DA-MergeLoRA: Hypernetwork-Based LoRA Merging for Few-Shot Test-Time Domain Adaptation

Siobhan Reid[1], Zhixiang Chi[2], Li Gu[1,3], Omid Reza Heidari[1], Ziqiang Wang[1,3], and Yang Wang[1,3]

[1] Concordia University, Montreal QC H3G 1M8, Canada
[2] University of Toronto, Toronto ON M5S 1A1, Canada
[3] Mila – Quebec AI Institute, Montreal QC H2S 3H1, Canada

Abstract. Few-shot Test-Time Domain Adaptation (FSTT-DA) seeks to adapt models to novel domains using only a handful of unlabeled target samples. This setting is more realistic than typical domain adaptation setups, which assume access to target data during source training. However, prior FSTT-DA approaches fail to effectively leverage source domain-specific knowledge, relying on shallow batch normalization updates, prompt-based methods that treat the model as a black box, or ensembling strategies that do not capture cross-domain relationships. To address these limitations, we introduce a new FSTT-DA framework that integrates LoRA fine-tuning with model merging. In our approach, separate LoRA modules are fine-tuned on CLIP's vision encoder for each source domain. Since LoRA modifies only a small fraction of the model's parameters, it retains the base model's generalized knowledge while internally learning domain-specific features. To adapt the learned knowledge to a specific target domain, we propose a hypernetwork trained via meta-learning that generates per-column merging factors to combine LoRA modules. Given a small batch of target images, the hypernetwork produces merging weights that fuse source LoRA modules into a single adapted representation. Our results demonstrate state-of-the-art performance across various domain adaptation datasets. Our code is publicly available at `https://github.com/nahbois4321/DA-MergeLoRA`.

Keywords: Few-shot Test Time Domain Adaptation · Hypernetwork Model Merging · CLIP

1 Introduction

Machine learning models typically assume that training and test data are identically distributed; when the test distribution differs (domain shift), performance degrades [47,62]. Test-Time Adaptation (TTA) addresses domain shift by adapting a model at inference time, typically operating in either an offline or online mode [12, 22, 48, 53]. In the offline setting, the model is given a large corpus of unlabeled target data and performs multi-step unsupervised adaptation before

inference. In the online setting, the model receives a stream of target samples and updates itself incrementally, often on a per-sample or per-mini-batch basis, throughout deployment [46]. Both modes assume access to many target samples and allow repeated adaptation steps.

In contrast, Few-Shot Test-Time Domain Adaptation (FSTT-DA) assumes that the model receives only a single small batch of unlabeled target images at deployment and must perform a one-shot adaptation step before processing the entire target domain, with no further updates [52]. In this setting, the model can be trained on labeled data from multiple source domains during development, but the target domain remains completely unseen until test time and is available only as a small unlabeled batch for adaptation. This differs from typical multi-source domain adaptation [37], which assumes access to the unlabeled target domain during source training for alignment or distribution matching.

FSTT-DA arises in real-world applications where data are limited, inference-time overhead must be minimal, or adaptation using target data is restricted by privacy or safety constraints. For example, medical imaging models deployed in new hospitals face domain shifts; however, regulations can prevent iterative or large-scale adaptation using patient data. Likewise, robots entering novel environments typically receive only a handful of initial observations before they must begin downstream tasks with minimal inference-time overhead. FSTT-DA is therefore a practical yet more challenging setting than typical TTA.

This setting poses two key difficulties: **(CI)** extracting domain-specific cues from diverse source domains without harming cross-domain generalization, and **(CII)** transferring that knowledge to a new target domain under severe data and label scarcity. Despite recent progress, current FSTT-DA methods remain limited. BatchNorm-based adaptation, MABN [52], is efficient but restricted to re-estimating BatchNorm statistics and affine parameters; with small test batches, these estimates become noisy and lead to unstable transfer. Meta-DMoE [61] employs ensemble/distillation strategies to train multiple source-domain experts via a teacher–student pipeline, incurring substantial compute, scaling poorly with the number of sources, and limiting cross-domain sharing. Prompt-based approaches [10, 11] treat the backbone model as a black box and adjust only external prompts, leaving internal representations fixed and restricting knowledge transfer. As a result, existing methods make shallow updates, scale poorly, or restrict knowledge sharing, and ultimately fall short of the two key requirements **(CI–CII)**.

To overcome **(CI)**, we draw inspiration from recent work showing that inserting lightweight low-rank adapters (LoRA) [19] in parallel with model weights can effectively capture domain-specific information [30]. We attach separate sets of LoRAs to a frozen CLIP backbone to model knowledge from multiple source domains. This design provides more expressive adaptation than shallow BatchNorm updates [52] and allows internal knowledge steering, unlike black-box prompting methods such as VDPG [11] and L2C [10]. Moreover, unlike ensemble methods that require a full model per source domain, our approach enables cross-domain knowledge sharing while retaining the generalization of the base model.

To address **(CII)**, we adopt parameter-space model merging to integrate domain-specific LoRA modules into a single model adapted to the target domain. Unlike prompt-based methods [10, 11], merging updates internal weights and aligns both low- and high-level features. Compared to ensembling [61], it avoids running multiple experts and produces one model with fixed memory and a single forward pass. Beyond efficiency, merging promotes direct knowledge sharing across domains and avoids the instability of retraining or distillation in the few-shot setting.

To further motivate our approach, parameter-space model merging has been successfully applied in several other areas. One line of work merges task-specific models into a single multi-task generalist model [18,23,33,34,43,56]. Unlike these approaches, which aim to unify different tasks into one generalist model, our goal is to create a specialized domain-specific model tailored to a novel visual domain. Another line of work merges pairs of target LoRAs at test time to produce mixed style-content LoRAs in diffusion models [39, 41]. In contrast, our setting has no target LoRA available; instead, we must synthesize a domain-specific LoRA by combining source-domain LoRAs guided by only a few unlabeled target samples. Model merging has also been explored for NLP task adaptation [1,20,51], where each LoRA corresponds to a different supervised task (QA, translation, NLI, etc.). Unlike previous task-centric merging, our work focuses on a fixed-task classification setting that undergoes visual distribution shifts across domains, motivating a merging strategy tailored for domain adaptation rather than task-specific tuning.

To this end, we introduce **DA-MergeLoRA**, an FSTT-DA framework that reformulates adaptation as parameter-space merging. For each source domain, we fine-tune a separate LoRA module on the CLIP vision encoder [38], while keeping the base parameters frozen to preserve cross-domain generalization. At test time, given a few unlabeled target images, a hypernetwork predicts merging weights to combine the source-domain LoRAs into a single target LoRA. The hypernetwork is conditioned on both target images and the pool of source LoRA weights, and is trained in a meta-learning fashion to learn effective merging policies. Empirically, our approach achieves state-of-the-art results across multiple FSTT-DA benchmarks, including a $+$**1.24%** gain in average accuracy on DomainNet, $+$**2.70%** on Camelyon17, and $+$**11.40%** in worst-case accuracy on FMoW.

