Supplementary Material for SCOPE: Shared Content Orchestration of Parameter-Efficient Experts for Federated Domain Generalization

Abstract

This document provides supplementary material for our main paper, organized to enhance clarity and ensure full reproducibility. We offer expanded details on our core architectural and algorithmic contributions and provide a qualitative t-SNE visualization to empirically validate the domain-invariant and class-discriminative properties of our learned feature space. For reproducibility, we include a full breakdown of our experimental setup, complete data from our ablation studies, a detailed communication cost analysis, and a comprehensive table of all codebase hyperparameters, along with the exact commands used to generate our main results.

A. Appendix Overview

This supplementary document is organized as follows:

- **Section B:** We provide an expanded discussion of the core components of the SCOPE framework, including the motivation and mechanics of the architectural partition, DRIFT, ELMA, and L-TTA.
- **Section C:** We detail our experimental setup, including dataset specifics, evaluation protocols, and hardware/software stack.
- **Section D:** We present the full, detailed results of our ablation studies in tabular format.
- **Section E:** We provide a comprehensive table of all configurable hyperparameters in our codebase for full reproducibility.
- **Section F:** We provide a detailed, step-by-step calculation of the per-round communication cost to ensure full transparency and reproducibility of our efficiency claims.
- **Section G:** We provide a qualitative analysis of the learned feature spaces via t-SNE visualizations to empirically validate our claims of improved class separability and domain invariance.

B. Detailed Framework Components

B.1. Architectural Partitioning

The central hypothesis of SCOPE is that the federated learning process should be aligned with the natural feature hierarchy of deep networks. Our architectural partition is the direct implementation of this principle. By designating the initial N layers as a **Private Trunk**, we create a structural information bottleneck. These layers, trained only on a client's local data and never aggregated, are compelled by the optimization process to specialize in modeling the low-level, domain-specific statistics (e.g., texture, color palettes, artistic style) characteristic of that client's domain.

Conversely, the deeper layers form the **Shared Trunk**. As these are the only parameters subjected to Federated Averaging, the aggregation process itself acts as an implicit domain filter. Parameter updates that are only beneficial for a single domain are attenuated, while updates corresponding to domain-invariant semantic content are consistently reinforced across the federation. This creates a powerful inductive bias, encouraging the shared trunk to learn a robust and generalizable representation of high-level concepts.

B.2. DRIFT: Federated Moment Matching

The implicit regularization from FedAvg can be a weak and noisy signal. DRIFT (**D**istributional **R**egularization for **I**nvariant **F**ederated **T**argets) provides a stronger, more explicit gradient towards domain invariance. It operationalizes the principle of moment matching by aligning the first two moments (mean and variance) of the feature distributions at each layer of the shared trunk. This is a principled and tractable surrogate for full distributional matching, as deep features often exhibit approximately Gaussian behavior. By creating global target statistics in each round, DRIFT provides a clear, domain-agnostic target for each client's local training, creating a "cascaded information filter" that progressively purifies the features of domain-specific information.

B.3. ELMA: Expert-Level Matched Aggregation

The use of Mixture-of-Experts (MoE) adapters is critical for parameter efficiency but introduces the problem of permutation ambiguity in federated settings. ELMA (**E**xpert-**L**evel **M**atched **A**ggregation) is our solution to this challenge. The key insight is to use the router's learnable gating vectors as semantic signatures for each expert. Since these vectors are optimized to route specific types of features to their corresponding expert, they serve as a compact and functionally meaningful representation of what each expert has specialized in. By solving the assignment problem on these signatures using the Hungarian algorithm, ELMA ensures that functionally equivalent experts are aligned before their parameters are averaged, preventing destructive interference and enabling constructive knowledge transfer.

B.4. L-TTA: Layer-Wise Test-Time Adaptation

A single, static global model represents an averaged consensus that may be suboptimal for a specific test sample. L-TTA (**L**ayer-**W**ise **T**est-**T**ime **A**daptation) is designed to leverage the full library of specialized client models generated during training. Its layer-wise nature is its defining characteristic. A test image might share low-level texture features with one source domain but high-level object composition with another. By independently computing similarity and blending parameters at each layer of the shared trunk, L-TTA can compose a unique, bespoke model that is optimally adapted to the specific multi-level characteristics of the input batch, fully realizing the potential of our hierarchical framework at inference time.

C. Experimental Setup

Datasets. We evaluate SCOPE on three standard domain generalization benchmarks: **PACS** [2], **OfficeHome** [3], and **VLCS** [1]. These datasets feature significant domain shifts across photographic, artistic, and sketch-based domains.

