.4
CML	13.1 ± 0.2	15.8 ± 0.1	48.9 ± 0.1	385.2 ± 8.0	55.2 ± 1.0	35.2 ± 0.2	131.8 ± 0.2	461.9 ± 7.2
BN	16.5 ± 0.7	19.2 ± 0.2	74.1 ± 0.1	345.2 ± 1.9	44.5 ± 0.3	31.6 ± 0.3	112.3 ± 0.3	432.2 ± 0.8
CXDA	17.8 ± 1.5	20.5 ± 0.2	77.9 ± 2.8	392.5 ± 1.3	110.0 ± 12.0	63.2 ± 0.6	167.9 ± 1.9	491.6 ± 1.0
FT-EM	352.7 ± 42.4	387.1 ± 10.5	840.2 ± 15.4	2044.7 ± 238.0	97.5 ± 6.4	156.6 ± 3.6	483.4 ± 4.9	907.1 ± 28.0
FT-IM	296.9 ± 3.1	385.9 ± 10.9	830.7 ± 16.6	1709.8 ± 16.5	78.2 ± 0.4	150.2 ± 3.9	470.8 ± 7.3	688.0 ± 14.8
SF-OCDA	294.0 ± 7.5	488.0 ± 8.8	980.8 ± 8.8	1708.6 ± 1.9	77.7 ± 1.5	229.5 ± 34.3	630.8 ± 23.8	599.8 ± 3.1
CoTTA	1126.2 ± 177.4	1390.9 ± 20.1	5876.5 ± 196.8	8790.9 ± 335.7	209.3 ± 29.1	492.2 ± 7.0	3009.4 ± 93.4	1823.5 ± 85.8
SLA	25.5 ± 0.8	25.1 ± 0.0	217.6 ± 0.5	317.3 ± 5.2	85.5 ± 2.3	49.7 ± 0.1	367.9 ± 3.9	489.4 ± 5.8

Table 3. Comparative computational cost of different adaptation methods for adaptation and pre-training.

Approach	1 domain	2 domains	5 domains	10 domains	20 domains
ERM	58.3 ± 0.4	54.2 ± 0.3	44.3 ± 0.5	37.9 ± 0.5	29.4 ± 0.4
CML	57.7 ± 0.7	54.3 ± 0.6	44.8 ± 0.5	39.2 ± 0.4	30.6 ± 0.4
BN	59.7 ± 0.5	55.4 ± 0.5	45.4 ± 0.7	38.9 ± 0.7	30.2 ± 0.6
CXDA	62.7 ± 0.4	58.7 ± 0.3	49.4 ± 0.6	43.0 ± 0.5	33.2 ± 0.3
FT-EM	49.4 ± 0.6	49.0 ± 0.5	44.9 ± 0.6	39.1 ± 0.5	30.5 ± 0.4
FT-IM	53.1 ± 0.5	50.7 ± 0.5	45.6 ± 0.5	39.6 ± 0.5	30.7 ± 0.4

Table 4. Analysis of the impact of variable number of domains in the support set – worst 10% tasks test accuracy on CIFAR-C (%).

Approach	1 domain	2 domains	5 domains	10 domains	20 domains
ERM	68.7 ± 0.3	68.7 ± 0.3	68.6 ± 0.3	68.5 ± 0.2	68.6 ± 0.2
CML	68.8 ± 0.6	69.2 ± 0.5	69.5 ± 0.5	69.4 ± 0.4	69.5 ± 0.4
BN	69.7 ± 0.4	69.4 ± 0.4	69.3 ± 0.4	69.1 ± 0.4	69.2 ± 0.4
CXDA	72.2 ± 0.2	72.1 ± 0.2	72.0 ± 0.3	71.9 ± 0.3	71.9 ± 0.3
FT-EM	69.0 ± 0.4	69.1 ± 0.3	69.2 ± 0.4	69.2 ± 0.3	69.3 ± 0.3
FT-IM	69.8 ± 0.3	69.6 ± 0.3	69.5 ± 0.3	69.4 ± 0.3	69.5 ± 0.3

Table 5. Analysis of the impact of variable number of domains in the support set – average test task accuracy on CIFAR-C (%).