Our work makes the following contributions: (i) Novel FSTT-DA framework: we formulate FSTT-DA as a parameter-space merging problem, yielding a single adapted model. (ii) Hypernetwork architecture: We introduce a meta-learned lightweight hypernetwork that generates per-column merge weights to combine multiple source LoRA modules into a specialized, domain-specific target model, conditioned on a small batch of unlabeled target data, enabling effective FSTT-DA. (iii) State-of-the-art results: Our approach achieves SOTA on the DomainNet and WILDS benchmarks, empirically demonstrating that parameter-space merging is an effective approach for FSTT-DA. Together, these contributions move beyond prior FSTT-DA methods (batch normalization, prompt-based, and

distillation ensembles) toward modular internal source-domain representations and an adaptive parameter-space mechanism for effective test-time adaptation.

2 Related Work

Test-time domain adaptation. Distribution shift between training and test data often degrades performance when models are deployed to new domains [15]. Domain adaptation algorithms address this by learning features shared between source and target domains, enabling better generalization [6, 17, 31]. Test-time adaptation (TTA) considers the setting where no target-domain data is available during model development [46]. After deployment to a new domain, the model may adapt in an offline manner, where the model is trained on all unlabeled target data before inference, or in an online manner, where the model is continually adapted to streaming test data [27]. Common TTA strategies include entropy minimization [46, 48], pseudo-labels [28], auxiliary tasks [29, 45], and contrastive learning [9].

Few-shot test-time domain adaptation. In FSTT-DA, models adapt to unseen domains using only a batch of unlabeled samples at inference [61]. Prior works approach this challenge in different ways. Meta-DMoE distills knowledge from a mixture of source-domain experts to a new target model via a meta-learned Transformer aggregator [61]. MABN adapts batch-normalization affine parameters using a self-supervised auxiliary branch [52]. VDPG generates domain-specific visual prompts from a knowledge bank conditioned on target samples [11]. L2C enhances VDPG by attaching a parallel branch to CLIP, learning directly from dataset-specific input knowledge and text semantics [10].

Foundation models. Recent FSTT-DA methods,VDPG [11] and L2C [10], use CLIP as a frozen backbone. CLIP is a vision–language model whose image and text encoders are trained using contrastive learning so that paired images and captions have high cosine similarity. Classification then selects the prompt "a photo of a <classname>" with the highest similarity [38]. Vision–language models are appealing for domain adaptation because of their strong out-of-distribution generalization. However, prior research shows that full fine-tuning of CLIP can degrade its generalization capabilities [50], motivating parameter-efficient methods such as prompt tuning [11] and LoRA [19], which keep the backbone frozen while learning domain-specific features. Similar to VDPG and L2C, our work builds on CLIP, but differs by encoding domain knowledge directly in LoRA modules, enabling richer and more flexible adaptation than prompt- or batch normalization-based approaches.

Model merging. Model-merging methods aim to combine multiple task-specific models into a single general-purpose model while mitigating task interference. Examples include parameter averaging [13, 49], Task Arithmetic [21], Fisher

Merging [34], RegMean [23], and structured merging methods such as TIES [56], which resolves sign conflicts and averages only aligned updates. Several recent advances further improve alignment and reduce interference: KnOTS [43] applies singular value decomposition (SVD) to concatenated LoRA updates before merging task-aligned components; Iso-C / Iso-CTS [33] use SVD and isotropic scaling to identify shared and task-specific directions; and TSV-M [18] extracts low-rank task directions via SVD and orthogonalizes them across tasks. These approaches all aim to produce a single multi-task model that preserves performance across the original training tasks. In contrast, our method focuses on generating a specialized domain-specific model, as opposed to a generalist model. Merging in prior work is unconditional (merge rules do not depend on target data) whereas our merging is conditional and meta-learned, driven by a target-domain embedding extracted from a few unlabeled target images.

LoRA merging. LoRA fine-tuning inserts low-rank adapters into frozen foundation models, enabling efficient adaptation with fewer trainable parameters. LoRA merging has recently been applied across different domains [8,20,25,26,40–42,54]. In diffusion models, LoRA.rar [41] and ZipLoRA [39] merge pairs of content and style LoRAs for personalized image generation, while MoLE [51] trains a gating network to combine multiple content-specific LoRAs into a mixed-content adapter. These approaches all assume that target LoRA modules already exist and focus on fusing them coherently. In contrast, we assume no target LoRA is available; our method must construct a target LoRA by merging source-domain LoRAs, guided only by a few unlabeled target images. Several recent works explore LoRA merging and LoRA generation in NLP and vision. LoRAHub [20] learns to merge LoRA modules for NLP task transfer but requires few-shot labeled examples, whereas our adaptation uses only unlabeled target data. SD-LoRA [54] incrementally incorporates new LoRA modules for continual learning by decoupling their direction and magnitude. Text-to-LoRA [8] uses a hypernetwork to generate LoRA weights from natural-language task descriptions. ICM-Fusion [40] fuses LoRA modules using a VAE-based latent representation, and RegCL [42] merges LoRA adapters for continual-learning domain-specific image segmentation tasks. These diverse successes from other research areas further motivate exploring LoRA merging within the FSTT-DA setting.

Meta-learning. Meta-learning is a training paradigm that enables a model to quickly adapt to new tasks or domains [16]. Previous FSTT-DA methods, such as VDPG [11], employ meta-learning to generate domain-specific visual prompts from a knowledge bank. In contrast to these prompt-based approaches which leave internal representations fixed, we use meta-learning to train a hypernetwork that predicts per-column merging weights. This allows our method to dynamically synthesize a specialized target LoRA module in the parameter space.

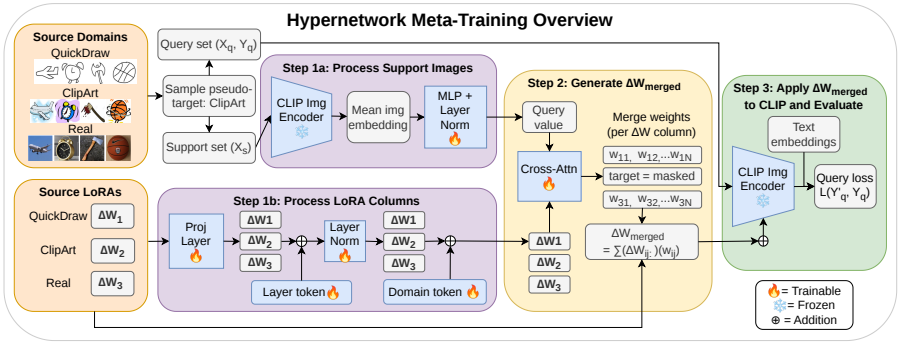

Fig. 1: Overview of our hypernetwork training procedure. Given pre-trained source LoRA modules, we meta-train a hypernetwork to dynamically predict optimal weights to merge the source LoRA modules into a target-domain specific model. At each meta-training step, one source domain is randomly sampled as the pseudo-target, while the remaining domains serve as source data. Cross-attention is computed between the processed pseudo-target support images and source LoRA columns to generate column-wise merge weights, yielding ΔW_{merged}. Finally, the merged model is applied to the frozen CLIP backbone and evaluated on query images to compute the meta-training loss.

3 Method

Problem setting. In this work, we follow the same problem setting as previous FSTT-DA methods [10, 11, 61]. The training set consists of N labeled source domains $\mathcal{D}_s = \{\mathcal{D}_s^n\}_{n=1}^N$, where each source domain $\mathcal{D}_s^n = (x_s, y_s)^n$ contains image–label pairs (x, y). The test set includes M target domains $\mathcal{D}_t = \{\mathcal{D}_t^m\}_{m=1}^M$, where each target domain $\mathcal{D}_t^m = (x_t)^m$ provides only unlabeled images x_t. We assume a distribution shift exists between any source and target domain, while all domains share the same label space. FSTT-DA trains exclusively on the source domains, without access to target data. At test time, when an unseen target domain $\mathcal{D}_t^m$ is encountered (deployed in a new environment), the model must adapt to this domain using only a few-shot batch of unlabeled samples. The adapted model is then evaluated on the entire target domain.

