

Efficient Fine-Tuning of Foundation Models: Generalization, Disparity, and Subgroup Adaptation

Artur Parkhimchyk

**A Thesis submitted to the Faculty of Graduate Studies in Partial Fulfillment of the
Requirements for the Degree of Master of Science**

**Graduate Program in Electrical Engineering and Computer Science (EECS)
York University
Toronto, Ontario**

May 2026

© Artur Parkhimchyk, 2026

Abstract

Foundation models (FMs) perform well in vision but remain difficult to deploy when compute is limited and data are demographically imbalanced. This thesis makes two contributions. First, it studies demographic adaptation with parameter-efficient fine-tuning (PEFT) under realistic constraints where full fine-tuning is infeasible and available data are skewed. We evaluate five adaptation strategies, three pretraining paradigms, and multiple medical imaging datasets, covering 19 dataset-demographic variants, 490 configurations, and over 3,500 GPU hours. Results show that demographic adaptation can improve minority-group performance, especially with E^2VPT , and clarify when PEFT offers strong efficiency–performance tradeoffs. Second, we propose **Low-Rank Adaptation with Sinusoidal Projection Sampling (LoRA-SPS)**, a sinusoidal-projection low-rank adapter that samples from pretrained weights, decouples parameter count from embedding dimension, and improves projection rank stability while remaining mergeable for zero-overhead inference. Across vision and language tasks, LoRA-SPS matches or outperforms strong baselines using substantially fewer parameters.

Originality Statement

This thesis is founded on the author’s published and submitted research articles. The key ideas, methodologies, and experimental findings presented throughout the thesis are drawn from the following papers:

- **Published (NeurIPS Workshop on Adaptive Foundation Models (AFM) 2024)**

A. Parkhimchyk, A. Naziri, and L. Seyyed-Kalantari, “*Exploring Visual Prompt Tuning for Demographic Adaptation in Foundation Models for Medical Imaging.*”

- **Under Review (Conference on Neural Information Processing Systems (NeurIPS) 2026)**

A. Parkhimchyk, M. Tavakoli, and L. Seyyed-Kalantari, “*LoRA-SPS: Low-Rank Adaptation with Sinusoidal Projections of Sampled Pretrained Weights.*”

- **Under Review (Machine Learning for Healthcare (MLHC) 2026)**

A. Parkhimchyk, M. Tavakoli, A. Naziri, J. Gichoya, and L. Seyyed-Kalantari, “*Foundation Models Demographic Adaptation in Medical Imaging: Utility, Disparity, and Compute Trade-off.*”

The thesis integrates, extends, and unifies the contributions from these works. Material from the published papers has been adapted with permission where required. All experimental implementations, analyses, and figures appearing in the thesis were produced by the author unless otherwise stated.

Acknowledgements

This research was made possible through the support of several institutions and funding sources. I am grateful to York University for providing a supportive academic environment in which this work could be carried out. I'm especially grateful to the Vector Institute for providing access to high-performance computing resources, which were essential for conducting the large-scale experiments in this study.

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) through a Discovery Grant and by a travel grant through the Connected Minds initiative. I am also grateful for having been awarded the Ontario Graduate Scholarship (OGS), which I respectfully declined. These forms of support played an important role in enabling the computational and methodological investigations undertaken in this research.

I would like to express my deepest gratitude to my supervisor, Prof. Laleh Seyyed-Kalantari, whose guidance, encouragement, and steadfast support have been central to this work. Her mentorship shaped the direction of this research and has had a significant influence on my development as a researcher.

I am also thankful to my colleagues, collaborators, and friends, especially Amirreza Naziri and Matin Tavakoli, for their thoughtful discussions, constructive feedback, and continued encouragement throughout this process.

Finally, I am deeply grateful to my family for their unwavering support, patience, and belief in me throughout this journey.

Contents

	Page
Abstract	ii
Originality Statement	iii
Acknowledgements	iv
Contents	v
List of Tables	ix
List of Figures	xiii
1 Introduction	1
1.1 Motivation and Problem Statement	1
1.2 Research Gaps and Thesis Objectives	2
1.3 Thesis Contributions	3
1.4 Thesis Organization	3
2 Technical background	5
2.1 Foundation Models (FMs)	5
2.1.1 Pretraining Techniques	5
2.1.2 Vision Transformers (ViT)	7
2.1.3 Masked Autoencoder (MAE)	8
2.1.4 Momentum Contrast V3 (MoCoV3)	9
2.2 Parameter Efficient Fine-Tuning	10
2.2.1 Vision Prompt Adaptation	11
2.2.1.1 Visual Prompt Tuning	12
2.2.1.2 Gated Visual Prompt Tuning	13
2.2.1.3 Effective & Efficient VPT	13
2.2.2 Low-Rank Adaptation	14
2.2.3 Other Paradigms	15
2.3 Evaluation Metrics	15
2.3.1 Utility	15
2.3.2 Disparity	15
2.3.3 Language	16

2.3.4	Generative	16
2.3.5	Efficiency	18
3	Empirical Study on Demographic-Specific Adaptation	19
3.1	Introduction	19
3.2	Related Work	20
3.3	Methodology	21
3.3.1	Datasets with Sensitive Attributes	22
3.3.2	Foundation Models	25
3.3.2.1	Pretraining Details	26
3.3.3	Adaptation Methods	26
3.3.3.1	Adaptation Details	26
3.3.3.2	Hyperparameter Search	27
3.4	Experiments	28
3.4.1	Demographic Adaptation Utility	29
3.4.1.1	Evaluation Overview	29
3.4.1.2	Aggregate vs. Demographic Performance	29
3.4.1.3	Imbalanced Settings	30
3.4.1.4	Balanced Setting	30
3.4.1.5	Utility Under Data Scarcity	30
3.4.2	Subgroup Performance Disparity	31
3.4.2.1	Evaluation Overview	31
3.4.2.2	Disparities & Under-diagnosis	31
3.4.2.3	Disparity Under Data Scarcity	32
3.4.3	Utility & Compute Tradeoff	32
3.4.3.1	Evaluation Overview	32
3.4.3.2	Overall Efficiency–Performance Trends	33
3.4.3.3	Dataset-Specific Behavior	33
3.4.3.4	In-Domain vs. Out-of-Domain Adaptation	34
3.4.3.5	Limitations of Linear Adaptation	35
3.4.3.6	Overall Comparison of Adaptation Strategies	35
3.4.4	Demographic Adaptation in Natural Domain	35
3.4.4.1	Demographic Adaptation Utility	35
3.4.4.2	Subgroup Performance Disparity	36
3.4.4.3	Utility & Compute Tradeoff	37
3.5	Summary & Discussion	38
4	Low-Rank Adaptation with Sinusoidal Projection Sampling (LoRA-SPS)	40
4.1	Introduction	40
4.2	Related Work	41

4.3	Methodology	43
4.3.1	Decoupled Low-Rank Adaptation via Weight Sampling	43
4.3.2	Non-Linear Transformation & Amplitude Mixing	44
4.4	LoRA-SPS	46
4.4.1	Rank-Efficient Budget Allocation	46
4.4.2	Intuition and Design Rationale	47
4.4.3	Algorithm	49
4.5	Datasets	50
4.5.1	Vision Datasets	50
4.5.1.1	MNIST & KMNIST	50
4.5.1.2	VTAB-1K	50
4.5.1.3	FGVC	50
4.5.1.4	ADE20K	50
4.5.1.5	DreamBooth	51
4.5.2	Language Datasets	51
4.5.2.1	GLUE	51
4.5.3	LoRA-SPS Setup	52
4.5.3.1	Demonstrative Example	52
4.5.3.2	Parameter Initialization	52
4.5.3.3	Hyperparameters for VTAB-1K & FGVC Benchmarks	53
4.5.3.4	Hyperparameters for ADE20K Semantic Segmentation	54
4.5.3.5	Hyperparameters for DreamBooth Text-to-Image Generation	54
4.5.3.6	Hyperparameters for GLUE Benchmark	55
4.5.4	Baseline Setup	56
4.5.4.1	LoRA-XS Setup	56
4.5.4.2	FouRA Setup	56
4.5.4.3	FourierFT Setup	56
4.5.4.4	LoRA-Sine Setup	56
4.6	Experiments	57
4.6.1	Classification Tasks	57
4.6.1.1	Setup	57
4.6.1.2	Results	57
4.6.2	Semantic Segmentation Tasks	59
4.6.2.1	Setup	59
4.6.2.2	Results	59
4.6.3	Text-to-Image Generation Task	59
4.6.3.1	Setup	59
4.6.3.2	Results	60
4.6.4	Language Tasks	60

4.6.4.1	Setup	60
4.6.4.2	Results	60
4.6.5	Ablation Studies	62
4.6.5.1	Effect of Rank & Sinusoidal Dimension	62
4.6.5.2	Effect of Frequency Scaling	62
4.6.5.3	Guide to Hyperparameter Selection	63
4.6.5.4	Effect of Model Size	63
4.6.5.5	Effect of Equal Parameter Budget Comparison	63
4.6.5.6	Effect of LoRA-SPS Components	64
4.6.5.7	Effect of Sinusoidal Transformation	64
4.6.5.8	Effect of Sampling Strategy, Performance & Compute	64
4.6.5.9	Effect of Targeted Layers	65
4.6.5.10	Effect of LoRA-SPS Layer Coverage	66
4.6.6	Summary & Discussion	66
5	Conclusion	67
5.1	Broader Impact	67
5.2	Limitations	67
5.3	Future Directions	68
	Bibliography	69
A	Supplementary Results	80
A.1	Detailed Tabulated Results for Pooled & Demographic Adaptation	80
A.2	VTAB-1K Detailed Tabulated Results	88
A.3	FGVC Detailed Tabulated Results	92

List of Tables

3.1	Chest X-Ray datasets, including CheXpert and MIMIC-CXR, along with their demographic breakdowns and dataset variations.	24
3.2	Retina datasets, including FairVLMed10K and GF3300, along with their demographic breakdowns and dataset variations.	25
3.3	Natural datasets, including CelebA and CelebA-1K, along with their demographic breakdowns and dataset variations.	26
3.4	Parameter Search for Pooled & Demographic adaptation.	27
4.1	Feature comparison. <i>Mergeable</i> : folds into frozen weights. <i>Decoupled emb.</i> m,n : parameter count is independent of embedding dimension. <i>Non-linear</i> : enriches the effective spectrum at fixed parameter cost. <i>Pretrained basis</i> : derived from frozen pretrained weights W . <i>Trainable A</i> : mixes sinusoidal features via a learnable amplitude matrix. Bottom row highlights the unique combination achieved by LoRA-SPS.	42
4.2	Core properties/mechanisms of LoRA-SPS.	43
4.3	Hyperparameters for demonstrative example. Configuration of the 3-layer ViT, LoRA & LoRA-SPS fine-tuning setup used in the two-phase demonstrative example.	52
4.4	Comparison of different initialization methods on VTAB-Caltech101. Each of s_1, s_2, s_3, s_4 is scaled to 1. We report Top-1 accuracy, with stability measured by how often the achieved accuracy stays above 90.0%. The chosen method is highlighted in bold	52
4.5	Summary of the hyperparameters used in LoRA-SPS search for VTAB-1K & FGVC. . .	54
4.6	Summary of the hyperparameters used in LoRA-SPS search for the DreamBooth dataset. . .	55
4.7	Summary of the hyperparameters used in LoRA-SPS for GLUE benchmark.	55
4.8	Image classification on FGVC and VTAB-1K. ViT-B/16 [1] pretrained on supervised ImageNet-21k; mean test top-1 accuracy over three runs. <i>% Params</i> is the percentage of trainable parameters relative to the backbone, the lower the better. Bold and <u>underline</u> mark best and second-best per column across PEFT methods. Per-task breakdowns in tables A.9 to A.12. . .	58
4.9	Semantic segmentation on ADE20K. SETR [2] with a ViT-L backbone; mIoU reported under single-scale (s.s.) and multi-scale (m.s.) evaluation. Fully fine-tuned ResNet-101 is shown for reference. Bold marks the best.	59

4.10	Text-to-image generation , DreamBooth, Stable Diffusion v1.5 [3]. DINO, CLIP-I, CLIP-T, and LPIPS Div. (higher is better); FID (lower is better). Bold : Best and <u>underline</u> : second-best.	60
4.11	RoBERTa-Large [4] performance . Fine-tuning spans various PEFT methods and is evaluated across six GLUE [5] tasks. Results for CoLA are reported using Matthew’s correlation (MCC), and STS-B using Pearson’s correlation (PCC). Each result reported is a median of five runs. Bold and <u>underline</u> indicate best and second-best performance across PEFT methods for the respective dataset.	61
4.12	ViT model-scale comparison on VTAB. Top-1 accuracy reported for Natural, Specialized, and Structured subsets. Tuned / Total (%) indicates the percentage of trainable parameters relative to the backbone. Results are shown for ViT-Small/16, ViT-Base/16, ViT-Large/16, and ViT-Huge/14. Bold indicates best performance across PEFT methods for each model size and VTAB subset.	63
4.13	Equivalent-budget transition from LoRA to LoRA-SPS components . Top-1 VTAB-1K accuracy with 0.165M trainable params. We compare LoRA, Sine-LoRA ($f = 400$), and LoRA-SPS ($f = 1024$): LoRA/Sine-LoRA use $r = 1$, while LoRA-SPS variants use matching parameter budgets. We only tune $\alpha \in \{0.01, 0.1, 1.0\}$ per row; normalization omitted from the method for clarity.	64
4.14	Effect of sinusoidal transformation function perturbations on LoRA-SPS performance (top-1 accuracy) on VTAB-1K using a ViT-B backbone. Rank (r), sinusoidal dimension (d), and frequency (f) are 32, 48, and 1024, respectively.	64
4.15	Effect of sampling strategy at moderate ($r = 32$) rank . We evaluate LoRA-SPS top-1 accuracy on VTAB-1K with rank $r = 32$, sinusoidal dimension $d = 48$, and frequency $f = 1024$ fixed.	65
4.16	Effect of sampling strategies & compute at large parameter count . Training overhead and top-1 accuracy on VTAB-1K for LoRA ($r = 32$) and LoRA-SPS ($r = d = 136$, $f = 1024$, $\alpha = 0.1$), average training time (in minutes) over 100 epochs, measured in wall-clock time across all specialized tasks.	65
4.17	Impact of different LoRA-SPS target layer configurations on VTAB-1K. Evaluated top-1 accuracy on Natural, Specialized, and Structured tasks, plus the overall mean. Tuned / Total (%) indicates the percentage of trainable parameters. We fix rank $r = 32$, sinusoidal dimension $d = 48$, frequency $f = 1024$, and scaling factor $\alpha = 0.1$. We only vary the target.	65
4.18	Impact of progressively expanding LoRA-SPS layers . LoRA-SPS modules are inserted at layers $\{1, 2, 4, 6, 12\}$ and all intermediate layers inherit the most recent LoRA-SPS weights. Evaluated on VTAB-1K top-1 accuracy across Natural, Specialized, and Structured tasks, as well as the overall mean accuracy. Tuned / Total (%) indicates the percentage of trainable parameters. We fix rank $r = 32$, sinusoidal dimension $d = 48$, frequency $f = 1024$, and scaling factor $\alpha = 0.1$. Only the number of enabled layers in LoRA-SPS is varied.	66

A.1	Performance of models pretrained on CXR and adapted to the CXR dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.	80
A.2	Performance of models pretrained on IN and adapted to the CXR dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.	81
A.3	Performance of models pretrained on CXR and adapted to the CXR-1K dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.	82
A.4	Performance of models pretrained on IN and adapted to the CXR-1K dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.	83
A.5	Performance of models pretrained on IN and adapted to the CelebA dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.	84
A.6	Performance of models pretrained on IN and adapted to the CelebA-1K dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.	85
A.7	Performance of models pretrained on IN and adapted to the FairVLMed10k dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.	86
A.8	Performance of models pretrained on IN and adapted to the GF3300 dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.	87
A.9	ViT-B/16 top-1 accuracy on VTAB-1K Natural (avg. over 3 runs). Tuned / Total (%) indicates the percentage of tunable parameters. Rank (r), sinusoidal dimension (d), frequency scale (f), and scaling factor (α) are reported per dataset. Bold and <u>underline</u> indicate best and second-best performance.	89
A.10	ViT-B/16 top-1 accuracy on VTAB-1K Specialized (avg. over 3 runs). Tuned / Total (%) indicates the percentage of tunable parameters. Rank (r), sinusoidal dimension (d), frequency scale (f), and scaling factor (α) are reported per dataset. Bold and <u>underline</u> indicate best and second-best performance.	90
A.11	ViT-B/16 top-1 accuracy on VTAB-1K Structured (avg. over 3 runs). Tuned / Total (%) indicates the percentage of tunable parameters. Rank (r), sinusoidal dimension (d), frequency scale (f), and scaling factor (α) are reported per dataset. Bold and <u>underline</u> indicate best and second-best performance.	91

A.12 ViT-B/16 top-1 accuracy on FGVC (avg. over 3 runs). Tuned / Total (%) indicates the percentage of tunable parameters. Rank (r), sinusoidal dimension (d), frequency scale (f), and scaling factor (α) are reported per dataset. **Bold** and underline indicate best and second-best performance. 92

List of Figures

2.1	Overview of the Vision Transformer (ViT) architecture , illustrating the sequence of steps: (1) dividing the image into patches, (2) applying a linear projection to each patch, (3) adding positional embeddings, (4) pre-pending the class token, and (5) feeding the resulting sequence into the Transformer encoder. Figure is from the original ViT paper [1].	7
2.2	Overview of the Masked Autoencoder (MAE) architecture . Figure is from the MAE paper [6].	8
2.3	Overview of the Momentum Contrast V3 (MoCoV3) architecture . Figure is from the MoCoV3 paper [7].	9
2.4	Visualization of standard adaptation methods . Left: illustration of patch embeddings. Center: linear probing (Linear), where only the classification head is trained while the backbone remains frozen. Right: full fine-tuning (Full), where all model weights are updated. Frozen parameters are indicated in blue with a snowflake icon, and trainable parameters are shown in orange with a flame icon.	10
2.5	Visualization of VPT . Learnable head and prompt tokens (orange) are prepended to the input sequence while the backbone (blue) remains frozen, allowing task-specific adaptation with minimal trainable parameters.	11
2.6	GVPT & E²VPT Overview . GVPT (left) uses gated prompts for layer-wise control, while E ² VPT (right) reduces trainable parameters via reparameterization and embedding optimizations. Frozen parameters are indicated in blue with a snowflake icon, and trainable parameters are shown in orange with a flame icon.	12
2.7	LoRA overview . LoRA represents the weight update as a low-rank factorization, using D to project the input into a compact subspace and U to project it back to the original dimensionality.	14
3.1	Overview of our experimental framework . We evaluate multiple foundation models (FMs), pretraining strategies, and adaptation methods under both pooled and demographic-specific training regimes, measuring utility, disparity and compute across diverse datasets and data regimes.	22

3.2	Utility on CXR, FairVLMed10K & GF3300 , comparing performance of FMs and adaptation methods across demographics (Aggregate, Asian, Black, White). In each cell, the top value is from a model trained on pooled data; bottom value is from a model trained only on the selected demographic. Colors Δ indicate <i>Demographic – Pooled</i> : green ($\Delta > 1.0$), red ($\Delta < -1.0$), and yellow ($ \Delta \leq 1.0$).	28
3.3	Utility on CXR-1K , comparing performance of FMs and adaptation methods across demographics (Aggregate, Asian, Black, White). In each cell, the top value is from a model trained on pooled data; bottom value is from a model trained only on the selected demographic. Colors Δ indicate <i>Demographic – Pooled</i> : green ($\Delta > 1.0$), red ($\Delta < -1.0$), and yellow ($ \Delta \leq 1.0$).	29
3.4	Subgroup disparity on CXR, GF3300 and FairVLMed10K , defined as the Δ AUC difference between the lowest and highest demographic group values (Asian, Black, White). Similarly, FPR Gap is reported only for No Finding label. For both Δ AUC and FPR Gap, smaller is better. In each cell, the top value is from a model trained on pooled data; bottom value is from a model trained only on the selected demographic. Colors Δ indicate <i>Demographic – Pooled</i> : green ($\Delta < -1.0$), red ($\Delta > 1.0$), and yellow ($ \Delta \leq 1.0$).	31
3.5	Subgroup disparity on CXR-1K , defined as the Δ AUC difference between the lowest and highest demographic group values (Asian, Black, White). Similarly, FPR Gap is reported only for No Finding label. For both Δ AUC and FPR Gap, smaller is better. In each cell, the top value is from a model trained on pooled data; bottom value is from a model trained only on the selected demographic. Colors Δ indicate <i>Demographic – Pooled</i> : green ($\Delta < -1.0$), red ($\Delta > 1.0$), and yellow ($ \Delta \leq 1.0$).	32
3.6	Utility-compute tradeoff on CXR across adaptation methods and pretraining paradigms. The y-axis shows per-demographic performance difference (adapter AUC minus pooled adaptation). Both performance deltas and training time savings are reported relative to each adapter method. Each subplot also reports full-data training time range for reference. We define a tradeoff boundary allowing up to a 1% performance drop for up to a 50% compute reduction. Points within this tolerance are colored green , and others are colored red	33
3.7	Utility-compute tradeoff on FairVLMed10k & GF3300 across adaptation methods and pretraining paradigms. The y-axis shows per-demographic performance difference (adapter AUC minus pooled adaptation). Both performance deltas and training time savings are reported relative to each adapter method. Each subplot also reports full-data training time range for reference. We define a tradeoff boundary allowing up to a 1% performance drop for up to a 50% compute reduction. Points within this tolerance are colored green , and others are colored red	34
3.8	Utility on CelebA , comparing performance of FMs and adaptation methods across demographics (Aggregate, Male, Female). In each cell, the top value is from a model trained on pooled data; bottom value is from a model trained only on the selected demographic. Colors Δ indicate <i>Demographic – Pooled</i> : green ($\Delta > 1.0$), red ($\Delta < -1.0$), and yellow ($ \Delta \leq 1.0$).	35

3.9	Utility on CelebA-1K , comparing performance of FMs and adaptation methods across demographics (Aggregate, Male, Female). In each cell, the top value is from a model trained on pooled data; bottom value is from a model trained only on the selected demographic. Colors Δ indicate <i>Demographic – Pooled</i> : green ($\Delta > 1.0$), red ($\Delta < -1.0$), and yellow ($ \Delta \leq 1.0$).	36
3.10	Subgroup disparity on CelebA , defined as the Δ AUC difference between the lowest and highest demographic group values (Male, Female). For Δ AUC, smaller is better. In each cell, the top value is from a model trained on pooled data; bottom value is from a model trained only on the selected demographic. Colors Δ indicate <i>Demographic – Pooled</i> : green ($\Delta < -1.0$), red ($\Delta > 1.0$), and yellow ($ \Delta \leq 1.0$).	36
3.11	Subgroup disparity on CelebA-1K , defined as the Δ AUC difference between the lowest and highest demographic group values (Male, Female). For Δ AUC, smaller is better. In each cell, the top value is from a model trained on pooled data; bottom value is from a model trained only on the selected demographic. Colors Δ indicate <i>Demographic – Pooled</i> : green ($\Delta < -1.0$), red ($\Delta > 1.0$), and yellow ($ \Delta \leq 1.0$).	37
3.12	Utility-compute tradeoff on CelebA across adaptation methods and pretraining paradigms. The y-axis shows per-demographic performance difference (adapter AUC minus pooled adaptation). Both performance deltas and training time savings are reported relative to each adapter method. Each subplot also reports full-data training time range for reference. We define a tradeoff boundary allowing up to a 1% performance drop for up to a 50% compute reduction. Points within this tolerance are colored green , and others are colored red	37
4.1	Overview of LoRA-SPS . Parameter–accuracy comparison on FGVC and VTAB-1K using reported results from the evaluated state-of-the-art PEFT methods. LoRA-SPS (gold) provides a strong trade-off between accuracy and trainable-parameter count.	41
4.2	Core LoRA-SPS properties visualized (a) Feasible rank under a fixed parameter budget. (b) Trainable parameters vs. rank. (c) Equal-parameter (256) reconstruction with/without trainable amplitudes. (d) Singular value spectra of full-, low-, and sine-activated low-rank matrices.	43
4.3	Comparing LoRA (left) and LoRA-SPS (right) . LoRA factorizes the update into down- and up-projections D , U . LoRA-SPS instead derives both from small subsets of pretrained weights (section 4.3.1): each subset is processed by a compact trainable F_D , transformed via a sinusoidal projection, and reweighted by an amplitude matrix A . The same procedure yields U (eq. (4.8)); Remaining updates follow LoRA. Orange : trainable parameters; blue : frozen pretrained weights.	46
4.4	LoRA-SPS expressiveness on MNIST. Weight updates compared to LoRA and pretrained weights, varying frequency (top) and parameter count (bottom). Unlike LoRA’s limited-structure updates, LoRA-SPS reuses pretrained-weight patterns; expressiveness grows with both frequency and parameter count. Details: section 4.5.3.1.	47

4.5	Signal reconstruction. We reconstruct a rank-1 LoRA weight update (dim. 48) on MNIST [8] using 50% of parameters. LoRA-SPS achieves a combined MSE+cosine loss of 0.27 ± 0.08 vs. 0.91 ± 0.00 for FourierFT, indicating higher effective rank per parameter. Details are provided in section 4.5.3.1.	47
4.6	Comparison of different scaling factors for Xavier Normal initialization on VTAB-1K–Caltech101. Each of s_1, s_2, s_3, s_4 is scaled by 0.01, 0.1, 1.0, or 10.0. The left heatmap shows results when s_1, s_2 are varied while s_3, s_4 are fixed at a scale of 1, whereas the right heatmap shows results when s_3, s_4 are varied with s_1, s_2 fixed at a scale of 1. Colors indicate top-1 accuracy. The selected configuration is $s_1 = 1.0, s_2 = 1.0, s_3 = 0.1, s_4 = 10.0$	53
4.7	Qualitative DreamBooth comparison. LoRA-SPS matches or exceeds the visual quality of LoRA, Sine-LoRA, and FouRA while preserving prompt alignment, at up to $25 \times$ fewer parameters.	61
4.8	Dimensionality search. Top-1 accuracy (color intensity) on VTAB-1K and total % of parameters tuned (shown in parentheses) for varying LoRA-SPS rank r and sinusoidal dimension d	62
4.9	Frequency search. Top-1 accuracy performance of LoRA-SPS on VTAB-1K for varied frequency, while rank r and sinusoidal dimension d are fixed to 32 and 48, respectively. . . .	62

Chapter 1

Introduction

1.1 Motivation and Problem Statement

Foundation models (FMs) have become a dominant paradigm in modern machine learning due to their ability to learn transferable representations from large-scale pretraining [9, 10, 11, 12]. In computer vision, transformer-based models such as Vision Transformers (ViT) [1], Masked Autoencoders (MAE) [6], and contrastive models such as MoCoV3 [7] have achieved strong performance across both natural and medical imaging tasks [1, 13, 14, 15, 16]. Their success has motivated rapid adoption in medical imaging, where FMs are increasingly used across different modalities, datasets, and clinical applications [14, 17, 18].

Despite this progress, adapting FMs to downstream clinical tasks remains difficult. The standard adaptation pipeline, consisting of large-scale pretraining followed by full fine-tuning, requires substantial computational resources, memory, and storage. This makes FM adaptation inaccessible for many healthcare institutions and research groups, especially when separate models are needed for multiple tasks, domains, or patient subgroups. Moreover, full fine-tuning often assumes access to large, diverse, and representative labeled datasets, an assumption that rarely holds in medical imaging, where expert annotations are expensive and data collection is often uneven across populations.

A second major challenge is demographic imbalance. Medical imaging datasets are frequently dominated by a limited number of demographic groups [19, 20, 21]. Prior work has shown that medical imaging models can encode demographic attributes such as race or sex directly from images [22, 23], and that these learned representations can contribute to systematic performance disparities across patient groups [24, 25, 26]. As a result, models trained or adapted on pooled datasets may achieve strong aggregate performance while still underperforming for minority or underrepresented groups.

Parameter-efficient fine-tuning (PEFT) offers a promising alternative to full fine-tuning by adapting pre-trained models through a small number of trainable parameters [27, 28]. PEFT methods can reduce computational and storage costs while preserving the pretrained backbone, making them attractive for resource-constrained deployment and demographic-specific adaptation. However, important questions remain: when

does subgroup-specific adaptation improve performance over pooled adaptation, how do PEFT methods behave under demographic imbalance and data scarcity, and how can low-rank adaptation methods be made more efficient without sacrificing downstream performance?

This thesis addresses these challenges from two complementary perspectives. First, it studies demographic-specific adaptation of medical foundation models under realistic data and compute constraints, evaluating the tradeoff between utility, subgroup disparity, and efficiency. Second, it proposes a new PEFT method, Low-Rank Adaptation with Sinusoidal Projection Sampling (LoRA-SPS), designed to improve the efficiency–performance tradeoff of low-rank adaptation while preserving mergeable, zero-overhead inference.

1.2 Research Gaps and Thesis Objectives

Despite substantial progress in parameter-efficient fine-tuning (PEFT), several challenges remain unresolved [29, 30, 31, 32, 33]. PEFT methods must reduce trainable parameters while preserving adaptation quality, but many approaches achieve efficiency by restricting the trainable update space. This can limit their ability to adapt to tasks that require larger representational changes. Other methods improve adaptation quality at the cost of additional memory, training overhead, or inference latency, reducing the practical benefits of PEFT.

These tradeoffs appear across major PEFT families. Prompt-based methods such as VPT and VFPT are efficient, but may have limited flexibility for tasks requiring substantial feature adaptation [31, 34]. Low-rank methods such as LoRA and LoRA-XS constrain updates to low-rank subspaces, which can create rank-capacity tradeoffs under tight parameter budgets [32, 35, 33]. Sensitivity-aware methods such as SPT and CDRA-SPT selectively update task-relevant parameters, improving efficiency but potentially limiting adaptation when broader changes are needed [36, 37].

In medical imaging, these limitations are especially important because PEFT methods may be used to specialize models for demographic subgroups, such as race or gender, using limited data. However, it remains unclear when demographic-specific adapters improve over pooled adaptation, how close they come to models trained under ideal conditions, and how their behavior depends on pretraining paradigm, domain shift, and subgroup sample size.

