MetaVFM: Meta-Learned Efficient Adaptation of Vision Foundation Models for Medical Imaging

Anonymized Authors

Anonymized Affiliations
email@anonymized.com

Abstract. Adapting vision foundation models (VFMs) to medical and other domain-shifted tasks involves trade-offs among target-task accuracy, the number of trainable parameters, and the number of optimization steps required for convergence, considerations that are particularly pronounced in medical imaging. While parameter-efficient fine-tuning (PEFT) methods substantially reduce trainable parameters, they typically do not reduce the number of optimization steps, leaving the training step budget a dominant computational cost. Moreover, PEFT performance is highly sensitive to adapter placement, rank allocation, and initialization. These design choices are often set through per-dataset hyperparameter search, making adaptation both configuration-sensitive and training-intensive. To address these challenges, we propose MetaVFM, a meta-learning framework that predicts adapter placement, rank, and initialization for unseen datasets with no per-task search. MetaVFM learns a meta-space from a shared-weight Super-Adapter trained on 90+ datasets, enabling fast retrieval of a configuration and initialization that speeds fine-tuning convergence. We evaluate MetaVFM on 17 datasets, including multiple medical imaging benchmarks. MetaVFM attains the highest average accuracy, ranking top-1 or top-2 on nearly all datasets, while other methods are often strong on some datasets but weak on others, reflecting sensitivity to configuration and initialization. Under a fixed epoch budget, MetaVFM converges 2–3× faster than strong PEFT baselines, with particularly pronounced improvements on medical datasets, highlighting its effectiveness and robustness in the medical domain. The full code and experimental setup will be made public.

Keywords: Efficient Adaptation · Meta Learning · Medical Imaging.

1 Introduction

Vision foundation models (VFMs), such as DINO [31, 28], are increasingly adopted in medical imaging [26] and other visual tasks due to their strong transferable representations. However, their zero-shot performance often remains insufficient for specialized tasks [37], making efficient adaptation essential. Traditional strategies lie at two extremes: full fine-tuning, which is computationally intensive and prone to overfitting in limited-data settings [23], and linear probing, which is efficient but typically underperforms due to limited feature adaptation [1].

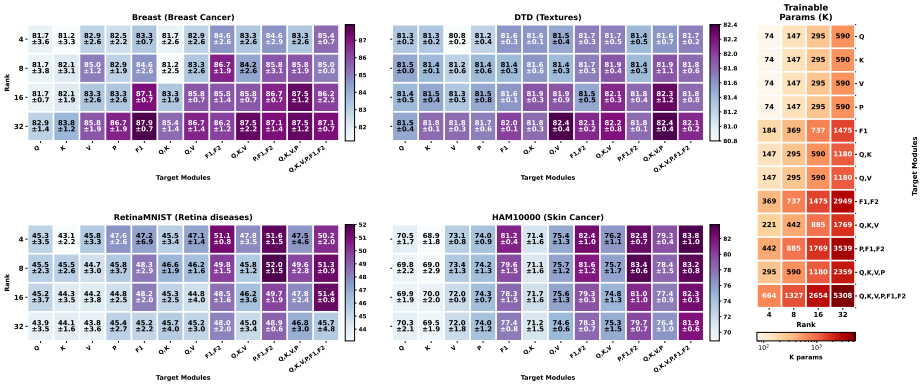


Fig. 1: LoRA configuration sensitivity across four datasets. Purple Heatmaps show validation accuracy (mean ± std over seeds) for different module–rank combinations. The right panel reports trainable parameter counts (K) for each configuration.

Parameter-Efficient Fine-Tuning (PEFT) methods provide a middle ground by updating only small adapter modules while freezing the backbone. Approaches such as LoRA [13], AdaptFormer [2], and Visual Prompt Tuning [14] substantially reduce trainable parameters and memory overhead. However, reducing parameter count does not necessarily reduce the number of optimization steps required for convergence. Even with a few trainable weights, adaptation still requires repeated passes through the backbone, leaving the training step budget a primary computational bottleneck.

Beyond step inefficiency, PEFT performance is sensitive to configuration choices. Accuracy depends critically on adapter placement, rank allocation, and initialization. As shown in Fig. 1, performance varies across module–rank combinations and does not change monotonically with capacity (Number of parameters). Moreover, optimal configurations differ across datasets; settings that perform well on one dataset may underperform on another, making per-task hyperparameter search common practice. This sensitivity reduces the reliability and scalability of PEFT adaptation across heterogeneous domains, especially in medical imaging, where shifts occur from several perspectives, including data, scanner, modality, and operator.

Several works have investigated how to select adapter placement and rank. Chen et al. [2] reported strong results from parallel adapters next to MLP layers, and Sorkhei et al. [32] highlighted post-attention projections as effective insertion points. On capacity, AutoLoRA [36] meta-learns per-layer ranks, while ARD-LoRA [19] adjusts ranks using learned scaling. However, these approaches treat placement, capacity, and initialization as largely independent design axes.