Evaluation Protocol. All experiments adhere to the standard leave-one-domain-out protocol. For a dataset with K domains, we conduct K independent experiments. In each experiment, one domain is held out as the unseen target domain for testing, while the remaining $K - 1$ domains are used for training, with each treated as a separate client in our federated setup. The final reported accuracy is the average across all K splits.

Hardware and Software. Experiments were conducted on a high-performance computing cluster equipped with NVIDIA A100 and V100 GPUs. Our implementation is built on PyTorch 1.12 and Python 3.9, with CUDA 11.3.

D. Full Ablation Study Results

This section contains the complete data from the ablation studies presented graphically in the main paper.

Table 1. Convergence analysis showing model accuracy (%) at each communication round across three datasets.

Round	PACS	OfficeHome	VLCS
0	97.36	88.17	84.52
1	97.75	88.93	84.96
2	97.94	88.96	85.13
3	97.88	89.22	85.38
4	97.81	89.19	85.40
5	97.75	89.16	**85.58**
6	97.88	88.84	84.96
10	97.81	88.73	83.91
15	97.49	88.55	83.59
20	97.45	88.54	82.75

Table 2. Sensitivity analysis of model accuracy (%) to the number of private blocks (N). Performance peaks at $N = 4$.

N (Private Blocks)	PACS	OfficeHome	VLCS
4	**97.94**	**89.22**	**85.58**
6	97.61	88.96	85.38
8	97.73	88.96	84.89
10	97.48	87.90	85.04

Table 3. Sensitivity analysis of model accuracy (%) to the DRIFT regularization weight (λ). Performance peaks at $\lambda = 5$.

λ (DRIFT Weight)	PACS	OfficeHome	VLCS
0.1	97.88	88.98	84.83
1.0	97.80	88.93	84.74
5.0	**97.94**	**89.22**	**85.58**
10.0	97.80	88.95	85.23

E. Hyperparameter Configuration

To ensure full reproducibility, Table 4 lists all configurable hyperparameters in our codebase and their default values.

Table 4. Full list of codebase hyperparameters and their default values.

Parameter	Default Value	Description
General Settings		
dataset	[officehome, PACS, VLCS]	Name of the dataset to use.
model	clip_moe_style	Model architecture name.
num_classes	[65, 7, 5]	Number of output classes.
batch_size	32	Batch size for local client training.
test_batch_size	32	Batch size for evaluation.
local_epochs	10	Number of local training epochs per communication round.
comm	20	Total number of communication rounds.
lr	0.001	Learning rate for the SGD optimizer.
lr_policy	step	Learning rate scheduler policy.
SCOPE Architecture Settings		
private_blocks_count	(4)	Number of transformer blocks in the Private Trunk.
where	every_qkv	Where to inject MoE layers in the transformer blocks.
agg_strategy	elma	Server-side aggregation strategy (e.g., fedavg, elma).
elma_temp	0.5	Temperature for probabilistic anchor selection in ELMA.
DRIFT Settings		
do_drift	(True)	Flag to enable DRIFT (moment matching).
drift_lambda_mean	(5.0)	Regularization weight λ for the mean matching loss.
drift_lambda_var	(5.0)	Regularization weight λ for the variance matching loss.
L-TTA (Test-Time Adaptation) Settings		
tta_blend_target	layer_wise	Which trunk(s) to adapt at test time.
domain_tracker	offline_cosine_muvar	Domain statistics tracking method.
avg_tokens	(False)	Whether to average token features.
batch_agg_type	pre_similarity	How to aggregate features in a batch for similarity computation.
inv_temp	2.0	Inverse temperature (1/T) for similarity calculation.
normalize_features	True	Whether to normalize features by standard deviation.
Other MoE Settings		
aux_loss_weight	0.01	Weight for the MoE auxiliary load-balancing loss (λ_{aux}).

F. Detailed Communication Cost Analysis

To ensure transparency and reproducibility, this section provides a detailed breakdown of the communication cost per round for the SCOPE framework. The total cost is the sum of all parameters and statistics that are transmitted from a client to the central server during the upload phase of each communication round. This includes the parameters of the shared trunk and the statistics required for the DRIFT mechanism.

F.1. Experimental Configuration

The following analysis is based on the specific configuration used for our experiments on the PACS dataset:

- **Base Model:** A standard ViT-Base architecture with 12 transformer blocks and a hidden dimension of $d_{\text{model}} = 768$.
- **Architectural Partition:** The first 4 blocks are designated as the private trunk. The remaining 8 blocks (**5 through 12**) constitute the public (shared) trunk.
- **MoE Injection:** Following the framework in Section 3 of the main paper, lightweight MoE adapters are injected into the transformer blocks. In our experimental configuration, adapters are placed at **4 blocks** within the shared trunk (blocks 6, 8, 10, and 11) to balance expressiveness with communication efficiency.
- **DRIFT Statistics:** For each of the 8 public trunk layers, the client computes and transmits a mean vector (μ) and a variance vector (σ^2), not a full covariance matrix.
- **Dataset:** PACS, which has 7 classes.

