Appendix Contents

A. Sharpness-aware Optimization

In order to properly finetune the new module $\theta(t)$ with the pseudo-label entropy minimization, we seek to make the model insensitive to large gradients by encouraging the model to converge to a flat area of the entropy loss surface, since a flat minimum leads to good generalization and robustness to large gradients [37, 38]:

$$\min_{\lambda} \mathcal{L}^{SA(t)}(\{x_i^{(t)}\}_{i=1}^B; \lambda), \tag{A}$$

$$\text{where } \mathcal{L}^{SA(t)} \triangleq \max_{\|\epsilon\|_2 \leq \rho} \mathcal{L}^{(t)}(\{x_i^{(t)}\}_{i=1}^B; \lambda + \epsilon) \tag{B}$$

in which $\mathcal{L}^{(t)}$ is defined in (5) in the main paper. In this context, the inner optimization aims to discover a perturbation ϵ of the module parameter $\theta(t)$ within a Euclidean ball of radius ρ that maximizes entropy. The degree of sharpness is measured by the maximum change in the Euclidean ball neighbourhood $N_\rho(\lambda)$. This bi-level problem incentivizes the optimization process to locate flat minima. Following SAM [37], we can approximately solve the inner optimization via a first-order Taylor expansion:

$$\begin{aligned}
\epsilon^*(\lambda) &\triangleq \arg\max_{\|\epsilon\|_2 \leq \rho} \mathcal{L}^{(t)}(\{x_i^{(t)}\}_{i=1}^B; \lambda + \epsilon) \\
&\approx \arg\max_{\|\epsilon\|_2 \leq \rho} \mathcal{L}^{(t)}(\{x_i^{(t)}\}_{i=1}^B; \lambda) + \epsilon^\intercal \nabla_\lambda \mathcal{L}^{(t)}(\{x_i^{(t)}\}_{i=1}^B; \lambda) \\
&= \arg\max_{\|\epsilon\|_2 \leq \rho} \epsilon^\intercal \nabla_\lambda \mathcal{L}^{(t)}(\{x_i^{(t)}\}_{i=1}^B; \lambda)
\end{aligned}$$

Let $\mathbf{v} = \nabla_\lambda \mathcal{L}^{(t)}(\{x_i^{(t)}\}_{i=1}^B; \lambda)$. Hölder's inequality implies that $\epsilon^\top \mathbf{v} \leq \|\epsilon\|_p \|\mathbf{v}\|_q \leq \rho \|\mathbf{v}\|_q$ $(1/p + 1/q = 1)$. For $p = q = 2$, the linear function achieves that bound $\epsilon^*(\lambda)^\intercal \mathbf{v} = \rho \|\mathbf{v}\|_2$, where $\epsilon^*(\lambda) = \rho \cdot \text{sgn}(\mathbf{v}) \cdot \frac{|\mathbf{v}|}{\|\mathbf{v}\|_2}$.
By substituting $\epsilon^*(\lambda)$ back into Eqn. 7 and differentiating both sides, the final gradient approximation is:

$$\nabla_\lambda \mathcal{L}^{SA(t)} \approx \nabla_\lambda \mathcal{L}^{(t)}(\{x_i^{(t)}\}_{i=1}^B; \lambda)|_{\lambda + \epsilon^*(\lambda)}. \tag{C}$$

B. Synthetic data generation

The procedure is outlined in Alg. A. In the first two steps, synthetic data points $z_1, \ldots, z_M$ are selected independently of the private dataset, relying solely on the database size N. In the main paper, we use $q \sim \mathcal{N}(0, 1)$. Steps 3 and 4 involve constructing the linear subspace $\mathcal{H}_M$ of $\mathcal{H}$ spanned by the feature maps of these synthetic points and computing a finite basis for this subspace. The private data is then accessed to calculate the empirical KME $\hat{\mu}_X$ (step 5), which is subsequently projected onto the subspace $\mathcal{H}_M$ and represented using the precomputed basis (step 6-7). The algorithm ensures that the number of synthetic data points M increases to infinity as N approaches infinity (step 1), guaranteeing that Algorithm 1 yields a consistent estimator of the true KME

Algorithm A Synthetic Data Subspace of the RKHS

Require: Dataset $\mathcal{D} = \{x_1, \ldots, x_N\} \subset \mathcal{X}$, kernel k on $\mathcal{X}$, number of synthetic data points M

Ensure: Weighted synthetic dataset (representing an estimate of μ_X in the RKHS $\mathcal{H}$ of k)

1: Initialize $z_1, \ldots, z_M$ deterministically or randomly from some distribution q on $\mathcal{X}$

2: $\mathcal{H}_M \leftarrow \text{Span}(\{k(z_1, \cdot), \ldots, k(z_M, \cdot)\}) \subseteq \mathcal{H}$

3: $b_1, \ldots, b_F$ form an orthonormal basis of $\mathcal{H}_M$ (obtained using Gram-Schmidt process)

4: $\hat{\mu}_X \leftarrow \frac{1}{N}\sum_{n=1}^{N} k(x_n, \cdot)$, empirical KME of X in $\mathcal{H}$

5: $\bar{\mu}_X \leftarrow \sum_{f=1}^{F} \langle b_f, \hat{\mu}_X \rangle b_f = \sum_{f=1}^{F} \alpha_f b_f$, projection of $\hat{\mu}_X$ onto $\mathcal{H}_M$

6: Re-express $\bar{\mu}_X \leftarrow \sum_{f=1}^{F} \beta_f b_f = \sum_{m=1}^{M} w_m k(z_m, \cdot)$ in terms of $k(z_m, \cdot)$

7: **return** $(z_1, w_1), \ldots, (z_M, w_M)$

μ_X, provided that the synthetic data points are sampled from a distribution with sufficiently large support.

Lemma B.1. *([44], Lemma 10) Let $\mathcal{X}$ be a compact metric space and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a continuous kernel on $\mathcal{X}$. Suppose that the synthetic data points $z_1, z_2, \ldots$ are sampled i.i.d. from a probability distribution q on $\mathcal{X}$. If the support supp$(\mathcal{X})$ of $\mathcal{X}$ is included in the support of q, then*

$$\|\bar{\mu}_X - \hat{\mu}_X\|_{\mathcal{H}} \xrightarrow{p} 0 \text{ as } N \to \infty. \tag{D}$$

Proof Sketch. Let $\epsilon > 0$. Since k is continuous on the compact space $\mathcal{X} \times \mathcal{X}$, it is uniformly continuous. Therefore, there exists $\delta > 0$ such that

$$|k(x, x') - k(y, y')| < \epsilon^2 \quad \text{whenever} \quad \|x - y\| + \|x' - y'\| < \delta.$$

The compactness of $\mathcal{X}$ implies it is totally bounded; thus, supp$(\mathcal{X})$ can be covered by finitely many balls $B_1, \ldots, B_K$ of radius $\delta/2$. Since supp$(\mathcal{X}) \subseteq$ supp(q) and each $q(B_k) > 0$, with high probability, each B_k contains at least one sample point z_m as $M \to \infty$.