In practice, adapter placement, rank capacity, and initialization are tightly coupled. The insertion location influences gradient flow and representational updates, thereby affecting both the required capacity and the speed of convergence. A poorly chosen placement can slow optimization even with sufficient rank, while

an excessively high rank may waste computation if inserted ineffectively. These interdependencies become more pronounced when vision foundation models are deployed across heterogeneous datasets, including medical imaging tasks with varying acquisition protocols and visual characteristics. Configurations that perform well on one dataset may converge slowly or degrade on another. This motivates a unified framework that jointly models adapter placement, rank allocation, and convergence efficiency to enable reliable and computationally efficient adaptation across clinically relevant domains.

To address these challenges, we propose MetaVFM (Meta-learned efficient adaptation of Vision Foundation Models), a unified meta-learning framework that jointly models adapter placement, rank, and initialization. The approach leverages a shared-weight Super-Adapter and a dataset–adapter meta-space to enable configuration selection and initialization without per-task search. Our key contributions are:

- **Super-Adapter.** We design a shared-weight supernet (Super-Adapter) that unifies LoRA ranks and insertion positions within a single adapter module, enabling efficient exploration of diverse configurations without retraining and making large-scale evaluation computationally feasible.
- **Adapter Zoo.** We construct a large-scale adapter zoo using the Super-Adapter spanning diverse datasets and configurations, providing supervision for learning dataset–configuration relationships.
- **Meta-Adapter Space.** We learn a joint embedding of datasets and adapter configurations aligned by performance, enabling retrieval of adapter structure and initialization for new datasets and supporting efficient, robust fine-tuning, particularly in medical imaging.

2 Methodology

We propose a meta-learning framework that jointly models adapter placement, rank, and initialization under a constrained budget. Let $\mathcal{S}$ denote the adapter-eligible weight matrices $\{W_s \in R^{d_s^{\text{out}} \times d_s^{\text{in}}}\}$ of a pretrained model M (attention and MLP projections). For each $s \in \mathcal{S}$, we choose a rank r_s from a predefined pool of ranks $\mathcal{R} = \{r_1 > \cdots > r_K > 0\}$ (0 means "no adapter at layer s") and an initialization Φ_0 for the adapter parameters. We denote by $M_{r,\Phi}$ the model obtained by augmenting M with adapters of ranks $r = \{r_s\}_{s\in\mathcal{S}}$ and parameters Φ. The goal is to select (r, Φ_0) such that the fine-tuned adapted model M_{r,Φ_T} minimizes the validation loss on D_{val} under a parameter budget B:

$$\min_{r\in\mathcal{R}^{|\mathcal{S}|},\,\Phi_0} \mathcal{S}_{\text{val}}\big(M_{r,\Phi_T}; D_{\text{val}}\big) \quad \text{s.t.} \quad \sum_{s\in\mathcal{S}} r_s(d_s^{\text{in}}+d_s^{\text{out}}) \le B, \quad \Phi_T = \text{FT}\big(M_{r,\Phi_0}; D_{\text{tr}}, T\big). \tag{1}$$

D_{val} and D_{tr} are the validation and training sets, B limits the number of trainable adapter parameters, and $\text{FT}(\cdot)$ is the fine-tuning operator that takes an initialized adapted model M_{r,Φ_0}, runs T optimization steps on D_{tr} updating

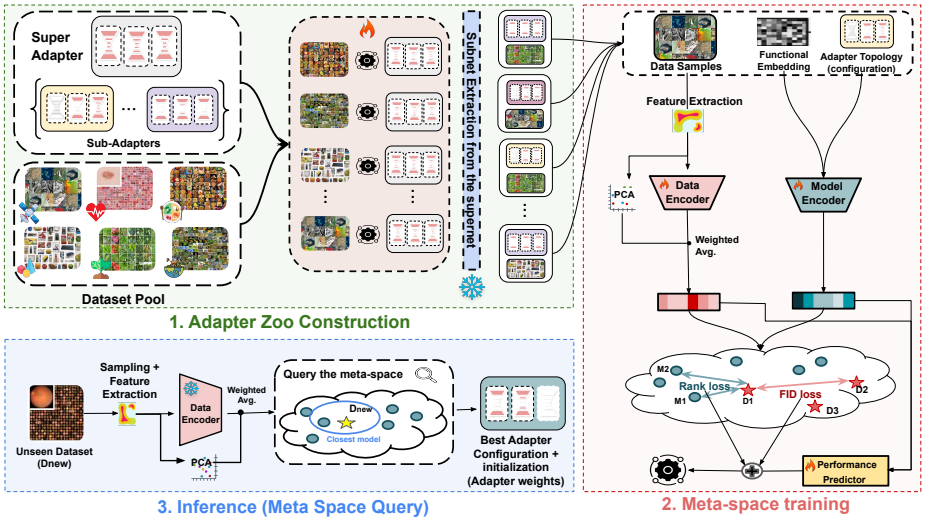


Fig. 2: Overview of MetaVFM. During training, a shared-weight Super-Adapter is trained per dataset to construct an adapter zoo. Datasets and adapter configurations are embedded into a joint meta-space optimized with performance, ranking, and FID losses. During inference, a new dataset embedding retrieves both adapter structure and initialization for efficient fine-tuning.

only adapter parameters, and returns its final state Φ_T. Prior work typically optimizes adapter placement or rank while relying on fixed or heuristic initialization, leaving structure and initialization decoupled. In contrast, we jointly meta-learn both. We first construct an adapter zoo using a weight-sharing Multi-Rank LoRA Super-Adapter (Fig. 2), then learn a dataset–adapter embedding aligned by performance. At inference, we retrieve both adapter structure and initialization for a new dataset via cosine similarity, followed by lightweight fine-tuning.