This thesis addresses the following research gaps:

- **Limited understanding of demographic-specific adaptation.** Existing studies often focus on pooled adaptation or assume idealized data availability. Less is known about subgroup-specific adaptation when fine-tuning data are scarce, imbalanced, or available for only some patient groups.
- **Unclear interaction between adaptation strategy and pretraining paradigm.** It remains unclear how supervised, masked-image-modeling, and contrastive pretraining support demographic adaptation, particularly under in-domain and out-of-domain transfer.
- **Insufficient analysis of utility, disparity, and compute tradeoffs.** Prior work has not systematically compared PEFT methods under matched experimental conditions while jointly considering subgroup utility, performance disparity, data scarcity, and training cost.

- **Structural limitations of low-rank adaptation.** Standard LoRA parameterizes updates as a linear low-rank product, which constrains the stable rank under a fixed parameter budget [33, 38]. In addition, the feasible rank of low-rank adapters is tied to the embedding dimension of the adapted layer [35, 39]. These constraints create an accuracy–parameter tradeoff, where reducing trainable parameters can directly reduce adaptation quality.

Based on these gaps, this thesis has two main objectives. The first is to systematically evaluate demographic-specific adaptation for medical foundation models under realistic data and compute constraints. The second is to improve the efficiency–performance tradeoff of low-rank adaptation through LoRA-SPS, a mergeable PEFT method that constructs compact adapter updates from sampled pretrained weights and sinusoidal projections.

1.3 Thesis Contributions

This thesis makes the following contributions:

- **Demographic-specific adaptation in medical imaging.** We provide a systematic evaluation of pooled and demographic-specific adaptation for medical foundation models across multiple datasets, demographic attributes, pretraining paradigms, and PEFT methods.
- **Joint analysis of utility, disparity, and compute.** We evaluate adaptation strategies using aggregate performance, subgroup performance gaps, false-positive-rate gaps, and training cost, providing practical guidance on when demographic-specific adaptation is beneficial.
- **LoRA-SPS for efficient low-rank adaptation.** We propose Low-Rank Adaptation with Sinusoidal Projection Sampling (LoRA-SPS), a PEFT method that constructs compact adapter updates from sampled pretrained weights and sinusoidal projections while remaining mergeable for zero-overhead inference.
- **Broad validation across tasks and modalities.** We evaluate LoRA-SPS on image classification, semantic segmentation, text-to-image generation, and language understanding, showing that it can match or improve upon strong PEFT baselines with fewer trainable parameters.
- **A unified view of efficient FM adaptation.** Together, the two contributions study efficient adaptation from both a deployment perspective, through subgroup-aware medical adaptation, and a methodological perspective, through improved low-rank adapter design.

1.4 Thesis Organization

The remainder of this thesis is organized as follows.

Chapter 2 provides the technical background required for the rest of the thesis. It first introduces the pre-training paradigms and foundation models used in this work, including supervised ViT pretraining, Masked

Autoencoders (MAE), and Momentum Contrast V3 (MoCoV3). It then reviews parameter-efficient fine-tuning methods, including full fine-tuning, linear probing, visual prompt tuning variants, and low-rank adaptation. The chapter concludes by defining the evaluation metrics used throughout the thesis, including utility, disparity, generative, language, and efficiency metrics.

Chapter 3 presents an empirical study of demographic-specific adaptation for foundation models in medical imaging. It compares pooled and demographic-specific adaptation across chest X-ray, retinal imaging, and natural-domain datasets. The chapter evaluates how different pretraining paradigms and PEFT methods affect subgroup utility, performance disparity, and compute efficiency under realistic data and resource constraints.

Chapter 4 introduces Low-Rank Adaptation with Sinusoidal Projection Sampling (LoRA-SPS), a parameter-efficient fine-tuning method designed to improve the efficiency–performance tradeoff of low-rank adaptation. The chapter presents the method, its design motivation, and its relationship to existing LoRA-style approaches. It then evaluates LoRA-SPS across image classification, semantic segmentation, text-to-image generation, and language understanding tasks, followed by ablation studies on its main components.

Chapter 5 concludes the thesis by summarizing the main findings from the demographic adaptation study and the LoRA-SPS method. It also discusses broader implications, limitations, and future directions for efficient and subgroup-aware foundation model adaptation.

Chapter 2

Technical background

2.1 Foundation Models (FMs)

Foundation models (FMs) are large neural networks pretrained on extensive datasets that often use self-supervised learning to learn general-purpose representations transferable to a wide range of downstream tasks [9, 10, 11, 12]. In computer vision, transformer-based FMs such as Vision Transformers (ViT) [1], Masked Autoencoders (MAE) [6], and contrastive models (e.g., MoCoV3) [7] have achieved strong performance across natural and medical imaging applications [1, 13, 14, 15, 16]. Their success has driven rapid adoption in medical imaging, where FMs now underpin state-of-the-art systems across a variety of modalities and domains [14, 17, 18].

2.1.1 Pretraining Techniques

Pretraining is the central mechanism through which foundation models acquire transferable representations. Broadly, pretraining strategies for vision models fall into two main categories: supervised learning and self-supervised learning. Both aim to learn generalizable feature representations, but they differ in their reliance on labeled data and training objectives.

Supervised Learning. In supervised pretraining, models are trained on large-scale labeled datasets using explicit target annotations. In computer vision, this paradigm has traditionally relied on datasets such as ImageNet [40], where models are optimized to minimize a classification loss (e.g., cross-entropy) over predefined categories. ViTs, for example, initially demonstrated strong performance when pretrained in a fully supervised manner on large image classification corpora.

Supervised pretraining can produce highly discriminative representations, particularly when labels are abundant and diverse. However, in medical imaging, acquiring large-scale, high-quality annotations is costly and often requires expert clinicians. Moreover, supervised objectives constrain the representation to the predefined label space, which may limit transferability to tasks that differ substantially from the original supervision.

Self-Supervised Learning. Self-supervised learning (SSL) removes the need for manual annotations by constructing surrogate (pretext) tasks directly from the data. Instead of predicting human-provided labels, the model learns to predict withheld or transformed parts of the input. SSL has become the dominant paradigm for modern vision foundation models due to its scalability and label efficiency [41, 42].

A number of self-supervised methods have emerged in vision, including:

- *Contrastive learning*, where the model is trained to bring representations of different views of the same image closer together while pushing apart representations of different images. Methods such as SimCLR [10] and MoCoV3 [7] exemplify this approach.
- *Masked image modeling*, inspired by masked language modeling in NLP (e.g., BERT) [43], where portions of the input image are masked and the model learns to reconstruct the missing content. Masked Autoencoders (MAE) [6] exemplify this approach in vision.
- *Other self-supervised methods*, which learn transferable representations through objectives that do not fit neatly into contrastive learning or masked image reconstruction. Methods such as DINO and BYOL [44, 9, 11] fall under this broader category and have shown strong performance in both natural and medical imaging.

Self-supervised pretraining is particularly attractive in medical imaging, where vast quantities of unlabeled data are available but labeled datasets remain limited. By leveraging large-scale unlabeled corpora, SSL enables models to learn modality-specific structure and semantic priors before being adapted to downstream clinical tasks.

Comparison and Practical Considerations. Supervised and self-supervised pretraining are not mutually exclusive; in practice, many foundation models combine both strategies, for example by first performing large-scale self-supervised pretraining followed by supervised refinement. Empirically, self-supervised approaches often yield more robust and transferable representations under domain shift, while supervised pretraining can provide strong task-specific discriminative signals when high-quality labels are available.

In the context of medical foundation models, the choice of pretraining strategy has important implications for fairness, generalization, and accessibility. Self-supervised learning reduces dependence on curated annotations and may improve inclusivity across underrepresented groups, yet it does not inherently eliminate biases present in the underlying data distribution. Consequently, careful dataset design, evaluation across demographic subgroups, and computational considerations remain critical regardless of the chosen pretraining paradigm.

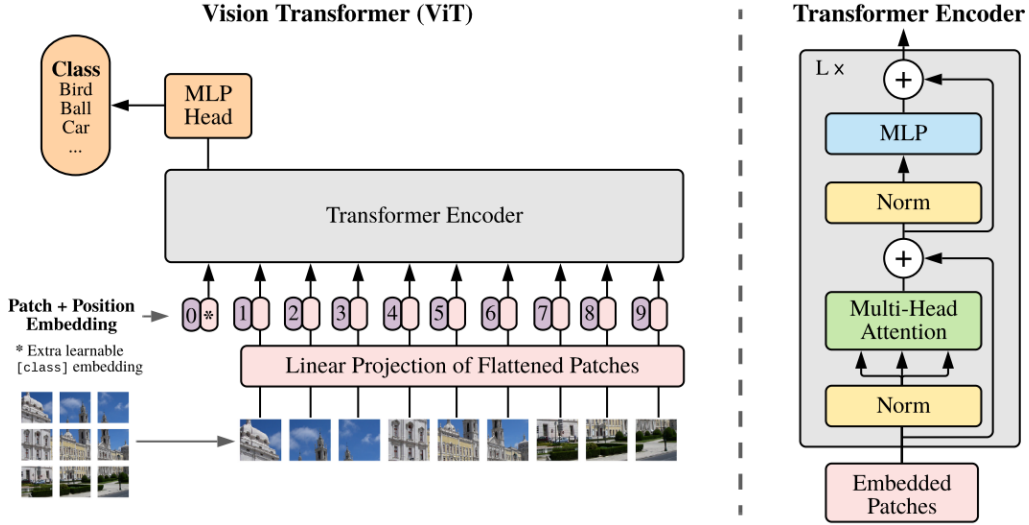


Figure 2.1: **Overview of the Vision Transformer (ViT) architecture**, illustrating the sequence of steps: (1) dividing the image into patches, (2) applying a linear projection to each patch, (3) adding positional embeddings, (4) pre-pending the class token, and (5) feeding the resulting sequence into the Transformer encoder. Figure is from the original ViT paper [1].

2.1.2 Vision Transformers (ViT)

Vision Transformers (ViT) [1] adapted the Transformer architecture from NLP [45] to images and are a major driver of the recent growth in vision foundation models. By treating an image as a sequence of learned tokens rather than relying on convolutional priors [46], ViTs offer a simple, scalable, and modality-agnostic backbone that benefits strongly from large-scale self-supervised or supervised pretraining; this ability to scale with data and compute made them a natural fit for the foundation-model paradigm and for transfer across many downstream tasks.

Architecturally, the ViT was first presented in [1] and further discussed in connection with VPT in [31]. Prior to ViT blocks, each image I of size $H \times W \times C$ is separated into smaller patches I_j of size $h \times w \times C$, the ratio of which results in $p = HW/hw$ number of patches.

$$I_j \in \mathbb{R}^{h \times w \times C} \quad j = 1, \dots, p \quad (2.1)$$

The positional encoder of the ViT block generates embeddings $e_{0,j}$ with latent dimension n . Input to the initial ViT block $\mathbf{X}_0$ is a concatenation of the class token $\mathbf{T}_0$, pre-pended to the embedding. The returned value $\mathbf{X}_1$ of the ViT block is then fed into the next block. This process continues throughout the blocks.

$$\mathbf{X}_l = [\mathbf{T}_l, \mathbf{E}_l] = [\mathbf{T}_l, e_{l,0}, \dots, e_{l,p}] \quad \mathbf{T}_l, e_{l,j} \in \mathbb{R}^n \quad (2.2)$$

$$\mathbf{X}_l = \text{Block}^l(\mathbf{X}_{l-1}) \quad l = 1, \dots, L \quad (2.3)$$

The key component inside the ViT block is the multi-head self-attention layer. The attention utilizes query,

key, and value matrices computed with learnable parameters $\mathbf{W}_Q$, $\mathbf{W}_K$, $\mathbf{W}_V$ respectively, described in eq. (2.4). The attention score is determined with softmax normalization of n scaled dot-product as described in eq. (2.5).

$$\mathbf{Q} = \mathbf{X}_{l-1} \mathbf{W}_Q \quad \mathbf{K} = \mathbf{X}_{l-1} \mathbf{W}_K \quad \mathbf{V} = \mathbf{X}_{l-1} \mathbf{W}_V \quad (2.4)$$

$$Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Softmax\left(\frac{\mathbf{QK}^\top}{\sqrt{n}}\right) \mathbf{V} \quad (2.5)$$

The $Block^l(\cdot)$ summarized in eq. (2.6), computes in order, layer normalization, attention, residual connection, followed by another layer normalization, feed forward network, and final residual connection.

$$\begin{aligned} \tilde{\mathbf{X}}_{l-1} &= LayerNorm(\mathbf{X}_{l-1}) \\ \mathbf{X}'_l &= \mathbf{X}_{l-1} + Attention(\tilde{\mathbf{X}}_{l-1}) \\ \tilde{\mathbf{X}}'_l &= LayerNorm(\mathbf{X}'_l) \\ \mathbf{X}_l &= \mathbf{X}'_l + FFN(\tilde{\mathbf{X}}'_l) \end{aligned} \quad (2.6)$$

After completing the iterations across ViT blocks, the final class token $\mathbf{T}_L$ is fed into the head, a feed-forward layer for a classification prediction $y = head(\mathbf{T}_L)$.

Figure 2.4 demonstrates two standard strategies of adapting the ViT model. LP would train only the head, a final feed-forward layer. FT trains the entire model.

2.1.3 Masked Autoencoder (MAE)

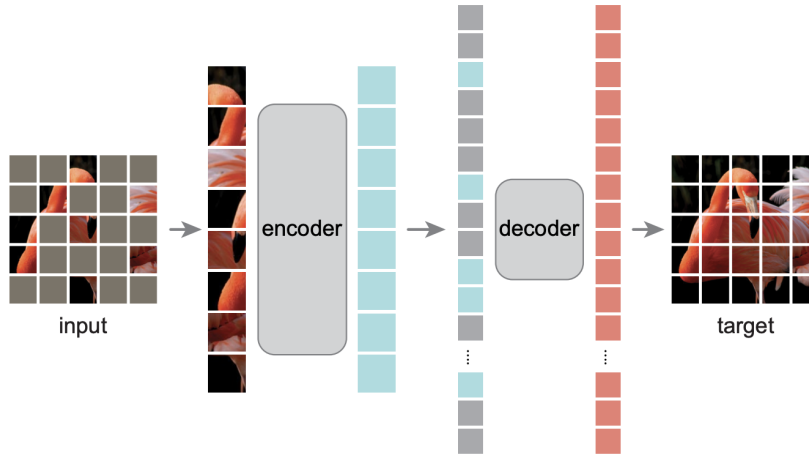


Figure 2.2: **Overview of the Masked Autoencoder (MAE) architecture.** Figure is from the MAE paper [6].

The Masked Autoencoder (MAE) [6] visualized in fig. 2.2 operates by masking a subset of image patches and training the model to reconstruct the missing content. Continuing from the ViT definition, given an input

image I , partitioned into m patches. A random subset $\mathcal{M} \subset \{1, \dots, p\}$ of patches is masked, and the remaining visible patches $\{I_j \mid j \notin \mathcal{M}\}$ are embedded and passed through the encoder to produce latent representations:

$$\mathbf{E}_{vis} = \text{Encoder}(I_j \mid j \notin \mathcal{M}) \quad (2.7)$$

A lightweight decoder then reconstructs the original pixel values of the masked patches using both the encoded visible embeddings and positional encodings p_j :

$$\hat{I}_j = \text{Decoder}(\mathbf{E}_{vis}, p_j), \quad j \in \mathcal{M} \quad (2.8)$$

The reconstruction loss is typically the mean squared error between reconstructed and original masked patches:

$$\mathcal{L}_{MAE} = \frac{1}{|\mathcal{M}|} \sum_{j \in \mathcal{M}} \|I_j - \hat{I}_j\|^2 \quad (2.9)$$

By pretraining the encoder in this masked reconstruction task, MAE learns general-purpose representations that can be adapted for downstream tasks.

2.1.4 Momentum Contrast V3 (MoCoV3)

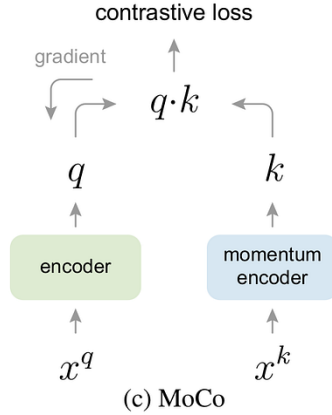


Figure 2.3: **Overview of the Momentum Contrast V3 (MoCoV3) architecture.** Figure is from the MoCoV3 paper [7].

MoCoV3 [7] visualized in fig. 2.3 is a self-supervised contrastive learning method for visual representation learning. Given an input image I , two augmented views I_q (query) and I_k (key) are generated using stochastic augmentations. The query view is encoded by an online encoder f_q , while the key view is encoded by a momentum-updated encoder f_k :

$$z_q = p_q(f_q(I_q)), \quad z_k = f_k(I_k), \quad (2.10)$$

where p_q denotes the prediction head applied only to the query branch. The query encoder f_q consists of a backbone, a projection head, and a prediction head, whereas the key encoder f_k consists of a backbone and

a projection head only. The parameters of f_k are updated by exponential moving average of f_q .

Unlike earlier MoCo variants, MoCoV3 does not rely on a memory queue; instead, negative samples are taken from the other examples in the current minibatch. Using a temperature hyperparameter τ_c , the contrastive objective can be written as

$$\ell(q_i, k_i) = -\log \frac{\exp(\text{sim}(z_{q_i}, z_{k_i})/\tau_c)}{\sum_{j=1}^N \exp(\text{sim}(z_{q_i}, z_{k_j})/\tau_c)}, \quad (2.11)$$

where $\text{sim}(\cdot, \cdot)$ denotes cosine similarity and N is the batch size. In practice, the loss is symmetrized over both views:

$$\mathcal{L}_{\text{MoCoV3}} = \frac{1}{2} (\ell(q_1, k_2) + \ell(q_2, k_1)). \quad (2.12)$$

Through this process, MoCoV3 learns embeddings that are invariant to augmentations and suitable for downstream transfer.

2.2 Parameter Efficient Fine-Tuning

Prior to newer PEFT paradigms, the most common approaches to adapting a foundation model were full fine-tuning (Full) and linear probing (Linear) visualized in fig. 2.4. In FT, all model weights are updated during training. In LP, only the final linear head is trained, while the backbone remains frozen. Backbone refers to the foundation model, a main body of the network that contains most of the layers and learns general-purpose feature representations. FT offers maximum flexibility and typically achieves the best task-specific performance, since all parameters can adapt to the new objective; however, it is computationally expensive, memory-intensive, and requires storing a separate full model for each task. In contrast, LP is lightweight and efficient, as it trains only a small task-specific head while keeping the pretrained backbone fixed, reducing compute and storage costs. However, its performance is often limited, especially when the target task differs substantially from the pretraining data, because the frozen backbone cannot adapt its representations.

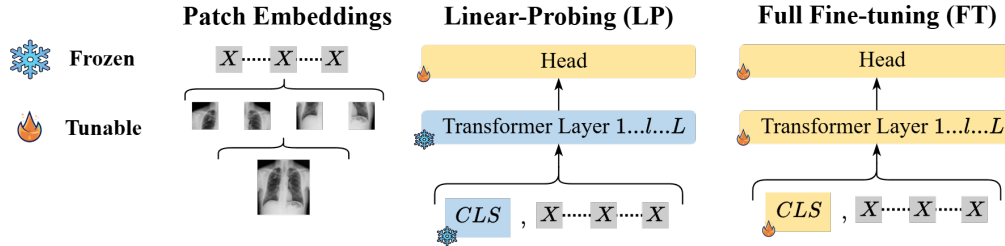


Figure 2.4: **Visualization of standard adaptation methods.** Left: illustration of patch embeddings. Center: linear probing (Linear), where only the classification head is trained while the backbone remains frozen. Right: full fine-tuning (Full), where all model weights are updated. Frozen parameters are indicated in blue with a snowflake icon, and trainable parameters are shown in orange with a flame icon.

PEFT methods offer a promising alternative to full fine-tuning under such constraints. PEFT methods adapt pretrained FMs by updating only a small subset of parameters, significantly reducing computational and

storage overhead [27, 28]. They have been successfully extended to a range of visual tasks, including image classification [31, 37], semantic segmentation [47, 48], and object detection [49]. Recent progress in PEFT has led to several influential methodological families.

2.2.1 Vision Prompt Adaptation

Vision Prompt Tuning (VPT) [31] adapts pretrained vision transformers by prepending learnable prompt tokens to the input sequence, while keeping the backbone largely frozen. These prompt tokens act as task-specific modifiers, guiding the frozen backbone to produce features suitable for the downstream task, enabling parameter-efficient adaptation without modifying the majority of model weights.

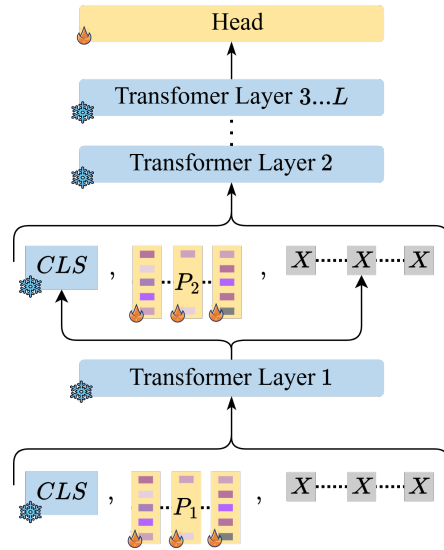


Figure 2.5: **Visualization of VPT.** Learnable head and prompt tokens (orange) are prepended to the input sequence while the backbone (blue) remains frozen, allowing task-specific adaptation with minimal trainable parameters.

Gated VPT (GVPT) [50] extends VPT by introducing a learnable gating mechanism that modulates the contribution of each prompt token at different layers of the transformer. This allows the model to dynamically control how much task-specific information is injected at each layer, improving flexibility and performance on tasks with high variability or limited data.

E²VPT [51] (Effective and Efficient VPT) further builds on these ideas by incorporating reparameterization and embedding-level optimizations. E²VPT enables even fewer trainable parameters while maintaining or improving downstream performance, making it particularly suitable for large-scale FMs and resource-constrained environments.

More recently, VFPT [34] (Vision Fourier Prompt Tuning) introduces prompts in the frequency domain by applying learnable Fourier-based perturbations to the input image embeddings. This approach allows the model to capture global, high-frequency and low-frequency patterns efficiently, enabling more effective adaptation with minimal trainable parameters while retaining the benefits of prompt-based PEFT.

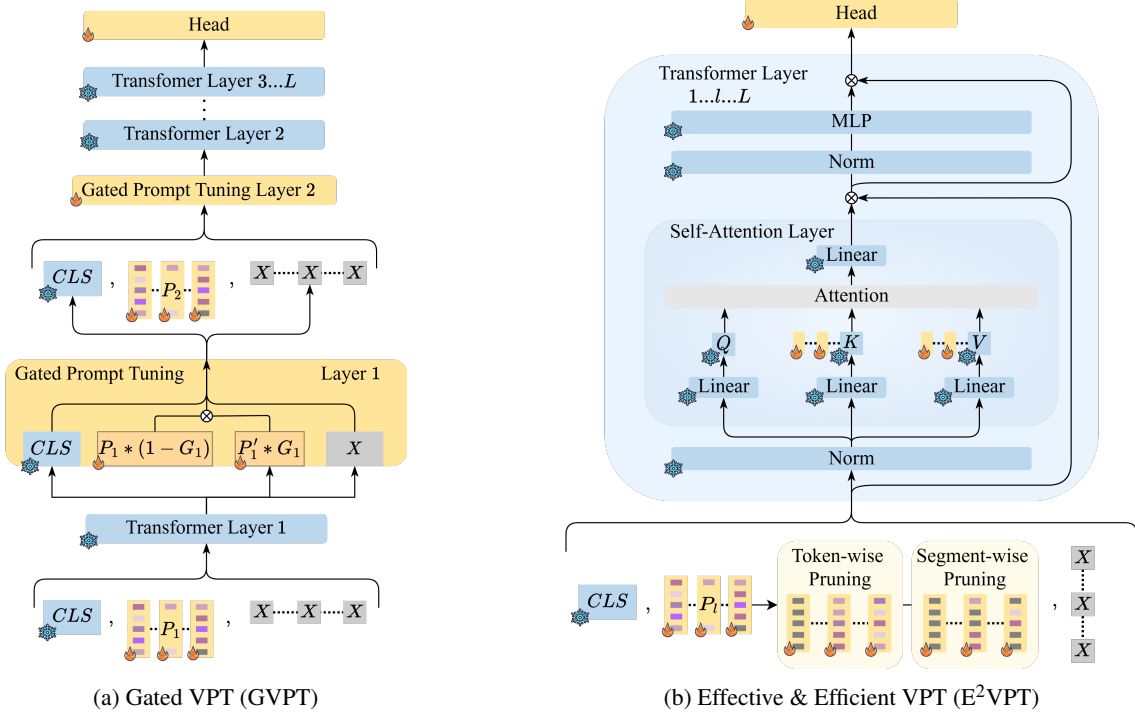


Figure 2.6: **GVPT & E²VPT Overview.** GVPT (left) uses gated prompts for layer-wise control, while E²VPT (right) reduces trainable parameters via reparameterization and embedding optimizations. Frozen parameters are indicated in blue with a snowflake icon, and trainable parameters are shown in orange with a flame icon.

Each FM is adapted to tasks using Full, Linear, and three chosen VPT variants: VPT, E²VPT, and GVPT. FT updates all parameters, whereas Linear only updates linear layers. VPT trains a small set of prompts prepended to the input image while freezing the rest of the model. GVPT introduces a gating mechanism to dynamically control prompt influence across transformer blocks, enabling more flexible and context-aware adaptation. Finally, E²VPT builds upon VPT by introducing separate trainable key and value prompts that are injected into self-attention layers, combined with prompt pruning for efficiency.

2.2.1.1 Visual Prompt Tuning

VPT [31] has two variants, Shallow-VPT and Deep-VPT. Deep-VPT introduces learnable parameters called prompts, denoted by $\mathbf{P}_l$, into each ViT block. The new prompt is packed between a classification token and the embeddings prior to each ViT block. The number of prompts p_n is a hyperparameter, and the size of each prompt is n , the same size as the class token and the embeddings.

$$\mathbf{X}_l = [\mathbf{T}_l, \mathbf{P}_l, \mathbf{E}_l] = [\mathbf{T}_l, p_{l,0}, \dots, p_{l,n}, e_{l,0}, \dots, e_{l,m}] \quad (2.13)$$

$$\mathbf{X}_l = [\mathbf{T}_l, _, \mathbf{E}_l] = \text{Block}^l(\mathbf{X}_{l-1}) \quad (2.14)$$

Shallow-VPT, in contrast, introduces prompt $\mathbf{P}_0$ at the first block, and then simply follows the ViT architecture.

$$\begin{aligned}\mathbf{X}_1 &= [\mathbf{T}_1, \mathbf{P}_1, \mathbf{E}_1] = \text{Block}^0([\mathbf{T}_0, \mathbf{P}_0, \mathbf{E}_0]) \\ \mathbf{X}_l &= [\mathbf{T}_l, \mathbf{P}_l, \mathbf{E}_l] = \text{Block}^l(\mathbf{X}_{l-1})\end{aligned}\tag{2.15}$$

Training VPT requires updating $\mathbf{P}_l$ as trainable weights alongside the final classification head, visualized in fig. 2.5.

2.2.1.2 Gated Visual Prompt Tuning

Building upon VPT, GVPT introduces learnable gate values g_l with gate priors s_l between blocks. Unlike standard VPT, the prompts are updated dynamically at each layer.

$$g_l = \text{sigmoid}(s_l) \quad l = 1, \dots, L-1\tag{2.16}$$

At each block l , the input prompt $\mathbf{P}_l$ is computed as the weighted sum of the current layer’s learnable prompt $\mathbf{P}'_l$ and the prompt propagated from the previous block ($l-1$). This weighted sum is determined by the gate value:

$$\begin{aligned}\mathbf{X}_1 &= [\mathbf{T}_1, \mathbf{P}_0, \mathbf{E}_1] = \text{Block}^0([\mathbf{T}_0, \mathbf{P}_0, \mathbf{E}_0]) \\ \mathbf{P}_l &= g_l \cdot \mathbf{P}'_{l-1} + (1 - g_l) \cdot \mathbf{P}_{l-1} \\ \mathbf{X}_l &= [\mathbf{T}_l, \mathbf{P}_l, \mathbf{E}_l] = \text{Block}^l([\mathbf{T}_l, \mathbf{P}_l, \mathbf{E}_l])\end{aligned}\tag{2.17}$$

Recall Shallow VPT, and observe the differences here. During training, the first prompt P_0 is updated jointly with the gate priors and a learnable attention temperature scalar τ (see eq. (2.18)). This scalar τ controls the sharpness of the attention distribution: smaller values of τ produce more peaked attention. See section 2.2.1 for the visualization. For further details, please refer to the original work [50].

$$\text{GatedAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\tau}\right)\mathbf{V}\tag{2.18}$$

2.2.1.3 Effective & Efficient VPT

E²VPT section 2.2.1 extends VPT by introducing additional trainable prompts, $\mathbf{P}_K$ and $\mathbf{P}_V$, which are appended to the key and value matrices, $\mathbf{K}$ and $\mathbf{V}$, respectively. These prompts introduce additional parameters to the key and value matrices in self-attention, which aid in capturing new attention patterns within the data.

$$\mathbf{K}' = [\mathbf{P}_K, \mathbf{K}] \quad \mathbf{V}' = [\mathbf{P}_V, \mathbf{V}]\tag{2.19}$$

$$\text{Attention}(\mathbf{Q}, \mathbf{K}', \mathbf{V}') = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}'^\top}{\sqrt{n}}\right)\mathbf{V}'\tag{2.20}$$

$$S_{P_k} = \mathbb{E}_{x \sim \mathcal{D}_x} \left| \frac{\partial \mathcal{L}(x)}{\partial \rho_k} \right|\tag{2.21}$$

Second, efficiency is obtained by eliminating less significant prompts. This is done by performing token-wise pruning, followed by segment-wise pruning, each with their respective binary masks with values $\{0, 1\}$. Each

pruning position is determined with an importance score computed on the expectation of the loss sensitivity to its respective masks over the training data distribution. See the original work for more information [51].