Overview. In this work, we develop a hypernetwork that generates merging weights to combine LoRA modules, conditioned on a batch of unlabeled target images. Sec. 3.1 outlines the training of individual LoRA modules. Sec. 3.2 presents the architecture of the LoRA hypernetwork, which produces per-column merging weights. Sec. 3.3 details the meta-learning setup used to train the hypernetwork and the inference procedure.

3.1 LoRA Module Training

To capture domain-specific knowledge, a LoRA model is trained on each source domain using CLIP [38]. LoRA adapts the vision encoder by adding trainable low-rank matrices to its attention and projection weight matrices [19]. Instead of fine-tuning the weight matrix $W \in \mathbb{R}^{d\times k}$, LoRA replaces the weight update with a low-rank approximation, learned through two smaller matrices A and B:

$$W' = W + \Delta W, \quad \Delta W = BA \tag{1}$$

where $A \in \mathbb{R}^{r\times k}$ and $B \in \mathbb{R}^{d\times r}$ with $r \ll \min(d, k)$. The rank r controls the trade-off between capacity and efficiency. During training, only A and B are updated, while W remains frozen. For an input h, the forward pass becomes:

$$h' = Wh + BAh = Wh + \Delta Wh \tag{2}$$

The backbone CLIP model is frozen throughout training, and only the LoRA parameters are updated. Before training, class-specific text prompt embeddings (e.g., "a photo of a <classname>") are precomputed using CLIP's text encoder. During training, the model uses the cross-entropy loss to learn to select the prompt embedding with the highest cosine similarity to the LoRA-modified image embedding.

3.2 LoRA Merging Hypernetwork

A hypernetwork is a neural network that generates the weights or parameters of another network [41]. Here, we use a hypernetwork that learns to generate merging weights to combine the columns of LoRA matrices ΔW trained on different source domains. Prior work [21] shows that models fine-tuned from a shared backbone (e.g., CLIP) remain compatible for arithmetic merging. The hypernetwork accepts as input a batch of target images and the LoRA ΔW columns from the different source domains, utilizing cross-attention to find the source columns that correlate best with the target images. Following LoRA.rar [41], we employ per-column ΔW merging rather than per-model or per-layer merging. Theoretically, this grants a more flexible weighting mechanism to selectively emphasize or suppress feature subspaces relevant to the target domain.

Offline LoRA preparation. Before hypernetwork training begins, the full product matrices $\Delta W = BA$ for all source domains are pre-calculated offline for each transformer attention and projection layer. We explicitly merge the columns of the full ΔW product, as opposed to merging the separate low-rank matrices A or B, because it allows for fine-grained merging and enables us to combine different LoRA models regardless of their rank r. For this column-wise merging, the hypernetwork generates weight vectors matching each layer's dimensionality ($d = 768$ for CLIP ViT-B/16).

Target image processing. During each forward pass of the hypernetwork, the current batch of target domain images is processed before being used for cross-attention. The images, X_d, are passed through the frozen CLIP backbone to generate embeddings, and the mean embedding, e_d, is found:

$$e_d = \text{mean}(\text{EncodeImage}(X_d)) \tag{3}$$

The mean embedding is then passed through a small multi-layer perceptron (MLP) network to reduce its dimension to $\mathbb{R}^{128}$ and extract domain-specific information. The MLP consists of two linear layers with a GELU activation and hidden dimension size of $\mathbb{R}^{256}$. The output from the MLP is then passed through a LayerNorm (LN) to normalize the output:

$$e'_d = \text{LN}(\text{MLP}(e_d)) \tag{4}$$

LoRA column processing. Before LoRA ΔW columns are used for cross-attention to calculate merge weights, they are also processed. Firstly, they are passed through a projection layer, which maps their input dimension to a uniform dimension of $\mathbb{R}^{128}$. Additionally, we add three content tokens to the columns to help the hypernetwork learn more fine-grained and context-aware merging weights: (i) Layer index token: a learnable embedding $u_\ell \in \mathbb{R}^{128}$ representing the transformer layer ℓ the column came from; (ii) Sub-layer type token: a learnable embedding $v_t \in \mathbb{R}^{128}$ indicating whether the column comes from an attention (`qkv`) or projection (`proj`) site of the transformer layer. These two tokens act as positional embeddings; (iii) Domain token: for each source domain d, a learnable embedding q_d is projected to $\mathbb{R}^{128}$ and added after normalization, providing a lightweight bias indicating which domain the column came from:

$$c^\star = \text{LN}(\tilde{c} + u_\ell + v_t) + \alpha P_{\text{dom}} q_d \tag{5}$$

Here, $\tilde{c} \in \mathbb{R}^{128}$ is the projected LoRA column, $P_{\text{dom}} \in \mathbb{R}^{128 \times E'}$ projects the domain embedding into the 128-dimensional space, and α is a small scaling factor that controls the influence of the domain token. The resulting $c^\star \in \mathbb{R}^{128}$ is the final context-enriched column representation used in cross-attention.

Cross-attention. Cross-attention is then used to produce the merge weights, where the query is the embedded target domain representation and the keys are the processed LoRA columns:

$$w^{(k)}_{\ell,t,c} = \text{softmax}_k \left(\frac{Q\, K^\top_{\ell,t,k,c}}{\tau} \right) \tag{6}$$

Here, Q is the projected target-domain embedding, $K_{\ell,t,k,c}$ is the key vector for the LoRA column at transformer layer ℓ, site t, domain k, and column index c, and τ is a temperature scaling factor. The softmax is taken along the domain axis k, producing normalized per-domain weights for each column.

Algorithm 1: Overview of our hypernetwork meta-training algorithm.

Input: $\mathcal{D}_S$: source domains; $\{\Delta W^{(k)}\}_{k=1}^{K}$: frozen LoRA weight matrices from the K source domains; frozen base weights W (CLIP); trainable hypernetwork $\mathcal{H}$; step size α
Output: Trained hypernetwork $\mathcal{H}$
Randomly initialize θ (params of $\mathcal{H}$)
while *not converged* **do**
  Sample pseudo-target domain index $d \sim \{1, \ldots, K\}$
  Sample support/query from domain d: $(X_{S_d}, Y_{S_d}), (X_{Q_d}, Y_{Q_d})$
  Compute support embedding with frozen CLIP: $e_{S_d} \leftarrow \text{mean}(\text{EncodeImage}(X_{S_d}))$
  Set pseudo-target LoRA columns to zero: $\Delta W^{(d)} \leftarrow 0$
  Generate per-column merge weights using the hypernet: $w \leftarrow \mathcal{H}\big(e_{S_d}, \{\Delta W^{(k)}\}_{k=1}^{K}\big)$
  foreach *layer, ℓ, layer type t, and column c* **do**
    Perform merge: $(\Delta W^{\text{merge}}_{\ell,t})_{:,c} = \sum_{k=1}^{K} w^{(k)}_{\ell,t,c} \, (\Delta W^{(k)}_{\ell,t})_{:,c}$
  end
  Apply merged LoRA matrices to base CLIP model: $W' \leftarrow W + \Delta W^{(\text{merge})}$
  Evaluate on query set with merged model: $\hat{Y}_{Q_d} \leftarrow f_{W'}(X_{Q_d})$
  Calculate loss: $\mathcal{L} \leftarrow \text{CE}(\hat{Y}_{Q_d}, Y_{Q_d})$
  Update hypernetwork parameters: $\theta_{\mathcal{H}} \leftarrow \theta_{\mathcal{H}} - \alpha \nabla_{\theta_{\mathcal{H}}} \mathcal{L}$
end