2.1 Adapter Zoo via Super-Adapter. Exhaustively training a separate LoRA adapter for every rank/insertion choice on every dataset is computationally prohibitive. We therefore use a shared-weight Super-Adapter that supports a discrete rank pool $\mathcal{R} = \{r_1 > \cdots > r_K > 0\}$ within a single module. Given an adapter-eligible matrix W, instead of training independent LoRA updates for each $r \in \mathcal{R}$, we parameterize a nested family of updates such that each rank corresponds to a prefix of shared factors:

$$y = \Big(W + \alpha\, \Delta W^{(k)}\Big)x, \quad \Delta W^{(k)} := B_1 \cdots B_k\, A_k \cdots A_1, \quad k \in \{1, \ldots, K\}. \tag{2}$$

Selecting prefix k yields a LoRA update with rank r_k, and higher-rank adapters reuse and extend the parameters of lower-rank ones. After training, the chosen prefix is collapsed into a standard LoRA layer by multiplying the factors, producing ordinary $(A^{(k)}, B^{(k)})$ for lightweight fine-tuning. In such a way, training one Super-Adapter per dataset yields an adapter zoo $\mathcal{Z} = \{(D_i, c_m, \widehat{P}_{i,m})\}$,

where $\widehat{P}_{i,m}$ is an approximation of the validation performance obtained by subsampling configuration c_m from the Super-Adapter trained on dataset D_i. This approach amortizes the evaluation of thousands of configurations across tasks.

2.2 Dataset–Adapter Meta-Space. We learn a joint embedding space that aligns datasets and adapter configurations using measured performance from the adapter zoo. This means that good configurations for a dataset lie close to its embedding, enabling retrieval-based adaptation. For adapter configurations, we combine structure (ranks and insertion placements) with a functional signature that measures how the configuration shifts backbone representations. Let M_{c,ϕ_c} be the model obtained by augmenting the frozen backbone M with configuration c and adapter weights ϕ_c. Using a probe set $\mathcal{X}_{\text{probe}}$, and penultimate-layer features $g(\cdot)$, we summarize the functional effect of c by the average feature shift Δ_c, and form the adapter embedding v_c by encoding the concatenation of this shift with configuration parameter $flat(c)$ using a neural network trained encoder E_m:

$$v_c = E_m\big([\text{flat}(c); \Delta_c]\big) \;\text{ s.t. }\; \Delta_c = \frac{1}{|\mathcal{X}_{\text{probe}}|} \sum_{x \in \mathcal{X}_{\text{probe}}} \Big[g\big(M_{c,\phi_c}(x)\big) - g\big(M(x)\big) \Big]. \quad (3)$$

Datasets are encoded from DINO features. For each D, we sample a small subset $D^{\text{sub}} \subset D$ and average their embeddings to form μ_D. We then map μ_D to the meta-space with a learned encoder $E_d(\cdot)$. To preserve cross-dataset geometry, we linearly blend the learned embedding with a PCA projection p_D computed from the same DINO statistics, yielding the final dataset embedding u_D:

$$\mu_D = \frac{1}{|D^{\text{sub}}|} \sum_{x \in D^{\text{sub}}} \phi_{\text{DINO}}(x), \qquad u_D = (1-\gamma) E_d(\mu_D) + \gamma p_D. \quad (4)$$

To align embeddings with performance, we train the meta-space with a composite objective $\mathcal{L} = \mathcal{L}_{\text{perf}} + \mathcal{L}_{\text{rank}} + \mathcal{L}_{\text{FID}}$ and score a dataset–configuration pair by cosine similarity $s_{i,m} = \cos(u_i, v_m)$, where u_i embeds dataset D_i and v_m embeds configuration c_m. $\mathcal{L}_{\text{perf}}$ fits a predictor of the measured zoo performance $\widehat{P}_{i,m}$ from (u_i, v_m) via mean-squared error. The rank loss enforces performance-consistent ordering in the meta-space: for each dataset i, if $\widehat{P}_{i,m} > \widehat{P}_{i,n}$ then $s_{i,m} > s_{i,n}$. To preserve inter-dataset geometry, we weight dataset pairs by $w_{ij} = \exp(-\text{FID}(D_i, D_j)/\tau)$ and pull similar datasets together:

$$\mathcal{L}_{\text{rank}} = -\frac{1}{|\mathcal{P}|} \sum_{(i,m,n) \in \mathcal{P}} \log \sigma\big(\beta(s_{i,m} - s_{i,n})\big), \quad \mathcal{L}_{\text{FID}} = \frac{1}{|\mathcal{Q}|} \sum_{(i,j) \in \mathcal{Q}} w_{ij} \, \|u_i - u_j\|_2^2. \quad (5)$$