For each x_n, select $z_{m(n)}$ within distance δ (possible due to the coverage). Then,

$$\|\hat{\mu}_X - \bar{\mu}_X\|_{\mathcal{H}} \leq \frac{1}{N}\sum_{n=1}^{N} \left\|k(x_n, \cdot) - k(z_{m(n)}, \cdot)\right\|_{\mathcal{H}}$$

$$= \frac{1}{N}\sum_{n=1}^{N} \left(k(x_n, x_n) - 2k(x_n, z_{m(n)}) + k(z_{m(n)}, z_{m(n)})\right)^{1/2}$$

$$< \epsilon.$$

The last inequality follows from the uniform continuity of k and the choice of δ. Therefore, as $N \to \infty$, we have $\|\hat{\mu}_X - \bar{\mu}_X\|_{\mathcal{H}} \xrightarrow{p} 0$. □

C. Efficiency of AdMiT

C.1. Parameter Efficiency

Table A. **Number of parameters** for different models' scales and their corresponding PET module size. ViT-T/S/B/L stands for "Tiny, Small, Base, Large", corresponding to different pretrained ViT sizes. The bolded number is the size of the PET modules we applied in our main paper experiments.

	ViT-T	ViT-S	ViT-B	ViT-L
Full model	5,543,716	21,704,164	85,875,556	303,404,132
Adapter	58,564	116,932	233,668	417,984
LoRA	93,028	185,956	**371,812**	888,932
VPT	37,732	75,364	150,628	299,108
Header	19,300	38,500	76,900	102,500

Table B. **Computational cost.** We evaluate the computational cost and adaptation accuracy of multi-source adaptation for the baseline methods compared to AdMiT on ViT-B/16, using a batch size of 128 and LoRA PET modules (as detailed in the main paper), under both static (S) and dynamic (D) adaptation settings.

Tuning Method (PET module: LoRA)		Avg. Tuned Params During Adaptation (M)	GFLOPs (Avg.)	Image Classification Avg. Acc.(BS=128) ImageNet-C (S)	Digits (S)	CIFAR-100 (D)
Full	-	85.87	17.58	61.9	84.1	45.3
Multi	π-tuning	0.25	18.24	62.4	85.8	64.5
	SESoM	1.16	18.32	62.6	86.0	60.5
	CONTRAST	0.25	19.21	63.1	86.1	72.5
	Model soup	0.23	22.14	53.4	69.7	57.2
	AdMiT	0.23	17.62	63.7	86.8	73.3
	AdMiT-ZeroShot	0	17.62	59.6	82.2	59.1

Table C. **Latency from KME matching**. We evaluate the adaptation speed on ViT-B/16 backbone for Static adaptation on ImageNet-C among different PET methods, to demonstrate the latency brought by KME matching. The inference speed is defined by images per second (imgs/sec). All results are the average of 5 runs.

Method	GFLOPs (Avg across BS)	Adaptation speed (img/sec) BS=1	BS=32	BS=128	Slowdown Percentage (%) BS=1	BS=32	BS=128
Full fine-tuning	17.58	123.4	305.3	308.2	-	-	-
LoRA	17.58	95.7	291.3	283.1	-	-	-
LoRA-AdMiT	17.62	94.5	289.5	282.6	1.26	0.62	0.18
Adapter	17.81	104.2	285.6	296.3	-	-	-
Adapter-AdMiT	17.85	102.5	285.1	295.9	1.63	0.18	0.13
VPT	18.32	117.3	248.3	251.4	-	-	-
VPT-AdMiT	18.38	116.6	247.5	250.7	0.60	0.32	0.28

Parameter-efficient tuning (PET) methods align naturally with model ensemble techniques[7], particularly in terms of parameter efficiency. In contrast to other models where an ensemble of N models results in N times more module parameters, the additional module parameters introduced by the module integration in Alg. 1 is only of the size of one PET module. This represents less than 0.4% of a pretrained ViT-base model ($\approx$ 86M, w.r.t. Table. A).

C.2. Computational Efficiency

In Table B, Table C, and Table D, we present a comprehensive comparison of inference speeds and adaptation performance across various benchmarks. As shown in Table D, we

Table D. **Edge device performance evaluation:** Adaptation performance and inference speed comparison on *Raspberry Pi 5* using Digit-5 dataset ($N = 4$) and ImageNet-C ($N = 14$), with the same experiment setting as the Figure 2 of the main paper. Existing PET methods exhibit slower adaptation speed than full fine-tuning despite tuning fewer parameters, as they require extra gradient propagation through the PET modules. Different source-target matching approaches further introduce varying computational overhead.

Tuning method PET module: LoRA	Num of Tuned Prams (M)	Adaptation Speed (img/100 second)	Avg. Acc. (BS=128) Digit-5	ImageNet-C
Full model	85.87	42	83.5	61.4
π-tuning	0.25	34	83.2	62.1
SESoM	1.16	25	84.1	62.3
CONTRAST	0.25	37	84.4	62.8
AdMiT	0.23	40	**85.3**	**63.5**
AdMiT-ZeroShot	**0**	**145**	81.5	58.2

evaluate the computational efficiency of different methods on edge devices (Raspberry Pi 5) using Digit-5 ($N = 4$) and ImageNet-C ($N = 14$) datasets.

The results reveal that existing PET methods exhibit slower adaptation speeds than full fine-tuning despite tuning fewer parameters, as they require extra gradient propagation through the PET modules. Our method, AdMiT, demonstrates (Table C) superior computational efficiency during deployment with only a minimal slowdown (less than 2%) compared to standard PET methods due to the empirical KME calculation for module matching.

Notably, AdMiT achieves the highest accuracy on both benchmarks (85.3% on Digit-5 and 63.5% on ImageNet-C) while maintaining competitive adaptation speed (40 img/100 second). For scenarios where fine-tuning is not feasible, AdMiT-ZeroShot enables remarkably efficient adaptation through direct weighted combination of stored modules, achieving 81.5% and 58.2% accuracy on Digit-5 and ImageNet-C respectively, while offering the fastest inference speed (145 img/100 second) with zero tunable parameters.

Unlike existing methods that require extra tuning parameters or routing network optimization, AdMiT's matching and weight assignment rely solely on empirical KME calculations, making our method more computationally efficient during deployment while achieving superior adaptation performance.

D. Additional results

As shown in table. A, we use LoRA in the main paper as PET modules. We provide the results for other PET modules in this sections.

D.1. Details on different PET methods

Visual-Prompt Tuning (VPT). With a pre-trained Transformer (ViT) model as our starting point, we introduce a set of p continuous embeddings in the input space, each of dimension d, referred to as "prompts." During fine-tuning, only the prompts specific to the task are updated, while the Transformer backbone remains frozen. We applied VPT-Shallow[21] as follows:

Table E. **Results on Digits**, same setup as for Fig 2.We train the source modules using 4 digits datasets to perform adaptation on the remaining dataset. All the results are the average of 5 runs. **Best** performance is bolded, and second-best performance is underlined. ($M = N = 4$).