After the merge weights are generated using the hypernetwork, the LoRA matrices ΔW from the different source domains are combined together:

$$\big(\Delta W^{\text{merge}}_{\ell,t}\big)_{:,c} = \sum_{k=1}^{K} w^{(k)}_{\ell,t,c} \, \big(\Delta W^{(k)}_{\ell,t}\big)_{:,c} \qquad c = 1, \ldots, C_{\ell,t} \tag{7}$$

Here, $\Delta W^{(k)}_{\ell,t}$ denotes the LoRA update at transformer layer ℓ and site $t \in \{\text{attn}, \text{proj}\}$ for source domain k. $w^{(k)}_{\ell,t,c}$ is the merging weight for column c. $C_{\ell,t}$ is the number of columns at that (ℓ, t) site. K is the number of source domains. After each merged ΔW is computed, they are applied to the frozen CLIP model to generate the new merged model.

3.3 Hypernetwork Meta-Training and Inference

Training. The hypernetwork is trained using a meta-learning framework, similar to [11]. At each meta-iteration, one source domain is randomly selected as a pseudo-target domain to simulate test-time evaluation on novel domains. The LoRA columns of the pseudo-target domain are masked with zeros, and its merge weight is fixed to zero. From this pseudo-target domain, a support batch and a query batch are sampled. The support set images and the LoRA columns from the other source domains are passed to the hypernetwork, which generates the column-wise merge weights. The resulting merged LoRA matrices are applied to the frozen CLIP backbone, and the merged model is evaluated on the query set. A standard cross-entropy classification loss is computed on the query set predictions, and the resulting gradients are backpropagated end-to-end, from the merged model to the hypernetwork. The hypernetwork's parameters are trainable, while all other parameters are frozen (CLIP backbone and source LoRA models). The training procedure is detailed in Algorithm 1 and visualized in Fig. 1.

Inference: At test time, an unlabeled batch of images from the target domain and all source LoRA modules are passed to the hypernetwork. The target domain is novel and excluded from the hypernetwork training process. The hypernetwork outputs the merging weights, which are used to combine the source LoRA modules into a new model optimized for the target domain. After this initial one-step adaptation, the merged model is then evaluated on the entire target domain with no further updates.

4 Experiments

Datasets. We follow prior FSTT-DA work [10, 11] and evaluate on the WILDS and DomainNet benchmarks. We evaluate on three WILDS [24] benchmark classification datasets (iWildCam [4], FMoW [14], and Camelyon17 [3]), as well as the WILDS PovertyMap [58] benchmark regression dataset. These datasets consist of real-world data with distribution shifts. We use the official WILDS evaluation metrics for each dataset (average accuracy, F1-score, worst-case (WC) accuracy, and WC Pearson correlation (r)) and the official train and out-of-distribution (OOD) test splits. We also evaluate on DomainNet [37], which contains roughly 600k images from 345 classes across six domains. We use the standard leave-one-out protocol for DomainNet, selecting one domain for testing and the remainder as source domains. Images are preprocessed using CLIP's standard train and test transforms. The PovertyMap [58] dataset uses 8-channel input images. We follow prior FSTT-DA work [10, 11] and use the first 3 out of 8 channels, so that its input is compatible with CLIP. Additionally, we use the English version of the class names for iWildCam.

Baselines: We compare our results against common CNN-based [2, 5, 35, 44, 52, 55, 59, 61] and CLIP-based [7, 10, 11, 60] domain adaptation methods, which include recent FSTT-DA algorithms (MABN [52], Meta-DMOE [61], VDPG [11], and L2C [10]). Additionally, we compare against zero-shot CLIP with no fine-tuning.

Training Details. We use OpenAI's pretrained CLIP ViT-B/16 backbone [38]. All LoRA models use a rank of 16 and $\alpha = 32$. Learning rates are set between 3×10^{-4} to 5×10^{-4} for LoRA fine-tuning, and between 1×10^{-5} to 5×10^{-5} for hypernetwork training, with a batch size of 16 for both query and support sets. Hypernetwork models are trained for 1 to 5 epochs, except for PovertyMap, which uses an additional MLP regression head and is trained for 50 epochs. The text prompts used for training are dataset-specific, following the templates from L2C [10]; the mean embedding from these text prompts is used as the final text embedding. Additionally, following Meta-DMoE [61], the iWildCam source domains are grouped into 10 clusters for training. Detailed hyperparameter configurations and training procedures are provided in Appendix A.3 of the supplementary material.

Table 1: Evaluation of our method (DA-MergeLoRA) on the (a) DomainNet dataset and (b) WILDS datasets. The best CNN and best ViT methods are bolded separately. Results are reported as mean (standard deviation) over three trials with different random seeds. The † symbol denotes the official dataset metric. Our method achieves SOTA results on five of the six DomainNet domains and on three of the four official WILDS metrics.

Method	Backbone	Clip	Info	Paint	Quick	Real	Sketch	Avg
ERM	CNNs	58.1 (0.3)	18.8 (0.3)	46.7 (0.3)	12.2 (0.4)	59.6 (0.1)	49.8 (0.4)	40.9
Mixup [55]		55.7 (0.3)	18.5 (0.5)	44.3 (0.5)	12.5 (0.4)	55.8 (0.3)	48.2 (0.5)	39.2
CORAL [44]		59.2 (0.1)	19.7 (0.2)	46.6 (0.3)	13.4 (0.4)	59.8 (0.2)	50.1 (0.6)	41.5
MTL [5]		57.9 (0.5)	18.5 (0.4)	46.0 (0.1)	12.5 (0.1)	59.5 (0.3)	49.2 (0.1)	40.6
SegNet [35]		57.7 (0.3)	19.0 (0.2)	45.3 (0.3)	12.7 (0.5)	58.1 (0.5)	48.8 (0.2)	40.3
ARM [59]		49.7 (0.3)	16.3 (0.5)	40.9 (1.1)	9.4 (0.1)	53.4 (0.4)	43.5 (0.4)	35.5
Meta-DMoE [61]		63.5 (0.2)	21.4 (0.3)	51.3 (0.4)	14.3 (0.3)	62.3 (1.0)	52.4 (0.2)	44.2
MABN [52]		**64.2**	**23.6**	**51.5**	**15.2**	**64.6**	**54.1**	**45.5**
DoPrompt [60]	ViT-B/16 IMN	67.6 (0.2)	24.6 (0.1)	54.9 (0.1)	17.5 (0.2)	69.6 (0.3)	55.2 (0.5)	48.3
Zero-shot [38]	ViT-B/16 CLIP	69.9	48.2	65.4	14.5	82.3	62.5	57.1
ERM		68.0 (0.1)	22.5 (0.6)	46.5 (4.2)	18.5 (0.9)	58.7 (2.7)	52.5 (1.2)	44.4
MIRO [7]		74.9 (0.2)	37.1 (0.4)	59.8 (0.6)	**18.7 (1.2)**	72.2 (0.2)	61.2 (0.9)	54.0
VDPG [11]		76.3 (0.2)	49.3 (0.1)	67.8 (0.1)	17.4 (0.2)	81.5 (0.3)	66.6 (0.2)	59.8
L2C [10]		75.6 (0.1)	52.1 (0.1)	69.4 (0.1)	17.3 (0.2)	85.5 (0.1)	67.1 (0.2)	61.2
DA-MergeLoRA		**76.66 (0.08)**	**54.49 (0.21)**	**70.91 (0.05)**	17.87 (0.72)	**85.52 (0.12)**	**69.17 (0.08)**	**62.44 (0.16)**

(a) Evaluation of our method (DA-MergeLoRA) on the DomainNet dataset using the leave-one-out domain testing protocol.