2.3 Inference via Retrieval. Given a new dataset D_{new}, we compute its embedding u_{new}. We then retrieve an adapter configuration by cosine similarity under a parameter budget. Let $\mathcal{C}_B = \{c_m : \text{Params}(c_m) \le B\}$ denote the set of budget-feasible configurations, where $\text{Params}(c_m)$ is the number of trainable adapter parameters in c_m. We select $c^\star = \arg\max_{c_m \in \mathcal{C}_B} \cos(u_{\text{new}}, v_m)$, instantiate $M_{c^\star,\phi_{c^\star}}$ using the retrieved structure and its meta-learned initialization, and fine-tune the adapter weights for T steps on D_{new}.

3 Results & Discussion

Implementation Details. Multi-rank LoRA modules are inserted into six matrices per layer (Q, K, V, MHA output, FFN1, FFN2), yielding 72 insertion sites for 12 layers, with ranks $r \in \{0, 2, 4, 8, 16, 32\}$ ($r = 0$ disables adaptation). The adapter zoo is built using multiple backbones (DINO v2-S/B and v3-S/S+/B), training one Super-Adapter per dataset–backbone pair and leveraging weight sharing across ranks to generate budget-stratified configurations. During training, N subnetworks are sampled per iteration and optimized jointly with shared weights under a curriculum schedule [3]. Full implementation, configuration files (all hyperparameters), and hardware/runtime details (GPU type, wall-clock) will be provided in the accompanying GitHub repository upon acceptance.

Evaluation Protocol. We evaluate on 17 image-classification datasets spanning fine-grained, general, medical, and scene domains. For fair comparison, Table 1 reports results within the DINOv2 ViT-B subspace, matching the one used by most PEFT baselines, while our meta-space is constructed across multiple backbone sizes and families to increase diversity and flexibility (see Tables 2 and 3 for DINOv3 results). We test two variants: MetaVFM$_{T1}$, which retrieves the nearest adapter and fine-tunes it, and MetaVFM$_{T5}$, which retrieves the top-5, briefly trains them jointly with weight sharing for one epoch (minimal overhead vs. $T1$), then continues with only the best for the remaining training. We compare against full fine-tuning and a broad range of PEFT baselines (Table 1). Importantly, none of the 17 evaluation datasets are included in meta-space training, so retrieval is performed on truly unseen datasets.

Overall performance (100 epoch fine-tuning). Across 17 datasets, MetaVFM consistently ranks among the top-performing methods, placing top-1 or top-2 on most tasks. The largest relative gains occur on cross-domain transfers, particularly scene and medical datasets, where conventional PEFT methods are more sensitive to configuration choices. These results support our central claim: retrieving both adapter configuration and meta-learned initialization improves robustness across heterogeneous domains. Notably, MetaVFM uses adapters ranging from 1.4K to 3M parameters (depending on the configuration), yet matches or surpasses baselines with substantially larger footprints (e.g., APLA at 7M), yielding a strong accuracy–parameter trade-off.

Convergence Under a Short Training Budget. To evaluate step efficiency, we fix a strict budget of 10 epochs and report accuracy at epochs 1, 5, and 10 across four representative datasets as shown in Table 2. Under this constrained setting, MetaVFM achieves higher early accuracy (79.1% average at epoch 1) and, on three of four datasets, reaches or surpasses LoRA's 10-epoch performance by epoch 5. By epoch 10, MetaVFM further widens the margin (88.5% vs. 86.6%). These results correspond to an effective 2–3× reduction in steps-to-target under the fixed short training budget.

Meta-Space Analysis. Fig. 3 shows the learned meta-space. Semantically related datasets cluster together, and higher-accuracy adapters lie closer to their corresponding dataset embeddings. This structured geometry explains the improved convergence: retrieval places each dataset near high-performing config-

Table 1: Comparison of MetaVFM to five families of fine-tuning methods across 17 datasets for 100 epochs. All approaches use ViT-B pretrained via DINOv2 self-supervision.