PET Module	Source	Method	MM	MT	UP	SV	SY	Avg
LoRA	Single	GT-Tuning	**70.4**	**99.6**	**93.1**	**90.5**	**97.2**	**90.2**
	Multi	π-tuning-PL	61.3	98.2	89.1	86.2	94.6	85.9
		SESoM-PL	62.1	98.3	88.6	87.1	94.5	86.1
		Model soup	52.3	83.2	71.5	66.4	75.2	69.7
		AdMiT	63.2	98.9	89.3	86.5	96.2	86.8
Adapter	Single	GT-Tuning	**71.3**	**99.8**	**94.1**	**89.9**	**97.0**	**90.4**
	Multi	π-tuning-PL	62.0	97.5	88.8	86.0	94.5	85.8
		SESoM-PL	61.2	97.5	89.4	85.3	94.7	85.6
		Model soup	53.3	83.2	70.0	65.1	74.9	69.3
		AdMiT	63.9	97.8	88.9	85.0	94.4	86.0
VPT	Single	GT-Tuning	**71.6**	**99.7**	**92.5**	**92.6**	**97.9**	**90.9**
	Multi	π-tuning-PL	63.5	98.8	88.6	85.4	96.1	86.5
		SESoM-PL	64.2	98.3	88.3	88.7	93.8	86.7
		Model soup	51.8	82.5	71.9	66.4	75.4	69.6
		AdMiT	65.7	99.4	89.9	88.5	97.2	88.1

VPT-Shallow. Prompts are inserted into the first Transformer layer only. Each prompt token is a learnable $d-$dimensional vector. A module, which is a collection of p (which is the prompt length) prompts is denoted as $\mathbf{P} = \{\mathbf{v} \in \mathbb{R}^d | k \in \mathbf{N}, 1 \leq k \leq p\}$, the shallow-prompted ViT is:

$$[x_1, \mathbf{Z}_1, \mathbf{E}_1] = L_1([x_0, \mathbf{P}, \mathbf{E}_0]) \quad \text{(E)}$$

$$[x_i, \mathbf{Z}_i, \mathbf{E}_i] = L_i([x_{i-1}, \mathbf{Z}_{i-1}, \mathbf{E}_{i-1}]), i = 2, \cdots, l \quad \text{(F)}$$

$$\mathbf{y} = \text{Head}(x_l), \quad \text{(G)}$$

where $\mathbf{Z}_i \in \mathbb{R}^{p \times d}$ represents the features computed by the i-th Transformer layer, and $[x_i, \mathbf{Z}_i, \mathbf{E}_i] \in \mathbb{R}^{(1+p+P) \times d}$ (P is the number of patches that a 2D image input is divided into). The colors above indicate learnable and frozen parameters. For ViTs, x_i is invariant to the location of prompts since they are inserted after positional encoding. The overall parameter count for Adapters in an l-layer Transformer can be calculated as $|\theta| = p \times d$.

Adapter. In the conventional configuration, a transformer model incorporates two Adapters per layer [51]. Each Adapter layer is composed of $2 \times k \times d$ parameters, accounting for both the down and up-projection matrices. Here, k represents the input dimension size, while d refers to the bottleneck dimension of the Adapter. The overall parameter count for Adapters in an l-layer Transformer can be calculated as $|\theta| = 2 \times l \times 2 \times k \times d$.

Results for different PET modules. Following the setup in main paper, we replace LoRA with Adapters,VPTs as the PET module, and demonstrate the stationary distribution adaptation results in Table. E. In can be concluded that our method, AdMiT, retains good performance across different PET modules.

E. Semantic Segmentation

AdMiT is not limited to image classification tasks and can be seamlessly extended to tasks like semantic segmentation. In this setting, we assume access to a collection of pre-trained PET modules $\{\theta^j\}_{j=1}^N$, where each module is fine-tuned on a distinct source distribution for pixel-wise classification. Specifically, each module outputs per-pixel probabilities for K classes, formulated as $f_{\theta^j} : \mathbb{R}^{H\times W} \rightarrow \mathbb{R}^{H\times W\times K}$. To adapt AdMiT for semantic segmentation, the entropy term in Eqn. 5 of the main paper is updated as follows:

$$\mathcal{L}_w^{(t)}(\mathbf{w}) = -\mathbf{E}_{\mathcal{D}_T^{(t)}} \sum_{h=1}^{H}\sum_{w=1}^{W}\sum_{c=1}^{K} \hat{y}_{\text{hwc}}^{(t)} \log(\hat{y}_{\text{hwc}}^{(t)}) \qquad \text{(H)}$$

Here, $\hat{y}_{\text{hwc}}^{(t)}$ represents the weighted probability output corresponding to class c for the pixel at location (h, w) at time-step t. The rest of the framework remains unchanged, ensuring consistency across tasks.

E.1. Datasets

Our experiments involve the following datasets:

• **Cityscapes:** The Cityscapes dataset [19] provides large-scale, densely annotated pixel-level data for 30 classes, grouped into 8 categories: flat surfaces, humans, vehicles, constructions, objects, nature, sky, and void. Simulated variants of this dataset include fog and rain conditions [52, 53].

• **ACDC:** The Adverse Conditions Dataset (ACDC) [20] includes pixel-level annotations for images captured under adverse conditions such as fog, nighttime, rain, and snow. The class structure aligns with the 19 semantic labels used in the Cityscapes evaluation, excluding the void class.

E.2. Experimental Setup

For all experiments, we use HRViT-b1 [54] as the segmentation model. We evaluate performance on 19 semantic labels, excluding the void label.

We consider a static target distribution setting for evaluation. Specifically, we train three source PET modules using the **clean**, **fog**, and **rain** splits of the Cityscapes dataset. After training, these modules are tested on the respective weather condition splits of the ACDC dataset. Using AdMiT, we dynamically integrate the source modules for each target condition and compare the results with baseline methods.

E.3. Results

Results on Cityscapes to ACDC: Table F shows the performance of AdMiT and baseline methods on ACDC test data under different weather conditions (static target distributions). The source modules are trained on Cityscapes and its simulated noisy variants. AdMiT significantly outperforms baseline adaptation methods, with results reported in terms of % mIoU, highlighting its effectiveness in leveraging multi-source knowledge for target adaptation.

Table F. **Semantic segmentation results.**

Source	Method	Fog	Rain	Snow	Night	Avg
Single	TENT-Best	25.4	21.7	19.7	13.5	20.0
	BECoTTA-Best	26.3	22.4	21.3	14.5	21.1
	SAR-Best	25.8	22.2	20.1	15.5	20.9
Multi	π-tuning-PL	28.3	23.0	24.2	17.4	23.2
	SESoM-PL	29.2	25.3	25.4	18.2	24.5
	CONTRAST	<u>32.4</u>	<u>29.4</u>	<u>25.2</u>	<u>18.7</u>	<u>26.4</u>
	Model soup	25.4	25.5	21.4	14.7	21.8
	AdMiT	**32.5**	**29.9**	**25.4**	**19.1**	**26.7**
	AdMiT-ZeroShot	27.5	22.3	19.9	14.7	21.1

F. Implementation details

We perform all the experiment on a single A100 GPU. We use ViT-Base-16 [55] model in all our experiments. For all experiments without extra clarification, we use a target batch size of $|T| = 128$, as used by TENT [30]. The experimental setup for tuning the integrated module is listed as in Table. G summarizing the optimization configurations we used. Implementation details for each tuning method apply to both source and target distributions.