2.2.2 Low-Rank Adaptation

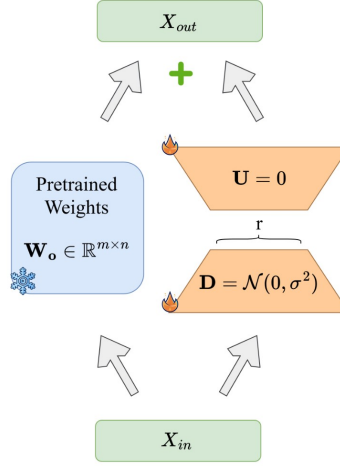


Figure 2.7: **LoRA overview.** LoRA represents the weight update as a low-rank factorization, using $\mathbf{D}$ to project the input into a compact subspace and $\mathbf{U}$ to project it back to the original dimensionality.

Low-Rank Adaptation (LoRA) [32] is a PEFT technique that adapts pretrained foundation models by injecting low-rank update matrices into selected weight layers, allowing task-specific adjustments while keeping the original backbone frozen. Standard extensions of LoRA [35, 52] explore variations in rank selection, initialization, and update distribution to improve efficiency and performance across downstream tasks. Spectral variants [53, 39, 54] perform adaptation in the frequency domain, decomposing weight updates along spectral bases to capture long-range or structured correlations in the model parameters. Non-linear variants [33, 38, 55] incorporate non-linear transformations or sinusoidal basis expansions into the low-rank updates, enabling the model to capture more complex relationships than standard linear projections. Finally, sensitivity-aware methods [36, 37] focus updates on the most task-relevant parameters or subspaces, often guided by gradient magnitudes or learned importance scores, concentrating adaptation capacity where it is most effective. Collectively, these LoRA extensions provide a flexible, computationally efficient framework for adapting large-scale vision and multimodal foundation models while maintaining the integrity of the pretrained backbone.

Low-Rank Adapter (LoRA) [32] (fig. 4.3) is defined with a pretrained weight matrix $\mathbf{W}_o \in \mathbb{R}^{m \times n}$, where n and m are the input and output embedding dimensions, respectively. The adapted weight takes the mergeable form $\mathbf{W} = \mathbf{W}_o + \alpha \Delta \mathbf{W}$, where the low-rank update matrix $\Delta \mathbf{W} = \mathbf{U}\mathbf{D}$ is factored into up- and down-projection matrices $\mathbf{U} \in \mathbb{R}^{m \times r}$ and $\mathbf{D} \in \mathbb{R}^{r \times n}$. Given input $\mathbf{x}_{in}$, the output $\mathbf{x}_{out}$ is:

$$\mathbf{x}_{out} = \mathbf{W}_o \mathbf{x}_{in} + \alpha \mathbf{U} \mathbf{D} \mathbf{x}_{in}. \quad (2.22)$$

Here $r \ll \min(m, n)$ is the adapter rank and α is a scaling factor [32]. This factorization reduces the trainable-

parameter count from $O(mn)$ to $O(r(m+n))$ [33]. However, the rank of the update matrix is bounded by $\text{rank}(\mathbf{UD}) \leq \min\{\text{rank}(\mathbf{U}), \text{rank}(\mathbf{D})\} \leq r$, so tight parameter budgets force small r , potentially limiting the adapter’s ability to capture task-specific structure.

2.2.3 Other Paradigms

Beyond VPT and LoRA methods, several other PEFT approaches have been proposed. Adapters [56] insert small bottleneck layers within each transformer block, enabling task-specific transformations without modifying the backbone. Side network adapters [57] add lightweight auxiliary networks in parallel to the main backbone, combining their outputs to enhance feature representations for downstream tasks. BitFit [58] restricts updates to only the bias terms of the model, providing an extremely parameter-efficient yet effective adaptation strategy. Similarly, bias-tuning [59] focuses exclusively on adjusting bias vectors in selected layers. While these methods are generally less popular than VPT or LoRA, they offer lightweight alternatives for task-specific adaptation of frozen backbones, allowing tradeoffs between flexibility, computational cost, and performance. A detailed comparison and exploration of these approaches is beyond the scope of this work.

2.3 Evaluation Metrics

These metrics evaluate the performance of the model according to specific criteria.

2.3.1 Utility

Accuracy (ACC). Accuracy is the proportion of the correct predictions out of the total number of predictions.

$$\text{ACC} = \frac{1}{n} \sum_{i=1}^n \mathbf{1}\{\hat{y}_i = y_i\} \quad (2.23)$$

where $\hat{y}_i$ is the predicted label and y_i is the true label for instance i .

Area Under Curve (AUC). AUC quantifies the probability that a randomly chosen positive instance is ranked higher than a randomly chosen negative instance:

$$\text{AUC} = \mathbb{P}(\hat{Y}_+ > \hat{Y}_-) \quad (2.24)$$

where $\hat{Y}_+$ and $\hat{Y}_-$ are the predicted scores for positive and negative samples, respectively.

2.3.2 Disparity

ΔAUC . For the following definition, we assume the set of subgroups $\mathcal{A} = \{a_1, a_2, \dots, a_k\}$. ΔAUC measures the maximum gap between AUC values across subgroups.

$$\Delta AUC = \max_{a_i, a_j \in \mathcal{A}} |AUC_a - AUC_{a'}| \quad (2.25)$$

True Positive Rate (TPR) & False Positive Rate (FPR). These are, respectively, the proportion of actual positives that are correctly identified by the model, and actual negatives that are incorrectly classified as positives.

$$\begin{aligned} TPR &= \mathbb{P}(\hat{Y} = 1 | Y = 1) = \frac{TP}{TP + FN} \\ FPR &= \mathbb{P}(\hat{Y} = 1 | Y = 0) = \frac{FP}{FP + TN} \end{aligned} \quad (2.26)$$

FPR Gap: FPR Gap measures disparity in false positive rates across subgroups.

$$\text{FPR Gap} = \max_{a \in \mathcal{A}} \text{FPR}_a - \min_{a \in \mathcal{A}} \text{FPR}_a \quad (2.27)$$

2.3.3 Language

Matthew's Correlation Coefficient (MCC). MCC is a balanced evaluation metric for binary classification that takes into account true and false positives and negatives. It is particularly useful for imbalanced datasets.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (2.28)$$

where TP , TN , FP , and FN denote the numbers of true positives, true negatives, false positives, and false negatives, respectively.

Pearson Correlation Coefficient (PCC). PCC measures the linear correlation between predicted values and ground-truth values.

$$PCC = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}} \quad (2.29)$$

where x_i and y_i denote the predicted and ground-truth values, while $\bar{x}$ and $\bar{y}$ are their corresponding means.

2.3.4 Generative

DINO Similarity (DINO) [60]. DINO similarity evaluates semantic consistency between generated and reference images using self-supervised visual embeddings extracted from a DINO encoder.

$$\text{DINO} = \frac{f_{\text{DINO}}(x_g) \cdot f_{\text{DINO}}(x_r)}{\|f_{\text{DINO}}(x_g)\| \|f_{\text{DINO}}(x_r)\|} \times 100 \quad (2.30)$$

where x_g and x_r denote the generated and reference images, respectively, and $f_{\text{DINO}}(\cdot)$ represents the DINO feature extractor.

CLIP Image Similarity (CLIP-I) [61]. CLIP-I measures visual similarity between generated and reference images using CLIP image embeddings.

$$\text{CLIP-I} = \frac{f_{\text{img}}(x_g) \cdot f_{\text{img}}(x_r)}{\|f_{\text{img}}(x_g)\| \|f_{\text{img}}(x_r)\|} \times 100 \quad (2.31)$$

where $f_{\text{img}}(\cdot)$ denotes the CLIP image encoder.

CLIP Text Similarity (CLIP-T) [61]. CLIP-T evaluates alignment between a generated image and its conditioning text prompt using CLIP embeddings.

$$\text{CLIP-T} = \frac{f_{\text{img}}(x_g) \cdot f_{\text{text}}(t)}{\|f_{\text{img}}(x_g)\| \|f_{\text{text}}(t)\|} \times 100 \quad (2.32)$$

where t is the text prompt, and $f_{\text{text}}(\cdot)$ is the CLIP text encoder.

Learned Perceptual Image Patch Similarity Diversity (LPIPS Div.) [62]. LPIPS Diversity measures perceptual diversity among generated samples using the Learned Perceptual Image Patch Similarity (LPIPS) metric.

$$\text{LPIPS Div.} = \frac{1}{N(N-1)} \sum_{i \neq j} d_{\text{LPIPS}}(x_i, x_j) \quad (2.33)$$

where $d_{\text{LPIPS}}(\cdot, \cdot)$ denotes the LPIPS perceptual distance between generated samples x_i and x_j .

Fréchet Inception Distance (FID) [63]. FID evaluates the distributional similarity between generated and real images based on feature statistics extracted from an Inception network.

$$\text{FID} = \|\mu_r - \mu_g\|_2^2 + \text{Tr} \left(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2} \right) \quad (2.34)$$

where (μ_r, Σ_r) and (μ_g, Σ_g) are the means and covariance matrices of feature embeddings for real and generated images, respectively.

Mean Intersection over Union (mIoU-s.s / mIoU-m.s) [64]. mIoU evaluates semantic segmentation quality by measuring the overlap between predicted and ground-truth segmentation masks. The single-scale variant (mIoU-s.s) performs inference at a single input resolution, while the multi-scale variant (mIoU-m.s) aggregates predictions across multiple image scales.

$$\text{mIoU} = \frac{1}{C} \sum_{c=1}^C \frac{TP_c}{TP_c + FP_c + FN_c} \quad (2.35)$$

where C is the number of classes, and TP_c , FP_c , and FN_c denote the true positives, false positives, and false negatives for class c .

2.3.5 Efficiency

We measure the computational cost of adaptation using wall-clock training time and GPU usage.

Parameter Efficiency. Parameter efficiency is the proportion of trainable parameters relative to the total number of model parameters.

$$\%Params = \frac{\theta_{\text{trainable}}}{\theta_{\text{total}}} \times 100 \quad (2.36)$$

where $\theta_{\text{trainable}}$ and θ_{total} denote the number of trainable and total parameters, respectively.

Utility–Efficiency Trade-off. We evaluate the tradeoff between performance (e.g., AUC) and training cost by comparing relative performance changes against compute savings. A method is considered efficient if it achieves substantial compute reduction with minimal performance degradation.

Chapter 3

Empirical Study on Demographic-Specific Adaptation

3.1 Introduction

Foundation Models (FMs) have demonstrated promising results in medical image analysis [65, 17, 15, 16]. However, training FMs is infeasible for institutions with limited computing resources. Additionally, training FMs requires access to large amounts of data, while real-world large-scale medical datasets such as MIMIC-CXR [20] and CheXpert [19] are severely suffering from demographic imbalances, for instance, White patients are overrepresented, Black are minorities, and Asians are rarely represented. Thus, *utility* and *disparity* of FMs trained on such imbalanced data may vary significantly across demographics.

Recognizing that a single model rarely generalizes equally across all populations, *demographic adaptation* offers a promising but underexplored approach to improving utility and reducing disparities for minority and rare demographics. Prior work has largely focused either on improving average performance or benchmarking fairness in isolation [13, 16]. However, the trade-offs between *utility*, *disparity*, and *compute*, under realistic constraints, such as dataset size, pretraining strategies, and the level of representativeness of the demographic in the FM pretraining data, have not been explored. In short, it is underexplored: *How demographic adaptation of FMs benefits decision-making in medical imaging with respect to utility, disparity, and compute, for majority, minority and rare demographics in FMs pretraining phase.*

Full fine-tuning (Full), as a common method of model adaptation, can help adapt FMs to demographics. However, it is not practical under compute constraints. Moreover, it increases the risk of catastrophic forgetting, where FMs deviate from the optimal learned parameters and lose their generalization. On the other hand, linear probing (Linear) is a common adaptation approach that trains only a single linear classifier on top of a pretrained model. However, due to its limited capacity to capture complex patterns, it often suffers from reduced performance. This limitation motivates the use of parameter-efficient fine-tuning (PEFT) methods, which can preserve much of the performance of Full while reducing catastrophic forgetting and maintaining computational efficiency. Among PEFT methods, Visual Prompt Tuning (VPT) and its variants

[31] introduce a small set of learnable prompts (typically $< 1\%$ of model parameters), reducing compute and storage costs. Compared to other PEFT methods such as LoRA or adapter layers, VPT operates directly in the input/token space, preserves pretrained representations, and avoids modifying internal network weights, properties that are especially beneficial where overfitting and catastrophic forgetting are major concerns. Extensions such as GVPT and E²VPT further increase the ability to learn complex patterns (expressivity) while maintaining efficiency.

In this work, we compare the conventional approach of adapting a pretrained FM on the entirety of the dataset, including all demographics, which we refer to as *Pooled Adaptation*, with an alternative adaptation strategy that adapts a pretrained FM to a specific demographic subgroup, referred to as *Demographic Adaptation*. We systematically study the impact of these strategies on utility, disparity, and compute across different models and adaptation methods, under realistic scenarios with majority, minority, and rare demographic representations. We place particular emphasis on VPT and its variants, given their suitability for low-resource compute and original model weight preservation capacity. We evaluate five adaptation strategies (Full, Linear, VPT, E²VPT, and GVPT) across three pretraining paradigms: supervised Vision Transformers (ViT) [1], Masked Autoencoders (MAE) [6, 13], and MoCoV3 [7]. To explicitly capture domain shift, we consider both in-domain (medical) and out-of-domain (ImageNet) pretraining. Our experiments span 19 splits across datasets and demographic variants in chest X-ray, retinal, and natural images, including both large-scale and data-scarce regimes, totaling over 490 experimental configurations.

3.2 Related Work

Foundation Models in Medical Imaging. FMs learn general-purpose representations that can be adapted to downstream tasks with minimal supervision. In medical imaging, self-supervised learning (SSL) has further improved representation quality, particularly through methods such as Masked Autoencoders (MAE) [6, 13] and contrastive approaches such as MoCoV3 [7]. Despite these advances, adapting FMs to downstream medical tasks typically requires full fine-tuning (FT), which is computationally expensive and often impractical in clinical environments. This limitation has motivated the development of parameter-efficient alternatives.

Parameter-Efficient Fine-Tuning (PEFT). Adaptation with PEFT updates a small fraction of parameters, reducing computational cost and memory footprint [27, 28]. Existing PEFTs are broadly categorized into (i) *weight-space methods*, such as adapters and low-rank updates (e.g., LoRA), which modify internal layers of the network, and (ii) *input-space methods*, such as Visual Prompt Tuning (VPT), which introduce learnable tokens without altering backbone weights. VPT [31] prepends trainable prompts to the input sequence, enabling efficient adaptation while preserving pretrained representations. Other variants of VPT, such as E²VPT [51], increase expressiveness, the ability to learn complex patterns, by introducing key-value prompts within attention layers, while GVPT [50] dynamically controls prompt influence across transformer blocks. Particularly, VPT families are beneficial in medical imaging as they preserve pretrained weights and reduce

catastrophic forgetting. However, their behavior under medical demographic imbalance constraints remains underexplored.

Fairness and Bias in Medical Imaging Models. Models trained on imbalanced medical datasets can exhibit systematic performance disparities across demographic groups [24, 25, 26, 66]. Recent benchmarks such as FairMedFM [16], MedFair [67], and FM-Bench [68] provide standardized evaluations of demographic bias in medical imaging FMs, primarily focusing on measurement rather than bias mitigation. Large-scale multi-modal FMs such as BiomedCLIP [69], trained on heterogeneous biomedical corpora, may not align well with realistic limitations of FMs’ pretraining data. FMs pretrained on large and noisy datasets perform poorly in utility and fairness compared to specialized FMs trained on smaller but high-quality in-domain data, with performance disparities persisting across demographic groups [66]. This highlights the importance of domain alignment and adapting FMs to diverse demographics, while motivating studying FMs adaptation under realistically imbalanced pretraining conditions, as a way to compensate disparities, an area that remains underexplored.

Demographic Adaptation. A key design choice in deploying FMs is whether to train a single generalist model on pooled data or use multiple specialist adaptation tailored to specific demographics. Full fine-tuning on pooled, large diverse datasets typically yields the best overall performance, but is computationally expensive. It remains unclear whether PEFT-based specialization can serve as a viable alternative to pooled adaptation in demographic-sensitive applications. Demographic adaptation aims to improve model performance for specific subgroups by training on subgroup-specific data. Prior work [70] on Alzheimer’s Disease and Related Dementia shows that fine-tuning on underrepresented populations can improve within-group performance, and in some cases, reduce disparities. This suggests that targeted specialization may offer a better compute-to-utility trade-off than training large generalist models. This has not been explored on FMs for medical imaging. Moreover, most existing studies focus on Full adaptation, which is computationally expensive and puts the model at the risk of catastrophic forgetting.

Positioning and Gap. To our knowledge, no prior study has addressed this gap: a systematic comparison between pooled training and demographic adaptation for FMs under realistic data imbalance and compute constraints, particularly with respect to their effects on overall utility and subgroup disparities across majority, minority, and rare populations. In particular, it is still unclear how much utility and subgroup disparity are affected when replacing pooled training with demographic adaptation under realistic data imbalance constraints for training FMs. This gap matters because such analysis can provide practical guidance for choosing between Full and lightweight adaptation strategies.

3.3 Methodology

Here, we describe our training and evaluation pipeline. The pipeline integrates three publicly available medical datasets (Chest X-Ray, FairVLMed10k, GF3300), and we also include a non-medical image dataset,

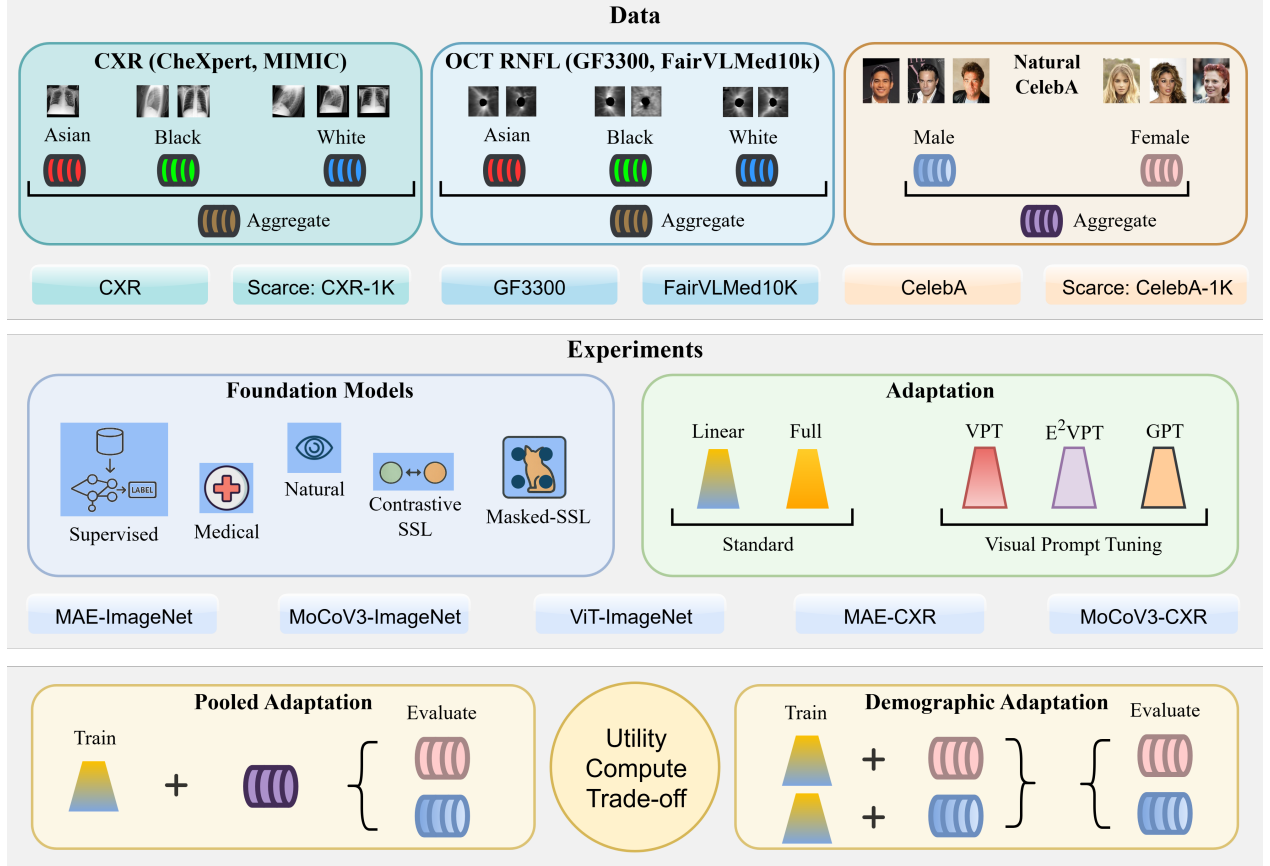


Figure 3.1: **Overview of our experimental framework.** We evaluate multiple foundation models (FMs), pretraining strategies, and adaptation methods under both pooled and demographic-specific training regimes, measuring utility, disparity and compute across diverse datasets and data regimes.

CelebA, in our appendix for completeness. We use three pretraining strategies (MAE, MoCoV3, and supervised ViT) from diverse sources, and five adaptation methods (linear probing, fine-tuning, and three variants of visual prompt tuning). Altogether, we evaluate 490 unique model-dataset-subset-method configurations. We explore fairness and utility jointly under both full-data and low-data regimes. The overall pipeline is illustrated in fig. 3.1.

3.3.1 Datasets with Sensitive Attributes

For demographic analysis in medical imaging, we focus on datasets that (a) are sufficiently large to support FMs pretraining, (b) include demographic attributes such as race and gender, and (c) are publicly available to ensure reproducibility. These conditions lead us to the MIMIC-CXR [20] and CheXpert [19], Chest X-Rays (CXR) datasets. CheXpert and MIMIC-CXR together provide $\sim 600K$ images. These datasets are widely used in fairness research [71, 17, 15, 13, 16, 20, 25, 24, 72, 67] due to their inclusion of race information [24]. Importantly, they are highly imbalanced across demographic groups, where White patients are the dominant group, Black patients are a minority, and Asian patients are relatively rare. We use this imbalance to study how demographic adaptation behaves under these three scenarios. Inspired by VTAB [73], we construct

a CXR-1K split to study data scarcity. In addition, we include an imbalanced FairVLMed10k [21] and balanced GF3300 [74] (i.e., all demographics are represented equally) retina datasets. We adopt stratified train/validation/test splits that preserve subgroup distributions. For evaluation, we compare two settings: *pooled adaptation*, where a model is trained on the entire dataset and then evaluated on specific demographic subgroups, and *demographic adaptation*, where models are adapted separately on each subgroup, aggregated into one, and then evaluated on the combined population; both are visualized in fig. 3.1. This design directly addresses our core question: *Is it better to do demographic adaptation for specialization or use a pooled adaptation that uses large but highly imbalanced demographics?*

CheXpert & MIMIC-CXR. CheXpert was generated at Stanford University Medical Center between October 2002 and July 2017, consisting of 223648 Chest X-rays associated with 65240 patients [19]. MIMIC-CXR dataset contains 227835 Chest X-ray imaging studies with 377110 corresponding images obtained from Beth Israel Deaconess Medical Center between 2011 and 2016 [20].

The CheXpert & MIMIC-CXR datasets were combined to create a single Chest X-Ray dataset. The dataset was further analyzed on the basis of race, where three of the most sizable races remained (Asian, Black, and White), while others were grouped under Other. The dataset stratified on Race was split following a 80% training, 10% validation, and 10% testing distribution.

In the end, we obtain 4 Chest X-Ray datasets, first that includes all the sensitive attributes, and then one for each sensitive attribute. The Chest X-Ray consists of 14 classes: Enlarged Cardiomeastinum, Cardiomegaly, Lung Opacity, Lung Lesion, Edema, Consolidation, Pneumonia, Atelectasis, Pneumothorax, Pleural Effusion, Pleural Other, Fracture, Support Devices, No Finding.

FairVLMed10K. The FairVLMed10K dataset consists of 10,000 visual-linguistic pairs collected for fairness research in medical AI [21]. It was originally introduced in the context of FairCLIP, where it was used to align medical images of the eye with descriptive text annotations. The dataset spans a variety of ophthalmic imaging modalities and textual descriptions, making it suitable for studying fairness in multimodal medical learning. In our work, however, we only utilize the image component to assess demographic adaptation of medical vision foundation models under limited-data conditions.

GF3300. GF3300 was curated for evaluating fairness in medical AI, with a particular focus on glaucoma detection [74]. It contains data from 3,300 subjects, providing both 2D and 3D retinal nerve imaging modalities. By combining structural and volumetric data, the dataset enables the study of diverse patient populations and supports fairness-aware evaluation across different demographic groups. In our setting, GF3300 is used as a lightweight ophthalmic dataset to test demographic adaptation in data-scarce scenarios, complementing large-scale medical image datasets.

CelebA. A face celebrity dataset containing 202,599 images, with 10,177 unique identities [75]. Similarly to previous dataset, we create 3 CelebA dataset variants, first that includes all the sensitive attributes, and

Table 3.1: **Chest X-Ray** datasets, including CheXpert and MIMIC-CXR, along with their demographic breakdowns and dataset variations.

Dataset	Training Set	Evaluation Set	Training	Validation	Testing
CheXpert	All	Asian	18,794	2,214	2,290
		Black	9,349	1,308	1,313
		White	100,401	12,481	12,742
		Other	50,499	6,268	5,989
	Asian	Asian	18,794	2,214	2,290
	Black	Black	9,349	1,308	1,313
	White	White	100,401	12,481	12,742
MIMIC-CXR	All	Asian	8,927	988	1,015
		Black	44,746	5,510	5,581
		White	175,721	21,047	21,435
		Other	73,455	9,575	9,110
	Asian	Asian	8,927	988	1,015
	Black	Black	44,746	5,510	5,581
	White	White	175,721	21,047	21,435
CXR	All	Asian	27,721	3,202	3,305
		Black	54,095	6,818	6,894
		White	276,122	33,528	34,177
		Other	109,024	15,843	15,099
	Asian	Asian	27,721	3,202	3,305
	Black	Black	54,095	6,818	6,894
	White	White	276,122	33,528	34,177
CXR-1K	All	Asian	200	50	3,305
		Black	200	50	6,894
		White	200	50	34,177
		Other	200	50	15,099
	Asian	Asian	800	200	3,305
	Black	Black	800	200	6,894
	White	White	800	200	34,177

Table 3.2: **Retina** datasets, including FairVLMed10K and GF3300, along with their demographic breakdowns and dataset variations.

Dataset	Training Set	Evaluation Set	Training	Validation	Testing
FairVLMed10k	All	Asian	601	60	158
		Black	1,027	159	305
		White	5,372	781	1,537
	Asian	Asian	601	60	158
	Black	Black	1,027	159	305
	White	White	5,372	781	1,537
GF3300	All	Asian	700	100	300
		Black	700	100	300
		White	700	100	300
	Asian	Asian	700	100	300
	Black	Black	700	100	300
	White	White	700	100	300

one for each sensitive attribute. There are 40 binary attributes per image. The dataset is further split into two based on gender (Male, Female). Attributes with low prevalence for either gender are omitted, including: 5 o'clock shadow, bald, goatee, heavy makeup, male, mustache, no beard, sideburns, wearing lipstick, wearing necktie. The rest of the attributes are used for multi-label classification: arched eyebrows, attractive, bags under eyes, bangs, big lips, big nose, black hair, blond hair, blurry, brown hair, bushy eyebrows, chubby, double chin, eyeglasses, gray hair, high cheekbones, mouth slightly open, narrow eyes, oval face, pale skin, pointy nose, receding hairline, rosy cheeks, smiling, straight hair, wavy hair, wearing earrings, wearing hat, wearing necklace, young.

Chest X-Ray 1K (CXR-1K) & CelebA-1K. Inspired by VTAB [73], we construct a subset from each corresponding dataset, consisting of 800 training samples, 200 validation samples, and the complete test set. For datasets that include all sensitive attributes, we ensure equal representation across these attributes. This yields an additional set of 7 datasets for our experiments. Refer to table 3.1 for a detailed breakdown of the 16 datasets. During hyperparameter tuning, we use only the training and validation splits. Once the optimal set of parameters is identified, we combine the training and validation samples resulting in a total of 1,000 examples to train the final model. This model is then evaluated on the full test set. The 1K variants include the same classes as the original complete datasets.

3.3.2 Foundation Models

We consider three pretraining strategies: MoCoV3, MAE, and ViT. All FMs were pretrained in the base-ViT configuration, either by us in-domain (Chest X-Ray) or using publicly available out-of-domain (ImageNet-1K, ImageNet-21K) FMs.

Table 3.3: **Natural** datasets, including CelebA and CelebA-1K, along with their demographic breakdowns and dataset variations.

Dataset	Training Set	Evaluation Set	Training	Validation	Testing
CelebA	All	Male	68,261	8,458	7,715
		Female	94,509	11,409	12,247
	Male	Male	68,261	8,458	7,715
	Female	Female	94,509	11,409	12,247
CelebA-1K	All	Male	400	100	7,715
		Female	400	100	12,247
	Male	Male	800	200	7,715
	Female	Female	800	200	12,247

3.3.2.1 Pretraining Details

MAE was pretrained following a setup similar to [13], using 4 NVIDIA RTX 6000 GPUs over approximately 15 days. For MoCoV3, we used a batch size of 4096 with the AdamW optimizer, a weight decay of 0.1, a learning rate of 1×10^{-5} , and a 40-epoch warm-up schedule, which produced the lowest training loss. All remaining hyperparameters and image transformations were kept consistent with the original MoCoV3 implementation, and the model was pretrained for 300 epochs. This pretraining process required 64 NVIDIA RTX 6000 GPUs and completed in approximately 3 days.

3.3.3 Adaptation Methods

Each FM is adapted to tasks using five adaptation strategies, full fine-tuning (Full), linear probing (Linear), visual prompt tuning (VPT), gated VPT (GVPT), and effective & efficient VPT (E^2 VPT). Full adaptation updates the entire backbone and linear classifier parameters, whereas Linear only updates the linear classifier. VPT trains a small set of prompts prepended to the input image while freezing the rest of the model. GVPT introduces a gating mechanism to dynamically control prompt influence across transformer blocks, enabling more flexible and context-aware adaptation. Finally, E^2 VPT builds upon VPT by introducing separate trainable key and value prompts that are injected into self-attention layers, combined with prompt pruning for efficiency. The flow of these strategies was illustrated and formulated earlier in chapter 2.