	Fine-grained						**General**					**Scene & Satellite**				**Medical**						
	CUB-200-2011[33]	NABirds[12]	StanfordDogs[20]	StanfordCars[21]	Aircraft[27]	Average	Caltech-256[8]	CIFAR-100[22]	Oxford-IIIT Pet[29]	DTD[4]	Average	MIT-Indoor[30]	SUN397[35]	AID[34]	Average	ISIC2019[5]	APTOS2019[16]	DDSM[24]	Coloretal[17]	Pneumonia[18]	Average	Total Average
Baselines																						
Full Fine-tuning	88.9	85.2	86.0	94.4	**87.5**	88.4	93.9	92.4	94.7	81.9	90.7	87.8	75.6	95.4	86.3	87.7	90.8	95.5	97.8	99.4	94.2	90.0
Linear Probing	89.1	86.6	87.6	88.4	76.5	85.6	94.9	88.9	95.8	81.1	90.2	88.9	76.4	91.2	85.5	55.3	90.4	89.4	94.0	97.9	85.4	87.2
MLP	89.1	86.4	87.8	88.3	77.7	85.9	94.4	89.2	96.0	80.9	90.1	88.6	76.2	91.6	85.5	71.9	90.7	93.4	95.8	97.9	89.9	88.1
Partial Adaptation	88.8	86.5	87.4	88.1	76.6	85.5	94.9	89.0	96.0	80.6	90.1	88.5	76.3	90.9	85.2	56.1	90.6	89.5	94.2	97.6	85.6	86.9
Adapters & Feature Modulation																						
Bias Tuning	89.4	87.9	87.8	92.5	83.2	88.2	95.2	93.1	95.7	82.2	91.6	89.3	77.7	95.2	87.4	79.0	90.1	96.4	97.2	99.1	92.4	90.3
Adapter	89.6	88.4	87.8	93.5	86.1	89.1	95.0	93.5	95.9	81.8	91.6	89.5	77.9	95.0	87.5	84.3	89.6	96.1	98.0	98.5	93.3	90.7
AdaptFormer[2]	89.7	88.4	88.1	93.1	85.4	89.0	95.6	93.2	95.9	82.8	91.9	89.9	78.1	95.4	87.8	85.6	90.8	97.3	97.4	98.6	93.9	91.0
SSF[25]	89.4	88.1	87.7	92.7	83.7	88.3	95.3	93.2	95.7	82.1	91.6	88.9	77.4	95.3	87.2	80.7	90.5	96.4	97.2	99.0	92.8	90.3
Consolidator[10]	89.7	87.4	87.1	93.0	83.4	88.1	94.5	92.7	95.8	81.8	91.2	89.5	77.0	94.6	87.0	81.9	91.1	96.9	96.2	99.4	93.1	90.0
Prompt-based Tuning																						
VPT-Shallow[14]	88.8	86.7	87.4	90.6	73.2	85.3	95.1	92.4	96.0	80.4	91.0	89.4	76.6	91.6	85.9	76.5	89.3	96.2	96.4	98.7	91.4	88.4
VPT-Deep[14]	89.1	87.3	87.2	91.5	81.7	87.4	95.3	92.7	95.7	80.5	91.1	90.0	77.0	94.4	87.1	79.6	91.0	96.2	97.6	98.8	92.6	89.7
E^2VPT [9]	88.3	86.6	87.4	91.2	81.0	86.9	94.5	92.7	95.4	79.6	90.6	88.7	76.3	93.7	86.2	80.9	90.6	96.2	96.8	98.6	92.6	89.2
Low-rank & Tensorized Reparameterization																						
LoRA[13]	88.7	87.5	86.3	93.4	86.4	88.5	94.3	93.0	94.0	80.4	90.4	88.5	76.4	95.4	86.8	86.5	91.1	95.1	97.4	98.9	93.8	90.1
SPT-LoRA[11]	89.2	87.9	87.5	92.8	86.3	88.7	**95.8**	92.6	95.7	82.3	91.6	89.9	77.7	95.4	87.7	82.2	89.3	96.2	97.4	99.1	92.8	90.5
FACT-TK[15]	88.8	87.8	87.5	93.0	85.4	88.5	95.3	92.8	95.5	81.5	91.3	88.9	77.4	95.4	87.2	85.1	91.5	96.3	97.2	97.9	93.6	90.3
FACT-TT[15]	88.8	87.6	87.1	92.9	84.3	88.1	94.9	92.4	95.5	81.7	91.1	89.4	77.1	94.5	87.0	81.5	90.9	97.0	97.4	98.6	93.1	90.0
RLRR[7]	88.9	87.9	87.6	92.4	83.7	88.1	95.2	93.1	95.8	82.0	91.5	89.5	77.6	95.0	87.4	81.7	90.4	96.3	96.6	98.6	92.7	90.1
Parameter Selection / Search / Composition																						
SPT-Adapter[11]	89.4	88.1	87.7	93.1	86.2	88.9	**95.8**	93.1	95.8	82.7	91.9	89.5	78.1	95.6	87.7	82.1	90.4	96.1	97.2	99.0	93.0	90.6
ARC[6]	89.4	88.2	88.1	92.6	84.1	88.5	**95.8**	92.9	96.0	82.8	91.9	89.8	78.2	95.6	87.9	82.5	90.7	97.3	97.0	98.8	93.3	90.7
GPS[38]	89.1	86.4	86.6	**94.7**	85.5	88.5	94.8	**94.0**	94.4	77.9	90.3	88.4	76.9	94.9	86.7	87.7	90.8	96.7	97.4	99.3	94.4	89.8
APLA[32]	89.6	88.0	88.5	94.0	86.7	89.4	95.7	93.4	96.1	**83.0**	**92.1**	**90.4**	78.3	96.0	88.2	**88.2**	92.1	97.2	98.4	**99.5**	95.1	91.6
Meta-Learning based (ours)																						
MetaVFM$_{T1}$	**90.0**	**88.5**	88.2	94.8	86.1	89.5	94.5	93.5	96.1	82.6	91.7	**90.4**	**79.3**	**98.3**	**89.3**	87.7	**92.3**	96.9	97.8	**99.5**	94.8	91.6
MetaVFM$_{T5}$	**90.0**	**88.5**	**89.2**	**95.3**	86.4	**89.9**	94.5	**94.0**	**96.6**	82.9	92.0	**90.4**	**79.3**	**98.3**	**89.3**	87.8	**92.3**	**97.7**	**98.7**	**99.5**	**95.2**	**91.8**