In this problem setting, we propose to adaptively combine multiple pre-trained parameter-efficient tuning (PET) modules during deployment through suitable combination weights, which are determined based on a limited number of target samples. Consider the scenario where we have a collection of N pre-trained PET modules, denoted as $\{\theta^j\}_{j=1}^N$, which are fine-tuned on distinct source distributions. During deployment, target data arrives in an online fashion as a sequence of batches $\{x_i^{(1)}\}_{i=1}^B \rightarrow \{x_i^{(2)}\}_{i=1}^B \rightarrow \ldots \{x_i^{(t)}\}_{i=1}^B \rightarrow \ldots$, where t represents the time-stamp and B is the number of samples in each target batch. The target distribution at time-stamp t is denoted as $\mathcal{D}_T^{(t)}$, implying $\{x_i^{(t)}\}_{i=1}^B \sim \mathcal{D}_T^{(t)}$.

Motivated by the multi-source adaptation framework, we model the target distribution at each time-stamp t as a linear combination of source distributions, with combination weights denoted as $\{w_j^{(t)}\}_{j=1}^N$. Using these weights, AdMiT integrates the pre-trained PET modules to form an adaptive module for the current target batch. Thus, the inference model for test batch t can be expressed as $f_T^{(t)} = f_{\theta(t)}$, where $\theta(t) = \sum_{j=1}^N w_j^{(t)}\theta^j$ is the dynamically integrated PET module for time-stamp t.

We implement the baselines as follows:

- **TENT.** TENT [30] adapts transformers by modifying only the LayerNorm statistics during test-time adaptation while keeping the PET modules and backbone weights unchanged. It minimizes the entropy of predictions for target batches, encouraging confident predictions. The LayerNorm parameters (mean and variance) are updated

Table G. **Hyperparameters** for tuning AdMiT

	Full, Adapter, LoRA	VPT
Optimizer	AdamW	SGD
Optimizer momentum	N/A	0.9
$base\_lr$ search range	{0.001, 0.0001, 0.0005, 0.005}	{50., 25., 10., 5., 2.5, 1.,0.5, 0.25, 0.1, 0.05}
Weight decay range	{0.01, 0.001, 0.0001, 0.0}	
LR schedule	cosine decay	
Warm up epochs	10	
Total epochs	100	

iteratively using gradients computed from the entropy loss. Key hyperparameters include the learning rate for updating LayerNorm statistics (1×10^{-4}) and the batch size for target adaptation ($B = 64$).

- **BECoTTA.** BECoTTA(-M) [33] integrates multiple PET modules using a MoDE (Mixture of Domain Experts) module, which applies a Top-K routing strategy to select the $K = 2$ most relevant PET modules based on input features. During pretraining, BECoTTA initializes with $D = 3$ proxy domains (source domain, darkness, and brightness) and trains the MoDE module alongside a domain discriminator and a synergy loss, freezing the backbone. In deployment, PET modules are activated for online adaptation, with entropy-based filtering used to refine pseudo-labels. The primary hyperparameters include the number of proxy domains ($D = 3$) and the Top-K selection parameter ($K = 2$).
- **SAR.** SAR [38] adapts transformers by restricting updates to LayerNorm statistics during test-time adaptation while freezing PET modules and backbone weights. Unlike TENT, SAR selectively filters high-entropy (low-confidence) samples, focusing on reliable predictions. Sharpness-aware optimization is applied to smooth the entropy loss, improving robustness against noisy target distributions. The key hyperparameters include the entropy threshold for filtering ($E_0 = 0.4 \times \ln(K)$, where K is the number of classes) and the sharpness radius ($\rho = 0.05$) for optimization.
- **π-Tuning.** π-Tuning [16] combines knowledge from multiple pre-trained PET modules by interpolating their outputs based on task similarity. Task embeddings are computed using the Fisher Information Matrix (FIM), with similarity calculated as cosine similarity between the embeddings of target and source tasks. The top k PET modules are selected for interpolation, and weights are optimized via pseudo-label entropy minimization. Key hyperparameters include the number of selected PET modules ($k = 3$), learning rate for fine-tuning (1×10^{-4}).
- **SESoM.** SESoM [12] integrates the outputs of multiple source models through the utilization of an additional attention-based routing network. Within our experimental framework, each PET module operates as a source-specific unit. The logits derived from these modules are transmitted to an attention-based routing network tasked with computing sample-specific weights. The routing network undergoes fine-tuning via pseudo-label entropy minimization, employing few-shot pseudo-labeled target data while maintaining the PET modules and backbone architecture in a fixed state. Key hyperparameters comprise the learning rate for the attention module (3×10^{-4}) and a dropout rate set at 0.1. An attentionn-based routing network of approximately $d \times d'_x + d'_x \times d' + v \times d'_l + d'_l \times d' + 4d' = 0.85M$ size (as defined in [12]) is employed to derive the attention weights.
- **CONTRAST.** CONTRAST [15] adapts to evolving target distributions by dynamically combining pre-trained source models and selectively updating the most relevant model. For ViT-based architectures, CONTRAST computes weights for PET modules based on LayerNorm statistics or feature embeddings and updates the PET module with the highest weight for each target batch. Key hyperparameters include learning rate for weight updates (1×10^{-4}), and test batch size ($B = 128$).
- **Model Soup.** Model Soup [46] improves performance by averaging the weights of multiple fine-tuned PET modules without additional inference cost. We adapt Model Soup by sequentially adding PET modules to the soup using the Greedy Soup strategy, retaining only modules that improve validation accuracy. Hyperparameters include learning rates ($\{10^{-4}, 10^{-5}\}$) and using 10% of training data for validation.

G. Details of dataset corruptions

We summarize these corruptions types by example in Fig. A. The order of these corruptions is the same as the order in Table. 2 and Figure. 3.

H. Theoretical Insights

H.1. Maximum Mean Discrepancy as distribution similarity metric

The objective of module selection in AdMiT is to quantify the similarity between the source distribution $\mathcal{D}_S$ and

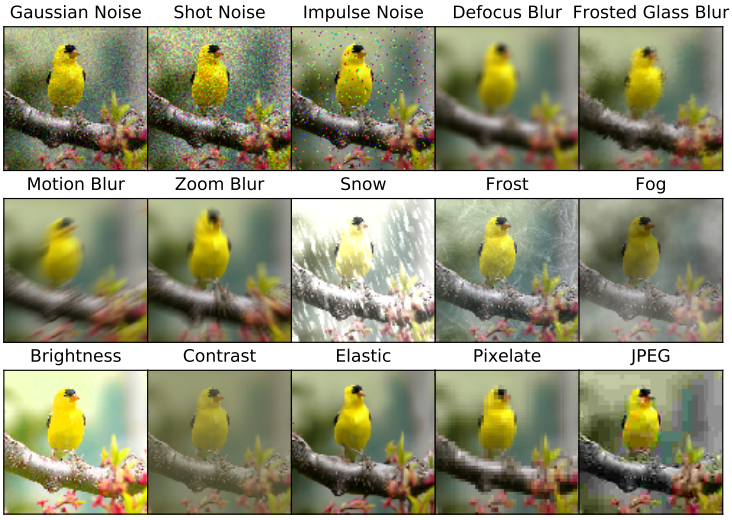

Figure A. Examples of each corruption type in the image corruptions benchmark. While synthetic, this set of corruptions aims to represent natural factors of variation like noise, blur, weather, and digital imaging effects.

the target distribution $\mathcal{D}_\mathcal{T}$ without access to the raw data. Rather than comparing moments of different orders, such as $\mathbf{E}_{\boldsymbol{x}\sim\mathcal{D}}(\boldsymbol{x}^n)$, we adopt a more comprehensive metric, $\mathbf{E}_{\boldsymbol{x}\sim\mathcal{D}}(f(\boldsymbol{x}))$, to universally characterize the properties of distributions.