F.2. Calculation of Trainable Model Parameters (Shared Trunk)

Only the parameters of the shared trunk are communicated to the server. This consists of the shared MoE adapters and the final classification head.

A. Shared MoE Adapter Parameters Each of the 4 shared MoE adapter layers has three trainable components: the expert weights, the router, and a bias adapter.

1. **Expert Weights:** Each expert implements the PHM-

based Kronecker factorization described in Eq. 3 of the main paper, with block dimension $p = 64$ and model dimension $d = 768$. The factored weight matrices A and B are implemented as low-rank projections. With $M = 4$ experts, the adapter projects across the Q, K, V subspaces (factor of 3) using an internal rank of 8:
   - Left factor (A): $M \times (p \times (d/p) \times \text{rank})$
     $4 \times (64 \times 12 \times 8) = 24,576$ parameters.
   - Right factor (B): $M \times (p \times \text{rank} \times 3(d/p))$
     $4 \times (64 \times 8 \times 36) = 73,728$ parameters.

   The total for the expert weights is $24,576 + 73,728 = 98,304$ parameters.
2. **Router (CosineTopKGate):** The router projects the input token features to a smaller dimension before computing similarity with the expert signature matrix.
   - Projector: $d_{\text{model}} \times d_{\text{proj}} + d_{\text{proj}}$
     $768 \times 256 + 256 = 196,864$ parameters.
   - Similarity Matrix: $d_{\text{proj}} \times M$
     $256 \times 4 = 1,024$ parameters.
   - Temperature: A single learnable scalar, 1 parameter.

   The total for the router is $196,864 + 1,024 + 1 = 197,889$ parameters.
3. **Bias Adapter:** A single learnable bias vector is added to the output of the MoE layer. Since adapters are injected across Q, K, V projections (`where=every_qkv`), the bias size is heads $\times$ head_dim $\times 3 = 12 \times 64 \times 3 = 2,304$.

The total parameter count for a single MoE adapter layer is therefore $98,304 + 197,889 + 2,304 = \mathbf{298,497}$. For all 4 shared MoE adapters, the total is $4 \times 298,497 = \mathbf{1,193,988}$ parameters.

B. Classification Head Parameters The final classification head is a standard linear layer. For the PACS dataset with 7 classes:

- Weights: $d_{\text{model}} \times \text{num\_classes} = 768 \times 7 = 5,376$.
- Bias: num_classes $= 7$.

The total for the classification head is $5,376 + 7 = \mathbf{5,383}$ parameters.

F.3. Calculation of DRIFT Statistics

The DRIFT mechanism requires clients to upload feature statistics for **every layer** in the public trunk, not just those with MoE adapters. In our configuration, this applies to all 8 public layers (blocks 5-12). We transmit the diagonal of the covariance matrix (i.e., the variance) to maintain communication efficiency.

- Mean Vector (μ): A vector of size $d_{\text{model}} = 768$.
- Variance Vector (σ^2): A vector of size $d_{\text{model}} = 768$.

The total statistics per layer are $768 + 768 = 1,536$ parameters. For all 8 public trunk layers, the total is $8 \times 1,536 = \mathbf{12,288}$ parameters.

F.4. Summary Table and Total Communication Cost

Table 5 summarizes the components of the per-round communication cost from one client to the server.

Table 5. Detailed breakdown of per-round communication cost.

Component	Sub-Component	Calculation	Parameters
Model Parameters	Shared MoE Adapters	4 layers $\times$ 298,497	1,193,988
	Classification Head	$768 \times 7 + 7$	5,383
	Subtotal		*1,199,371*
DRIFT Statistics	Mean Vectors	8 layers $\times$ 768	6,144
	Variance Vectors	8 layers $\times$ 768	6,144
	Subtotal		*12,288*
Total Cost		$\mathbf{1,199,371 + 12,288}$	**1,211,659**

The total communication cost per client per round for this configuration is **1.21M** parameters. This is a substantial reduction compared to fine-tuning the entire ViT-Base model ($\sim$86M parameters), demonstrating the parameter efficiency of our approach.

G. Qualitative Analysis of Feature Spaces

To provide a qualitative validation of our framework's ability to learn a domain-invariant and class-discriminative feature space, we present t-SNE visualizations of the feature embeddings from the final shared trunk layer in Figure 1. We compare the feature space learned by our full SCOPE framework against that of a standard `FedAvg` baseline on the PACS dataset. In the plots, colors represent semantic classes, while marker shapes represent the source domains.