urations from similar domains, reducing the magnitude of corrective updates required during fine-tuning. The smooth local neighborhoods in the meta-space are consistent with the observed early gains and reduced steps-to-target.

Ablation Study. We ablate MetaVFM's main components: meta-learned initialization, rank loss, FID loss, and PCA-based dataset similarity, under 10 epochs using a DINOv3-B backbone (Table 3). Removing meta-initialization produces the largest early degradation (epoch-1 average drops from 76.0 to 71.2, 4.8 points), highlighting its central role in accelerating convergence through a strong starting point. Rank and FID losses provide smaller but consistent improvements, enforcing performance-aware alignment and capturing inter-dataset similarity, which primarily refine accuracy at later epochs. PCA-based similarity yields modest yet steady gains, improving the global structure of the meta-space. Together, these results demonstrate the complementary importance of each component in achieving both fast adaptation and robust final performance.

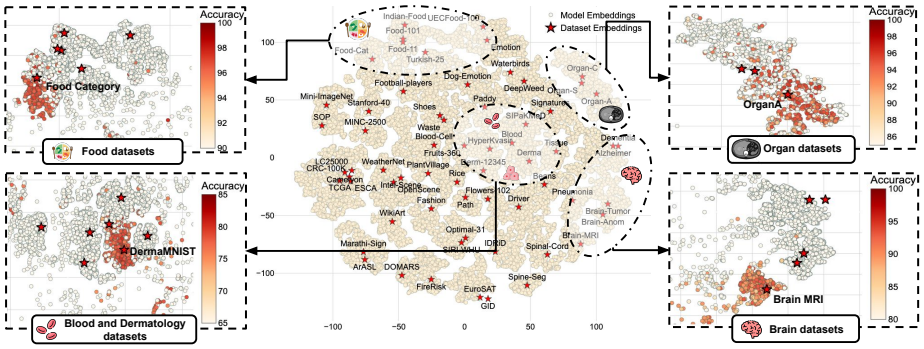


Fig. 3: T-SNE map of the meta-space. Stars denote dataset embeddings; points denote model embeddings colored by accuracy on the specified datasets (OrganA, Derma, Food Category, and Brain MRI) in the zoomed areas, illustrating clustering of similar datasets and a monotonic relationship between proximity to the dataset embedding and accuracy.

Table 2: Comparative analysis of convergence speed under 10-epoch budget based on DINOv3-B backbone. Top-1 accuracy at epochs 1, 5, and 10 on four datasets.

Method	Epoch	Accuracy (%)				
		Breast	DTD	Pneumonia	CUB-200-2011	Avg
Linear Probing	1	73.7 ± 0.3	64.1 ± 0.5	88.3 ± 1.3	73.8 ± 0.8	75.0 ± 0.7
	5	76.5 ± 0.5	79.1 ± 0.5	89.6 ± 0.4	87.8 ± 0.2	83.2 ± 0.4
	10	77.1 ± 1.0	79.5 ± 0.2	90.0 ± 0.2	88.8 ± 0.2	83.9 ± 0.4
MLP	1	73.1 ± 0.1	69.8 ± 0.4	84.4 ± 2.1	75.7 ± 0.6	75.8 ± 0.8
	5	74.6 ± 1.5	76.5 ± 0.5	86.0 ± 3.3	84.2 ± 0.5	80.3 ± 1.5
	10	77.8 ± 3.0	79.2 ± 0.5	88.9 ± 2.8	87.3 ± 0.4	83.3 ± 1.7
Full Fine-tuning	1	73.1 ± 0.3	22.1 ± 21.0	64.9 ± 9.4	1.3 ± 0.3	40.4 ± 7.8
	5	67.1 ± 7.1	60.3 ± 6.9	66.1 ± 4.1	10.1 ± 3.2	50.9 ± 5.3
	10	73.1 ± 0.0	72.1 ± 2.6	71.8 ± 13.3	20.0 ± 6.0	59.3 ± 5.5
LoRA	1	**76.5 ± 0.4**	67.7 ± 0.7	**89.9 ± 3.1**	**78.7 ± 0.9**	78.2 ± 1.3
	5	85.0 ± 0.5	80.9 ± 0.5	89.7 ± 1.8	**89.1 ± 0.3**	86.2 ± 0.8
	10	85.0 ± 0.7	81.7 ± 0.5	90.2 ± 0.7	**89.6 ± 0.5**	86.6 ± 0.6
MetaVFM (ours)	1	**76.5 ± 0.2**	**72.8 ± 0.6**	88.3 ± 3.1	**78.7 ± 0.3**	**79.1 ± 1.1**
	5	**85.5 ± 0.5**	**82.1 ± 0.2**	**91.8 ± 2.1**	88.6 ± 0.5	**87.0 ± 0.8**
	10	**88.0 ± 1.1**	**82.7 ± 0.3**	**93.5 ± 0.8**	**89.6 ± 0.1**	**88.5 ± 0.6**

Table 3: Ablation of MetaVFM components under a 10-epoch budget (DINOv3-B backbone). Top-1 accuracy (%) at epochs 1, 5, and 10 is reported on three datasets.