Approach	10 examples	20 examples	50 examples	100 examples	200 examples	500 examples
ERM	0.1 ± 0.0	21.7 ± 0.1	38.2 ± 0.5	44.3 ± 0.5	48.2 ± 0.6	50.2 ± 0.7
CML	0.0 ± 0.0	21.7 ± 0.2	38.9 ± 0.5	44.8 ± 0.5	48.6 ± 0.5	50.6 ± 0.4
BN	1.2 ± 1.0	22.8 ± 0.9	39.5 ± 0.7	45.4 ± 0.7	49.4 ± 0.7	51.3 ± 0.7
CXDA	1.8 ± 0.9	26.4 ± 0.7	43.0 ± 0.5	49.4 ± 0.6	53.8 ± 0.6	56.2 ± 0.8
FT-EM	0.8 ± 0.7	22.4 ± 0.6	39.0 ± 0.5	44.9 ± 0.6	48.9 ± 0.5	50.8 ± 0.5
FT-IM	1.3 ± 0.9	22.9 ± 0.8	39.7 ± 0.5	45.6 ± 0.5	49.6 ± 0.5	51.7 ± 0.4

Table 6. Analysis of the impact of variable number of examples in the support set – worst 10% tasks test accuracy on CIFAR-C (%).

Approach	10 examples	20 examples	50 examples	100 examples	200 examples	500 examples
ERM	68.7 ± 0.3	68.7 ± 0.3	68.6 ± 0.3	68.6 ± 0.3	68.6 ± 0.3	68.6 ± 0.3
CML	67.7 ± 0.3	68.7 ± 0.4	69.3 ± 0.4	69.5 ± 0.5	69.5 ± 0.5	69.6 ± 0.5
BN	69.3 ± 0.4	69.3 ± 0.4	69.2 ± 0.4	69.3 ± 0.4	69.2 ± 0.4	69.3 ± 0.4
CXDA	69.6 ± 0.3	70.9 ± 0.3	71.7 ± 0.3	72.0 ± 0.3	72.1 ± 0.2	72.2 ± 0.3
FT-EM	69.1 ± 0.3	69.2 ± 0.3	69.1 ± 0.3	69.2 ± 0.4	69.2 ± 0.3	69.2 ± 0.3
FT-IM	69.4 ± 0.3	69.5 ± 0.3	69.4 ± 0.3	69.5 ± 0.3	69.4 ± 0.3	69.5 ± 0.3

Table 7. Analysis of the impact of variable number of examples in the support set – average test task accuracy on CIFAR-C (%).

	FEMNIST		**CIFAR-C**		**TinyImageNet-C**		**iWildCam**	
Approach	**W10%**	**Avg**	**W10%**	**Avg**	**W10%**	**Avg**	**W10%**	**Avg**
ERM	52.7 ± 1.4	77.2 ± 0.9	44.3 ± 0.5	68.6 ± 0.3	4.8 ± 0.2	26.4 ± 0.4	0.0 ± 0.0	38.7 ± 0.8
CXDA	53.3 ± 0.6	78.3 ± 0.0	49.4 ± 0.6	72.0 ± 0.3	6.5 ± 0.2	28.6 ± 0.3	3.6 ± 1.5	43.5 ± 1.5
CXDA P2P	52.0 ± 0.7	77.3 ± 0.3	45.6 ± 0.3	69.8 ± 0.2	6.8 ± 0.2	28.9 ± 0.1	4.1 ± 0.9	42.9 ± 1.3

Table 8. Comparison of image-to-image and patch-to-patch (P2P) attention: average and worst-case (worst 10% tasks) test performance, with standard error of the mean across 3 random seeds. Accuracy is reported for all except iWildCam, where F1 score is used (%). Our image-to-image CXDA performs better than the patch-to-patch alternative in general.