The visualization yields two critical insights:

- **Superior Class Separability:** The `FedAvg` baseline (left panel) produces a highly entangled feature space. Class clusters (indicated by color) are poorly separated and exhibit significant overlap. In stark contrast, the SCOPE framework (right panel) learns a feature space with high class separability, where features form dense, compact, and well-separated clusters corresponding to their semantic class.
- **Demonstrable Domain Invariance:** The inclusion of domain markers provides strong evidence of domain invariance. In the SCOPE feature space, within each distinct class cluster, multiple domain markers (e.g., circles for photo, triangles for cartoon, squares for 'art_painting') are present and intermingled. This empirically demonstrates that SCOPE successfully maps samples from different domains to a common, shared representation based on their semantic content. The `FedAvg` baseline fails to achieve this, showing no discernible structure with respect to either class or domain.

In summary, this visualization provides strong qualitative evidence that SCOPE's hierarchical design and orchestration mechanisms successfully produce a feature space that

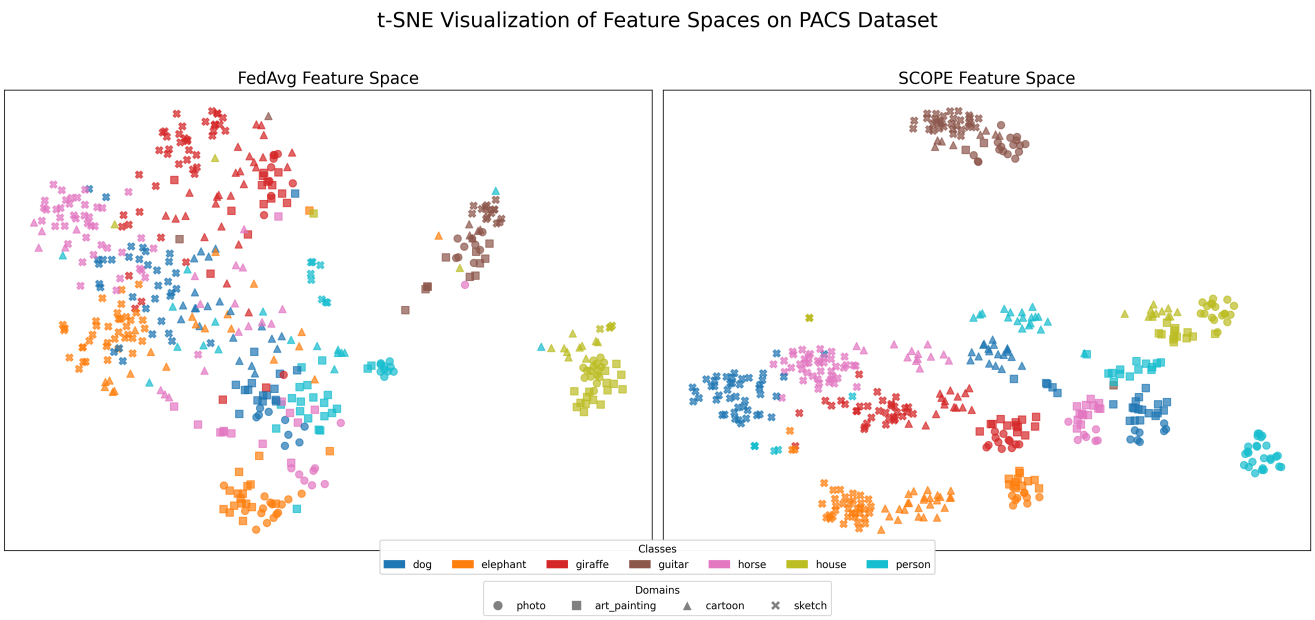

Figure 1. t-SNE visualization of feature spaces from the final shared trunk layer, evaluated on the PACS dataset. Colors correspond to semantic classes and marker shapes correspond to source domains. **(Left)** The `FedAvg` baseline learns an entangled representation where classes and domains are heavily mixed. **(Right)** The SCOPE framework learns a superior representation with distinct, compact class clusters that are demonstrably invariant to the source domain.

disentangles style (domain) from content (class), making it robust to domain shifts and highly effective for generalization.

References

[1] C. Fang, Y. Xu, and D. N. Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *Proceedings of the IEEE international conference on computer vision*, pages 1657–1664, 2013. 2

[2] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017. 2

[3] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sujoy Sethi. Deep hashing network for unsupervised domain adaptation. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5018–5027, 2017. 2