Method	Backbone	iWildCam		Camelyon17	FMoW		PovertyMap	
		Acc	Macro F1†	Acc†	WC Acc†	Acc	WC Pearson r†	Pearson r
ERM	CNNs	71.6 (2.5)	31.0 (1.3)	70.3 (6.4)	32.3 (1.25)	53.0 (0.55)	0.45 (0.06)	0.78 (0.04)
CORAL [44]		73.3 (4.3)	32.8 (0.1)	59.5 (7.7)	31.7 (1.24)	50.5 (0.36)	0.44 (0.06)	0.78 (0.05)
IRM [2]		59.8 (3.7)	15.1 (4.9)	64.2 (8.1)	30.0 (1.37)	50.8 (0.13)	0.43 (0.07)	0.77 (0.05)
ARM-CML [59]		70.5 (0.6)	28.6 (0.1)	84.2 (1.4)	27.2 (0.38)	45.7 (0.28)	0.37 (0.08)	0.75 (0.04)
ARM-BN [59]		70.3 (2.4)	23.7 (2.7)	87.2 (0.9)	24.6 (0.04)	42.0 (0.21)	0.49 (0.21)	0.84 (0.05)
Meta-DMoE [61]		77.2 (0.3)	34.0 (0.6)	91.4 (1.5)	35.4 (0.58)	52.5 (0.18)	0.51 (0.04)	0.80 (0.03)
MABN [52]		**78.4 (0.6)**	**38.3 (1.2)**	**92.4 (1.9)**	**36.6 (0.41)**	**53.2 (0.52)**	0.56 (0.05)	0.84 (0.04)
Zero-shot [38]	ViT-B/16 CLIP	14.9	9.7	50.1	14.5	16.3	0.27	0.58
VDPG [11]		71.4 (0.2)	30.1 (0.3)	93.2 (0.3)	37.8 (0.5)	52.7 (0.3)	0.38 (0.02)	0.77 (0.02)
L2C [10]		73.4 (0.4)	**35.2 (0.3)**	94.2 (0.2)	40.9 (0.4)	54.8 (0.1)	0.50 (0.02)	0.80 (0.03)
DA-MergeLoRA		**73.70 (1.49)**	33.62 (1.63)	**96.90 (0.14)**	**52.26 (0.16)**	**57.60 (0.14)**	**0.56 (0.01)**	**0.84 (0.01)**

(b) Evaluation of our method (DA-MergeLoRA) on the WILDS datasets (iWildCam, Camelyon17, FMoW, and PovertyMap) under out-of-distribution (OOD) test conditions.

4.1 Main Results

Table 1a and Table 1b present the performance of our hypernetwork on DomainNet and WILDS, respectively.

DomainNet. Compared to the state-of-the-art (SOTA) ViT-B/16 FSTT-DA method L2C [10], our approach improves average accuracy on DomainNet by 1.24%. Our model outperforms all methods across all domains except "quickdraw," where it performs slightly lower (−0.83%) than MIRO [7]. Relative to the SOTA CNN-based architecture MABN [52], our method improves average accuracy on DomainNet by 16.94%.

WILDS. Compared to the SOTA ViT-B/16 FSTT-DA method L2C [10], our approach improves average accuracy by 2.80% on FMoW, 2.70% on Camelyon17,

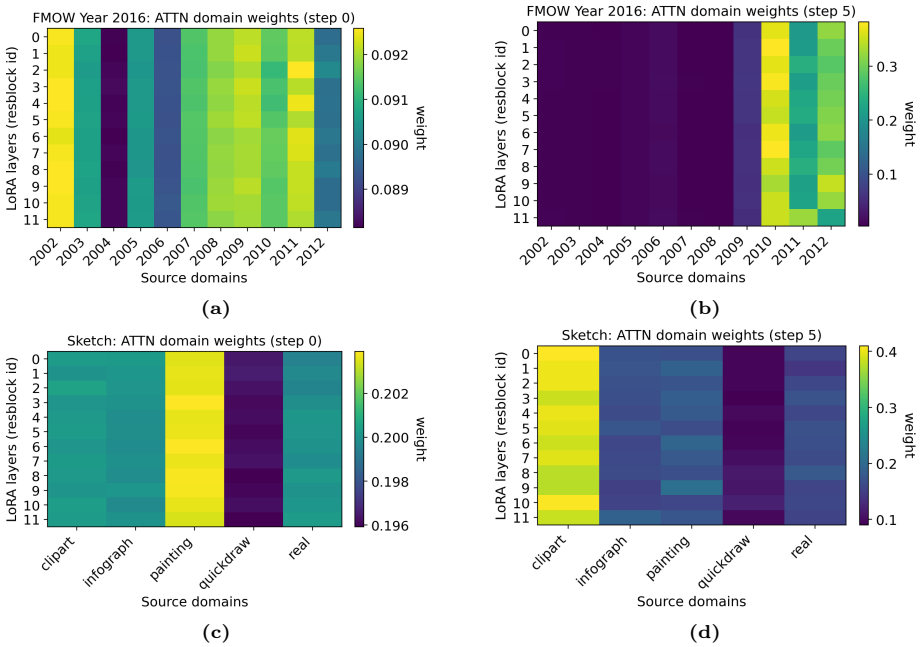

Fig. 2: Heatmaps of average merge weights. These plots illustrate the average merge weights per-attention-layer produced by the hypernetwork for each of the source domains. The x-axis shows the source domains and the y-axis shows the transformer layer. Plots (a) and (c) show the merge weights before training on FMoW ("Year 2016" target domain) and DomainNet ("Sketch" target domain). Plots (b) and (d) show results after training. Pre-training merge weights are nearly uniform; post-training, the hypernetwork prioritizes visually similar source domains over dissimilar or difficult domains.

and 0.24% on iWildCam, along with an 11.36% gain in worst-case accuracy on FMoW. For the PovertyMap regression task, our method increases WC Pearson correlation by 0.06 and average Pearson correlation by 0.04. Our model outperforms L2C on all metrics except the iWildCam F1-score, where it performs slightly worse (-1.58%). Relative to the SOTA CNN-based architecture MABN [52], our method improves average accuracy by 4.40% on FMoW, 4.50% on Camelyon17, and boosts worst-case accuracy on FMoW by 15.66%, while underperforming only on iWildCam and tying MABN on the PovertyMap regression task.

4.2 Ablation and Analyses

We conduct analyses and ablation studies examining merge weights before and after hypernetwork training, evaluating various LoRA training techniques, and assessing the impact of different architectural components. Additional analyses are provided in the supplementary material (Appendix A.4).