For CelebA, we used publicly available MAE and MoCoV3 models pretrained on ImageNet-1K, along with a ViT model pretrained by Google on ImageNet-21K [40]. Across all foundation models, the base-ViT configuration was used. Additionally, only the original image transformations and preprocessing procedures from the respective pretraining methods were applied.

3.3.3.1 Adaptation Details

For the large-scale CXR setting, models were trained for 75 epochs with a 5-epoch warm-up phase. For all remaining datasets, including the data-scarce variants, training was conducted for 100 epochs with a 10-

Table 3.4: **Parameter Search** for Pooled & Demographic adaptation.

Parameter	Full	Linear	VPT [31]	E ² VPT [51]	GVPT [50]
Learning Rate	[0.01-10.0]	[0.01-100.0]	[0.01-100.0]	[0.01-2.5]	[0.01-5.0]
Weight Decay	[0.0,0.00001,0.0001,0.001,0.01,0.1]				
Prompt Size			[10,25,50,100,150,200]	[10,25,50,100,150]	
K – V Prompt Size				[5,10,25,50,100,150]	
Gate Initialization					[1,5,10]

epoch warm-up. A batch size of 128 was used for the full datasets, while the 1K variants used a batch size of 64. All images were resized to 224×224 and normalized using standard preprocessing procedures, with no additional image augmentations or transformations applied. Experiments were conducted on NVIDIA RTX 6000 GPUs.

3.3.3.2 Hyperparameter Search

The initial hyperparameter search was conducted on the 1K dataset variants using the FM models. For both Full and Linear adaptation, the best-performing configurations were generally consistent across demographic subsets, suggesting that hyperparameters selected on one subset often transfer well to others. In the Chest X-Ray classification setting, Full adaptation also revealed a clear separation between stable low-loss configurations and unstable training regimes, where some runs failed due to exploding gradients.

For VPT-based methods, we first identified suitable learning rate and weight decay values for each pretrained model and adaptation setting using a grid search tailored to the downstream task. We then performed a more detailed sweep over adaptation-specific parameters. In contrast to learning rate and weight decay, parameters such as prompt size exhibited greater variability, with strong configurations distributed more irregularly across the search space. This suggests that VPT and its variants are more sensitive to initialization and interactions between hyperparameters, particularly when adapting ImageNet-pretrained MAE models to domain-shifted tasks such as Chest X-Ray interpretation.

Overall, we performed a comprehensive hyperparameter search for both CXR-1K and CelebA-1K across all demographic subsets. The insights from these smaller-scale experiments were then used to guide training on the full-scale datasets, including CXR, FairVLMed10k, GF3300, and CelebA.

For the full-scale datasets, the hyperparameter search was restricted to a refined set of configurations informed by the preliminary experiments and the ranges reported in table 3.4. This stage used broader step sizes for learning rate and weight decay to reduce computational cost while preserving coverage of promising regions. Tuning was first performed on the smallest demographic subset using a fine-grained search strategy. The top five or more configurations were then selected and evaluated on progressively larger demographic subsets.

		Aggregate					ASIAN					BLACK					WHITE				
CXR	MAE	81.1	79.3	81.2	81.7	81.0	81.5	79.3	80.7	81.2	80.4	82.1	80.4	82.5	82.8	82.3	80.6	78.8	80.7	81.3	80.6
	(IN)	82.2	78.9	80.7	81.5	81.6	80.7	78.8	79.6	79.9	80.0	81.9	79.7	81.3	81.3	81.5	82.2	79.0	80.5	81.5	81.5
	MoCoV3	81.6	79.2	81.5	80.2	80.7	82.0	78.8	81.1	79.5	80.0	82.5	80.2	82.5	81.2	81.4	81.1	78.8	81.1	79.8	80.3
	(IN)	80.6	79.1	81.2	80.0	80.5	79.8	77.8	79.1	78.8	79.1	79.9	80.7	80.7	80.4	80.5	80.4	78.8	81.2	79.9	80.5
	MAE	83.1	72.5	79.0	79.2	79.1	83.0	72.3	78.6	78.8	78.2	84.3	73.1	79.7	80.2	79.8	82.7	72.0	78.5	78.7	78.7
	(IN)	81.1	72.9	78.0	79.1	80.2	78.8	73.6	75.6	77.2	77.1	80.9	74.8	77.9	78.7	78.8	81.1	72.8	77.9	79.2	80.5
FairVLMed10k	MAE	81.7	74.3	80.6	78.6	78.8	82.4	74.3	80.1	77.9	78.3	82.7	75.2	81.5	79.6	79.6	81.2	73.8	80.1	78.1	78.4
	(IN)	79.7	74.5	80.0	78.7	79.4	78.2	74.1	77.0	76.9	77.2	79.9	75.8	79.1	78.9	79.3	79.6	74.0	80.3	78.6	79.5
	ViT	75.6	74.4	78.4	79.0	80.2	75.4	74.3	77.7	78.7	79.6	77.2	75.3	79.4	79.8	80.8	75.0	73.9	77.9	78.6	79.8
	(IN)	78.4	74.4	77.8	78.5	79.1	76.4	73.5	75.7	76.2	76.5	77.5	75.0	77.5	78.4	78.8	78.7	73.9	77.7	78.7	79.5
	MAE	80.5	77.4	79.8	81.2	80.8	82.6	78.0	84.8	83.9	86.4	74.3	75.2	74.8	78.0	75.6	81.1	77.4	80.1	81.5	81.1
	(IN)	80.0	66.0	79.5	80.2	80.7	81.5	77.6	79.7	82.0	82.6	73.8	72.3	76.9	73.4	75.0	81.2	77.5	81.1	80.6	81.6
GF3300	MAE	81.0	80.1	81.8	80.5	81.3	82.7	81.4	86.4	85.1	86.6	79.4	77.4	77.7	76.5	78.6	81.2	80.4	82.0	80.5	81.2
	(IN)	79.8	77.4	81.7	81.6	82.1	84.4	81.6	85.3	87.0	86.8	73.8	75.8	75.6	77.0	77.1	80.9	81.0	82.5	81.8	82.7
	ViT	81.3	75.0	77.6	76.0	78.5	83.3	76.4	82.7	82.6	84.9	75.9	69.4	72.6	70.2	74.5	82.2	75.6	78.1	76.5	78.7
	(IN)	78.3	75.3	75.7	79.6	79.7	81.1	75.6	80.2	80.6	82.2	70.0	70.6	72.4	74.2	74.7	81.2	75.7	79.5	79.9	80.4
	MAE	85.9	85.2	85.1	85.8	86.0	89.1	87.5	89.0	89.4	89.7	80.8	82.4	81.1	81.3	81.8	86.5	85.0	84.3	85.6	85.4
	(IN)	80.3	85.3	86.8	86.9	86.9	89.7	87.7	89.4	89.8	89.3	79.9	81.8	82.2	82.8	83.4	87.4	85.2	87.1	87.7	87.1
GF3300	MAE	84.5	85.6	86.1	85.7	86.1	87.9	88.1	89.1	88.8	89.1	80.2	81.0	80.5	81.0	80.5	84.3	86.2	87.3	86.0	87.0
	(IN)	85.8	85.2	85.6	86.3	84.7	88.2	87.9	88.8	89.4	89.8	81.8	80.5	82.3	82.5	82.8	87.0	86.0	87.3	87.8	88.7
	ViT	84.8	84.4	85.7	85.9	85.2	88.1	87.2	89.5	88.7	89.5	80.5	80.0	80.1	81.2	79.9	84.8	85.0	86.5	86.6	85.2
	(IN)	86.5	79.6	85.9	84.2	86.5	89.2	88.1	89.1	89.1	90.5	80.9	82.4	81.9	83.4	83.4	86.7	85.0	87.8	88.3	88.6
	MAE	85.9	85.2	85.1	85.8	86.0	89.1	87.5	89.0	89.4	89.7	80.8	82.4	81.1	81.3	81.8	86.5	85.0	84.3	85.6	85.4
	(IN)	80.3	85.3	86.8	86.9	86.9	89.7	87.7	89.4	89.8	89.3	79.9	81.8	82.2	82.8	83.4	87.4	85.2	87.1	87.7	87.1

Figure 3.2: **Utility on CXR, FairVLMed10K & GF3300**, comparing performance of FMs and adaptation methods across demographics (Aggregate, Asian, Black, White). In each cell, the **top value** is from a model trained on pooled data; **bottom value** is from a model trained only on the selected demographic. Colors Δ indicate *Demographic – Pooled*: **green** ($\Delta > 1.0$), **red** ($\Delta < -1.0$), and **yellow** ($|\Delta| \leq 1.0$).

3.4 Experiments

We evaluate five adaptation strategies, Full, Linear, VPT, GVPT, and E²VPT, across three foundation model pretraining paradigms: ViT, MAE, and MoCoV3. Our experiments span both medical and natural image domains, including CXR, CXR-1K, FairVLMed10k, and GF3300. For completeness, we also include additional natural-domain results covering CelebA and CelebA-1K. Performance is measured using AUC, reported both per demographic subgroup and aggregated across all groups. Subgroup disparities are quantified as Δ AUC between the best- and worst-performing demographic groups. To assess whether a configuration mitigates underdiagnosis [24], we additionally report the False Positive Rate (FPR) gap for the “No Finding” label in CXR datasets.

Across all plots, results are color-coded to indicate the relative effect of demographic-specific adaptation. **Green** denotes cases where demographic adaptation improves performance over pooled adaptation, whereas **red** denotes cases where pooled adaptation performs better. **Yellow** indicates cases where the absolute performance difference is within a 1% margin, which we treat as a practical equivalence region. This margin is used to distinguish meaningful performance changes from small fluctuations that may arise from training stochasticity, demographic subset sampling, or hyperparameter sensitivity. Therefore, yellow cells indicate

		Aggregate					ASIAN					BLACK					WHITE				
CXR-1K	MAE	68.8	71.8	72.0	70.5	72.4	69.0	72.4	72.7	71.3	73.4	69.5	72.3	72.6	71.1	73.2	68.3	71.2	71.4	69.9	71.8
	(IN)	69.0	71.1	71.7	69.5	71.8	69.2	72.7	73.1	72.2	72.9	69.4	72.3	73.7	71.7	74.1	68.8	70.4	70.9	68.9	71.0
	MoCoV3	67.9	72.0	70.1	71.8	73.2	67.8	72.1	70.4	72.2	73.7	69.1	73.7	72.2	73.3	74.9	67.4	71.3	69.3	71.1	72.4
	(CXR)	67.9	70.7	71.4	70.4	70.6	66.8	71.9	72.2	71.7	72.0	68.3	72.1	71.9	72.4	72.4	67.7	70.1	71.0	69.7	70.0
	MAE	64.6	68.0	67.8	66.3	68.5	64.2	67.8	68.1	65.6	67.9	65.6	68.7	69.4	66.8	70.0	64.1	67.4	67.1	65.8	67.9
	(IN)	65.0	65.5	66.4	66.3	66.8	64.4	66.5	67.2	65.5	67.5	66.5	67.4	69.9	68.6	70.0	64.6	64.6	65.3	65.7	66.1
	ViT	64.0	69.3	66.9	67.7	69.1	63.9	68.7	67.1	67.0	68.9	64.7	70.1	68.8	69.5	71.1	63.7	68.8	66.2	67.1	68.3
	(IN)	64.1	67.8	66.2	66.1	66.7	63.1	67.0	66.8	66.8	65.2	66.9	69.3	69.3	68.8	68.5	63.0	67.2	65.1	65.4	66.2
	MAE	65.3	67.5	64.8	64.7	66.5	64.7	66.9	64.0	65.3	65.2	65.5	67.3	65.5	65.9	67.8	65.0	67.1	64.3	64.1	66.0
	(IN)	66.6	66.7	66.7	64.9	66.6	64.8	66.8	66.6	64.9	67.4	66.7	68.7	69.6	66.8	68.6	66.2	65.9	65.7	64.4	65.8

■ ($\Delta < -1.0$)
 ■ ($|\Delta| \leq 1.0$)
 ■ ($\Delta > 1.0$)

Figure 3.3: **Utility on CXR-1K**, comparing performance of FMs and adaptation methods across demographics (Aggregate, Asian, Black, White). In each cell, the **top value** is from a model trained on pooled data; **bottom value** is from a model trained only on the selected demographic. Colors Δ indicate *Demographic – Pooled*: **green** ($\Delta > 1.0$), **red** ($\Delta < -1.0$), and **yellow** ($|\Delta| \leq 1.0$).

settings where demographic adaptation achieves comparable performance to pooled adaptation without a practically significant loss.

3.4.1 Demographic Adaptation Utility

3.4.1.1 Evaluation Overview

The fig. 3.2 summarizes the performance of different adaptation methods across FMs, demographic groups, and the aggregate population. CXR experiments represent a large and highly imbalanced setting, providing a clear view of how demographic-specific adaptation behaves under unequal subgroup distributions. In this domain, White patients constitute the majority of the training data, while Black patients are minorities and Asian patients are rarely represented. Leveraging this, we are evaluating whether subgroup-specific adaptation can recover performance for minority groups without compromising overall utility. In contrast, FairVLMed10k and GF3300 provide a complementary setting with smaller-scale datasets, where GF3300 is balanced. This allows us to study adaptation behavior under more uniform subgroup representation and to isolate the effect of data distribution on model performance and disparities.

3.4.1.2 Aggregate vs. Demographic Performance

While in fig. 3.2 we can observe that demographic adaptation can match or even improve performance within individual groups (Asian, Black, White), in some rare cases these gains do not translate to improved aggregate performance. For instance, we can see this in the case of ViT Linear and GVPT adaptation on GF3300 in fig. 3.2, the demographic adaptation has consistently better performance on Asian, Black and White, but under the aggregate it underperforms. This occurs due to differences in score distributions across demographic-specific adaptation, which can reduce overall AUC when predictions are aggregated. This underscores the importance of evaluating models at the demographic level rather than relying solely on

aggregate metrics. *In summary, due to global thresholding effects, there is a need to report demographic-level evaluation rather than aggregate metrics.*

3.4.1.3 Imbalanced Settings

Despite the difference in scale between the large CXR dataset and the smaller FairVLMed10K dataset, both exhibit similar patterns. The majority subgroup (White) retains most of its performance (see CXR subplot in fig. 3.2), never underperforms across adaptation methods and FM models (no occurrences of red blocks). This is even consistent when the models trained on reduced data (see FairVLMed10K subplot in fig. 3.2). On the other hand, the minority Black subgroup demographic adaptation experiences moderate degradation under simpler prompt-based methods such as VPT (see both CXR & FairVLMed10K fig. 3.2) where in all cases for imbalanced CXR demographic adaptation hurts utility. However, higher-capacity VPT variants such as E²VPT are able to recover much of this loss in both in-domain and out-of-domain settings, where only the ViT-IN demographic adaptation underperforms. In contrast, the rare subgroup (Asian) shows the largest performance drop under demographic-specific training. The main exception is Linear adaptation (across all datasets in fig. 3.2), where demographic adaptation consistently does not hurt performance. We should note that Linear adaptation reaches these results at the cost of a lower AUC in general compared to other adaptation methods. This occurs because Linear adaptation due to its low parameter capacity has a limited capacity to learn, beyond its backbone performance. Finally, we would like to highlight E²VPT when paired with in-domain pretraining, preserving both the performance and not being hurt by demographic adaptation (see MAE-CXR across all demographics in fig. 3.2). These results suggest that, even in low-data settings, higher-capacity prompt-based adaptation methods can effectively specialize to a target demographic when the pretrained model is well aligned with the downstream domain. *In summary, these findings highlight that leveraging the in-domain foundation model with a VPT variant E²VPT demographic adaptation performs comparatively to pooled adaptation even for the smallest demographic groups.*

3.4.1.4 Balanced Setting

In the more balanced GF3300 dataset (see GF3300 in fig. 3.2), demographic-specific adaptation behaves consistently better and does not hurt. Here, VPT and its variants are able to match or even exceed pooled performance across multiple subgroups. *In summary, these findings highlight that a balanced data distribution may lead to a better impact of demographic adaptation across groups even when the foundation model is out-of-distribution.*

3.4.1.5 Utility Under Data Scarcity

Under data-scarce conditions (fig. 3.3), only a small number of labelled examples are available for adaptation. In this setting, demographic adaptation yields the greatest benefits for the minority demographics, Asian and Black, whereas the majority demographic, White, tends to benefit more from pooled adaptation. Across both pooled and demographic settings, Full adaptation is often the weakest method, likely due to overfitting in the low-data regime. In contrast, E²VPT consistently achieves the strongest overall performance, indicating

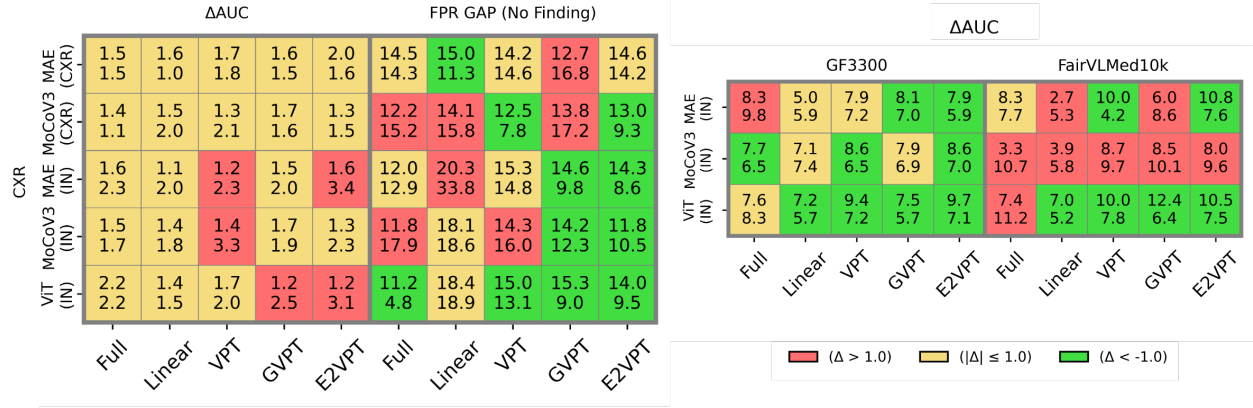


Figure 3.4: **Subgroup disparity on CXR, GF3300 and FairVLMed10K**, defined as the Δ AUC difference between the lowest and highest demographic group values (Asian, Black, White). Similarly, FPR Gap is reported only for No Finding label. For both Δ AUC and FPR Gap, smaller is better. In each cell, the **top value** is from a model trained on pooled data; **bottom value** is from a model trained only on the selected demographic. Colors Δ indicate *Demographic – Pooled*: **green** ($\Delta < -1.0$), **red** ($\Delta > 1.0$), and **yellow** ($|\Delta| \leq 1.0$).

its ability to effectively leverage pretrained representations in data scarcity. All in all, these findings amplify the importance of methods that can efficiently reuse pretrained knowledge while avoiding overfitting. Among PEFT approaches, higher-capacity prompt-based methods, E²VPT, provide the most reliable balance between computational efficiency and predictive performance, even when subgroup data is extremely limited. *In summary, under data-scarce conditions, demographic adaptation provides the largest gains for minority groups (Asian and Black), highlighting its effectiveness in improving performance where data is most limited.*

3.4.2 Subgroup Performance Disparity

3.4.2.1 Evaluation Overview

Here we report subgroup performance differences measured via Δ AUC for each adaptation strategy, and under-diagnosis rate, FPR gap for the "No Finding" label in the CXR setting.

3.4.2.2 Disparities & Under-diagnosis

In fig. 3.4 (left subplot for CXR, representing a highly imbalanced dataset setting), subgroup disparities Δ AUC, are very similar (yellow) between demographic and pooled adaptation, with demographic adaptation performing poorly (red) in a few out-of-domain VPT variant cases. Next, in the smaller imbalanced retina dataset, FairVLMed10K, demographic adaptation has inconsistent disparities (see the right subplot of fig. 3.4). This is due to a larger gap between the Asian and White demographic performance in FairVLMed10K as opposed to CXR due to very limited dataset size fig. 3.2. In the more balanced retina dataset, GF3300 (see the right subplot of fig. 3.4), disparity patterns are more consistent, with demographic adaptation consistently outperforming pooled adaptation. Lastly, in the same left CXR subplot in fig. 3.4, we report FPR gap for "No Finding" label. Here, we observe that most methods (e.g., E²VPT) for demographic

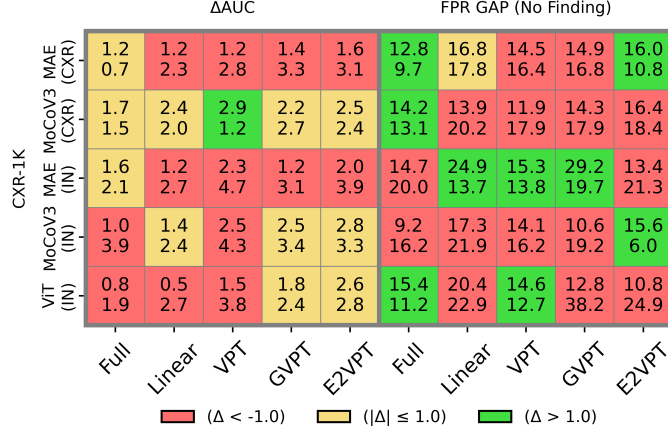


Figure 3.5: **Subgroup disparity on CXR-1K**, defined as the Δ AUC difference between the lowest and highest demographic group values (Asian, Black, White). Similarly, FPR Gap is reported only for No Finding label. For both Δ AUC and FPR Gap, smaller is better. In each cell, the **top value** is from a model trained on pooled data; **bottom value** is from a model trained only on the selected demographic. Colors Δ indicate *Demographic – Pooled*: **green** ($\Delta < -1.0$), **red** ($\Delta > 1.0$), and **yellow** ($|\Delta| \leq 1.0$).

adaptation consistently outperform their pooled-adaptation counterparts. *In summary, demographic adaptation reduces subgroup disparities and under-diagnosis, especially in balanced settings, by better aligning model performance across demographics.*

3.4.2.3 Disparity Under Data Scarcity

The overall lower AUC under data scarcity (see fig. 3.3) propagates to disparity metrics in fig. 3.5. demographic-specific adaptation generally leads to increased subgroup disparity, as improvements for certain groups (Asian and Black) do not extend to others (White). The under-diagnosis rate (FPR Gap for the No Finding label) does not exhibit a consistent trend across methods. This is likely because the models’ performance is generally low, which increases the overall false outcome; however, the lowest FPR gaps are often achieved by E²VPT under demographic adaptation, suggesting some robustness in controlling error disparities. *In summary, under data scarcity, demographic adaptation can improve minority subgroup performance but often increases overall disparity due to uneven gains across groups.*

3.4.3 Utility & Compute Tradeoff

3.4.3.1 Evaluation Overview

Across figs. 3.6 and 3.7, we analyze the performance deltas, indicating the difference between demographic and pooled adaptation, with respect to the training time of each. Using this we construct an example of a tradeoff line by allowing up to a 1% performance drop, linearly scaled against reductions of up to 50% in training compute time. This enables a consistent comparison of efficiency-performance tradeoffs across methods, datasets, and demographics. We omit utility-compute tradeoff analysis in the data-scarce setting, as performance is primarily limited by the small number of labeled examples, making differences in computational cost less informative than variability in AUC.

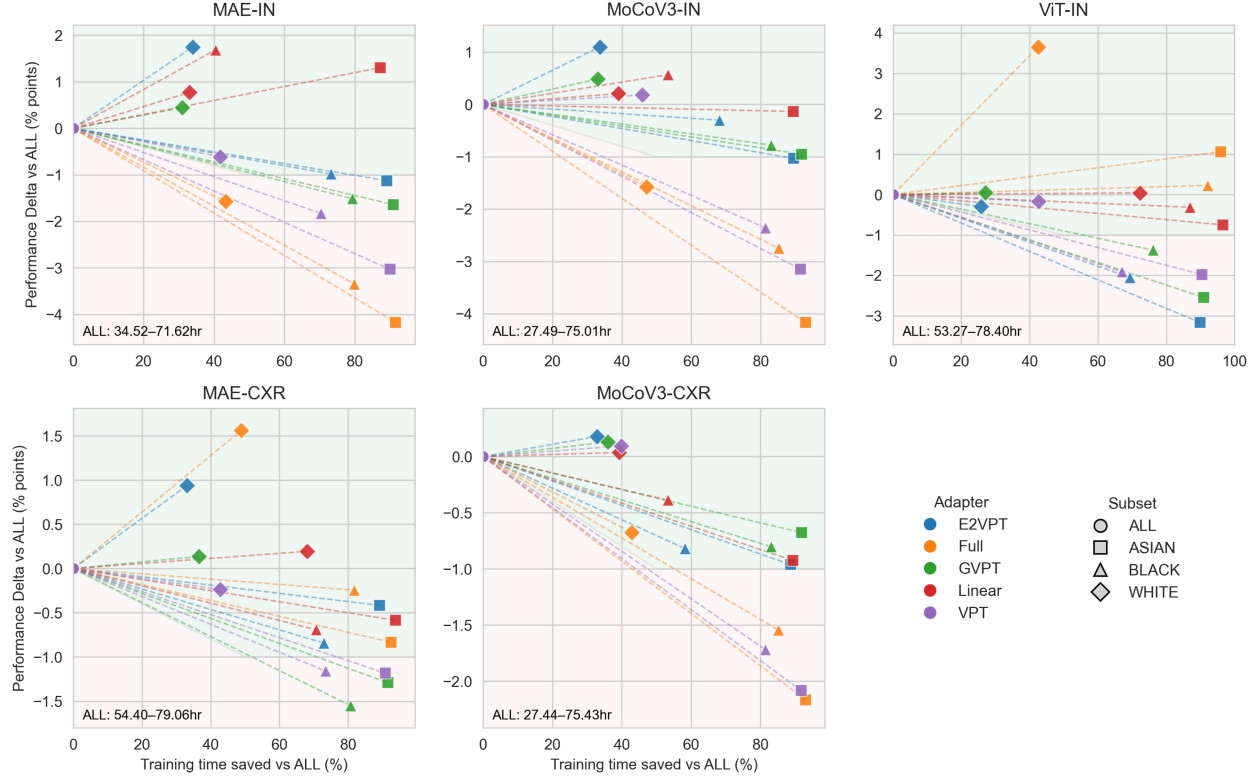


Figure 3.6: **Utility-compute tradeoff on CXR** across adaptation methods and pretraining paradigms. The y-axis shows per-demographic performance difference (adapter AUC minus pooled adaptation). Both performance deltas and training time savings are reported relative to each adapter method. Each subplot also reports full-data training time range for reference. We define a tradeoff boundary allowing up to a 1% performance drop for up to a 50% compute reduction. Points within this tolerance are colored **green**, and others are colored **red**.

3.4.3.2 Overall Efficiency-Performance Trends

Across all datasets in figs. 3.6 and 3.7, demographic adaptation demonstrates a favorable tradeoff: as training time approaches that of the pooled-data baseline, performance remains comparable or even improved. As training time is further reduced, more advanced methods, particularly E^2VPT , are better able to preserve performance while achieving substantial computational savings. For example, using the E^2VPT adapter on MAE-IN and MoCoV3-IN in an out-of-domain setting (trained on the Asian CXR subset), performance drops by only 1%, while training time is reduced by nearly 90% (from 47.81 hours to 5.1 hours). *In summary, demographic adaptation, especially with E^2VPT , achieves strong performance while substantially reducing training time, offering an efficient performance-compute tradeoff.*

3.4.3.3 Dataset-Specific Behavior

We observe similar trends in fig. 3.7. On FairVLMed10K, the smaller dataset size leads to greater performance variability, especially when training on the smallest demographic subset (Asian), resulting in larger swings in performance. In contrast, on the more balanced GF3300 dataset, where demographic subsets are of comparable size, these instabilities are reduced. Here, the benefits of demographic adaptation are more con-

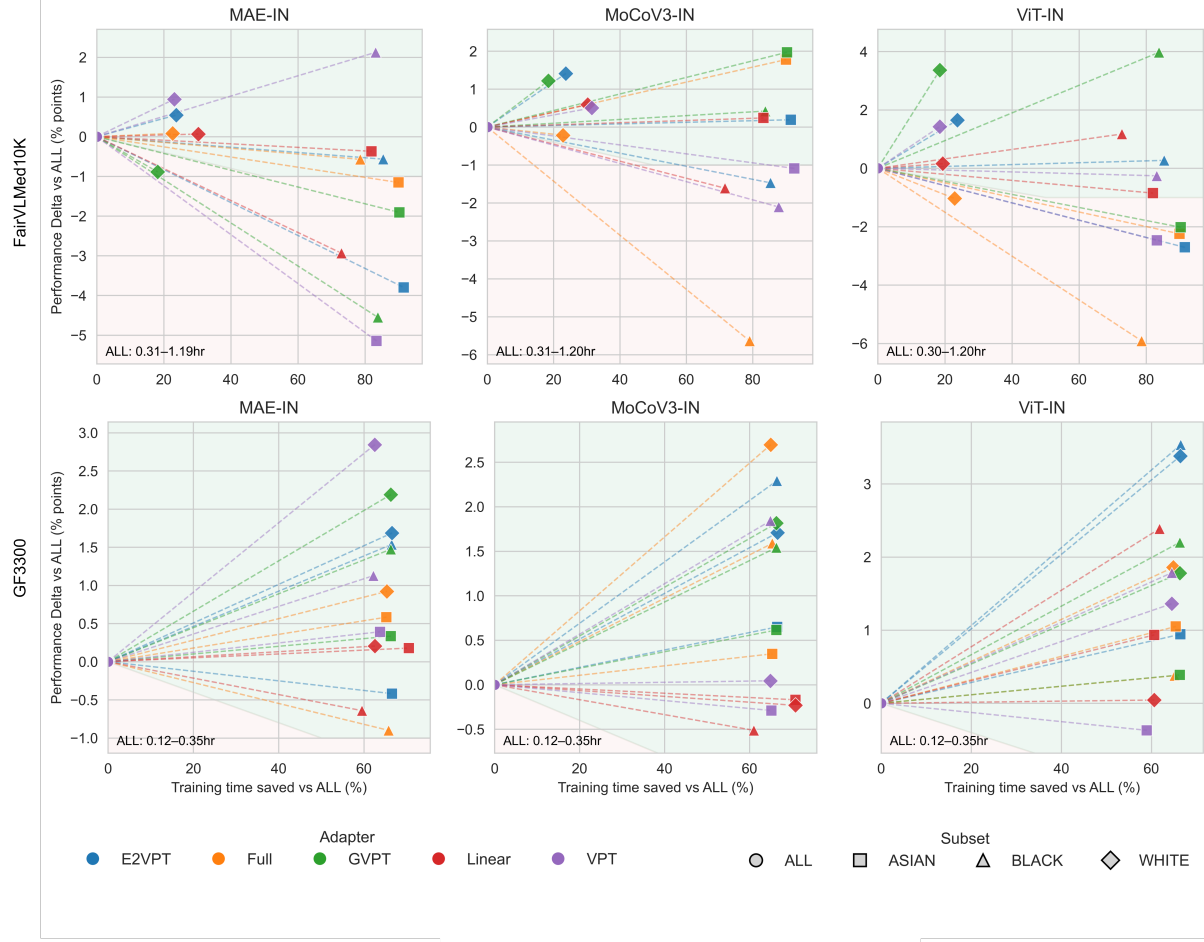


Figure 3.7: **Utility-compute tradeoff on FairVLMed10k & GF3300** across adaptation methods and pretraining paradigms. The y-axis shows per-demographic performance difference (adapter AUC minus pooled adaptation). Both performance deltas and training time savings are reported relative to each adapter method. Each subplot also reports full-data training time range for reference. We define a tradeoff boundary allowing up to a 1% performance drop for up to a 50% compute reduction. Points within this tolerance are colored **green**, and others are colored **red**.

sistent: performance remains within a narrow margin of the pooled-data baseline or improves, particularly when using higher-capacity methods such as E^2VPT . *In summary, demographic adaptation is consistently beneficial in balanced datasets, while smaller, imbalanced datasets lead to higher performance variability.*

3.4.3.4 In-Domain vs. Out-of-Domain Adaptation

For in-domain adaptation (MAE-CXR and MoCoV3-CXR) in fig. 3.6, performance degradation is notably smaller compared to the out-of-domain settings. E^2VPT consistently operates within a negligible margin of the pooled-data baseline, while other methods also exhibit competitive behavior with only minor losses. This suggests that demographic adaptation is especially effective when the domain shift is limited. *In summary, demographic adaptation is most effective under minimal domain shift, maintaining near-baseline performance while reducing training cost, particularly with E^2VPT .*

3.4.3.5 Limitations of Linear Adaptation

As observed earlier (see section 3.4.1), Linear adaptation is clearly constrained by its limited parameter capacity, resulting in very similar performance relative to its baseline (see figs. 3.6 and 3.7). In this scenario, under strict compute constraints, demographic adaptation can still provide meaningful speedups, offering a practical pathway to reduce training cost when more advanced adapters are not feasible.