Method	Epoch	Accuracy (%)			
		Breast	DTD	CUB-200-2011	Avg
MetaVFM w/o Rank Loss	1	75.4 ± 0.3	71.6 ± 0.7	77.5 ± 0.4	74.8 ± 1.3
	5	84.4 ± 0.6	81.0 ± 0.4	87.8 ± 0.3	84.4 ± 1.0
	10	87.0 ± 0.9	81.9 ± 0.5	88.9 ± 0.4	85.9 ± 0.8
MetaVFM w/o FID Loss	1	75.5 ± 0.4	72.0 ± 0.6	76.7 ± 0.5	74.7 ± 1.3
	5	84.3 ± 0.6	81.6 ± 0.4	87.1 ± 0.3	84.3 ± 1.0
	10	86.8 ± 0.9	82.1 ± 0.4	88.9 ± 0.4	85.9 ± 0.8
MetaVFM w/o Meta-init	1	71.2 ± 0.5	68.4 ± 0.8	73.9 ± 0.6	71.2 ± 1.4
	5	83.2 ± 0.7	80.1 ± 0.5	87.6 ± 0.4	83.6 ± 1.0
	10	87.1 ± 1.0	82.0 ± 0.4	88.8 ± 0.4	86.0 ± 0.8
MetaVFM w/o PCA Similarity	1	75.8 ± 0.3	71.0 ± 0.7	77.8 ± 0.4	74.9 ± 1.3
	5	84.8 ± 0.6	80.8 ± 0.5	88.0 ± 0.5	84.5 ± 1.0
	10	87.6 ± 1.0	81.7 ± 0.5	89.1 ± 0.4	86.1 ± 0.8
MetaVFM (ours)	1	**76.5 ± 0.2**	**72.8 ± 0.6**	**78.7 ± 0.3**	**76.0 ± 1.2**
	5	**85.5 ± 0.5**	**82.1 ± 0.2**	**88.6 ± 0.5**	**85.4 ± 0.9**
	10	**88.0 ± 1.1**	**82.7 ± 0.3**	**89.6 ± 0.1**	**86.8 ± 0.7**

4 Conclusion

We introduce MetaVFM, a Super-Adapter–based approach that pre-builds an adapter–dataset meta-space to quickly retrieve an effective adapter setup (placement, rank, and initialization) for new datasets with no per-task search, achieving higher accuracy, faster convergence, and better cross-domain transfer than leading PEFT baselines, especially on diverse medical imaging datasets where domain shifts are common. By reducing configuration sensitivity and training steps, MetaVFM offers a practical way to adapt foundation models when efficiency and reliability matter, such as in clinical workflows and multi-center deployments. Looking ahead, we will broaden the meta-space to more modalities and backbone families, extend evaluation to dense prediction tasks (e.g., segmentation), and add hardware-aware objectives like latency and energy. We also plan to explore uncertainty-aware retrieval and continual meta-space updates to improve robustness in unseen or highly heterogeneous domains.