Table 2: Ablation experiments on the WILDS classification datasets. Results are reported as mean (standard deviation) over three trials with different random seeds. The † symbol denotes the official dataset metric.

Method	iWildCam Acc	iWildCam Macro F1†	Camelyon17 Acc†	FMoW WC Acc†	FMoW Acc
LoRA-universal	72.77 (3.13)	28.88 (1.12)	89.81 (6.89)	51.02 (1.28)	55.69 (0.24)
LoRA-average	14.98 (0.19)	10.21 (0.05)	95.26 (0.12)	20.63 (0.08)	21.95 (0.11)
LoRA-entropy-average	41.89 (0.81)	12.24 (0.22)	95.28 (0.09)	21.71 (0.50)	23.15 (0.25)
LoRA-TIES	49.85 (3.25)	12.94 (0.84)	95.95 (0.18)	38.54 (1.27)	43.98 (1.06)
LoRA-TIES-per-column	51.25 (3.09)	13.44 (0.20)	95.35 (0.45)	40.34 (1.12)	45.35 (0.89)
LoRA-AdaMerge	58.02 (0.92)	18.66 (0.54)	96.10 (0.63)	44.85 (0.19)	50.19 (0.08)
DA-MergeLoRA	**73.70 (1.49)**	**33.62 (1.63)**	**96.90 (0.14)**	**52.26 (0.16)**	**57.60 (0.14)**

(a) Comparison of LoRA training strategies: LoRA-universal (a single LoRA model trained on all source data combined), LoRA-average (simple parameter averaging), LoRA-entropy-average (few-shot entropy-weighted averaging), LoRA-TIES (TIES merging applied to all LoRA parameters at once), LoRA-TIES-per-column (TIES merging applied to each LoRA ΔW column separately), LoRA-AdaMerge (few-shot AdaMerging applied to LoRA parameters), and ours (hypernetwork-based merging).

Method	iWildCam Acc	iWildCam Macro F1†	Camelyon17 Acc†	FMoW WC Acc†	FMoW Acc
No layer and layer type token	72.31 (0.43)	28.95 (0.78)	96.86 (0.34)	49.16 (0.34)	54.79 (0.05)
No domain token	67.21 (1.50)	25.72 (1.81)	**96.92 (0.12)**	48.80 (0.38)	54.16 (0.02)
No target images (noise)	55.86 (1.00)	17.39 (0.66)	95.86 (0.48)	43.04 (0.28)	48.43 (0.18)
No target images (trainable vector)	63.47 (1.30)	19.92 (0.65)	**96.92 (0.14)**	51.98 (0.18)	57.36 (0.13)
DA-MergeLoRA	**73.70 (1.49)**	**33.62 (1.63)**	96.90 (0.14)	**52.26 (0.16)**	**57.60 (0.14)**

(b) Ablation studying the effect of: (i) removing the layer and layer-type tokens, which indicate the transformer layer from which each LoRA column originates; (ii) removing the domain tokens, which identify the source domain of each column; (iii) providing random noise as input to the hypernetwork instead of target images for the support set; and (iv) replacing the target-embedding branch with a trainable query vector that learns a general merging policy.

Merging weight heatmaps. To visualize whether the hypernetwork learns meaningful merging weights, we plot heatmaps of average per-domain merging weights before and after training for 5 epochs. Fig. 2 shows results for FMoW ("Year 2016" target domain) and DomainNet ("Sketch" target domain). The x-axis denotes source domains (FMoW: years; DomainNet: image styles), and the y-axis denotes the CLIP transformer layers. Before training, weights are nearly uniform ($\sim$ 0.09 for FMoW, $\sim$ 0.20 for DomainNet). After training, the hypernetwork shifts merge weights towards domains visually similar to the target domain (later years for FMoW, "ClipArt" for "Sketch"), and down-weights dissimilar and difficult domains (earlier years for FMoW, "QuickDraw" for sketch), indicating it learns to emphasize relevant domains.

LoRA training techniques. We compare our method against five additional techniques for training LoRA on the source data: (i) LoRA-universal – Train a single LoRA model on all source domains combined into one dataset. This yields a domain-invariant LoRA module, rather than domain-specific modules; (ii) LoRA-average – Combine domain-specific LoRA modules by simple parameter-wise averaging; (iii) LoRA-entropy-average – A simple few-shot method where we use a batch of target images to compute the entropy using each source LoRA model. The inverse entropy (normalized) is used as the merging weight,

where lower entropy implies higher weight; (iv) LoRA-TIES and LoRA-TIES-per-column – We implement the TIES [56] merging procedure to merge LoRA modules. We evaluate two variants: one that merges all LoRA ΔW matrices jointly, and one that merges individual ΔW columns separately (similar to our approach). We use the same hyperparameter settings recommended in the original paper: trim step K=20%, sign election using resolve = mass, and disjoint merging using merge = dis-mean; (v) LoRA-AdaMerge – An adaptation of AdaMerging [57] to our few-shot setting, where entropy minimization is performed to determine the merge weights for each LoRA model.

Results are shown in Table 2a on the WILDS classification datasets. Our method outperforms the other LoRA techniques. On harder datasets (iWildCam, FMoW), simple averaging, entropy averaging, TIES, and AdaMerging perform markedly worse than the hypernetwork (between 15% to 23% decrease on iWildCam F1-score, and between 7% to 32% decrease on FMoW worst-case accuracy). On datasets with fewer source domains and more balanced per-domain classes (Camelyon17), simpler merging methods degrade less ($\sim$ 2% decrease), suggesting the hypernetwork excels in more complex merging situations. Our model also outperforms the universal LoRA in all cases, highlighting the benefit of domain-specific knowledge.

Component analysis. We remove specific components of the hypernetwork to evaluate their impact on performance. The components examined are: (i) Layer and layer-type tokens – We remove the learnable transformer layer ID token embeddings and layer-type (attention or projection) token embeddings appended to the LoRA columns; (ii) Domain tokens – We remove the learnable domain-specific token embeddings appended to the LoRA columns; (iii) Conditional target images (replaced with noise) – We remove the pseudo-target and target images as inputs to the hypernetwork during training and inference, replacing them with random noise; and (iv) Conditional target images (replaced with a trainable vector) – We remove the target-image embedding branch and replace it with a trainable vector; instead of using pseudo-target images as the support set during training, we train a learnable query vector, which represents learning a general merging policy.

Table 2b reports the results on WILDS classification datasets. Removing the conditional target images reduces F1 performance by 13.7%–16.2% on iWildCam, which is the most difficult dataset with 48 target domains exhibiting significant domain shifts. Removing conditional images from FMoW also decreases performance, though not as severely. The learnable query vector reduces performance less than noise as input, indicating that a general merging policy can learn useful priors from the source domains. However, as seen in iWildCam, a domain-specific model performs better on complex domain shifts. Removing the layer tokens and domain tokens also lowers performance for both iWildCam and FMoW, showing that appending layer- and domain-specific tokens provides useful information. For Camelyon17, architectural modifications have negligible impact. As Camelyon17 is a less challenging dataset with performance nearing

saturation (∼96%), the optimal merge weights are inherently near-uniform, as evidenced by the strong performance of simple parameter-wise averaging (Table 2a).