The discrepancy between the expected function values across two distributions, $\mathcal{D}_\mathcal{T}$ and $\mathcal{D}_\mathcal{S}$, is captured by the Maximum Mean Discrepancy (MMD). Mathematically, MMD is defined in a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ as:

$$\text{MMD}(\mathcal{F}, \mathcal{D}_\mathcal{T}, \mathcal{D}_\mathcal{S}) = \sup_{f\in\mathcal{F}} \left(\mathbf{E}_{\boldsymbol{x}\sim\mathcal{D}_\mathcal{S}}(f(\boldsymbol{x})) - \mathbf{E}_{\boldsymbol{x}\sim\mathcal{D}_\mathcal{T}}(f(\boldsymbol{x}))\right).$$

Without loss of generality, f is assumed to reside within a unit ball , i.e., $||f||_\mathcal{H} \leq 1$. The RKHS is structured using an orthogonal basis derived from the decomposition of a symmetric and positive semi-definite kernel function $k(\boldsymbol{x}, \boldsymbol{y})$. In the main paper, we employ a Gaussian radial basis kernel $k(\boldsymbol{x}, \boldsymbol{y}) = \exp(-||\boldsymbol{x} - \boldsymbol{y}||^2)$.

A symmetric and positive semi-definite kernel function $k(\boldsymbol{x}, \boldsymbol{y})$ can be decomposed [40] into a set of eigenvalues $\{\lambda_i\}_{i=1}^\infty$ and corresponding orthogonal eigenfunctions $\{\psi_i(\cdot)\}_{i=1}^\infty$:

$$k(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^\infty \lambda_i \psi_i(\boldsymbol{x})\psi_i(\boldsymbol{y}).$$

These eigenfunctions form an orthogonal basis $\{\sqrt{\lambda_i}\psi_i(\cdot)\}$ used to construct the Hilbert space $\mathcal{H}$. Any function f within this space can be expressed either as a linear combination of these basis functions:

$$f(\cdot) = \sum_{i=1}^\infty f_i \sqrt{\lambda_i}\psi_i(\cdot),$$

or represented as an infinite-dimensional vector in $\mathcal{H}$: $f = (f_1, f_2, \ldots)_\mathcal{H}^\top$. When one parameter of the kernel function is fixed to $\boldsymbol{x}$, it behaves like a function with a single variable or an infinite vector:

$$k(\boldsymbol{x}, \cdot) = \sum_{i=1}^\infty \lambda_i \psi_i(\boldsymbol{x})\psi_i(\cdot) = (\sqrt{\lambda_1}\psi_1(\boldsymbol{x}), \sqrt{\lambda_2}\psi_2(\boldsymbol{x}), \ldots)_\mathcal{H}^\top.$$

This leads to the following computation for the inner product of these two functions, illustrating the reproducing property of RKHS:

$$\begin{aligned}\langle f, k(\boldsymbol{x}, \cdot)\rangle_\mathcal{H} &= (f_1, f_2, \ldots)_\mathcal{H} \cdot (\sqrt{\lambda_1}\psi_1(\boldsymbol{x}), \sqrt{\lambda_2}\psi_2(\boldsymbol{x}), \ldots)_\mathcal{H}^\top \\ &= \sum_{i=1}^\infty f_i \sqrt{\lambda_i}\psi_i(\boldsymbol{x}) = f(\boldsymbol{x}),\end{aligned}$$

which effectively captures the essence of the reproducing property within the RKHS framework.

Furthermore, for a given distribution $\mathcal{D}$, we introduce the kernel mean embedding (KME), defined as:

$$\mu_\mathcal{D} = \mathbf{E}_{\boldsymbol{x}\sim\mathcal{D}}[k(\boldsymbol{x}, \cdot)].$$

This allows the expressions within the Maximum Mean Discrepancy (MMD) to be rewritten in terms of inner products in the RKHS:

$$\begin{aligned}\mathbf{E}_{\boldsymbol{x}\sim\mathcal{D}}(f(\boldsymbol{x})) &= \mathbf{E}_{\boldsymbol{x}\sim\mathcal{D}}\langle f, k(\boldsymbol{x}, \cdot)\rangle_\mathcal{H} = \langle f, \mathbf{E}_{\boldsymbol{x}\sim\mathcal{D}} k(\boldsymbol{x}, \cdot)\rangle_\mathcal{H} \\ &= \langle f, \mu_\mathcal{D}\rangle_\mathcal{H}.\end{aligned}$$

In the context of MMD, we assess the supremum with

these inner products:

$$\begin{aligned}
&\text{MMD}(\mathcal{F}, \mathcal{D}_\mathcal{T}, \mathcal{D}_\mathcal{S}) = \sup_{||f||_\mathcal{H} \leq 1} (\mathbf{E}_{\boldsymbol{x} \sim \mathcal{D}_\mathcal{S}}(f(\boldsymbol{x})) - \mathbf{E}_{\boldsymbol{x} \sim \mathcal{D}_\mathcal{T}}(f(\boldsymbol{x})))\\
&= \sup_{||f||_\mathcal{H} \leq 1} \langle \mu_{\mathcal{D}_\mathcal{T}}, f \rangle_\mathcal{H} - \langle \mu_{\mathcal{D}_\mathcal{S}}, f \rangle_\mathcal{H}\\
&= \sup_{||f||_\mathcal{H} \leq 1} \langle \mu_{\mathcal{D}_\mathcal{T}} - \mu_{\mathcal{D}_\mathcal{S}}, f \rangle_\mathcal{H} \leq \sup_{||f||_\mathcal{H} \leq 1} ||\mu_{\mathcal{D}_\mathcal{T}} - \mu_{\mathcal{D}_\mathcal{S}}||_\mathcal{H} \cdot ||f||_\mathcal{H}\\
&= ||\mu_{\mathcal{D}_\mathcal{T}} - \mu_{\mathcal{D}_\mathcal{S}}||_\mathcal{H}.
\end{aligned}$$

We work with several source datasets $S_j = \{(x_i, y_i)\}_{i=1}^{|S_j|} \sim \mathcal{D}_{\mathcal{S}j}$, $j \in [N]$ in the pretraining stage, and an unlabeled target batch $T = \{(x_i, \cdot)\}_{i=1}^{|T|} \sim \mathcal{D}_\mathcal{T}$ in the deployment stage. The empirical KME at these phases can be estimated as:

$$\widehat{\mu(T)} = \frac{1}{|T|} \sum_{x_n \in T} k(x_n, \cdot), \quad \widehat{\mu(S_j)} = \frac{1}{|S_j|} \sum_{x_n \in S_j} k(x_n, \cdot). \tag{I}$$