References

1. Alain, G., Bengio, Y.: Understanding intermediate layers using linear classifier probes (2017), https://openreview.net/forum?id=ryF7rTqgl
2. Chen, S., Ge, et al.: Adaptformer: Adapting vision transformers for scalable visual recognition. Advances in Neural Information Processing Systems **35**, 16664–16678 (2022)
3. Chu, X., Zhang, B., Xu, R.: Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. In: Proceedings of the IEEE/CVF International Conference on computer vision. pp. 12239–12248 (2021)
4. Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., Vedaldi, A.: Describing textures in the wild. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3606–3613 (2014)
5. Combalia, M., Codella, et al.: Bcn20000: Dermoscopic lesions in the wild. arXiv preprint arXiv:1908.02288 (2019)
6. Dong, W., Yan, D., Lin, Z., Wang, P.: Efficient adaptation of large vision transformer via adapter re-composing. Advances in Neural Information Processing Systems **36**, 52548–52567 (2023)
7. Dong, W., Zhang, et al.: Low-rank rescaled vision transformer fine-tuning: A residual design approach. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16101–16110 (2024)
8. Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset (2007)
9. Han, C., Wang, et al.: E^ 2vpt: An effective and efficient approach for visual prompt tuning. arXiv preprint arXiv:2307.13770 (2023)
10. Hao, T., Chen, H., Guo, Y., Ding, G.: Consolidator: Mergeable adapter with grouped connections for visual adaptation. arXiv preprint arXiv:2305.00603 (2023)
11. He, H., Cai, J., Zhang, J., Tao, D., Zhuang, B.: Sensitivity-aware visual parameter-efficient fine-tuning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 11825–11835 (2023)
12. Horn, V., et al.: Building a bird recognition app and large-scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 595–604 (2015)
13. Hu, E.J., Shen, et al.: Lora: Low-rank adaptation of large language models. ICLR **1**(2), 3 (2022)
14. Jia, M., Tang, L., Chen, B.C., Cardie, C., Belongie, S., Hariharan, B., Lim, S.N.: Visual prompt tuning. In: European conference on computer vision. pp. 709–727. Springer (2022)
15. Jie, S., Deng, Z.H.: Fact: Factor-tuning for lightweight adaptation on vision transformer. In: Proceedings of the AAAI conference. vol. 37, pp. 1060–1068 (2023)
16. Karthik, Maggie, S.D.: Aptos 2019 blindness detection (2019), https://kaggle.com/competitions/aptos2019-blindness-detection
17. Kather, J.N., Weis, et al.: Multi-class texture analysis in colorectal cancer histology. Scientific reports **6**(1), 1–11 (2016)
18. Kermany, D.S., Goldbaum, et al.: Identifying medical diagnoses and treatable diseases by image-based deep learning. cell **172**(5), 1122–1131 (2018)
19. Khan, H.U.S., Usama, M.: Ard-lora: Dynamic rank allocation for parameter-efficient fine-tuning of foundation models with heterogeneous adaptation needs. arXiv preprint arXiv:2506.18267 (2025)
20. Khosla, A., Jayadevaprakash, N., Yao, B., Li, F.F.: Novel dataset for fine-grained image categorization: Stanford dogs. In: Proc. CVPR workshop on fine-grained visual categorization (FGVC). vol. 2. Citeseer (2011)

21. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3d object representations for fine-grained categorization. In: Proceedings of the IEEE international conference on computer vision workshops. pp. 554–561 (2013)
22. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
23. Kumar, A., Raghunathan, A., Jones, R.M., Ma, T., Liang, P.: Fine-tuning can distort pretrained features and underperform out-of-distribution. In: ICLR. OpenReview.net (2022), http://dblp.uni-trier.de/db/conf/iclr/iclr2022.htmlKumarRJ0L22
24. Lee, R.S., Gimenez, F., Hoogi, A., Miyake, K.K., Gorovoy, M., Rubin, D.L.: A curated mammography data set for use in computer-aided detection and diagnosis research. Scientific data **4**(1), 1–9 (2017)
25. Lian, D., Zhou, et al.: Scaling & shifting your features: A new baseline for efficient model tuning. Advances in Neural Information Processing Systems **35**, 109–123 (2022)
26. Liang, P., Pu, et al.: Vision foundation models in medical image analysis: Advances and challenges. In: International Conference on Blockchain and Trustworthy Systems. pp. 170–181. Springer (2025)
27. Maji, S., et al.: Fine-grained visual classification of aircraft. arXiv preprint arXiv:1306.5151 (2013)
28. Oquab, M., Darcet, T., et al.: Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193 (2023)
29. Parkhi, O.M., Vedaldi, et al.: Cats and dogs. In: 2012 IEEE conference on computer vision and pattern recognition. pp. 3498–3505. IEEE (2012)
30. Quattoni, A., Torralba, A.: Recognizing indoor scenes. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 413–420. IEEE (2009)
31. Siméoni, O., Vo, et al.: Dinov3. arXiv preprint arXiv:2508.10104 (2025)
32. Sorkhei, M., Konuk, E., Smith, K., Matsoukas, C.: Apla: A simple adaptation method for vision transformers (2025), https://arxiv.org/abs/2503.11335
33. Wah, Catherine, B., et al.: The caltech-ucsd birds-200-2011 dataset. Tech. Rep. CNS-TR-2011-001, California Institute of Technology (2011)
34. Xia, G.S., Hu, et al.: Aid: A benchmark data set for performance evaluation of aerial scene classification. IEEE Transactions on Geoscience and Remote Sensing **55**(7), 3965–3981 (2017)
35. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: 2010 IEEE computer society conference on computer vision and pattern recognition. pp. 3485–3492. IEEE (2010)
36. Zhang, R., et al.: AutoLoRA: Automatically tuning matrix ranks in low-rank adaptation based on meta learning. In: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers). pp. 5048–5060. Association for Computational Linguistics, Mexico City, Mexico (Jun 2024). https://doi.org/10.18653/v1/2024.naacl-long.282, https://aclanthology.org/2024.naacl-long.282/
37. Zhang, Y., Doughty, H., Snoek, C.G.: Low-resource vision challenges for foundation models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21956–21966 (2024)
38. Zhang, Z., Zhang, Q., Gao, Z., Zhang, R., Shutova, E., Zhou, S., Zhang, S.: Gradient-based parameter selection for efficient fine-tuning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 28566–28577 (2024)