3.4.3.6 Overall Comparison of Adaptation Strategies

Overall, when restricted to linear adaptation, its limited parameter capacity allows for further reduction in training data by focusing on a single demographic subset with minimal additional loss. In contrast, Full adaptation, with its substantially higher capacity, is more prone to overfitting and therefore typically requires larger datasets and longer training times. VPT variants, and especially E²VPT offer a more balanced alternative: by combining parameter-efficient adaptation with demographic specialization, they achieve significant reductions in training time while maintaining only marginal performance degradation, effectively balancing efficiency and performance.

3.4.4 Demographic Adaptation in Natural Domain

The CelebA experiments provide a natural-image counterpart to the medical settings. Compared with CXR, CelebA is less domain-specific and more balanced across its demographic split, allowing us to examine whether the patterns observed in medical imaging persist when the downstream task is closer to generic visual recognition and the subgroup distributions are less extreme. As in the medical experiments, we compare pooled adaptation against demographic adaptation under both full-data and 1K data-scarce regimes.

3.4.4.1 Demographic Adaptation Utility

	Aggregate					FEMALE					MALE				
	Full	Linear	VPT	GVPT	E ² VPT	Full	Linear	VPT	GVPT	E ² VPT	Full	Linear	VPT	GVPT	E ² VPT
CelebA MAE (IN)	93.1	86.2	90.4	90.5	93.2	92.1	84.0	88.8	89.0	92.3	92.5	83.0	88.5	88.7	92.4
MoCoV3 (IN)	92.8	87.1	90.6	90.9	93.0	91.8	85.0	89.0	89.2	92.1	91.9	85.2	89.6	89.7	92.1
VIT (IN)	92.5	88.1	92.4	90.7	91.3	91.4	86.3	91.3	89.0	90.0	91.7	85.7	91.3	89.3	89.7
MoCoV3 (IN)	92.7	88.3	92.2	90.9	92.5	91.5	86.5	91.0	89.2	91.5	91.8	86.4	90.8	89.4	91.4
VIT (IN)	90.8	89.8	92.2	92.5	93.1	89.6	88.3	91.0	91.3	92.0	89.3	88.3	91.3	91.6	92.3
MoCoV3 (IN)	88.4	90.1	92.3	92.5	92.9	89.7	88.5	91.1	91.3	91.9	89.1	89.0	91.4	91.5	92.0

■ ($\Delta < -1.0$)
 ■ ($|\Delta| \leq 1.0$)
 ■ ($\Delta > 1.0$)

Figure 3.8: **Utility on CelebA**, comparing performance of FMs and adaptation methods across demographics (Aggregate, Male, Female). In each cell, the **top value** is from a model trained on pooled data; **bottom value** is from a model trained only on the selected demographic. Colors Δ indicate *Demographic – Pooled*: **green** ($\Delta > 1.0$), **red** ($\Delta < -1.0$), and **yellow** ($|\Delta| \leq 1.0$).

			Aggregate					FEMALE					MALE				
CelebA-1K	VIT	MoCoV3	MAE	(IN)													
	Full	Linear	VPT	GVPT	E2VPT	Full	Linear	VPT	GVPT	E2VPT	Full	Linear	VPT	GVPT	E2VPT	Full	Linear
	70.1	81.6	85.5	81.6	85.6	66.8	78.8	82.8	78.0	83.0	64.5	77.7	82.9	77.8	82.9	77.7	82.9
	77.2	80.8	86.1	75.5	86.6	66.9	78.7	82.4	56.4	84.1	80.3	77.7	83.4	83.8	84.1	83.4	84.1
	84.4	85.3	87.4	85.9	84.5	81.5	82.8	84.9	83.0	81.3	81.9	82.7	85.4	84.4	82.2	85.4	82.2
	84.8	86.0	84.3	83.5	85.5	82.9	83.1	85.1	83.6	82.4	81.5	82.8	71.1	71.1	83.3	81.5	83.3
	86.8	85.4	87.7	85.9	87.2	84.2	82.5	85.2	82.8	84.5	84.8	82.7	86.1	83.7	85.4	84.8	85.4
	87.7	85.8	87.8	87.0	87.5	85.6	82.9	84.7	84.0	85.2	85.5	81.8	85.9	84.3	84.7	85.5	84.7

■ ($\Delta < -1.0$) ■ ($|\Delta| \leq 1.0$) ■ ($\Delta > 1.0$)

Figure 3.9: **Utility on CelebA-1K**, comparing performance of FMs and adaptation methods across demographics (Aggregate, Male, Female). In each cell, the **top value** is from a model trained on pooled data; **bottom value** is from a model trained only on the selected demographic. Colors Δ indicate *Demographic – Pooled*: green ($\Delta > 1.0$), red ($\Delta < -1.0$), and yellow ($|\Delta| \leq 1.0$).

The CelebA utility results in figs. 3.8 and 3.9 stay consistent with the medical results. There is little difference between pooled adaptation and demographic adaptation across all foundation models, adapter and demographic, staying consistently yellow. In the full-data regime, the demographic adaptation using E²VPT is able to match pooled adaptation and is even often competitive to Full adaptation. In the CelebA-1K setting, the low-data regime follows the same pattern. Among the lightweight methods, the higher-capacity prompt-based method E²VPT tends to preserve performance better than other baselines. Overall, CelebA & CelebA-1K confirm that demographic adaptation is a viable alternative to pooled adaptation, where demographic adaptation can match pooled adaptation.

3.4.4.2 Subgroup Performance Disparity

			Δ AUC				
CelebA	VIT	MoCoV3	MAE	(IN)			
	Full	Linear	VPT	GVPT	E2VPT	Full	Linear
	0.4	1.0	0.3	0.2	0.1	0.4	0.3
	0.2	0.3	0.6	0.5	0.0	0.2	0.0
	0.3	0.6	0.0	0.3	0.3	0.3	0.3
	0.3	0.1	0.2	0.2	0.1	0.3	0.1
	0.4	0.1	0.3	0.3	0.3	0.4	0.3
	0.6	0.5	0.3	0.2	0.1	0.6	0.1

■ ($\Delta < -1.0$) ■ ($|\Delta| \leq 1.0$) ■ ($\Delta > 1.0$)

Figure 3.10: **Subgroup disparity on CelebA**, defined as the Δ AUC difference between the lowest and highest demographic group values (Male, Female). For Δ AUC, smaller is better. In each cell, the **top value** is from a model trained on pooled data; **bottom value** is from a model trained only on the selected demographic. Colors Δ indicate *Demographic – Pooled*: green ($\Delta < -1.0$), red ($\Delta > 1.0$), and yellow ($|\Delta| \leq 1.0$).

		Δ AUC				
CelebA-1K	MAE (IN)	2.3 13.5	1.1 1.1	0.1 1.0	0.2 27.4	0.0 0.0
	MoCoV3 (IN)	0.4 1.4	0.1 0.3	0.5 14.0	1.4 12.6	0.9 0.9
	ViT (IN)	0.6 0.1	0.2 1.1	0.9 1.2	0.9 0.3	1.0 0.5
		Full	Linear	VPT	GVPT	E2VPT
		$(\Delta < -1.0)$	$(\Delta \leq 1.0)$	$(\Delta > 1.0)$		

Figure 3.11: **Subgroup disparity on CelebA-1K**, defined as the Δ AUC difference between the lowest and highest demographic group values (Male, Female). For Δ AUC, smaller is better. In each cell, the **top value** is from a model trained on pooled data; **bottom value** is from a model trained only on the selected demographic. Colors Δ indicate *Demographic – Pooled*: **green** ($\Delta < -1.0$), **red** ($\Delta > 1.0$), and **yellow** ($|\Delta| \leq 1.0$).

Similar to medical results, the CelebA disparity stays consistent throughout in fig. 3.10 with minimal differences between the adaptation strategies. This trend continues in CelebA-1K in fig. 3.11 staying mostly consistent.

3.4.4.3 Utility & Compute Tradeoff

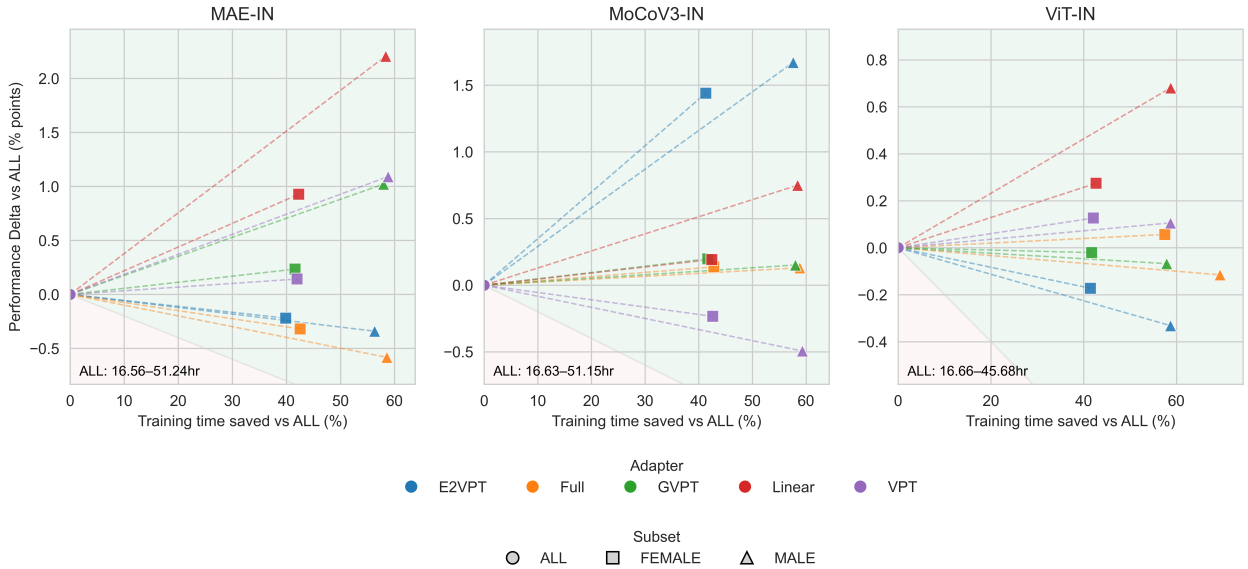


Figure 3.12: **Utility-compute tradeoff on CelebA** across adaptation methods and pretraining paradigms. The y-axis shows per-demographic performance difference (adapter AUC minus pooled adaptation). Both performance deltas and training time savings are reported relative to each adapter method. Each subplot also reports full-data training time range for reference. We define a tradeoff boundary allowing up to a 1% performance drop for up to a 50% compute reduction. Points within this tolerance are colored **green**, and others are colored **red**.

The tradeoff plot for CelebA (see fig. 3.12) highlights the same practical pattern seen throughout the study: demographic adaptation provides a more efficient alternative when compute constraints matter. In the

natural-domain setting, this efficiency comes with less of a performance penalty than in the most imbalanced medical settings.

3.5 Summary & Discussion

Our results provide a comprehensive empirical study of demographic adaptation under realistic compute and demographic distribution constraints, explicitly demonstrating the trade-off between performance and training cost across adaptation strategies. We found that leveraging PEFT for demographic adaptation through VPT variants (especially E^2 VPT), not only assists in retaining the performance of Full, but also exposes a favorable trade-off between utility, disparity, and compute, enabling effective subgroup specialization with reduced data, preserving much of the performance of pooled adaptation, and achieving substantial gains in training efficiency. Our analysis enables practitioners to understand how much predictive performance may be sacrificed in exchange for substantial reductions in compute, and under what conditions such trade-offs remain favorable. Therefore, we offer practical guidance for deploying FMs in resource-constrained settings, where both efficiency and subgroup performance are critical.

Broader impact

This work provides broadly applicable insights for machine learning in healthcare beyond medical imaging FMs. We show that for both in- and out-of-distribution settings:

- Demographic adaptation introduces a trade-off across utility, disparity, and compute: it can improve within-group utility for majority populations, maintain performance for minority groups, and (when using in-domain pretraining) better preserve performance even for rare demographics, all while achieving meaningful compute savings compared to pooled adaptation.
- The shift in pretraining domain impacts the adaptation outcome: in-domain pretrained models yield more stable and consistent gains with smallest utility performance differences between pooled and demographic adaptation, whereas out-of-domain models lead to greater instability and performance degradation, particularly for the smallest of underrepresented groups.
- Under data-scarce regimes, although demographic adaptation can improve performance for some minority groups, it often increases disparities and the risk of overfitting, making higher-capacity parameter-efficient methods better suited to preserve utility.
- Differences across adaptation strategies are primarily driven by their parameter capacity and the extent of parameter updates: Full offers the highest capacity but is computationally expensive and more prone to overfitting, while Linear represents the opposite extreme with a limited parameter budget that bounds its utility across both chest X-ray and retinal datasets; in contrast, prompt-based PEFT methods enable efficient adaptation while preserving pretrained knowledge, leading to more favorable outcomes in both utility and disparity.

- Among the adaptation methods, E^2VPT consistently delivers the strongest utility, reduces disparity and mitigates under-diagnosis, and most effectively matches or exceeds pooled adaptation across varying levels of demographic imbalance.

Together, these findings deepen our understanding of how model pretraining, adaptation, data distribution, and resource constraints interact in healthcare settings, offering principled guidance for developing machine learning systems that are both efficient and equitable across diverse patient populations.

Chapter 4

Low-Rank Adaptation with Sinusoidal Projection Sampling (LoRA-SPS)

4.1 Introduction

Adapting large pretrained models under tight parameter budgets is a deployment constraint, as frozen base models serve many tasks. Low-rank adaptation (LoRA) [32] is a dominant parameter-efficient fine-tuning (PEFT) method. However, LoRA is structurally limited where it is expected to operate: (a) its update is a linear low-rank product [33, 38] whose stable rank is capped by the fixed parameter cost, and (b) the feasible rank of low-rank adapters is tied to the embedding dimension [35, 39]. These limits produce an accuracy-parameter trade-off, visible in fig. 4.1: shrinking the trainable-parameter budget directly reduces accuracy.

Can we overcome LoRA’s dependence on the embedding dimension, thereby enabling higher feasible adapter rank under a similar parameter budget, as well as its learned-from-scratch adapter projections and limited expressiveness, while retaining zero-overhead inference?

In this chapter, we introduce Low-Rank Adaptation with Sinusoidal Projection Sampling (LoRA-SPS), an adaptation method that simultaneously (i) derives projection matrices from the frozen pretrained weights, (ii) decouples the trainable-parameter count from the embedding dimension, enabling a higher feasible adapter rank under a similar parameter budget, (iii) enriches the effective spectrum of the induced adapter projections through nonlinear sinusoidal transformations, and (iv) preserves LoRA’s mergeable, zero-overhead inference. We target structural limits of standard LoRA that no prior low-rank adapter relaxes *in combination*. We derive the projection matrices by sampling frozen pretrained weights as an informative basis for the update. The sampled weights are transformed via compact sinusoidal frequency–amplitude projections, increasing the feasible rank under the same parameter cost. The trainable-parameter count is set by the rank and the sinusoidal dimension, making it independent of the embedding dimension. Although LoRA-SPS uses a nonlinear sinusoidal transformation to construct the adapter factors, the resulting update is still a static additive matrix that is shape-compatible with the frozen weight matrix. Therefore, after training, the update can

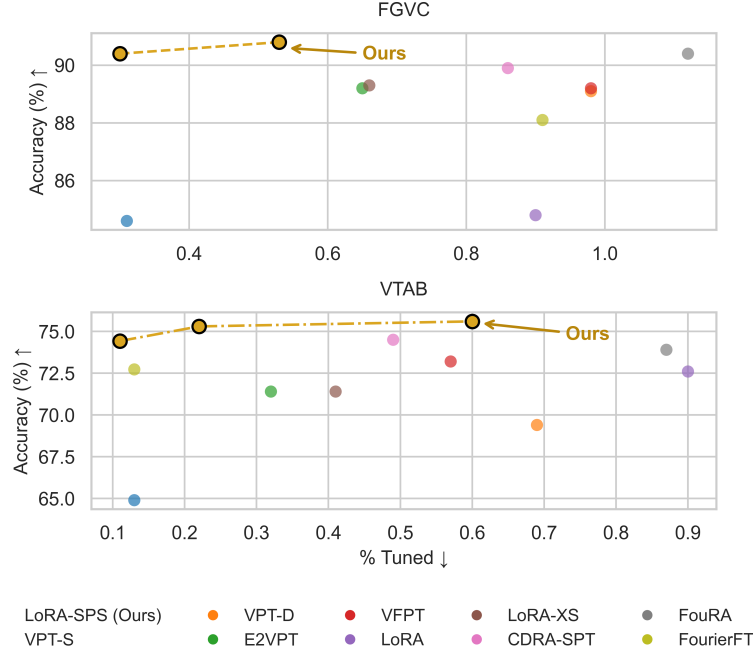


Figure 4.1: **Overview of LoRA-SPS.** Parameter–accuracy comparison on FGVC and VTAB-1K using reported results from the evaluated state-of-the-art PEFT methods. LoRA-SPS (gold) provides a strong trade-off between accuracy and trainable-parameter count.

be folded into the base weights, preserving zero-overhead inference. Applied without architecture-specific modifications, LoRA-SPS scales smoothly across tasks and matches or improves over baselines while using substantially fewer trainable parameters (fig. 4.1).

4.2 Related Work

Table 4.1 summarizes representative PEFT properties versus LoRA-SPS across the categories below.

Reparameterization-based adaptation. LoRA [32] factorizes weight updates into low-rank matrices, reducing fine-tuning cost while preserving zero-overhead inference. Subsequent works improve stability [76, 77], integrate quantization [78, 79], compress or share parameters (LoRA-XS [35], VeRA [80]), or extend adaptability through higher-rank or dynamic allocation (MoRA [81], CDRA-SPT [37]). These methods all rely on linear low-rank products, capping the stable rank at a fixed parameter count and tying the trainable-parameter count to the embedding dimension. In contrast, LoRA-XS uses frozen SVD-derived projections with only a small middle matrix trainable. Thus, LoRA-XS decouples trainable parameters from embedding size, but its full adapter footprint still scales with layer dimensions. LoRA-SPS (Ours) retains this mergeable, zero-overhead inference while breaking both structural limits.

Spectral and non-linear adaptation. Spectral methods (FourierFT [39], FouRA [53]) learn adapters in the Fourier basis; FreqFiT [54], F-Adapter [82], and WST [83] use frequency- or wavelet- modules for global or localized structure. Inspired by Implicit Neural Representations [84, 52, 55], LoRAN [38] and

Table 4.1: **Feature comparison.** *Mergeable*: folds into frozen weights. *Decoupled emb.* m, n : parameter count is independent of embedding dimension. *Non-linear*: enriches the effective spectrum at fixed parameter cost. *Pretrained basis*: derived from frozen pretrained weights W . *Trainable A*: mixes sinusoidal features via a learnable amplitude matrix. Bottom row highlights the unique combination achieved by LoRA-SPS.

Method	Mergeable	Decoupled embedding dim. m, n	Non-linear update	Pretrained basis	A
LoRA [ICLR22] [32]	✓	✗	✗	✗	✗
SPT-LoRA [ICCV23] [36]	✓	✗	✗	✗	✗
CDRA-SPT [CVPR25] [37]	✓	✗	✗	✗	✗
LoRA-XS [ECAI25] [35]	✓	✓	✗	✓	✗
FourierFT [ICML24] [39]	✓	✓	✗	✗	✗
FouRA [NeurIPS24] [53]	✓	✓	✗	✗	✗
Sine-LoRA [ICLR25] [33]	✓	✗	✓	✗	✗
VPT [ECCV22] [31]	✗	✗	✗	✗	✗
GVPT [ICML23] [50]	✗	✗	✗	✗	✗
E ² VPT [ICCV23] [51]	✗	✗	✗	✗	✗
VFPT [NeurIPS24] [34]	✗	✗	✓	✗	✗
LoRA-SPS (ours)	✓	✓	✓	✓	✓

Sine-LoRA [33] use sinusoidal activations to increase the stable rank and expressiveness, with extensions to quantization-robust settings [85]. Unlike these methods, which primarily apply nonlinearities to learned-from-scratch factors, tying the parameter count to the embedding dimension. LoRA-SPS applies sinusoidal transformations to sampled pretrained weights and learns amplitude mixing on top of this basis. This changes the parameterization regime (decoupling the parameter count from the embedding dimension) rather than only changing the adapter basis.

Prompt-based adaptation. Originating in NLP [86, 87, 88], prompt-based methods adapt frozen models via lightweight learnable tokens at the input or feature level. In vision, VPT [31] and its variants E²VPT [51], VFPT [34], and GVPT [50] enhance expressiveness through architectural or spectral extensions. Because the backbone weights remain frozen, prompt-based methods cannot induce new feature interactions inside the model and lag on tasks that require substantial weight-level adaptation (fig. 4.1). LoRA-SPS, like other reparameterization methods, modifies the weight matrices themselves while remaining mergeable into the base model at inference.

4.3 Methodology

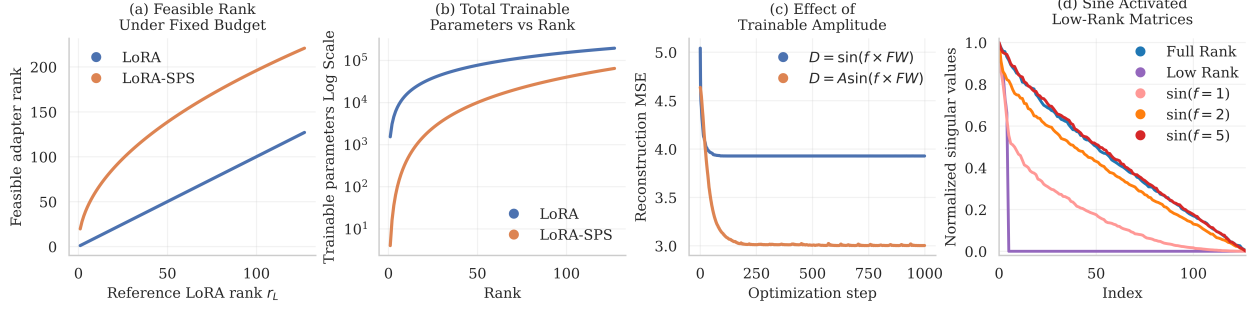


Figure 4.2: **Core LoRA-SPS properties visualized** (a) Feasible rank under a fixed parameter budget. (b) Trainable parameters vs. rank. (c) Equal-parameter (256) reconstruction with/without trainable amplitudes. (d) Singular value spectra of full-, low-, and sine-activated low-rank matrices.

Table 4.2: **Core properties/mechanisms of LoRA-SPS.**

LoRA-SPS Property	Mechanism Description
(a) Decoupled emb. dimension m, n fig. 4.2.a,b	Trainable parameters scale with the adapter rank r and sinusoidal dimension d , rather than directly with m, n ; it allows a larger feasible rank under the same budget.
(b) Pretrained weight-derived basis	Projection matrices $\mathbf{U}, \mathbf{D}$ are generated from sampled subsets of the frozen pretrained weight $\mathbf{W}_o$, rather than learned directly from scratch.
(c) Trainable amplitude mixing $\mathbf{A}$ fig. 4.2.c	The amplitude matrix learns how to combine sine-activated components into the final adapter projections, improving the quality of the induced updates.
(d) Sine-enriched effective spectrum fig. 4.2.d	Sinusoidal projection raises stable rank relative to the linear projection baseline, as reflected by a fuller singular-value spectrum, without adding trainable parameters.
(e) Architecture-agnostic	LoRA-SPS replaces LoRA’s $\mathbf{UD}$ with the construction above, while leaving the rest of the model unchanged; thus, serving as a drop-in replacement for LoRA.

4.3.1 Decoupled Low-Rank Adaptation via Weight Sampling

Inspired by [39, 35] that reduce trainable-parameter count by anchoring adaptation to a fixed representation, we address LoRA’s constraint that the trainable-parameter count scales as $r(m + n)$ by instead anchoring adaptation to frozen bases sampled from $\mathbf{W}_o$ (table 4.2(a,b), fig. 4.2(a,b)). We use compact trainable matrices $\mathbf{F} \in \mathbb{R}^{d \times r}$, where d is the sinusoidal dimension, to modulate these frozen bases. For each adapted $\mathbf{W}_o$, we construct two frozen sampled bases using a sampling operator S :

$$\mathcal{W}_D = S(\mathbf{W}_o; r) \in \mathbb{R}^{r \times n}, \quad \mathcal{W}_U = S(\mathbf{W}_o^\top; r) \in \mathbb{R}^{r \times m}. \quad (4.1)$$

Here, S constructs the compact pretrained basis of rank r . Thus, $\mathcal{W}_D$ and $\mathcal{W}_U$ sample rows and columns of $\mathbf{W}_o$, respectively. Both $\mathcal{W}_D$ and $\mathcal{W}_U$ are frozen and used to generate $\mathbf{D}$ and $\mathbf{U}^\top$, respectively. This produces intermediate features of the form $\mathbf{F}\mathcal{W}_D$ and $\mathbf{F}\mathcal{W}_U$, which are then transformed to construct the adapter factors $\mathbf{D}$ and $\mathbf{U}$. For simplicity, we drop the subscript and use $\mathbf{F}\mathcal{W}$, unless needed.

We evaluate four sampling strategies: selecting the first r rows (*top rows*, eq. (4.2)); sampling rows at fixed intervals (*stride*, eq. (4.3)); selecting rows with highest variance (*top variance*, eq. (4.4)); and a variance-maximized selection (*SVD-based*, eq. (4.5)). Results are largely insensitive to the choice of sampling strategy (see table 4.16).

$$S_{\text{top}}(\mathbf{W}_o, r) = \{w_i \in \mathbf{W}_o \mid i \leq r\} \quad (4.2)$$

$$S_{\text{stride}}(\mathbf{W}_o, r) = \{w_i \in \mathbf{W}_o \mid i \bmod \lfloor k/r \rfloor = 0\}_{[r]}, \quad k \in \{n, m\} \quad (4.3)$$

$$S_{\text{var}}(\mathbf{W}_o, r) = \{w_i \in \mathbf{W}_o \mid \text{Var}(w_i) \text{ is among the top } r\} \quad (4.4)$$

$$S_{\text{svd}}(\mathbf{W}_o, r) = \Sigma_{[r,:]} \mathbf{V}_{[r]}^\top, \quad \mathbf{W}_o = \mathbf{U} \Sigma \mathbf{V}^\top \quad (4.5)$$

4.3.2 Non-Linear Transformation & Amplitude Mixing

Inspired by [38, 33], where adding nonlinearity raises stable rank, we introduce the non-linear projection $\mathbf{Y} = \sum_{p \in P} p(f \mathbf{F} \mathcal{W})$, $P \subseteq \{\sin, \cos\}$, where f is a fixed frequency scaling factor. Figure 4.2.c visualizes the amplitude-mixing property (table 4.2.c), comparing reconstruction with and without trainable amplitudes at a fixed parameter budget. Figure 4.2.d visualizes the sine-enriched effective-spectrum property (table 4.2.d), contrasting a linear sampled-weight projection against sinusoidal projections at different frequencies. The sinusoidal transformation broadens the induced singular-value spectrum, raising the stable rank and the effective expressiveness of the adapter factors at fixed trainable-parameter cost.