We can also compute the squared Maximum Mean Discrepancy (MMD^2) by expanding the definition:

$$\begin{aligned}
&\text{MMD}^2(\mathcal{D}_\mathcal{T}, \mathcal{D}_{\mathcal{S}j}) = ||\mu_{\mathcal{D}_\mathcal{T}} - \mu_{\mathcal{D}_{\mathcal{S}j}}||_\mathcal{H}^2\\
&= ||\mu_{\mathcal{D}_\mathcal{T}}||_\mathcal{H}^2 - 2\langle \mu_{\mathcal{D}_\mathcal{T}}, \mu_{\mathcal{D}_{\mathcal{S}j}} \rangle_\mathcal{H} + ||\mu_{\mathcal{D}_{\mathcal{S}j}}||_\mathcal{H}^2\\
&= \mathbf{E}_{\boldsymbol{x},\boldsymbol{y} \sim \mathcal{D}_\mathcal{T}} k(\boldsymbol{x}, \boldsymbol{y}) - 2\mathbf{E}_{\boldsymbol{x} \sim \mathcal{D}_\mathcal{T}, \boldsymbol{y} \sim \mathcal{D}_{\mathcal{S}j}} k(\boldsymbol{x}, \boldsymbol{y})\\
&+ \mathbf{E}_{\boldsymbol{x},\boldsymbol{y} \sim \mathcal{D}_{\mathcal{S}j}} k(\boldsymbol{x}, \boldsymbol{y}),
\end{aligned}$$

which results in the following empirical estimate:

$$\begin{aligned}
&\widehat{\text{MMD}}^2(T, S_j)\\
&= \frac{1}{|T|^2} \sum_{x_n, x_m \in T} k(x_n, x_m) - \frac{2}{|T||S_j|} \sum_{x_n \in T, x_m \in S_j} k(x_n, x_m)\\
&+ \frac{1}{|S_j|^2} \sum_{x_n, x_m \in S_j} k(x_n, x_m).
\end{aligned} \tag{J}$$

We provide proofs for the following properties:

- If the kernel $k(\cdot, \cdot)$ is universal, the mapping from $\mathcal{D}$ to $\mu(\mathcal{D})$ via KME is injective.
- $\text{MMD}(\mathcal{D}_\mathcal{T}, \mathcal{D}_\mathcal{S}) = 0$ if and only if $\mathcal{D}_\mathcal{T} = \mathcal{D}_\mathcal{S}$.

These properties enable the measurement of distribution similarity through MMD and KME.

Theorem H.1. *If the kernel k is universal, then the mean map $\mu : \mathcal{P}_X \to \mathcal{H}$ is injective.*

Proof. We will use a proof by contradiction to establish the theorem.

Assume that $\mu : \mathcal{P}_X \to \mathcal{H}$ is not injective. Then, there exist two different probability measures p and q such that $\mu[p] = \mu[q]$, i.e.,

$$\mu[p] = \mu[q].$$

The mean map $\mu[p]$ is represented as:

$$\mu[p](\cdot) = \int_X k(x, \cdot)\, dp(x),$$

and similarly,

$$\mu[q](\cdot) = \int_X k(x, \cdot)\, dq(x).$$

For any $f \in \mathcal{H}$, we have:

$$\langle f, \mu[p] \rangle_\mathcal{H} = \langle f, \mu[q] \rangle_\mathcal{H}.$$

The inner product $\langle f, \mu[p] \rangle_\mathcal{H}$ can be written as:

$$\langle f, \mu[p] \rangle_\mathcal{H} = \mathbb{E}_{X \sim p}[f(X)],$$

and similarly,

$$\langle f, \mu[q] \rangle_\mathcal{H} = \mathbb{E}_{X \sim q}[f(X)].$$

Since $\mu[p] = \mu[q]$, it follows that:

$$\mathbb{E}_{X \sim p}[f(X)] = \mathbb{E}_{X \sim q}[f(X)] \quad \forall f \in \mathcal{H}.$$

By the universality of the kernel k, the RKHS $\mathcal{H}$ is dense in $C(X)$. This means that the above equality holds for all continuous functions $f \in C(X)$.

By the uniqueness theorem for measures, if two measures p and q agree on all continuous functions, then $p = q$. This contradicts our assumption that p and q are different.

Therefore, the assumption that μ is not injective must be false, and hence μ is injective. □

Theorem H.2. *Let $\mathcal{F}$ be a unit ball in a universal RKHS $\mathcal{H}$, defined on the compact metric space $\mathcal{X}$, with associated continuous kernel $k(\cdot, \cdot)$. Then MMD $\{\mathcal{F}, p, q\} = 0$ if and only if $p = q$.*

Proof. First, it is clear that $p = q$ implies MMD $\{\mathcal{F}, p, q\}$ is zero. We now prove the converse.

By the universality of $\mathcal{H}$, for any given $\epsilon > 0$ and $f \in C(\mathcal{X})$, there exists a $g \in \mathcal{H}$ such that

$$\|f - g\|_\infty \leq \epsilon.$$

We next make the expansion

$$\begin{aligned}
|\mathbb{E}_x f(x) - \mathbb{E}_y f(y)| \leq |\mathbb{E}_x f(x) - \mathbb{E}_x g(x)| +\\
|\mathbb{E}_x g(x) - \mathbb{E}_y g(y)| + |\mathbb{E}_y g(y) - \mathbb{E}_y f(y)|.
\end{aligned}$$

The first and third terms satisfy

$$|\mathbb{E}_x f(x) - \mathbb{E}_x g(x)| \leq \mathbb{E}_x |f(x) - g(x)| \leq \epsilon.$$

Next, write

$$\mathbb{E}_x g(x) - \mathbb{E}_y g(y) = \langle g, \mu_p - \mu_q \rangle_\mathcal{H} = 0,$$

since MMD $\{\mathcal{F}, p, q\} = 0$ implies $\mu_p = \mu_q$. Hence

$$|\mathbb{E}_x f(x) - \mathbb{E}_y f(y)| \leq 2\epsilon$$

for all $f \in C(\mathcal{X})$ and $\epsilon > 0$, which implies $p = q$. □

H.2. w-convergence bound

For a bounded kernel $0 \leq k(\cdot,\cdot) \leq K$, the biased empirical estimator of MMD is

$$\text{MMD}_b[\mathcal{F}, X, Y] = \sup_{f\in\mathcal{F}} \left(\frac{1}{m}\sum_{i=1}^{m} f(x_i) - \frac{1}{n}\sum_{i=1}^{n} f(y_i) \right),$$

where X, Y are random variables from distribution p, q, and $\{x_i\}_{i=1}^m, \{y_i\}_{i=1}^n$ are samples drawn from these two distributions.