Additional ablations and analyses. We provide additional ablations and analyses in the supplementary material (Appendix A.4). These include architectural comparisons between MLP and cross-attention hypernetworks (Suppl. Table 2) and the impact of merging granularity (per-model, per-layer, and per-column) (Suppl. Table 3). Sensitivity analyses on support set batch size and number of source domains show our method remains stable across different scales (Suppl. Tables 4, 5). Analyzing the hardest cases for our hypernetwork (iWildCam F1-score, DomainNet "QuickDraw" accuracy), we find iWildCam's lower performance stems from class imbalance, while QuickDraw's stems from inherent domain difficulty. Qualitative examples and t-SNE visualizations further show improved class separation and error reduction over baseline CLIP (Suppl. Figs. 1, 2), with predicted merge weights positively correlating with domain similarity (Suppl. Fig. 3). Finally, computational analysis shows that DA-MergeLoRA is roughly 50% faster and uses 50% less memory at inference than prior FSTT-DA frameworks (Suppl. Table 7).

5 Conclusion

In this work, we introduce a model-merging framework for few-shot test-time domain adaptation. We capture domain-specific information by training a LoRA module for each source domain. A hypernetwork, trained via meta-learning, then generates merging weights to optimally combine these modules for adaptation to a target domain, given a batch of unlabeled target images. Our method effectively adapts to novel domains and achieves state-of-the-art performance across diverse benchmarks. This work also opens several avenues for future research, including: (i) exploring merging strategies beyond meta-learning (e.g., reinforcement learning); (ii) developing mechanisms to handle class imbalance and novel classes; (iii) leveraging CLIP's text encoder for stronger vision–language alignment; (iv) increasing the diversity of pseudo-target domains during meta-training through data augmentation or style mixing; (v) designing richer hypernetwork architectures; and (vi) employing LoRA column-alignment strategies to improve merging.

Acknowledgements

This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

1. Akiba, T., Shing, M., Tang, Y., Sun, Q., Ha, D.: Evolutionary optimization of model merging recipes. Nature Mach. Intell. **7**(2), 195–204 (2025)
2. Arjovsky, M., Bottou, L., Gulrajani, I., Lopez-Paz, D.: Invariant risk minimization. arXiv preprint arXiv:1907.02893 (2019)
3. Bandi, P., Geessink, O., Manson, Q., Van Dijk, M., Balkenhol, M., Hermsen, M., Bejnordi, B.E., Lee, B., Paeng, K., Zhong, A., et al.: From detection of individual metastases to classification of lymph node status at the patient level: the camelyon17 challenge. IEEE transactions on medical imaging **38**(2), 550–560 (2018)
4. Beery, S., Agarwal, A., Cole, E., Birodkar, V.: The iwildcam 2021 competition dataset. arXiv preprint arXiv:2105.03494 (2021)
5. Blanchard, G., Lee, G., Scott, C.: Generalizing from several related classification tasks to a new unlabeled sample. Adv. Neural Inform. Process. Syst. **24** (2011)
6. Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., Erhan, D.: Domain separation networks. Adv. Neural Inform. Process. Syst. **29** (2016)
7. Cha, J., Lee, K., Park, S., Chun, S.: Domain generalization by mutual-information regularization with pre-trained models. In: Eur. Conf. Comput. Vis. pp. 440–457. Springer (2022)
8. Charakorn, R., Cetin, E., Tang, Y., Lange, R.T.: Text-to-lora: Instant transformer adaption. arXiv preprint arXiv:2506.06105 (2025)
9. Chen, D., Wang, D., Darrell, T., Ebrahimi, S.: Contrastive test-time adaptation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 295–305 (2022)
10. Chi, Z., Gu, L., Liu, H., Wang, Z., Wu, Y., Wang, Y., Plataniotis, K.N.: Learning to adapt frozen clip for few-shot test-time domain adaptation. In: Int. Conf. Learn. Represent. (2025), https://openreview.net/forum?id=TD3SGJfBC7
11. Chi, Z., Gu, L., Zhong, T., Liu, H., Yu, Y., Plataniotis, K.N., Wang, Y.: Adapting to distribution shift by visual domain prompt generation. In: Int. Conf. Learn. Represent. (2024), https://proceedings.iclr.cc/paper_files/paper/2024/file/440f269a4a6b9d51c51b4997963761ff-Paper-Conference.pdf
12. Chi, Z., Wu, Y., Gu, L., Liu, H., Wang, Z., Zhang, Y., Wang, Y., Plataniotis, K.: Plug-in feedback self-adaptive attention in clip for training-free open-vocabulary segmentation. In: Int. Conf. Comput. Vis. pp. 22815–22825 (2025)
13. Choshen, L., Venezian, E., Slonim, N., Katz, Y.: Fusing finetuned models for better pretraining. arXiv preprint arXiv:2204.03044 (2022)
14. Christie, G., Fendley, N., Wilson, J., Mukherjee, R.: Functional map of the world. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 6172–6180 (2018)
15. Csurka, G., et al.: Domain adaptation in computer vision applications, vol. 2. Springer (2017)
16. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: Int. Conf. Mach. Learn. pp. 1126–1135. PMLR (2017)
17. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., March, M., Lempitsky, V.: Domain-adversarial training of neural networks. J. Mach. Learn. Res. **17**(59), 1–35 (2016)
18. Gargiulo, A.A., Crisostomi, D., Bucarelli, M.S., Scardapane, S., Silvestri, F., Rodola, E.: Task singular vectors: Reducing task interference in model merging. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 18695–18705 (2025)
19. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al.: Lora: Low-rank adaptation of large language models. Int. Conf. Learn. Represent. **1**(2), 3 (2022)

20. Huang, C., Liu, Q., Lin, B.Y., Pang, T., Du, C., Lin, M.: Lorahub: Efficient cross-task generalization via dynamic lora composition. In: Conf. on Language Modeling (COLM) (2024), https://openreview.net/forum?id=TrloAXEJ2B
21. Ilharco, G., Ribeiro, M.T., Wortsman, M., Gururangan, S., Schmidt, L., Hajishirzi, H., Farhadi, A.: Editing models with task arithmetic. arXiv preprint arXiv:2212.04089 (2022)
22. Jiang, L., Ma, R., Gu, L., Wang, Z., Zuo, X., Wang, Y.: Pointmac: Meta-learned adaptation for robust test-time point cloud completion. Adv. Neural Inform. Process. Syst. **38**, 44992–45017 (2026)
23. Jin, X., Ren, X., Preotiuc-Pietro, D., Cheng, P.: Dataless knowledge fusion by merging weights of language models. arXiv preprint arXiv:2212.09849 (2022)
24. Koh, P.W., Sagawa, S., Marklund, H., Xie, S.M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R.L., Gao, I., et al.: Wilds: A benchmark of in-the-wild distribution shifts. In: Int. Conf. Mach. Learn. pp. 5637–5664. PMLR (2021)
25. Lee, G., Jang, W., Kim, J., Jung, J., Kim, S.: Domain generalization using large pretrained models with mixture-of-adapters. arXiv preprint arXiv:2310.11031 (2024), https://arxiv.org/abs/2310.11031
26. Li, Y., Gao, V., Zhang, C., Torkamani, M.: Ensembles of low-rank expert adapters. In: Int. Conf. Learn. Represent. (2025), https://openreview.net/forum?id=l0gZS0sAlf
27. Liang, J., He, R., Tan, T.: A comprehensive survey on test-time adaptation under distribution shifts. Int. J. Comput. Vis. **133**(1), 31–64 (2025)
28. Liang, J., Hu, D., Feng, J.: Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In: Int. Conf. Mach. Learn. pp. 6028–6039. PMLR (2020)
29. Liu, H., Wu, Z., Li, L., Salehkalaibar, S., Chen, J., Wang, K.: Towards multi-domain single image dehazing via test-time training. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5831–5840 (2022)
30. Liu, J., Yang, S., Jia, P., Zhang, R., Lu, M., Guo, Y., Xue, W., Zhang, S.: Vida: Homeostatic visual domain adapter for continual test time adaptation. In: Int. Conf. Learn. Represent. (2024)
31. Long, M., Cao, Y., Wang, J., Jordan, M.: Learning transferable features with deep adaptation networks. In: Int. Conf. Mach. Learn. pp. 97–105. PMLR (2015)
32. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
33. Marczak, D., Magistri, S., Cygert, S., Twardowski, B., Bagdanov, A.D., van de Weijer, J.: No task left behind: Isotropic model merging with common and task-specific subspaces. arXiv preprint arXiv:2502.04959 (2025)
34. Matena, M.S., Raffel, C.A.: Merging models with fisher-weighted averaging. Adv. Neural Inform. Process. Syst. **35**, 17703–17716 (2022)
35. Nam, H., Lee, H., Park, J., Yoon, W., Yoo, D.: Reducing domain gap by reducing style bias. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 8690–8699 (2021)
36. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Adv. Neural Inform. Process. Syst. **32** (2019)
37. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: Int. Conf. Comput. Vis. pp. 1406–1415 (2019)
38. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: Int. Conf. Mach. Learn. pp. 8748–8763. PmLR (2021)