Although this enriches the feature space, the bounded sinusoidal range limits feature reweighting, and each row of F receives gradient signal only from its corresponding output coordinate. We address both issues by introducing a trainable amplitude matrix $\mathbf{A} \in \mathbb{R}^{r \times d}$, corresponding to the amplitude-mixing property in table 4.2.c and visualized in fig. 4.2.c. The matrix $\mathbf{A}$ learns how to rescale and combine the sine-activated components before forming the final adapter projection, improving the expressiveness of the induced update under the same parameter budget. In addition, $\mathbf{A}$ enables gradients from different output coordinates to be mixed across sinusoidal features, rather than restricting each row of $\mathbf{F}$ to receive signal only from its corresponding output coordinate:

$$\mathbf{Y} = \mathbf{A} \sum_{p \in P} p(f \mathbf{F} \mathcal{W}). \quad (4.6)$$

Matrix $\mathbf{A}$ both rescales sinusoidal features and learns how output-gradient signals should be routed to frequency components.

For notational simplicity, we present the analysis for a single column of $\mathcal{W}$ and the square case $d = r$; the full matrix case follows column-wise.

Proposition 1 (Gradient Aggregation via $\mathbf{A}$). *Let $\delta = \partial \mathcal{L} / \partial \mathbf{y} \in \mathbb{R}^r$. In the baseline $\mathbf{y} = \sin(f \mathbf{F} \mathcal{W})$, the i -th row gradient of $\mathbf{F}$, $\mathbf{f}_i$ is scaled only by δ_i . In the amplitude-mixed model eq. (4.6), the gradient of row $\mathbf{f}_j$ is scaled by $(\mathbf{A}^\top \delta)_j$, a learned linear combination of all output gradients.*

Proof. Assuming $P = \{\sin\}$, the baseline and amplitude-mixed parameterizations give:

$$\begin{aligned} \text{baseline: } y_i &= \sin\left(f \mathbf{f}_i^\top \mathcal{W}\right), & \nabla_{\mathbf{f}_i} \mathcal{L} &= \delta_i f \cos\left(f \mathbf{f}_i^\top \mathcal{W}\right) \mathcal{W}, \\ \text{mixed: } y_i &= \sum_{j=1}^d A_{ij} \sin\left(f \mathbf{f}_j^\top \mathcal{W}\right), & \nabla_{\mathbf{f}_j} \mathcal{L} &= \sum_{i=1}^r \delta_i A_{ij} f \cos\left(f \mathbf{f}_j^\top \mathcal{W}\right) \mathcal{W}. \end{aligned}$$

The mixed gradient simplifies to eq. (4.7). In the baseline, row $\mathbf{f}_i$ is scaled by the single coordinate δ_i . In the mixed model, row $\mathbf{f}_j$ is scaled by $\mathbf{A}^\top \boldsymbol{\delta}$, a learned linear combination of all output-gradient coordinates. Thus $\mathbf{A}^\top$ implements a trainable map from output-gradient space to frequency-update space, strictly generalizing the coordinate-wise routing structure of the baseline.

$$\nabla_{\mathbf{f}_j} \mathcal{L} = \left(\mathbf{A}^\top \boldsymbol{\delta}\right)_j f \cos\left(f \mathbf{f}_j^\top \mathcal{W}\right) \mathcal{W}, \quad \left(\mathbf{A}^\top \boldsymbol{\delta}\right)_j = \sum_{i=1}^r A_{ij} \delta_i. \quad (4.7)$$

Implications. Figure 4.2(c) shows that, under an equal parameter budget, trainable amplitude mixing improves reconstruction quality compared with sinusoidal components without learned amplitudes. This suggests that A is not merely a scalar rescaling term, but a local trainable mechanism for combining sine-activated basis components inside the LoRA-SPS adapter projection. Proposition 1 supports this interpretation from the optimization perspective: within this local parameterization, the update to each mixed component depends on $(\mathbf{A}^\top \boldsymbol{\delta})_j$, allowing output-gradient signals to be redistributed across sinusoidal basis components before forming the projection. Thus, amplitude mixing helps LoRA-SPS use its sinusoidal basis more effectively under the same trainable-parameter budget.

4.4 LoRA-SPS

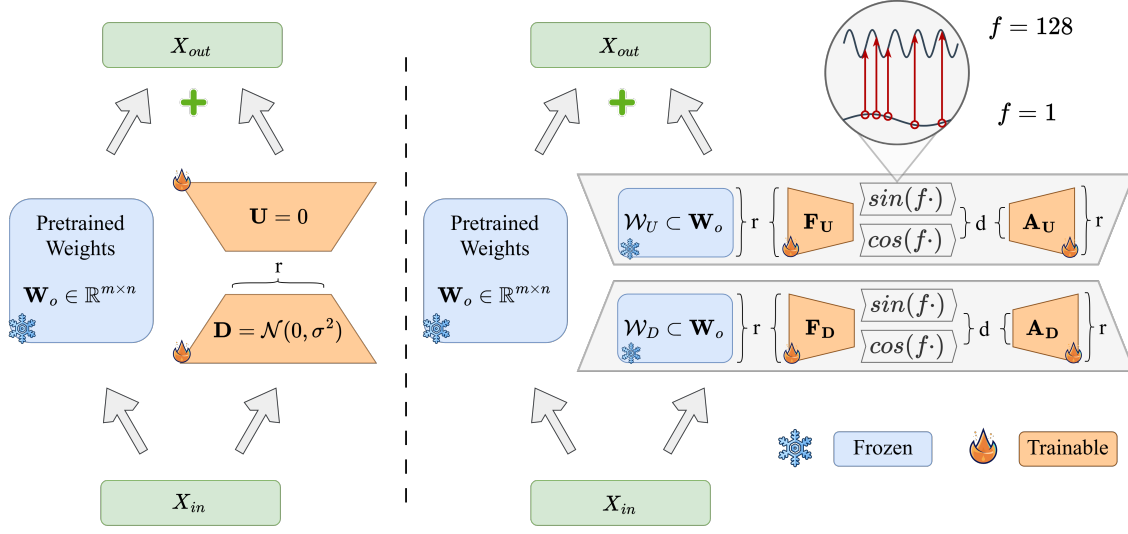


Figure 4.3: **Comparing LoRA (left) and LoRA-SPS (right).** LoRA factorizes the update into down- and up-projections $\mathbf{D}, \mathbf{U}$. LoRA-SPS instead derives both from small subsets of pretrained weights (section 4.3.1): each subset is processed by a compact trainable $\mathbf{F}_\mathbf{D}$, transformed via a sinusoidal projection, and reweighted by an amplitude matrix $\mathbf{A}$. The same procedure yields $\mathbf{U}$ (eq. (4.8)); Remaining updates follow LoRA. **Orange**: trainable parameters; **blue**: frozen pretrained weights.

We generate $\mathbf{D}$ by applying the stride sampling strategy in eq. (4.3) to obtain the sampled weights eq. (4.1), followed by eq. (4.6), using the corresponding frequency $\mathbf{F}_\mathbf{D}$ that expands sampled pretrained weights into a d -dimensional sinusoidal feature space and amplitude $\mathbf{A}_\mathbf{D}$ that mixes these features back to rank r . Similarly, we generate $\mathbf{U}$ using $\mathbf{F}_\mathbf{U}$ and $\mathbf{A}_\mathbf{U}$. The formulation for down/up-projections $\mathbf{D}$ and $\mathbf{U}$ is eq. (4.8). fig. 4.3 illustrates the pipeline, with the full algorithm in the following sections, see algorithm 1. Figure 4.4 compares pretrained weights, LoRA, and LoRA-SPS across parameter budgets and frequencies: while LoRA shows limited structure, LoRA-SPS preserves and reuses pretrained weight patterns under tight budgets, with expressiveness amplified by higher frequency or parameter count.

$$\mathbf{D} = \mathbf{A}_\mathbf{D} \sum_{p \in P} p(f \mathbf{F}_\mathbf{D} \mathcal{W}_\mathbf{D}) \frac{1}{\sqrt{r} \sqrt{d}}, \quad \mathbf{U}^\top = \mathbf{A}_\mathbf{U} \sum_{p \in P} p(f \mathbf{F}_\mathbf{U} \mathcal{W}_\mathbf{U}) \frac{1}{\sqrt{r} \sqrt{d}}. \quad (4.8)$$

4.4.1 Rank-Efficient Budget Allocation

For a square pretrained layer $\mathbf{W}_o \in \mathbb{R}^{n \times n}$, standard LoRA with rank r_L uses $P_L = 2r_L n$ trainable parameters. For LoRA-SPS at the common configuration $d = r$, the four matrices $\mathbf{F}_\mathbf{D}, \mathbf{F}_\mathbf{U}, \mathbf{A}_\mathbf{D}, \mathbf{A}_\mathbf{U} \in \mathbb{R}^{r \times r}$ contribute $P_{\text{SPS}} = 4r^2$ parameters. (i) At equal adapter rank ($r = r_L$), $P_{\text{SPS}}/P_L = 2r/n$, LoRA-SPS is smaller than LoRA in the typical low-rank regime $r \ll n$. (ii) Under equal parameter budget ($P_{\text{SPS}} = P_L$), solving $4r^2 = 2r_L n$ gives $r = \sqrt{r_L n/2}$, which strictly exceeds r_L whenever $n > 2r_L$. Hence, in the $d = r$ setting, LoRA-SPS either reduces the trainable parameter count at fixed adapter rank or supports a higher feasible rank at fixed

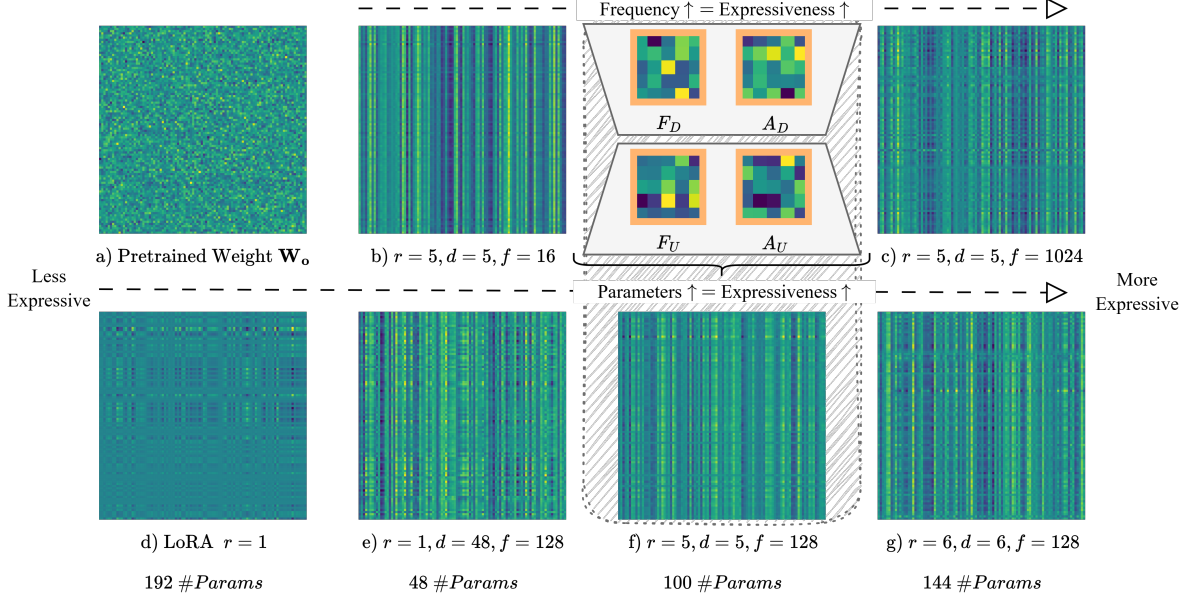


Figure 4.4: **LoRA-SPS expressiveness** on MNIST. Weight updates compared to LoRA and pretrained weights, varying frequency (top) and parameter count (bottom). Unlike LoRA’s limited-structure updates, LoRA-SPS reuses pretrained-weight patterns; expressiveness grows with both frequency and parameter count. Details: section 4.5.3.1.

parameter budget. While $d = r$ is the default and most parameter-efficient setting, LoRA-SPS also permits a tradeoff between sinusoidal dimension d and adapter rank r , enabling flexible budget allocation, see fig. 4.8.

4.4.2 Intuition and Design Rationale

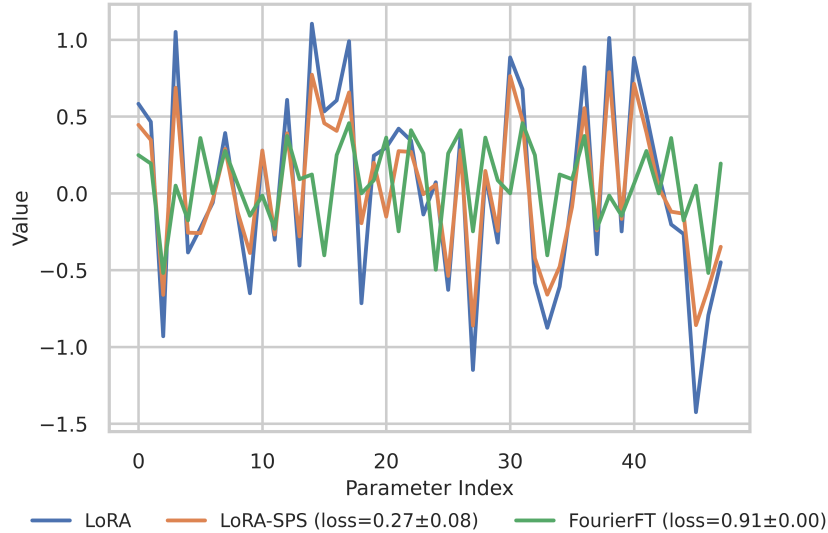


Figure 4.5: **Signal reconstruction.** We reconstruct a rank-1 LoRA weight update (dim. 48) on MNIST [8] using 50% of parameters. LoRA-SPS achieves a combined MSE+cosine loss of 0.27 ± 0.08 vs. 0.91 ± 0.00 for FourierFT, indicating higher effective rank per parameter. Details are provided in section 4.5.3.1.

Beyond the properties shown in fig. 4.2 and table 4.2, fig. 4.5 further illustrates that LoRA-SPS is able to reconstruct a rank-1 LoRA update on MNIST [8] with over $3\times$ lower error than FourierFT when evaluated using a combined MSE and cosine-similarity loss. This suggests more faithful reconstruction and improved exploitation of pretrained weight structure under identical parameter budgets.

The design of LoRA-SPS follows the mechanisms in table 4.2. Sampling controls adapter size independently of embedding dimension, avoiding proportional scaling with layer width. Because the sampled directions come from the frozen pretrained weights, the update is also anchored in representations already learned by the base model. Sinusoidal projection nonlinearly transforms sampled directions, enriching the effective spectrum of the adapter factors without requiring dense trainable adapters. The amplitude matrix learns how strongly each transformed direction should contribute and how these features should be mixed for the downstream task. Together, these components yield a LoRA-compatible adapter whose parameter count is controlled by sampling and frequency size, with updates grounded in the pretrained weight structure.

4.4.3 Algorithm

Algorithm 1 LoRA-SPS PyTorch-Style Pseudocode

Require: Linear layer $layer$, alpha a , rank r , sinusoidal dimension d , dropout $dropout$, frequency scale f

```
class LoRA_SPS(Linear):
    def __init__(self, layer, a, r, d, dropout, f):
        super(LoRA_SPS, self).__init__()
        self.copy(layer) # Copy weights, and bias
        self.a, self.r, self.d, self.f = a, r, d, f
        self.norm = sqrt(self.d) * sqrt(self.r)
        self.DA = self.reset(self.r, self.d)
        self.DF = self.reset(self.d, self.r)
        self.UA = self.reset(self.r, self.d, 10.0)
        self.UF = self.reset(self.d, self.r, 0.1)
        self.dropout = Dropout(dropout)
        self.sample ∈ {TopRows, Stride, TopVar, SVD}
        self.projections ⊆ {cos, sin}

    def reset(self, d1, d2, scale = 1):
        W = nn.Parameter(torch.empty(d1, d2))
        nn.init.xavier_normal_(W)
        return W × scale

    def compute(self, x, W, freq, ampl, transpose):
        SW = self.sample(W)
        proj = (self.f × freq) @ SW
        proj = sum(f(proj) for f in self.projections)
        NW = (ampl @ proj) / self.norm
        if transpose:
            return F.linear(x, NW.T, self.bias)
        return F.linear(x, NW, self.bias)

    def forward(self, x):
        x_orig = F.linear(x, self.W, self.bias)
        x = self.compute(x, W, self.DF, self.DA, False)
        x = self.dropout(x)
        x = self.compute(x, W.T, self.UF, self.UA, True)
        return x_orig + (self.a × x)
```

4.5 Datasets

4.5.1 Vision Datasets

4.5.1.1 MNIST & KMNIST

MNIST [8] is a handwritten digit classification dataset consisting of 60,000 training images and 10,000 test images, featuring grayscale digits from 0–9. It serves as a foundational benchmark for evaluating image classification models. KMNIST [89] is similar to MNIST, but composed of grayscale images of Japanese Kuzushiji characters. It matches MNIST in size and format, providing a more challenging benchmark while retaining the same experimental setup.

4.5.1.2 VTAB-1K

The Visual Task Adaptation Benchmark (VTAB-1K) [90] consists of 19 datasets grouped into three categories. Natural datasets include CIFAR100 [91], Caltech101 [92], DTD [93], Flowers102 [94], Pets [95], SVHN [96], and SUN397 [97]. Specialized datasets comprise Camelyon [98], EuroSAT [99], RESISC45 [100], and Retinopathy [101]. Structured datasets contain Clevr-Count [102], Clevr-Dist [102], DMLab [103], KITTI-Dist [104], dSpr-Loc [105], dSpr-Ori [105], sNORB-Azim [106], and sNORB-Ele [106].

From each dataset, 800 images were sampled for training and 200 for validation, while leaving the test set unchanged. For validation, we follow the standard VTAB-1K split, using 800 images for training and 200 images for validation during hyperparameter selection. After selecting the hyperparameters, we combine the training and validation subsets into a single 1,000-image training set and evaluate the final model on the full test set. Details for reproducing the VTAB setup are provided in the original VPT protocol [31, 90]. Performance is measured using top-1 accuracy, and shown per dataset, accompanied by the category-wise averages for Natural, Specialized, and Structured groups. The overall mean accuracy is computed across all individual datasets. Raw images are obtained from previous work [107].

4.5.1.3 FGVC

The fine-grained visual classification (FGVC) benchmark consists of five datasets in natural image domain: CUB-2011 [108], NABirds [109], Oxford Flowers 102 [110], Stanford Dogs [111], and Stanford Cars [112]. See individual papers for dataset details. Performance is measured using top-1 accuracy, shown per dataset, and accompanied by overall mean accuracy computed across all individual datasets.

4.5.1.4 ADE20K

ADE20K [113] is a large-scale scene parsing dataset that provides densely annotated images spanning a broad range of everyday environments. It contains 20K training images, 2K validation images, and 3K test images, each labeled with both object and stuff categories at the pixel level. The dataset covers diverse scenes, from indoor environments to complex outdoor settings, making it a standard benchmark for semantic segmentation and scene understanding tasks. For details on dataset structure and annotation protocols, please

refer to the original ADE20K documentation. Semantic segmentation performance is assessed using two evaluation strategies, quantified by mean Intersection-over-Union (mIoU) under single-scale (s.s.) and multi-scale (m.s.) conditions. Single-scale results correspond to the original resolution, while multi-scale results aggregate outputs from multiple input sizes for enhanced robustness. As done in previous works, evaluation is performed using the validation set [31, 36].

4.5.1.5 DreamBooth

DreamBooth [114] is a subject-driven text-to-image personalization benchmark built on small per-subject image collections and prompt-based evaluation. In our setup, we use the five standard subjects (*dog*, *cat*, *backpack*, *berry bowl*, and *can*) from the public DreamBooth release and follow the same subject-specific prompt templates used in prior diffusion PEFT comparisons. For each subject, we train on the corresponding DreamBooth training images and evaluate generation quality on a fixed grid of 25 prompts with 4 images per prompt (100 generated images per method/subject). Evaluation uses the reference images from the same subject folder and reports DINO subject fidelity, CLIP image-image and text-image alignment, LPIPS diversity, and FID, with per-subject scores averaged to obtain the main table means.

4.5.2 Language Datasets

4.5.2.1 GLUE

The General Language Understanding Evaluation (GLUE) benchmark [5] is a collection of diverse natural-language understanding (NLU) tasks designed to evaluate models’ generalization across syntax, semantics, inference, and commonsense reasoning. It aggregates datasets spanning acceptability judgments, sentiment, paraphrase detection, natural-language inference, and question answering. See individual works for details. We adhere to the FourierFT [39] protocol, using the validation set for evaluation, and exclude the MNLI [115], WNLI [116], and QQP [5] datasets from our study.

GLUE includes:

- CoLA: Corpus of Linguistic Acceptability [117].
- SST-2: Stanford Sentiment Treebank [118].
- MRPC: Microsoft Research Paraphrase Corpus [119].
- STS-B: Semantic Textual Similarity Benchmark [120].
- QNLI: Question Natural-Language Inference [121].
- RTE: Recognizing Textual Entailment [122, 123, 124, 125].

Together, these tasks test core NLU capabilities and support unified evaluation of modern language models. We evaluate this benchmark’s dataset CoLA using Matthew’s correlation, STS-B using Pearson’s correlation, and all others using accuracy.

Table 4.3: **Hyperparameters for demonstrative example.** Configuration of the 3-layer ViT, LoRA & LoRA-SPS fine-tuning setup used in the two-phase demonstrative example.

Parameter	Value	Parameter	Value	Parameter	Value
Image size	32×32	LoRA rank	1	Target modules	query, value
Patch size	16	LoRA-SPS rank	4, 5, 6	Optimizer	AdamW
Hidden size	96	d	4, 5, 6, 7	Learning rate	10^{-3}
Attention heads	3	f	16, 128, 1024	Weight decay	10^{-4}
MLP dimension	192	α	1	Pretrain epochs	10
Hidden layers	3	dropout	0.1	Fine-tune epochs	10

4.5.3 LoRA-SPS Setup

4.5.3.1 Demonstrative Example

The objective of this experiment was to explore PEFT on a lightweight ViT architecture and to visualize how weight structures evolve under LoRA & LoRA-SPS, seen in fig. 4.4, fig. 4.5. We employed a compact 3-layer ViT trained first on the KMNIST dataset for pretraining, followed by fine-tuning on MNIST to examine low-rank adaptation behavior. A small-scale configuration was chosen to facilitate interpretability and efficient computation. The model and training hyperparameters are summarized in table 4.3. To provide a clearer visualization of the reconstruction, we use a hidden size of 48 instead of 96, and perform the reconstruction using only half the number of parameters.

4.5.3.2 Parameter Initialization

Table 4.4: **Comparison of different initialization methods on VTAB–Caltech101.** Each of s_1, s_2, s_3, s_4 is scaled to 1. We report Top-1 accuracy, with stability measured by how often the achieved accuracy stays above 90.0%. The chosen method is highlighted in **bold**.

Initialization Method	Accuracy	Stability Count
Normal	91.2	11
Uniform	91.3	12
Kaiming Normal	91.0	13
Kaiming Uniform	91.2	14
Xavier Normal	91.2	15
Xavier Uniform	90.9	11

We first conducted an extensive search on VTAB-1K–Caltech101 over different initialization methods, including Normal, Uniform, Kaiming Normal [126], Kaiming Uniform [126], Xavier Normal [127], and

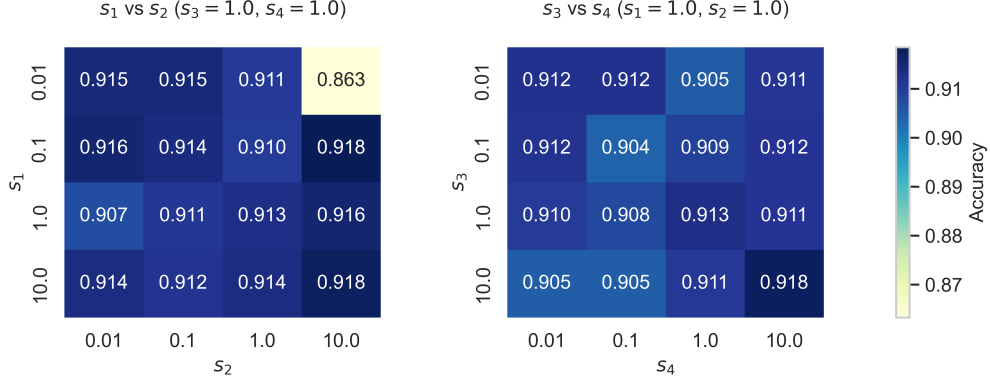


Figure 4.6: **Comparison of different scaling factors** for Xavier Normal initialization on VTAB-1K–Caltech101. Each of s_1, s_2, s_3, s_4 is scaled by 0.01, 0.1, 1.0, or 10.0. The left heatmap shows results when s_1, s_2 are varied while s_3, s_4 are fixed at a scale of 1, whereas the right heatmap shows results when s_3, s_4 are varied with s_1, s_2 fixed at a scale of 1. Colors indicate top-1 accuracy. The selected configuration is $s_1 = 1.0, s_2 = 1.0, s_3 = 0.1, s_4 = 10.0$.

Xavier Uniform [127], to determine the most effective initialization for our learnable weights table 4.4. We kept the rest of the hyperparameters frozen: rank $r = 32$, sinusoidal dimension $d = 48$, frequency $f = 256$, and $\alpha = 0.1$.

We further explore the effect of scaling factors s_1, s_2, s_3, s_4 for parameters F_D, A_D, F_U, A_U eqs. (4.9) to (4.12) from [127] (where n and m represent input and output feature dimensions) across a wide range of values: $s \in \{0.01, 0.1, 1.0, 10.0\}$ in fig. 4.6.

$$F_D \sim \mathcal{N}\left(-s_1 \sqrt{\frac{6}{n+m}}, s_1 \sqrt{\frac{6}{n+m}}\right) \quad (4.9)$$

$$A_D \sim \mathcal{N}\left(-s_2 \sqrt{\frac{6}{n+m}}, s_2 \sqrt{\frac{6}{n+m}}\right) \quad (4.10)$$

$$F_U \sim \mathcal{N}\left(-s_3 \sqrt{\frac{6}{n+m}}, s_3 \sqrt{\frac{6}{n+m}}\right) \quad (4.11)$$

$$A_U \sim \mathcal{N}\left(-s_4 \sqrt{\frac{6}{n+m}}, s_4 \sqrt{\frac{6}{n+m}}\right) \quad (4.12)$$

Through this systematic evaluation, we selected the configuration $s_1 = 0.1, s_2 = 10.0, s_3 = 1.0, s_4 = 1.0$, which achieved the best performance in terms of accuracy on VTAB-1K–Caltech101. This specific initialization and scaling configuration was subsequently employed for all experiments reported in this study, ensuring consistency and reproducibility across our evaluations.

4.5.3.3 Hyperparameters for VTAB-1K & FGVC Benchmarks

To adapt LoRA-SPS (Efficient) we perform a subset search of ranks and sinusoidal dimensions in table 4.5, with a threshold of 768 maximum number of parameters per layer. For LoRA-SPS (Best), we perform a

Table 4.5: **Summary of the hyperparameters** used in LoRA-SPS search for VTAB-1K & FGVC.

Parameter	Value	Parameter	Value	Parameter	Value
r	2, 4, 8, 16, 32, 64, 96	α	0.01, 0.1, 1.0	Projection	cos
d	12, 24, 48, 96	dropout	0.1	Optimizer	AdamW
f	128, 256, 512, 1024	Target modules	All Linear	Learning rate	10^{-3}
Epochs	100	Weight decay	10^{-4}	LR Schedule	Cosine

complete search.

4.5.3.4 Hyperparameters for ADE20K Semantic Segmentation

When adapting ViT-L [1] backbone, specifically SETR-PUP [2], using LoRA-SPS to ADE20K [113], we follow prior work [31, 36] and employ the MMSegmentation [128] code base. Training is performed using AdamW with a learning rate of 10^{-4} for both the LoRA-SPS modules and the classifier, a weight decay of 0.05, rank $r = 32$, and a sinusoidal dimension of $d = 64$ with a frequency scale of $f = 1024$. The only hyperparameter we varied was the scaling factor $\alpha \in \{0.01, 0.1, 1.0\}$.

4.5.3.5 Hyperparameters for DreamBooth Text-to-Image Generation

Unless noted otherwise, all DreamBooth experiments fine-tune Stable Diffusion v1.5 with resolution 512, AdamW on adapter parameters ($\beta=(0.9, 0.999)$, weight decay 10^{-2}), cosine learning-rate decay, learning rate 10^{-4} , batch size 1, gradient accumulation 4, and mixed precision fp16 for adapter methods (FourierFT does not use mixed precision). Target modules are the UNet attention projections. For the quantitative comparisons we train baselines and LoRA-SPS sweeps for **800 optimization steps** with final checkpoints saved at step 800; Full / Linear UNet fine-tuning follow the same recipe except all (or all linear) UNet weights are trainable. At inference, we generate 25 prompts $\times$ 4 images per subject (50 diffusion steps, classifier-free guidance 7.5). table 4.6 lists hyperparameters used in LoRA-SPS’s search.

For LoRA-SPS we search over the rank and the sinusoidal feature dimension, tied as $r=d$, the scaling α , and the frequency scale f . LoRA-SPS (Efficient) model uses $r=d=16$, $\alpha=0.1$, $f=16384$; LoRA-SPS (Best) uses $r=d=32$, $\alpha=0.1$, $f=16384$. In this task, LoRA-SPS greatly benefits from a higher f .

Table 4.6: **Summary of the hyperparameters** used in LoRA-SPS search for the DreamBooth dataset.