We want to show that the absolute difference between $\text{MMD}(\mathcal{F}, p, q)$ and $\text{MMD}_b(\mathcal{F}, X, Y)$ is close to its expected value, independent of the distributions p and q. To this end, we prove three intermediate results, which we then combine. The first result we need is an upper bound on the absolute difference between $\text{MMD}(\mathcal{F}, p, q)$ and $\text{MMD}_b(\mathcal{F}, X, Y)$. We have

$$\begin{aligned}
&|\text{MMD}(\mathcal{F}, p, q) - \text{MMD}_b(\mathcal{F}, X, Y)| \\
&= |\sup_{f\in\mathcal{F}} \left(\mathbb{E}_x(f) - \mathbb{E}_y(f)\right) \\
&- \sup_{f\in\mathcal{F}} \left(\frac{1}{m}\sum_{i=1}^{m} f(x_i) - \frac{1}{n}\sum_{i=1}^{n} f(y_i) \right)| \\
&\leq \underbrace{\sup_{f\in\mathcal{F}} \left| \mathbb{E}_x(f) - \mathbb{E}_y(f) - \frac{1}{m}\sum_{i=1}^{m} f(x_i) + \frac{1}{n}\sum_{i=1}^{n} f(y_i) \right|}_{\Delta(p,q,X,Y)}
\end{aligned} \tag{K}$$

Then, we provide an upper bound on the difference between $\Delta(p, q, X, Y)$ and its expectation. Changing either of x_i or y_i in $\Delta(p, q, X, Y)$ results in changes in magnitude of at most $2K^{1/2}/m$ or $2K^{1/2}/n$, respectively. We can then apply McDiarmid's theorem [56], given a denominator in the exponent of

$$\begin{aligned}
&m\left(\frac{2K^{1/2}}{m}\right)^2 + n\left(\frac{2K^{1/2}}{n}\right)^2 \\
&= 4K\left(\frac{1}{m} + \frac{1}{n}\right) = \frac{4K(m+n)}{mn},
\end{aligned}$$

to obtain

$$\begin{aligned}
&\Pr_{X,Y}\left(\Delta(p, q, X, Y) - \mathbb{E}_{X,Y}[\Delta(p, q, X, Y)] > \epsilon\right) \leq \\
&\qquad \exp\left(-\frac{\epsilon^2 mn}{2K(m+n)}\right).
\end{aligned} \tag{L}$$

Next, we exploit symmetrisation, following [57], to upper bound the expectation of $\Delta(p, q, X, Y)$. Denoting by X' an i.i.d sample of size m drawn independently of X (and likewise for Y'), we have

$$\begin{aligned}
&\mathbb{E}_{X,Y}[\Delta(p, q, X, Y)] \leq \\
&\mathbb{E}_{X,Y} \sup_{f\in\mathcal{F}} \left| \mathbb{E}_x(f) - \frac{1}{m}\sum_{i=1}^{m} f(x_i) - \mathbb{E}_y(f) + \frac{1}{n}\sum_{i=1}^{n} f(y_i) \right| \\
&= \mathbb{E}_{X,Y} \sup_{f\in\mathcal{F}} \Big| \mathbb{E}_{X'}\left(\frac{1}{m}\sum_{i=1}^{m} f(x_i') \right) - \frac{1}{m}\sum_{i=1}^{m} f(x_i) \\
&- \mathbb{E}_y(f) + \frac{1}{n}\sum_{i=1}^{n} f(y_i) \Big| \\
&= \mathbb{E}_{X,Y} \sup_{f\in\mathcal{F}} \left| \mathbb{E}_{X'}\left(\frac{1}{m}\sum_{i=1}^{m} f(x_i') - \frac{1}{m}\sum_{i=1}^{m} f(x_i) \right) \right. \\
&\left. + \frac{1}{n}\sum_{i=1}^{n} f(y_i) - \mathbb{E}_y(f) \right| \\
&\leq \mathbb{E}_{X,Y,X',Y'} \sup_{f\in\mathcal{F}} \left| \frac{1}{m}\sum_{i=1}^{m} f(x_i') - \frac{1}{m}\sum_{i=1}^{m} f(x_i) \right| \\
&+ \mathbb{E}_{Y,Y'} \sup_{f\in\mathcal{F}} \left| \frac{1}{n}\sum_{i=1}^{n} f(y_i') - \mathbb{E}_y(f) \right| \\
&= \mathbb{E}_{X,Y,X',Y'} \sup_{f\in\mathcal{F}} \left| \frac{1}{m}\sum_{i=1}^{m} \sigma_i \left(f(x_i') - f(x_i)\right) \right| + \\
&\mathbb{E}_{Y,Y',\sigma'} \sup_{f\in\mathcal{F}} \left| \frac{1}{n}\sum_{i=1}^{n} \sigma_i' \left(f(y_i') - f(y_i)\right) \right| \\
&\leq \mathbb{E}_{X,X',\sigma} \sup_{f\in\mathcal{F}} \left| \frac{1}{m}\sum_{i=1}^{m} \sigma_i \left(f(x_i') - f(x_i)\right) \right| + \\
&\mathbb{E}_{Y,Y',\sigma'} \sup_{f\in\mathcal{F}} \left| \frac{1}{n}\sum_{i=1}^{n} \sigma_i' \left(f(y_i') - f(y_i)\right) \right| \\
&\leq 2\left[R_m(\mathcal{F}, p) + R_n(\mathcal{F}, q)\right] \\
&\leq 2\left[\left(\frac{K}{m}\right)^{1/2} + \left(\frac{K}{n}\right)^{1/2}\right]
\end{aligned} \tag{M}$$

The first step applies Jensen's inequality to simplify the expression. Next, the triangle inequality is used to separate the terms, followed by substituting the Rademacher average to connect the empirical and expected values. Finally, the Rademacher averages are bounded as defined in Definition 30 of [18]. Combining Eqn. L with the bounded term

$\mathbb{E}_{X,Y}[\Delta(p,q,X,Y)]$ derived from Eqn. M, we can obtain

$$\Pr_{X,Y}\left(\Delta(p,q,X,Y) - 2\left[\left(\frac{K}{m}\right)^{1/2} + \left(\frac{K}{n}\right)^{1/2}\right] > \epsilon\right) \leq \exp\left(-\frac{\epsilon^2 mn}{2K(m+n)}\right). \tag{N}$$

Combining Eqn. N with Theorem 7 in [18], leads to the following Corollary:

Corollary H.3. *If $p = q$, then with probability at least $1-\delta$, the (biased) empirical MMD (obtained by drawing m samples from p and n samples from q) is bounded by:*

$$\frac{1}{2}MMD_b(\hat{p},\hat{q}) < \sqrt{\frac{K}{m}} + \sqrt{\frac{K}{n}} + \sqrt{\frac{K(m+n)\log\frac{1}{\delta}}{2mn}}$$

for any arbitrarily small $\delta > 0$.

Setting $p = \mathcal{D}_{\mathcal{T}}, q = \sum_{j=1}^{N} w_j \mathcal{D}_{\mathcal{S}j}, \sum_{j=1}^{N} w_j = 1$ leads to:

$$\boxed{\frac{1}{2}\left\|\widehat{\mu(T)} - \sum_{j=1}^{N} w_j \widehat{\mu(S_j)}\right\|_{\mathcal{H}} < \sqrt{\frac{K}{m}} + \sqrt{\frac{K}{n}} + \sqrt{\frac{K(m+n)\log\frac{1}{\delta}}{2mn}}.}$$

The above result upper-bounds the error of estimating empirical target KME with weighted average of empirical source KME in the optimization in 4 of the main paper. Note that n is the number of samples from q, i.e. the number of samples from the **involved** source datasets, thus as the number of modules N increases, n also increases.