39. Shah, V., Ruiz, N., Cole, F., Lu, E., Lazebnik, S., Li, Y., Jampani, V.: Ziplora: Any subject in any style by effectively merging loras. In: Eur. Conf. Comput. Vis. pp. 422–438. Springer (2024)
40. Shao, Y., Lin, X., Long, X., Chen, S., Yan, M., Liu, Y., Yan, Z., Ma, A., Tang, H., Guo, J.: Icm-fusion: In-context meta-optimized lora fusion for multi-task adaptation. arXiv preprint arXiv:2508.04153 (2025)
41. Shenaj, D., Bohdal, O., Ozay, M., Zanuttigh, P., Michieli, U.: Lora. rar: Learning to merge loras via hypernetworks for subject-style conditioned image generation. In: Int. Conf. Comput. Vis. pp. 16132–16142 (2025)
42. Shu, Y.C., Lin, Z., Wang, Y.: Regcl: Continual adaptation of segment anything model via model merging. arXiv preprint arXiv:2507.12297 (2025)
43. Stoica, G., Ramesh, P., Ecsedi, B., Choshen, L., Hoffman, J.: Model merging with svd to tie the knots. arXiv preprint arXiv:2410.19735 (2024)
44. Sun, B., Saenko, K.: Deep coral: Correlation alignment for deep domain adaptation. In: Eur. Conf. Comput. Vis. pp. 443–450. Springer (2016)
45. Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A., Hardt, M.: Test-time training with self-supervision for generalization under distribution shifts. In: Int. Conf. Mach. Learn. pp. 9229–9248. PMLR (2020)
46. Wang, D., Shelhamer, E., Liu, S., Olshausen, B., Darrell, T.: Tent: Fully test-time adaptation by entropy minimization. In: Int. Conf. Learn. Represent. (2021)
47. Wang, J., Lan, C., Liu, C., Ouyang, Y., Qin, T., Lu, W., Chen, Y., Zeng, W., Yu, P.: Generalizing to unseen domains: A survey on domain generalization. IEEE Trans. on Knowledge and Data Engineering (2022)
48. Wang, Z., Chi, Z., Wu, Y., Gu, L., Liu, Z., Plataniotis, K., Wang, Y.: Distribution alignment for fully test-time adaptation with dynamic online data streams. In: Eur. Conf. Comput. Vis. pp. 332–349. Springer (2024)
49. Wortsman, M., Ilharco, G., Gadre, S.Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A.S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al.: Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In: Int. Conf. Mach. Learn. pp. 23965–23998. PMLR (2022)
50. Wortsman, M., Ilharco, G., Kim, J.W., Li, M., Kornblith, S., Roelofs, R., Lopes, R.G., Hajishirzi, H., Farhadi, A., Namkoong, H., Schmidt, L.: Robust fine-tuning of zero-shot models. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 7959–7971 (June 2022)
51. Wu, X., Huang, S., Wei, F.: Mixture of lora experts. arXiv preprint arXiv:2404.13628 (2024)
52. Wu, Y., Chi, Z., Wang, Y., Plataniotis, K.N., Feng, S.: Test-time domain adaptation by learning domain-aware batch normalization. In: AAAI. vol. 38, pp. 15961–15969 (2024)
53. Wu, Y., Yan, Y., Chen, T., Chi, Z., Wu, Z., Jin, Y., Wang, Y., Li, Z.: Talon: Test-time adaptive learning for on-the-fly category discovery. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 22259–22269 (2026)
54. Wu, Y., Piao, H., Huang, L.K., Wang, R., Li, W., Pfister, H., Meng, D., Ma, K., Wei, Y.: Sd-lora: Scalable decoupled low-rank adaptation for class incremental learning. arXiv preprint arXiv:2501.13198 (2025)
55. Xu, M., Zhang, J., Ni, B., Li, T., Wang, C., Tian, Q., Zhang, W.: Adversarial domain adaptation with domain mixup. In: AAAI. vol. 34, pp. 6502–6509 (2020)
56. Yadav, P., Tam, D., Choshen, L., Raffel, C.A., Bansal, M.: Ties-merging: Resolving interference when merging models. Adv. Neural Inform. Process. Syst. **36**, 7093–7115 (2023)

57. Yang, E., Wang, Z., Shen, L., Liu, S., Guo, G., Wang, X., Tao, D.: Adamerging: Adaptive model merging for multi-task learning. In: Int. Conf. Learn. Represent. vol. 2024, pp. 22743–22763 (2024)
58. Yeh, C., Perez, A., Driscoll, A., Azzari, G., Tang, Z., Lobell, D., Ermon, S., Burke, M.: Using publicly available satellite imagery and deep learning to understand economic well-being in africa. Nature communications **11**(1), 2583 (2020)
59. Zhang, M., Marklund, H., Dhawan, N., Gupta, A., Levine, S., Finn, C.: Adaptive risk minimization: Learning to adapt to domain shift. Adv. Neural Inform. Process. Syst. **34**, 23664–23678 (2021)
60. Zheng, Z., Yue, X., Wang, K., You, Y.: Prompt vision transformer for domain generalization. arXiv preprint arXiv:2208.08914 (2022)
61. Zhong, T., Chi, Z., Gu, L., Wang, Y., Yu, Y., Tang, J.: Meta-dmoe: Adapting to domain shift by meta-distillation from mixture-of-experts. Adv. Neural Inform. Process. Syst. **35**, 22243–22257 (2022)
62. Zhou, K., Liu, Z., Qiao, Y., Xiang, T., Loy, C.C.: Domain generalization: A survey. IEEE Trans. Pattern Anal. Mach. Intell. **45**(4), 4396–4415 (2022)