Parameter	Setting
Base model	Stable Diffusion v1.5
Train steps (adapters)	800 (final checkpoint). Full / Linear tune the full resp. linear UNet with the same recipe
Resolution	512×512
Learning rate	10^{-4}
LR schedule	Cosine decay (no warmup for DreamBooth sweeps)
Optimizer	AdamW ($\beta=(0.9,0.999)$)
Weight decay	10^{-2} (adapter parameters only where applicable)
Mixed precision	fp16 for adapter methods; FourierFT runs without fp16
Batch $\times$ accum.	1×4
Eval generation	25 prompts $\times$ 4 images per subject
Sampling	50 diffusion steps, classifier-free guidance 7.5
LoRA-SPS search space	
Rank / degree $r=d$	$\{16, 24, 32, 64\}$
Scaling α	$\{0.005, 0.01, 0.05, 0.1\}$
Frequency f (n_{freq})	$\{1024, 4096, 16384\}$

4.5.3.6 Hyperparameters for GLUE Benchmark

Table 4.7: **Summary of the hyperparameters** used in LoRA-SPS for GLUE benchmark.

Hyperparameter	SST-2 (Acc.)	MRPC (Acc.)	CoLA (MCC)	QNLI (Acc.)	RTE (Acc.)	STS-B (PCC)
$r = 8, d = 24$						
- L.R. classifier	1×10^{-3}	5×10^{-4}	1×10^{-3}	5×10^{-4}	1×10^{-3}	10^{-4}
- L.R. adapter	1×10^{-3}	5×10^{-4}	1×10^{-3}	5×10^{-4}	1×10^{-3}	10^{-3}
- α	0.01	0.01	0.005	0.005	0.01	0.01
$r = d = 16$						
- L.R. classifier	10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	1×10^{-4}	10^{-4}
- L.R. adapter	5×10^{-5}	5×10^{-4}	10^{-3}	5×10^{-4}	1×10^{-3}	10^{-3}
- α	0.01	0.01	0.01	0.005	0.05	0.01

To adapt LoRA-SPS to the GLUE datasets, we follow the procedure described in [39]. We maintain a comparable number of trainable parameters by fixing both the rank and the sinusoidal dimension to 16, and by applying adaptation only to the query and value attention layers. We also attempt to examine the performance when the number of parameters is further reduced with rank 8 and the sinusoidal dimension 24. We match the maximum sequence length used in [39]. The same epoch values are adapted as in [39].

We further set the frequency scale to $f = 1024$. The remaining tuned hyperparameters are the scaling factor $\alpha \in \{0.005, 0.01, 0.05, 0.1\}$, the learning rate for the classification head $\in \{10^{-3}, 5 \times 10^{-4}, 10^{-4}\}$, and the learning rate for LoRA-SPS $\in \{10^{-3}, 5 \times 10^{-4}, 10^{-4}\}$. Individual parameters per dataset are described in table 4.7.

4.5.4 Baseline Setup

All reported results for the baseline methods are taken from the original works that introduced the corresponding methods. When a paper does not report results on a specific dataset, we cite an additional source that provides them; if no such source is available, we implement and train the baseline ourselves. Below, we describe the experimental setup used for all baselines trained in this work. When available, we use the official Hugging Face implementation; otherwise, we rely on the authors public repositories, or re-implement the method based on the description in the paper.

4.5.4.1 LoRA-XS Setup

The source code for LoRA-XS¹ [35] is publicly available on GitHub. For experiments on the VTAB-1K and FGVC datasets, we use a fixed learning rate of 10^{-3} , weight decay of 10^{-4} , and dropout rate of 0.1. We tune the rank $r \in \{4, 8, 12, 16, 20, 25, 32, 64, 128\}$ and the scaling factor $\alpha \in \{0.01, 0.1, 1.0, 10.0, 25.0, 50.0, 75.0, 100.0\}$. LoRA-XS modules are applied to all linear layers except the classifier.

4.5.4.2 FouRA Setup

FouRA [53] is implemented by following the steps outlined in the paper. When training on VTAB-1K and FGVC datasets, we used the same learning rate 10^{-3} , weight decay 10^{-4} , and dropout 0.1 throughout. We tuned the following parameters, gating dimension $\in \{1, 2, 4, 8, 17\}$, rank $r \in \{1, 2, 4, 8, 16\}$, and scaling factor $\alpha \in \{0.01, 0.1, 1.0, 10.0\}$. Similarly, FouRA is attached to all linear layers, excluding the classifier.

4.5.4.3 FourierFT Setup

The FourierFT² implementation [39] is available through the Hugging Face PEFT library [129]. For FGVC experiments, we adopt fixed hyperparameters consisting of a learning rate of 10^{-3} and a weight decay of 10^{-4} . We tune the number of frequencies ranging from 1000 to 10000 in increments of 1000 and the scaling factor $\alpha \in \{10, 50, 100, 250, 500, 750, 10000\}$. Consistent with prior setups, FourierFT modules are applied to all linear layers except the classifier.

4.5.4.4 LoRA-Sine Setup

The LoRA-Sine³ implementation [33] is available through their public repository, but is also simple enough to implement above any existing LoRA. For VTAB-1K and FGVC experiments, we adopt fixed hyperpa-

¹<https://github.com/MohammadrezaBanaei/LoRA-XS>

²https://huggingface.co/docs/peft/main/en/package_reference/fourierft

³<https://github.com/yipingji/Sine-Low-Rank>

rameters consisting of a learning rate of 10^{-3} and a weight decay of 10^{-4} . We tune rank $r \in \{1, 2, 4\}$, the number of frequencies ranging from $f \in \{100, 200, 400, 600, 800, 1000\}$ and the scaling factor $\alpha \in \{0.01, 0.1, 1.0, 10.0\}$. Consistent with prior setups, LoRA-SPS modules are applied to all linear layers except the classifier.

4.6 Experiments

4.6.1 Classification Tasks

4.6.1.1 Setup

Following [31, 34], we evaluate our method on the Visual Task Adaptation Benchmark (VTAB-1K) and fine-grained visual classification (FGVC) datasets. VTAB-1K [90] comprises 19 diverse image classification tasks grouped into three categories: Natural, Specialized, and Structured. Each task contains 1,000 training examples, allowing us to assess the adaptability of our method across a diverse set of visual tasks. The FGVC benchmark includes five widely-used datasets: CUB-2011 [108], NABirds [109], Oxford Flowers 102 [110], Stanford Dogs [111], and Stanford Cars [112].

We compare our PEFT methods on the ViT backbone [1]. Specifically, we use ViT-B with 16×16 input patches pretrained on ImageNet-21k. The pretrained backbone provides a shared visual representation for all experiments; unless otherwise noted, the backbone weights are kept frozen and only the parameter-efficient modules (prompts/adapters/low-rank updates) are trained. We adapt standard Full and Linear fine-tuning baselines. Among prompt-based approaches, we include the original VPT-Shallow (VPT-S), VPT-Deep (VPT-D) [31], E²VPT [51], and VFPT [34]. For reparameterization-based methods, we evaluate against LoRA [32], SPT-LoRA [36], FouRA [53], CDRA-SPT [37], Sine-LoRA [33], and lightweight variants such as LoRA-XS [35], and FourierFT [39]. We evaluate performance using the top-1 accuracy averaged over three runs and report the mean and standard deviation for our main results.

Following [37], we train all models using the AdamW optimizer [130]. The learning rate is linearly warmed up over the first 10 epochs and decayed via a cosine schedule for a total of 100 epochs. We fix the learning rate and weight decay at 10^{-3} and 10^{-4} , respectively, and use only standard data augmentations on VTAB-1K and FGVC as in prior work [31]. All experiments are conducted on a single NVIDIA L40S GPU.

4.6.1.2 Results

Table 4.8 reports classification accuracy on FGVC (5 datasets) and VTAB-1K (19 datasets) using a frozen ViT-B/16 [1] backbone. Following [31], we report mean top-1 accuracy. LoRA-SPS consistently matches or improves over the strongest PEFT baselines at substantially smaller *trainable-parameter counts*. Across three VTAB-1K categories, either the Efficient or Best variant is among the top two PEFT methods in terms of accuracy, while having substantially fewer parameters than competing baselines. For example, LoRA-SPS (Efficient) reaches 75.3% mean VTAB-1K accuracy at 0.22% trainable parameters, against CDRA-SPT’s 74.5% at 0.49%. On FGVC, the Efficient variant achieves 90.4% accuracy with only 0.30% trainable

Table 4.8: **Image classification** on FGVC and VTAB-1K. ViT-B/16 [1] pretrained on supervised ImageNet-21k; mean test top-1 accuracy over three runs. *% Params* is the percentage of trainable parameters relative to the backbone, the lower the better. **Bold** and underline mark best and second-best per column across PEFT methods. Per-task breakdowns in tables A.9 to A.12.

ViT-Base/16 (85.8M)	FGVC		VTAB-1K				
	% Params	Mean Acc.	% Params	Natural	Specialized	Structured	Mean Acc.
# of Tasks	5		7		4	8	
Full [CVPR22]	100	88.5	100	75.9	83.4	47.6	65.6
Linear [CVPR22]	0.21	79.3	0.044	68.9	77.2	26.8	52.9
<i>Visual Prompt Tuning</i>							
VPT-S [ECCV22]	0.31	84.6	0.13	76.8	79.7	47.0	64.9
VPT-D [ECCV22]	0.98	89.1	0.69	78.5	82.4	55.0	69.4
E ² VPT [ICCV23]	0.65	89.2	0.32	80.0	84.4	57.4	71.4
VFPT [NeurIPS24]	0.98	89.2	0.57	81.4	84.9	60.2	73.2
<i>Reparameterization-Based Adaptation</i>							
LoRA [ICLR22]	0.90	84.8	0.90	79.8	84.9	60.2	72.6
SPT-LoRA [ICCV23]	0.60	90.1	0.63	81.9	85.9	61.3	74.1
FouRA [NeurIPS24]	1.12	<u>90.4</u>	0.96	80.9	85.5	62.0	73.9
FourierFT [ICML24]	0.91	88.1	0.13	80.8	83.9	53.7	72.8
LoRA-XS [ECAI25]	0.66	89.3	0.41	80.1	83.5	57.8	71.4
CDRA-SPT [CVPR25]	0.86	89.9	0.49	<u>82.4</u>	<u>86.3</u>	61.8	74.5
Sine-LoRA [ICLR25]	0.90	89.9	0.58	82.0	85.5	62.5	74.5
<i>Ours</i>							
LoRA-SPS (Efficient)	0.30	<u>90.4±.06</u>	0.22	82.3±.11	<u>86.3±.09</u>	<u>63.6±.21</u>	<u>75.3±.22</u>
LoRA-SPS (Best)	0.53	90.8±.08	0.60	82.7±.10	86.6±.07	63.8±.19	75.6±.19

parameters, outperforming all baselines in efficiency and matching the closest PEFT method, FouRA, while requiring $3.7\times$ fewer trainable parameters.

4.6.2 Semantic Segmentation Tasks

4.6.2.1 Setup

CDRA-SPT reports ADE20K [113] segmentation using DINOv2 [44] within the open-source code base [131]; however, a documented implementation bug⁴ renders these metrics unreliable. We therefore exclude CDRA-SPT from baselines and instead compare to SPT-LoRA [36]. Following SPT-LoRA, we adopt SETR-PUP [2] with a ViT-L [1] backbone and evaluate on the ADE20K validation set. SD is omitted due to the high computational cost of segmentation experiments. Hyperparameter details are in section 4.5.3.4.

4.6.2.2 Results

We evaluate LoRA-SPS on semantic segmentation and report mean Intersection-over-Union (mIoU) under single-scale (s.s.) and multi-scale (m.s.) settings. The s.s. metric measures performance at the original input resolution, while m.s. aggregates predictions from multiple input scales to enhance accuracy. As shown in table 4.9, LoRA-SPS consistently outperforms all other PEFT baselines and rivals the performance of full fine-tuning, while using $22.2\times$ fewer parameters. Note that full fine-tuning is not a PEFT method.

Table 4.9: **Semantic segmentation** on ADE20K. SETR [2] with a ViT-L backbone; mIoU reported under single-scale (s.s.) and multi-scale (m.s.) evaluation. Fully fine-tuned ResNet-101 is shown for reference. **Bold** marks the best.

Method	mIoU-s.s	mIoU-m.s	# Params (M)
Full (ResNet-101) [31]	45.5	46.3	63.0
Full (ViT-L) [31]	48.3	50.1	318.3
Linear [31]	35.1	37.5	13.2
VPT-D [ECCV22] [31]	44.0	45.6	15.8
LoRA [ICLR22] [32]	43.9	45.9	14.7
SPT-LoRA [ICCV23] [36]	45.4	47.5	14.6
LoRA-SPS (Ours)	47.5	48.1	14.3

4.6.3 Text-to-Image Generation Task

4.6.3.1 Setup

We evaluate subject-driven image generation on the DreamBooth dataset [114] with Stable Diffusion v1.5 [3] across five subjects: *dog*, *cat*, *backpack*, *berry bowl*, *can*, and compare to LoRA, Sine-LoRA, FouRA, and LoRA-SPS. Adapters are trained with AdamW [130] on diffusion loss; generation uses 50 DDIM steps, classifier-free guidance 7.5, and 25 prompts with 4 renders each [53]. Metrics are subject-wise means over

⁴CDRA-SPT, DINOv2 IoU metric computation issue: <https://github.com/RobvanGastel/dinov3-finetune/issues/9>.

prompts: fidelity via DINOv2 ViT-B/14 CLS cosine, alignment via CLIP ViT-B/32 cosine, dispersion via LPIPS [62], and distribution shift via FID [63]. Cosine scores are reported in %. Details in section 4.5.3.5.

4.6.3.2 Results

Table 4.10: **Text-to-image generation**, DreamBooth, Stable Diffusion v1.5 [3]. DINO, CLIP-I, CLIP-T, and LPIPS Div. (higher is better); FID (lower is better). **Bold**: Best and underline: second-best.

Method	# Params (M)	DINO (%) $\uparrow$	CLIP-I (%) $\uparrow$	CLIP-T (%) $\uparrow$	LPIPS Div. $\uparrow$	FID $\downarrow$
Full	859.52	80.4	90.2	27.3	44.91	64.80
LoRA [ICLR22] [32]	0.80	61.8	82.9	27.0	<u>65.98</u>	167.05
Sine-LoRA [arXiv24] [33]	0.80	<u>67.2</u>	84.1	<u>27.2</u>	65.22	<u>144.58</u>
FouRA [NeurIPS24] [53]	13.29	61.2	83.3	27.1	68.82	164.41
LoRA-SPS (Efficient)	0.13	64.2	<u>84.9</u>	27.4	63.74	149.64
LoRA-SPS (Best)	<u>0.52</u>	73.7	89.2	25.6	62.17	74.06

Table 4.10 reports DreamBooth-style fine-tuning on Stable Diffusion v1.5 [3] UNet. LoRA-SPS (Best) achieves the strongest result on fidelity (DINO 73.7%), visual alignment (CLIP-I 89.2%), and distributional quality (FID 74.06), surpassing Sine-LoRA, FouRA, and LoRA. The Efficient variant matches or beats Sine-LoRA on textual alignment, diversity, and FID at $6\times$ fewer trainable parameters (0.13M vs. 0.80M). Furthermore, LoRA-SPS (Efficient) outperforms the strongest spectral baseline FouRA in 4/5 metrics, while requiring $100\times$ fewer parameters. Thus, LoRA-SPS captures complex generative features under tight parameter budgets; qualitative comparisons for generated DreamBooth subjects are provided in fig. 4.7.

4.6.4 Language Tasks

4.6.4.1 Setup

We evaluate on the GLUE benchmark [5], a suite of diverse natural language understanding tasks (see table 4.7). Following the FourierFT protocol [39, 32, 37], the results are reported as the median over five runs, and with the best epoch selected for each run. Similarly, we use the validation set for evaluation and exclude MNLI [115], WNLI [116], and QQP. Hyperparameter details are provided in section 4.5.3.6.

4.6.4.2 Results

we extend our vision experiments by conducting a comprehensive evaluation on the GLUE benchmark. Detailed results are reported in table 4.11. Our method consistently outperforms the comparable FourierFT baseline. Compared to full fine-tuning, it achieves comparable or superior performance while utilizing only a small fraction of the parameters. Against the more recent CDRA-SPT [37], our approach requires $16\times$ fewer parameters while maintaining comparable or better performance. These results demonstrate the effectiveness of our method in vision tasks and its potential for extension to the language domain.

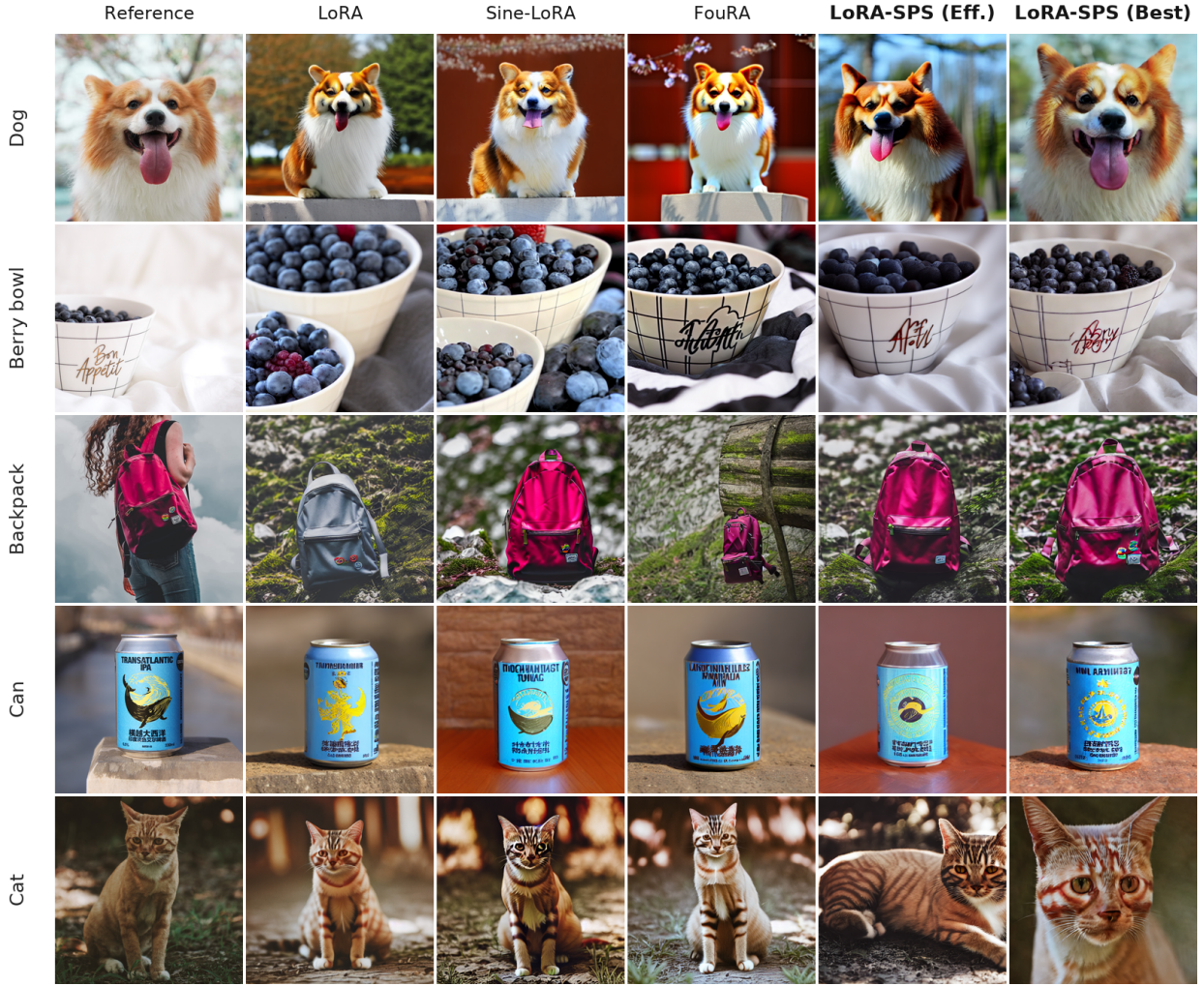


Figure 4.7: **Qualitative DreamBooth comparison.** LoRA-SPS matches or exceeds the visual quality of LoRA, Sine-LoRA, and FouRA while preserving prompt alignment, at up to $25\times$ fewer parameters.

Table 4.11: **RoBERTa-Large [4] performance.** Fine-tuning spans various PEFT methods and is evaluated across six GLUE [5] tasks. Results for CoLA are reported using Matthew’s correlation (MCC), and STS-B using Pearson’s correlation (PCC). Each result reported is a median of five runs. **Bold** and underline indicate best and second-best performance across PEFT methods for the respective dataset.

RoBERTa-Large (356M)	# Params (M)	SST-2 (Acc.)	MRPC (Acc.)	CoLA (MCC)	QNLI (Acc.)	RTE (Acc.)	STS-B (PCC)	Mean
Full [CVPR22] [4]	356	96.4	<u>90.9</u>	68.0	94.7	86.6	92.4	88.2
<i>Reparameterization-Based Adaptation</i>								
LoRA [ICLR22] [32]	0.8	<u>96.2</u>	90.2	<u>68.2</u>	<u>94.8</u>	85.2	<u>92.3</u>	87.8
FourierFT [ICML24] [39]	0.048	96.0	<u>90.9</u>	67.1	94.4	87.4	91.9	88.0
CDRA-SPT [CVPR25] [37]	0.8	95.5	<u>90.9</u>	68.5	94.7	<u>88.4</u>	92.4	<u>88.4</u>
<i>Ours</i>								
LoRA-SPS ($r = 8, d = 24$)	0.037	96.1 $\pm$.17	90.4 $\pm$.71	67.5 $\pm$.98	94.3 $\pm$.14	88.1 $\pm$.75	92.1 $\pm$.11	88.1 $\pm$.24
LoRA-SPS ($r = 16, d = 16$)	0.049	<u>96.2$\pm$.20</u>	91.2$\pm$.72	<u>68.2$\pm$.59</u>	94.7$\pm$.20	89.2$\pm$.90	92.2 $\pm$.60	88.6$\pm$.24

4.6.5 Ablation Studies

4.6.5.1 Effect of Rank & Sinusoidal Dimension

As shown in fig. 4.8, we analyze how varying the rank r (y-axis) and sinusoidal dimension d (x-axis) influences accuracy. The results indicate that often performance increases as the number of tunable parameters increases. However, on average, the performance plateaus around 0.3% tunable parameters likely due to overfitting, specifically at 0.23% for the Natural subset, 0.27% for Specialized, and 0.27% for Structured tasks. This motivates the *Efficient* configuration in table 4.8, which targets this regime and recovers nearly all *Best* accuracy at roughly half the parameter count (0.22% vs 0.6%).

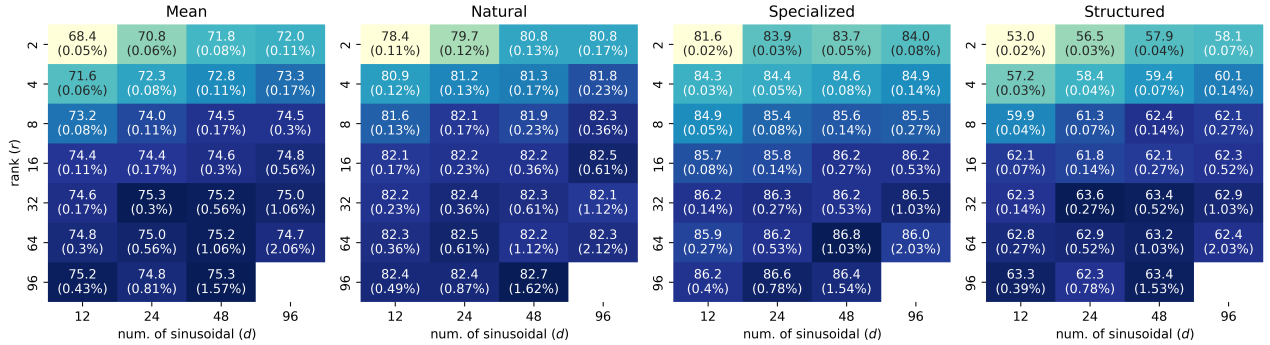


Figure 4.8: **Dimensionality search.** Top-1 accuracy (color intensity) on VTAB-1K and total % of parameters tuned (shown in parentheses) for varying LoRA-SPS rank r and sinusoidal dimension d .

4.6.5.2 Effect of Frequency Scaling

The optimal frequency scale varies by domain (see fig. 4.9): Natural peaks at 512, Specialized at 2048, and Structured at 256, with minimal overall performance difference between 1024 and 2048. Therefore, we adopt frequencies $f \in \{128, 256, 512, 1024\}$. Likely, Natural images favor lower frequencies due to alignment with the model’s pretraining distribution, while Specialized tasks, being distributionally shifted, require higher frequencies for effective generalization.

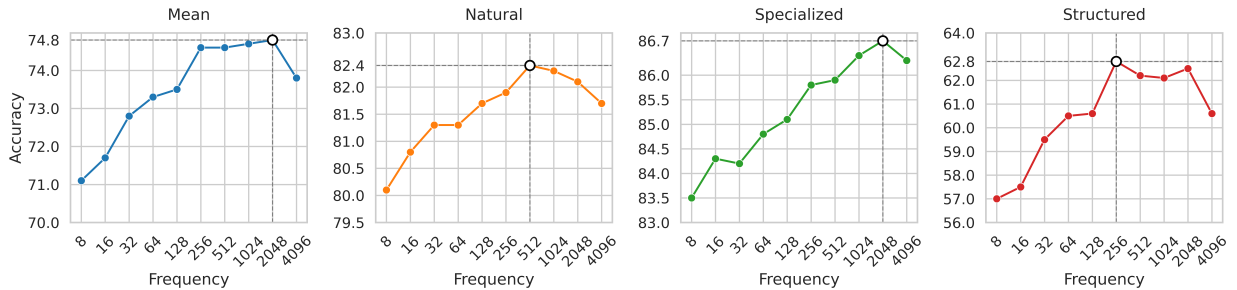


Figure 4.9: **Frequency search.** Top-1 accuracy performance of LoRA-SPS on VTAB-1K for varied frequency, while rank r and sinusoidal dimension d are fixed to 32 and 48, respectively.

4.6.5.3 Guide to Hyperparameter Selection

As demonstrated by the ablation studies in figs. 4.8 and 4.9, the hyperparameter selection in LoRA-SPS is intuitive and stable. In practice, choosing the sinusoidal dimension d to be comparable to the rank r (i.e., $d \approx r$) provides a reliable tradeoff between expressiveness and parameter efficiency. For frequency scaling, a moderate value (e.g., 512) consistently performs well when the target domain is close to the pretraining distribution, while larger frequency scales can be beneficial as the domain diverges further from the pretrained data.

4.6.5.4 Effect of Model Size

Table 4.12 compares LoRA-SPS vs LoRA/Sine-LoRA across four ViT scales in the same search space. LoRA-SPS gets the best performance at all sizes with the fewest parameters. The benefit widens with scale: on ViT-Huge/14, LoRA-SPS gains 1.5 points over LoRA at $4\times$ fewer parameters, consistent with stronger impact from embedding-dimension decoupling as d grows.

Table 4.12: **ViT model-scale comparison** on VTAB. Top-1 accuracy reported for Natural, Specialized, and Structured subsets. Tuned / Total (%) indicates the percentage of trainable parameters relative to the backbone. Results are shown for ViT-Small/16, ViT-Base/16, ViT-Large/16, and ViT-Huge/14. **Bold** indicates best performance across PEFT methods for each model size and VTAB subset.

ViT-Small/16 (22M)							ViT-Base/16 (85.8M)						
	LoRA [32]		Sine-LoRA [33]		LoRA-SPS			LoRA [32]		Sine-LoRA [33]		LoRA-SPS	
VTAB	Acc.	%Params	Acc.	%Params	Acc.	%Params	VTAB	Acc.	%Params	Acc.	%Params	Acc.	%Params
Natural	78.7	3.52	79.1	4.73	79.2	1.23	Natural	80.8	1.53	80.7	1.85	82.1	0.47
Specialized	85.5	1.90	86.3	1.62	85.9	0.55	Specialized	85.0	0.97	85.0	0.94	86.3	0.48
Structured	60.7	2.31	60.9	1.68	61.8	1.21	Structured	61.2	1.01	62.2	1.03	62.4	0.44
Mean	72.6	2.67	72.5	2.79	73.3	1.08	Mean	73.4	1.19	73.8	1.25	74.7	0.46
ViT-Large/16 (304M)							ViT-Huge/14 (632M)						
	LoRA [32]		Sine-LoRA [33]		LoRA-SPS			LoRA [32]		Sine-LoRA [33]		LoRA-SPS	
VTAB	Acc.	%Params	Acc.	%Params	Acc.	%Params	VTAB	Acc.	%Params	Acc.	%Params	Acc.	%Params
Natural	82.8	1.20	83.2	0.74	84.0	0.53	Natural	78.2	1.04	78.1	0.91	80.3	0.23
Specialized	86.2	0.37	87.0	1.05	87.4	0.49	Specialized	83.5	0.52	83.8	0.41	85.4	0.20
Structured	57.4	1.08	59.7	0.94	60.0	0.38	Structured	58.4	0.85	58.3	0.88	59.3	0.17
Mean	72.8	0.97	74.1	0.89	74.6	0.46	Mean	71.0	0.85	71.0	0.79	72.5	0.20

4.6.5.5 Effect of Equal Parameter Budget Comparison

Table 4.13 compares LoRA, Sine-LoRA, and LoRA-SPS under the same parameter budgets. LoRA-SPS reaches a substantially higher rank ($r = d = 24$) than rank-1 LoRA/Sine-LoRA, improving accuracy by +2.7 and +1.4, respectively.

4.6.5.6 Effect of LoRA-SPS Components

Table 4.13 ablates each component at a fixed 0.165M parameter budget. Removing the frequency matrix F is most damaging (68.7%); removing sin falls below LoRA (71.3%); removing the amplitude A yields 73.9%. The full construction reaches 74.8%, confirming that F , sin, and A contribute complementary capacity, with F the most impactful. We further analyze the impact of sinusoidal non-linearity in table 4.14, which confirms that using sin is impactful, while there is not much difference between using sin, cos, or both.

Table 4.13: **Equivalent-budget transition from LoRA to LoRA-SPS components.** Top-1 VTAB-1K accuracy with 0.165M trainable params. We compare LoRA, Sine-LoRA ($f = 400$), and LoRA-SPS ($f = 1024$): LoRA/Sine-LoRA use $r = 1$, while LoRA-SPS variants use matching parameter budgets. We only tune $\alpha \in \{0.01, 0.1, 1.0\}$ per row; normalization omitted from the method for clarity.