Similarly, we can derive the following estimation error bound for KME and empirical KME:

Corollary H.4. *([58], Theorem 1) With probability at least $1-\delta$ we have*

$$\|\mu(p) - \widehat{\mu(p)}\|_{\mathcal{H}} \leq 2\sqrt{\frac{K}{n}} + \sqrt{\frac{2\log\frac{1}{\delta}}{n}}.$$

Proof. Proof can be found in [58], section B.1. □

H.3. Loss bound

H.3.1. Assumptions

Suppose we are given a pretrained frozen backbone transformer model f, and there are N source domains in the upload phase. We build PET modules on their own domains and upload them to the module store for future users. Each source domain has a corresponding dataset $\{S_j\}_{j=1}^{N}$, reflecting the distribution $\mathcal{D}_{\mathcal{S}j}$. The source datasets will be inaccessible after the pretraining stage.

We also assume that a global optimal rule function $g : \mathcal{X} \to \mathcal{Y}$ exists for the domain adaptation problem,

$$\forall j \in [N], \forall (x,y) \in S_j, g(x) = y.$$

We assume that all sources are competent, and the source datasets are sufficient to solve their domains. Formally speaking, the modules $\hat{\theta}_j$ can help the pretrained model to reach a small error rate $\epsilon > 0$ with respect to a certain loss function L (upper-bounded by $|L|$) on their domain distribution $\mathcal{D}_{\mathcal{S}j}$ when applied with the pretrained backbone model f:

$$\forall j \in [N], \mathcal{L}(\mathcal{D}_{\mathcal{S}j}, f_{\hat{\theta}_j}) = \mathbb{E}_{x\sim\mathcal{D}_{\mathcal{S}j}}[L(f_{\hat{\theta}_j}(x), y)] \leq \epsilon. \tag{O}$$

In this context, the loss function $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ can be either a regression loss or classification loss. Since the domains $\{S_j\}_{j=1}^{N}$ are equipped with low-error pre-trained modules, they are referred to as solved domains.

In the deployment phase, a new user wants the model to solve the current domain with only unlabeled testing data $x \sim \mathcal{D}_{\mathcal{T}}$. Thus the target is to learn a good model $\hat{f}_t$ which minimizes $\mathcal{L}(\mathcal{D}_{\mathcal{T}}, \hat{f}_t)$, utilizing the information contained in pre-trained modules $\{\hat{\theta}_j\}_{j=1}^{N}$.

Theorem H.5 (Zero-shot adaptation loss bound)**.** *Assume that the assumptions in corollary H.3 hold. The module θ_j from each source dataset satisfies $\forall j \in [N], \mathcal{L}(\mathcal{D}_{\mathcal{S}j}, f_{\hat{\theta}_j}) = \mathbb{E}_{x\sim\mathcal{D}_{\mathcal{S}j}}[L(f_{\hat{\theta}_j}(x), y)] \leq \epsilon$. Assume that the loss function $L(f_{\hat{\theta}_j}(x), f(x)) \in \mathcal{H}_k$. The empirical MMD between distribution mixture $\sum_{j=1}^{N} w_j \mathcal{D}_{\mathcal{S}j}$ and current distribution $\mathcal{D}_{\mathcal{T}}$ can be estimated from*

$$MMD_b = \left\|\widehat{\mu(T)} - \sum_{j=1}^{N} w_j \widehat{\mu(S_j)}\right\|_{\mathcal{H}}$$

Then the finite sample loss satisfies:

$$\mathcal{L}(\mathcal{D}_{\mathcal{T}}, g, f_{\sum \boldsymbol{w}_j \hat{\theta}_j}) = \sum_{x_i \sim \mathcal{D}_{\mathcal{T}}} \left[L(f_{\sum \boldsymbol{w}_j \hat{\theta}_j}(x_i), g(x_i))\right] \leq \epsilon + O(\sqrt{\frac{1}{m}} + \sqrt{\frac{1}{n}})$$

Proof. By the reproducing property, the loss function can be written as:

$$L(f_{\sum \boldsymbol{w}_j \hat{\theta}_j}, g(x)) = L_{\sum \boldsymbol{w}_j \hat{\theta}_j}(x) = \langle L_{\sum \boldsymbol{w}_j \hat{\theta}_j}, k(x,\cdot)\rangle.$$

We then can represent the error of each model in the form of KME. For the current mixture:

$$\mathbb{E}_{x\sim\mathcal{D}_{\mathcal{T}}}[L_{\sum \boldsymbol{w}_j \hat{\theta}_j}(x)] = \langle L_{\sum \boldsymbol{w}_j \hat{\theta}_j}, \mu_{\mathcal{D}_{\mathcal{T}}}\rangle$$

The empirical loss can also be represented by:

$$\begin{aligned}
&\left|\left\langle L_{\sum w_j\hat{\theta}_j}, \widehat{\mu(T)}\right\rangle\right| \leq \\
&\underbrace{\left|\left\langle L_{\sum w_j\hat{\theta}_j}, \widehat{\mu(T)}\right\rangle - \left\langle L_{\sum w_j\hat{\theta}_j}, \sum_j w_j\widehat{\mu(S_j)}\right\rangle\right|}_{(A)} + \\
&\underbrace{\left|\left\langle L_{\sum w_j\hat{\theta}_j}, \sum_j w_j\widehat{\mu(S_j)}\right\rangle\right|}_{(B)}
\end{aligned} \tag{P}$$

We then bound (A) and (B) separately.

By the convergence rate for empirical MMD,

$$\begin{aligned}
(A) &= \left|\langle L_{\sum \boldsymbol{w}_j\hat{\theta}_j}, \widehat{\mu(T)} - \sum_{j=1}^{N} w_j\widehat{\mu(S_j)}\rangle\right| \\
&\leq \|L_{\sum \boldsymbol{w}_j\hat{\theta}_j}\|\|\widehat{\mu(T)} - \sum_{j=1}^{N} w_j\widehat{\mu(S_j)}\| \\
&\leq O(m,n)
\end{aligned} \tag{Q}$$

$$\begin{aligned}
(B) &\leq \left|\langle L_{\sum \boldsymbol{w}_j\hat{\theta}_j}, \sum w_j\mu_{\mathcal{D}_{\mathcal{S}_j}}\rangle\right| \\
&+ \left|\langle L_{\sum \boldsymbol{w}_j\hat{\theta}_j}, \sum_j w_j\widehat{\mu(S_j)} - \sum w_j\mu_{\mathcal{D}_{\mathcal{S}_j}}\rangle\right| \\
&\leq \sum_{j=1}^{N} w_j \left|\langle L_{\hat{\theta}_j}, \mu_{\mathcal{D}_{\mathcal{S}_j}}\rangle\right| \\
&+ |L| \,\|\sum_j w_j\widehat{\mu(S_j)} - \sum w_j\mu_{\mathcal{D}_{\mathcal{S}_j}}\| \\
&\leq \epsilon + 2\sqrt{\frac{K}{m}} + \sqrt{\frac{2\log\frac{1}{\delta}}{m}}
\end{aligned} \tag{R}$$

The last steps of the estimation (A) and (B) can be obtained directly from corollary H.4 and corollary H.3, and this completes the proof.

□

Theorem H.5 estimates the loss in directly applying the integrated module $\theta(t)$ to the target domain without any tuning on the target dataset, i.e. the zero-shot performance. It ensures a good initialization of the tuning in Sec. 3.2, leading to a good adaptation performance.