Method	Dimensions	sin(X)	A	F	$\mathcal{S}_{\text{stride}}$	Natural	Specialized	Structured	Mean Acc.
LoRA [ICLR22] [32]	$r = 1$					79.9	84.2	60.2	72.1
Sine-LoRA [ICLR25] [33]	$r = 1$	✓		✓		81.0	84.8	61.1	73.4
$\mathbf{y} = \mathbf{A}\mathbf{F}\mathcal{W}$	$d = r = 24$		✓	✓	✓	80.1	83.6	57.6	71.3
$\mathbf{y} = \sin(f \times \mathbf{F}\mathcal{W})$	$d = r = 32$	✓		✓	✓	81.9	86.1	60.6	73.9
$\mathbf{y} = \mathbf{A} \sin(f \times \mathcal{W})$	$d = r = 32$	✓	✓		✓	77.3	84.7	50.7	68.7
LoRA-SPS (Ours) eq. (4.6)	$d = r = 24$	✓	✓	✓	✓	82.3	86.2	62.7	74.8

4.6.5.7 Effect of Sinusoidal Transformation

We evaluate the effect of sinusoidal transformation function perturbations on LoRA-SPS performance (VTAB-1K, ViT-B). The sinusoidal activations (sin, cos, or both) consistently outperform no activation (see results in table 4.14). Omitting sinusoidal transformation reduces performance across all datasets except Caltech101 and KITTI-Dist, which instead show a performance increase.

Table 4.14: **Effect of sinusoidal transformation function** perturbations on LoRA-SPS performance (top-1 accuracy) on VTAB-1K using a ViT-B backbone. Rank (r), sinusoidal dimension (d), and frequency (f) are 32, 48, and 1024, respectively.

Activation	Mean Acc.	Natural	Specialized	Structured	caltech101	KITTI-Dist
sin, cos	74.8	82.4	86.6	62.3	90.4	79.6
sin	74.9	82.4	86.3	62.7	89.9	79.6
cos	75.1	82.6	86.5	62.9	89.8	79.9
none	70.1	80.2	84.2	58.7	95.5	84.0

4.6.5.8 Effect of Sampling Strategy, Performance & Compute

All four strategies achieve nearly identical accuracy (tables 4.15 and 4.16) and outperform LoRA by 1.2–1.5% at a large rank of 136, while incurring a minimal increase in wall-clock training time over 100 epochs, suggesting that LoRA-SPS is robust to the sampling rule.

Table 4.15: **Effect of sampling strategy at moderate ($r = 32$) rank.** We evaluate LoRA-SPS top-1 accuracy on VTAB-1K with rank $r = 32$, sinusoidal dimension $d = 48$, and frequency $f = 1024$ fixed.

Sampling	Natural	Specialized	Structured	Mean Acc.
Random	82.0	86.2	61.3	74.2
Stride, eq. (4.3)	82.3	86.3	63.1	75.1
Top-Var, eq. (4.4)	82.4	86.6	62.2	74.8
Top, eq. (4.2)	82.3	86.6	62.3	74.8
SVD, eq. (4.5)	82.6	86.5	63.4	75.3

Table 4.16: **Effect of sampling strategies & compute at large parameter count.** Training overhead and top-1 accuracy on VTAB-1K for LoRA ($r = 32$) and LoRA-SPS ($r = d = 136$, $f = 1024$, $\alpha = 0.1$), average training time (in minutes) over 100 epochs, measured in wall-clock time across all specialized tasks.

Sampling	# Params (M)	Mem. (MB)	Time (mm:ss)	Acc.
None, LoRA [32]	5.386	10290	7:15	85.1
Top, eq. (4.2)	5.327	8714	7:43	86.3
Stride, eq. (4.3)	5.327	8714	7:42	86.5
Top-Var, eq. (4.4)	5.327	8714	7:54	86.4
SVD, eq. (4.5)	5.327	8714	7:58	86.6

4.6.5.9 Effect of Targeted Layers

Table 4.17 shows that targeted layers in LoRA-SPS affect performance and efficiency. Adapting all layers achieves the highest performance at higher parameter cost, while selective targeting (e.g., key attention pathways) remains competitive and more efficient, making it advantageous under limited parameter/compute budgets.

Table 4.17: **Impact of different LoRA-SPS target layer configurations** on VTAB-1K. Evaluated top-1 accuracy on Natural, Specialized, and Structured tasks, plus the overall mean. Tuned / Total (%) indicates the percentage of trainable parameters. We fix rank $r = 32$, sinusoidal dimension $d = 48$, frequency $f = 1024$, and scaling factor $\alpha = 0.1$. We only vary the target.

Targets	Natural	Specialized	Structured	Mean Acc.
All-Linear	82.3	86.5	62.4	74.8
- Tuned / Total (%)	.61	.53	.52	.56
Q, V, MLP	81.9	86.0	62.0	74.4
- Tuned / Total (%)	.44	.36	.35	.39
K, V, MLP	82.0	86.0	62.4	74.6
- Tuned / Total (%)	.44	.36	.35	.39
Q, K, V	81.7	85.5	62.0	74.2
- Tuned / Total (%)	.36	.27	.27	.30
Q, V	81.4	85.6	61.4	73.9
- Tuned / Total (%)	.27	.19	.18	.22

4.6.5.10 Effect of LoRA-SPS Layer Coverage

Progressively expanding LoRA-SPS layers leads to steady improvements in accuracy across all VTAB-1K task types, see table 4.18. The Structured category benefits the most, reflecting the value of deeper adaptation for complex visual transformations.

Table 4.18: **Impact of progressively expanding LoRA-SPS layers.** LoRA-SPS modules are inserted at layers $\{1, 2, 4, 6, 12\}$ and all intermediate layers inherit the most recent LoRA-SPS weights. Evaluated on VTAB-1K top-1 accuracy across Natural, Specialized, and Structured tasks, as well as the overall mean accuracy. Tuned / Total (%) indicates the percentage of trainable parameters. We fix rank $r = 32$, sinusoidal dimension $d = 48$, frequency $f = 1024$, and scaling factor $\alpha = 0.1$. Only the number of enabled layers in LoRA-SPS is varied.

# Layers	Natural	Specialized	Structured	Mean Acc.
1	81.6	85.9	60.4	73.6
- Tuned / Total (%)	.14	.06	.05	.09
2	82.0	85.9	61.5	74.2
- Tuned / Total (%)	.19	.10	.10	.13
4	82.0	86.4	61.5	74.3
- Tuned / Total (%)	.27	.19	.18	.22
6	82.1	86.1	62.1	74.5
- Tuned / Total (%)	.36	.27	.27	.30
12	82.3	86.5	62.4	74.8
- Tuned / Total (%)	.61	.53	.52	.56

4.6.6 Summary & Discussion

The experiments show that LoRA-SPS is an effective and highly parameter-efficient alternative to standard LoRA-style adaptation. Across image classification, semantic segmentation, text-to-image generation, and language understanding tasks, LoRA-SPS consistently achieves competitive or improved performance while using substantially fewer trainable parameters. This indicates that sampling from pretrained weights and applying sinusoidal projections can produce effective adapter updates without relying on large low-rank matrices.

The ablation studies further show that LoRA-SPS performance depends on the interaction of its main components. The sinusoidal transformation, frequency scaling, trainable amplitude mixing, and pretrained-weight sampling each contribute to the final results. In particular, the frequency-based projection is important for improving adaptation quality, while the amplitude matrix helps reweight the projected features for the downstream task.

Overall, the results suggest that LoRA-SPS improves the efficiency–performance tradeoff of PEFT by making better use of pretrained weight structure. The main takeaway is that compact adapters can remain effective when their update space is built from informative pretrained-weight samples rather than learned entirely from scratch.

Chapter 5

Conclusion

This thesis investigated the efficient adaptation of foundation models under realistic constraints of limited compute and demographically imbalanced data, with a focus on medical imaging. Through a large-scale empirical study, we showed that PEFT methods offer a practical alternative to full fine-tuning, while revealing key tradeoffs between utility, subgroup disparity, and computational cost, especially within smaller demographic groups. Motivated by the limitations of existing approaches, we introduced LoRA-SPS, a novel low-rank adaptation method that leverages sampled pretrained weights and sinusoidal projections to enhance expressiveness under tight parameter budgets, consistently achieving strong performance across vision and language tasks. Overall, this work provides both practical insights and methodological contributions toward developing efficient, generalizable, and equitable adaptation strategies for foundation models in real-world settings.

5.1 Broader Impact

This work contributes to making foundation models more accessible and practical for real-world deployment, particularly in resource-constrained healthcare settings where full model retraining is often infeasible. By improving the efficiency of model adaptation and highlighting how performance varies across demographic groups, it supports the development of systems that can better serve diverse patient populations while reducing computational and environmental costs. The proposed methods enable practitioners to adapt models with limited data and infrastructure, potentially expanding the reach of AI-driven tools to smaller institutions. Overall, this work promotes more efficient, inclusive, and scalable use of foundation models, with the potential to improve reliability and equity in applied machine learning systems.

5.2 Limitations

First, our empirical study mainly focuses on medical imaging and on VPT-based methods. The choice of the VPT family was due to preserving the expensive learned pretrained parameters of FMs. The pretraining of FMs are inherently computationally expensive. Despite the scale of our study (490 configurations and over 3,500 GPU hours), covering all potential adaptation methods is computationally prohibitive. Moreover,

we have been limited by the availability of demographic data in large, public datasets for training FMs and assessing demographic disparities, while maintaining reproducibility. Extending this work to other demographics and clinical domains remains an important direction for future research.

LoRA-SPS has two main limitations. First, it uses sinusoidal transformations to construct adapter projection matrices during training, which are less hardware-optimized than matrix multiplications and convolutions; however, prior work [84, 55, 85, 33, 38] shows growing adoption of sinusoidal activations in deep learning, motivating future hardware optimization. Furthermore, this overhead is limited to training, as our method can be merged into the frozen backbone after adaptation, introducing no inference-time cost. Second, LoRA-SPS’s sinusoidal dimension and frequency hyperparameters remain empirically chosen per task. Automatic per-task budget allocation is an important direction for future work. We provide ablation studies and practical guidance to help select these hyperparameters.

5.3 Future Directions

This thesis opens several directions for future research in efficient and fair adaptation of foundation models. First, future work should extend this study beyond fixed demographic splits to intersectional patient populations, while developing adaptive PEFT methods that dynamically allocate capacity based on task complexity and data availability. Second, future work should explore dynamic and adaptive PEFT methods that can adjust parameter allocation based on data availability and task difficulty. Such approaches may improve the efficiency–performance tradeoff observed in static methods such as LoRA-SPS even further. Third, while this thesis focuses on vision and language tasks in medical settings, extending these methods to multimodal clinical systems combining imaging, text, and structured data represents an important next step. Efficient adaptation in multimodal settings remains underexplored and computationally challenging. Last, stronger integration of fairness constraints directly into the optimization process of PEFT methods could help mitigate the tradeoffs observed between efficiency & performance. In particular, jointly optimizing for overall and demographic performance during adaptation may reduce reliance on post-hoc analysis of disparities.

Bibliography

- [1] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, “An image is worth 16x16 words: Transformers for image recognition at scale,” in *International Conference on Learning Representations*, 2021.
- [2] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr, *et al.*, “Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6881–6890, 2021.
- [3] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.
- [4] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “Roberta: A robustly optimized bert pretraining approach,” *arXiv preprint arXiv:1907.11692*, 2019.
- [5] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, “GLUE: A multi-task benchmark and analysis platform for natural language understanding,” 2019. In the Proceedings of ICLR.
- [6] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, “Masked autoencoders are scalable vision learners,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16000–16009, 2022.
- [7] X. Chen, S. Xie, and K. He, “An empirical study of training self-supervised vision transformers,” in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9640–9649, 2021.
- [8] Y. LeCun, C. Cortes, and C. Burges, “Mnist handwritten digit database,” *ATT Labs [Online]*. Available: <http://yann.lecun.com/exdb/mnist>, vol. 2, 2010.
- [9] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa, *et al.*, “Dinov3,” *arXiv preprint arXiv:2508.10104*, 2025.

- [10] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in *International conference on machine learning*, pp. 1597–1607, PmLR, 2020.
- [11] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, *et al.*, “Bootstrap your own latent-a new approach to self-supervised learning,” *Advances in neural information processing systems*, vol. 33, pp. 21271–21284, 2020.
- [12] M. Singh, Q. Duval, K. V. Alwala, H. Fan, V. Aggarwal, A. Adcock, A. Joulin, P. Dollár, C. Feichtenhofer, R. Girshick, *et al.*, “The effectiveness of mae pre-pretraining for billion-scale pretraining,” in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5484–5494, 2023.
- [13] A. Gupta, I. Osman, M. S. Shehata, W. J. Braun, and R. E. Feldman, “Medmae: A self-supervised backbone for medical imaging tasks,” *Computation*, vol. 13, no. 4, 2025.
- [14] A. B. Sellergren, C. Chen, Z. Nabulsi, Y. Li, A. Maschinot, A. Sarna, J. Huang, C. Lau, S. R. Kalidindi, M. Etemadi, F. Garcia-Vicente, D. Melnick, Y. Liu, K. Eswaran, D. Tse, N. Beladia, D. Krishnan, and S. Shetty, “Simplified transfer learning for chest radiography models using less data,” *Radiology*, vol. 305, no. 2, pp. 454–465, 2022.
- [15] L. Zhou, H. Liu, J. Bae, J. He, D. Samaras, and P. Prasanna, “Self pre-training with masked autoencoders for medical image classification and segmentation,” in *2023 IEEE 20th International Symposium on Biomedical Imaging (ISBI)*, IEEE, 2023.
- [16] R. Jin, Z. Xu, Y. Zhong, Q. Yao, D. QI, S. K. Zhou, and X. Li, “Fairmedfm: fairness benchmarking for medical imaging foundation models,” *Advances in Neural Information Processing Systems*, vol. 37, 2024.
- [17] G. Bahre, H. Hamidihesarsorkh, A. Sellergren, L. A. Celi, F. Calimeri, and L. Seyyed-Kalantari, “Representation is all we need: Performance and fairness of google x-ray foundation model representations – a preliminary study,” in *Proceedings of the 13th IEEE International Conference on Healthcare Informatics (IEEE ICHI)*, (Rende, Italy), March 2025.
- [18] R. B. et al., “On the opportunities and risks of foundation models,” 2022.
- [19] J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Illcus, C. Chute, H. Marklund, B. Haghgoo, R. Ball, K. Shpanskaya, *et al.*, “Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, 2019.
- [20] A. E. W. Johnson, T. J. Pollard, N. R. Greenbaum, M. P. Lungren, C. ying Deng, Y. Peng, Z. Lu, R. G. Mark, S. J. Berkowitz, and S. Horng, “Mimic-cxr-jpg, a large publicly available database of labeled chest radiographs,” 2019.
- [21] Y. Luo, M. Shi, M. O. Khan, M. M. Afzal, H. Huang, S. Yuan, Y. Tian, L. Song, A. Kouhana, T. Elze, *et al.*, “Fairclip: Harnessing fairness in vision-language learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

- [22] J. W. Gichoya et al., “Ai recognition of patient race in medical imaging: a modelling study,” *The Lancet. Digital health*, vol. 4, pp. e406–e414, 2022.
- [23] E. Abbasi Babil, M. Ahluwalia, L. Seyyed-Kalantari, B. Fine, and M. Abdalla, “Body mass index prediction from chest radiographs and associated performance gaps in chest radiograph abnormality prediction,” in *CAR 2024 Annual Scientific Meeting (ASM)*, (Montréal, Canada), 2024.
- [24] L. Seyyed-Kalantari, H. Zhang, M. B. McDermott, I. Y. Chen, and M. Ghassemi, “Underdiagnosis bias of artificial intelligence algorithms applied to chest radiographs in under-served patient populations,” *Nature medicine*, vol. 27, no. 12, 2021.
- [25] L. Seyyed-Kalantari, G. Liu, M. B. A. McDermott, and M. Ghassemi, “Chexclusion: Fairness gaps in deep chest x-ray classifiers,” *Pacific Symposium on Biocomputing*, vol. 26, 2021.
- [26] M. K. Cho, “Rising to the challenge of bias in health care ai,” *Nature Medicine*, vol. 27, no. 12, 2021.
- [27] Z. Han, C. Gao, J. Liu, J. Zhang, and S. Q. Zhang, “Parameter-efficient fine-tuning for large models: A comprehensive survey,” *arXiv preprint arXiv:2403.14608*, 2024.
- [28] Y. Xin, S. Luo, H. Zhou, J. Du, X. Liu, Y. Fan, Q. Li, and Y. Du, “Parameter-efficient fine-tuning for pre-trained vision models: A survey,” *arXiv preprint arXiv:2402.02242*, 2024.
- [29] L. Wang, S. Chen, L. Jiang, S. Pan, R. Cai, S. Yang, and F. Yang, “Parameter-efficient fine-tuning in large language models: a survey of methodologies,” *Artificial Intelligence Review*, vol. 58, no. 8, p. 227, 2025.
- [30] N. J. Prottasha, U. R. Chowdhury, S. Mohanto, T. Nuzhat, A. A. Sami, M. S. Ali, M. S. I. Sobuj, H. Raman, M. Kowsher, and O. O. Garibay, “Peft a2z: Parameter-efficient fine-tuning survey for large language and vision models,” 2025.
- [31] M. Jia, L. Tang, B.-C. Chen, C. Cardie, S. Belongie, B. Hariharan, and S.-N. Lim, “Visual prompt tuning,” in *European Conference on Computer Vision (ECCV)*, 2022.
- [32] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “LoRA: Low-rank adaptation of large language models,” in *International Conference on Learning Representations*, 2022.
- [33] Y. Ji, H. Saratchandran, C. Gordon, Z. Zhang, and S. Lucey, “Efficient learning with sine-activated low-rank matrices,” in *The Thirteenth International Conference on Learning Representations*, 2025.
- [34] R. Zeng, C. Han, Q. Wang, C. Wu, T. Geng, L. Huang, Y. N. Wu, and D. Liu, “Visual fourier prompt tuning,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 5552–5585, 2024.
- [35] K. Balazy, M. Banaei, K. Aberer, and J. Tabor, “LoRA-XS: Low-rank adaptation with extremely small number of parameters,” 2024.

- [36] H. He, J. Cai, J. Zhang, D. Tao, and B. Zhuang, “Sensitivity-aware visual parameter-efficient fine-tuning,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11825–11835, 2023.
- [37] T. Chen, J. Chen, B. Zhang, Z. Yu, S. Chen, R. Ye, X. Li, and Y. Ye, “Sensitivity-Aware Efficient Fine-Tuning via Compact Dynamic-Rank Adaptation,” in *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (Los Alamitos, CA, USA), pp. 9655–9664, IEEE Computer Society, June 2025.
- [38] Y. Li, L. Song, and H. Hou, “LoRAN: Improved low-rank adaptation by a non-linear transformation,” in *Findings of the Association for Computational Linguistics: EMNLP 2024* (Y. Al-Onaizan, M. Bansal, and Y.-N. Chen, eds.), (Miami, Florida, USA), pp. 3134–3143, Association for Computational Linguistics, Nov. 2024.
- [39] Z. Gao, Q. Wang, A. Chen, Z. Liu, B. Wu, L. Chen, and J. Li, “Parameter-efficient fine-tuning with discrete fourier transform,” in *Forty-first International Conference on Machine Learning*, 2024.
- [40] T. Ridnik, E. Ben-Baruch, A. Noy, and L. Zelnik-Manor, “Imagenet-21k pretraining for the masses,” 2021.
- [41] I. Utseth, A. H. Vedal, S. Thomas, and L. Eikvil, “Comparing foundation models for medical images: A study on limited data and generalization,” in *Northern Lights Deep Learning Conference 2026*, 2026.
- [42] S.-C. Huang, A. Pareek, M. Jensen, M. P. Lungren, S. Yeung, and A. S. Chaudhari, “Self-supervised learning for medical image classification: a systematic review and implementation guidelines,” *npj Digital Medicine*, vol. 6, p. 74, Apr 2023.
- [43] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.
- [44] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. HAZIZA, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, “DINOv2: Learning robust visual features without supervision,” *Transactions on Machine Learning Research*, 2024. Featured Certification.
- [45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.
- [46] K. O’shea and R. Nash, “An introduction to convolutional neural networks,” *arXiv preprint arXiv:1511.08458*, 2015.

- [47] W. Xu, R. Xu, C. Wang, S. Xu, L. Guo, M. Zhang, and X. Zhang, “Spectral prompt tuning: Unveiling unseen classes for zero-shot semantic segmentation,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 38, pp. 6369–6377, 2024.
- [48] M. Zhu, H. Li, H. Chen, C. Fan, W. Mao, C. Jing, Y. Liu, and C. Shen, “Segprompt: Boosting open-world segmentation via category-level prompt learning,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 999–1008, 2023.
- [49] X. Nie, B. Ni, J. Chang, G. Meng, C. Huo, S. Xiang, and Q. Tian, “Pro-tuning: Unified prompt tuning for vision tasks,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 6, pp. 4653–4667, 2023.
- [50] S. Yoo, E. Kim, D. Jung, J. Lee, and S. Yoon, “Improving visual prompt tuning for self-supervised vision transformers,” in *International Conference on Machine Learning*, pp. 40075–40092, PMLR, 2023.
- [51] C. Han, Q. Wang, Y. Cui, Z. Cao, W. Wang, S. Qi, and D. Liu, “E²VPT: An effective and efficient approach for visual prompt tuning,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.
- [52] S. Ramasinghe and S. Lucey, “Beyond periodicity: Towards a unifying framework for activations in coordinate-mlps,” in *European Conference on Computer Vision*, pp. 142–158, Springer, 2022.
- [53] S. Borse, S. Kadambi, N. P. Pandey, K. Bhardwaj, V. Ganapathy, S. Priyadarshi, R. Garrepalli, R. Esteves, M. Hayat, and F. Porikli, “FouRA: Fourier low-rank adaptation,” in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.
- [54] S. T. Ly and H. V. Nguyen, “Enhancing parameter-efficient fine-tuning of vision transformers through frequency-based adaptation,” 2024.
- [55] V. Saragadam, D. LeJeune, J. Tan, G. Balakrishnan, A. Veeraraghavan, and R. G. Baraniuk, “Wire: Wavelet implicit neural representations,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18507–18516, 2023.
- [56] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, “Parameter-efficient transfer learning for nlp,” in *International conference on machine learning*, pp. 2790–2799, PMLR, 2019.
- [57] J. O. Zhang, A. Sax, A. Zamir, L. Guibas, and J. Malik, “Side-tuning: a baseline for network adaptation via additive side networks,” in *European conference on computer vision*, pp. 698–714, Springer, 2020.
- [58] E. B. Zaken, Y. Goldberg, and S. Ravfogel, “Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models,” in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1–9, 2022.

- [59] H. Cai, C. Gan, L. Zhu, and S. Han, “Tinytl: Reduce memory, not parameters for efficient on-device learning,” in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 11285–11297, Curran Associates, Inc., 2020.
- [60] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, “Emerging properties in self-supervised vision transformers,” in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.
- [61] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, “Learning transferable visual models from natural language supervision,” in *International conference on machine learning*, pp. 8748–8763, PmLR, 2021.
- [62] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.
- [63] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “Gans trained by a two time-scale update rule converge to a local nash equilibrium,” *Advances in neural information processing systems*, vol. 30, 2017.
- [64] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.
- [65] A. B. Sellergren, C. Chen, Z. Nabulsi, Y. Li, A. Maschinot, A. Sarna, J. Huang, C. Lau, S. R. Kalidindi, M. Etemadi, *et al.*, “Simplified transfer learning for chest radiography models using less data,” *Radiology*, vol. 305, no. 2, 2022.
- [66] G. Bahre *et al.*, “Underdiagnosis bias mitigation with expert foundation model’s representation,” *IEEE Access*, June 2025.
- [67] Y. Zong, Y. Yang, and T. Hospedales, “Medfair: benchmarking fairness for medical imaging,” *arXiv preprint arXiv:2210.01725*, 2022.
- [68] P. Wu, C. Liu, C. Chen, J. Li, C. I. Bercea, and R. Arcucci, “Fmbench: Benchmarking fairness in multimodal large language models on medical tasks,” *arXiv preprint arXiv:2410.01089*, 2024.
- [69] S. Zhang, Y. Xu, N. Usuyama, H. Xu, J. Bagga, R. Tinn, S. Preston, R. Rao, M. Wei, N. Valluri, and C. Wong, “Biomedclip: a multimodal biomedical foundation model pretrained from fifteen million scientific image-text pairs,” *arXiv preprint arXiv:2303.00915*, 2024.
- [70] Y. Wang, R. Bone, W. Fleisher, C. R. Gresenz, J. Mitchell, W. van der Klaauw, C. Wang, and L. Singh, “Using under-represented subgroup fine tuning to improve fairness for disease prediction,” in *Biomedical engineering systems and technologies, international joint conference, BIOSTEC... revised selected papers. BIOSTEC (Conference)*, vol. 2, p. 240, 2025.

- [71] T. Adewumi, L. Alkhaled, N. Gurung, G. van Boven, and I. Pagliai, “Fairness and bias in multimodal ai: A survey,” *arXiv preprint arXiv:2406.19097*, 2024.
- [72] I. Straw and H. Wu, “Investigating for bias in healthcare algorithms: a sex-stratified analysis of supervised machine learning models in liver disease prediction,” *BMJ health & care informatics*, vol. 29, no. 1, 2022.
- [73] X. Zhai, J. Puigcerver, A. Kolesnikov, P. Ruysen, C. Riquelme, M. Lucic, J. Djolonga, A. S. Pinto, M. Neumann, A. Dosovitskiy, L. Beyer, O. Bachem, M. Tschannen, M. Michalski, O. Bousquet, S. Gelly, and N. Houlsby, “The visual task adaptation benchmark,” 2019. cite arxiv:1910.04867.
- [74] Y. Luo, Y. Tian, M. Shi, L. R. Pasquale, L. Q. Shen, N. Zebardast, T. Elze, and M. Wang, “Harvard glaucoma fairness: a retinal nerve disease dataset for fairness learning and fair identity normalization,” *IEEE Transactions on Medical Imaging*, vol. 43, no. 7, pp. 2623–2633, 2024.
- [75] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.
- [76] S. Hayou, N. Ghosh, and B. Yu, “LoRA+: Efficient low rank adaptation of large models,” in *Forty-first International Conference on Machine Learning*, 2024.
- [77] F. Meng, Z. Wang, and M. Zhang, “Pissa: Principal singular values and singular vectors adaptation of large language models,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 121038–121072, 2024.
- [78] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, “Qlora: Efficient finetuning of quantized llms,” *Advances in neural information processing systems*, vol. 36, pp. 10088–10115, 2023.
- [79] Y. Li, Y. Yu, C. Liang, N. Karampatziakis, P. He, W. Chen, and T. Zhao, “Loftq: LoRA-fine-tuning-aware quantization for large language models,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [80] D. J. Kopiczko, T. Blankevoort, and Y. M. Asano, “VeRA: Vector-based random matrix adaptation,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [81] T. Jiang, S. Huang, S. Luo, Z. Zhang, H. Huang, F. Wei, W. Deng, F. Sun, Q. Zhang, S. Wang, deqing wang, and F. Zhuang, “MoRA: High-rank updating for parameter-efficient fine-tuning,” 2024.
- [82] H. Zhang, KangChun, Y. Wang, and D. Zou, “F-adapter: Frequency-adaptive parameter-efficient fine-tuning in scientific machine learning,” in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.
- [83] J. Zeng, L. Huang, and K. Wang, “Wst: Wavelet-based multi-scale tuning for visual transfer learning,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, pp. 22317–22325, 2025.

- [84] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, “Implicit neural representations with periodic activation functions,” *Advances in neural information processing systems*, vol. 33, pp. 7462–7473, 2020.
- [85] C. Gordon, Y. Ji, H. Saratchandran, P. Albert, and S. Lucey, “Compressing sine-activated low-rank adapters through post-training quantization,” *arXiv preprint arXiv:2505.21895*, 2025.
- [86] B. Lester, R. Al-Rfou, and N. Constant, “The power of scale for parameter-efficient prompt tuning,” in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, eds.), (Online and Punta Cana, Dominican Republic), pp. 3045–3059, Association for Computational Linguistics, Nov. 2021.
- [87] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, “Pre-trained models for natural language processing: A survey,” *Science China technological sciences*, vol. 63, no. 10, pp. 1872–1897, 2020.
- [88] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, “Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing,” *ACM computing surveys*, vol. 55, no. 9, pp. 1–35, 2023.
- [89] T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha, “Deep learning for classical japanese literature,” 2018.
- [90] X. Zhai, J. Puigcerver, A. Kolesnikov, P. Ruysen, C. Riquelme, M. Lucic, J. Djolonga, A. S. Pinto, M. Neumann, A. Dosovitskiy, *et al.*, “A large-scale study of representation learning with the visual task adaptation benchmark,” *arXiv preprint arXiv:1910.04867*, 2019.
- [91] A. Krizhevsky, V. Nair, and G. Hinton, “Cifar-10 and cifar-100 datasets,” *URL: <https://www.cs.toronto.edu/kriz/cifar.html>*, vol. 6, no. 1, p. 1, 2009.
- [92] F.-F. Li, M. Andreeto, M. Ranzato, and P. Perona, “Caltech 101,” Apr 2022.
- [93] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, “Describing textures in the wild,” in *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [94] M.-E. Nilsback and A. Zisserman, “Automated flower classification over a large number of classes,” in *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, Dec 2008.
- [95] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, “Cats and dogs,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.
- [96] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, “Reading digits in natural images with unsupervised feature learning,” 2011.
- [97] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, “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, June 2010.

- [98] J. W. T. C. M. W. B. S. Veeling, J. Linmans, “Rotation equivariant cnns for digital pathology,” Sept. 2018.
- [99] P. Helber, B. Bischke, A. Dengel, and D. Borth, “Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification,” 2017.
- [100] G. Cheng, J. Han, and X. Lu, “Remote sensing image scene classification: Benchmark and state of the art,” *Proceedings of the IEEE*, vol. 105, pp. 1865–1883, Oct 2017.
- [101] Kaggle and EyePacs, “Kaggle diabetic retinopathy detection,” jul 2015.
- [102] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick, “CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [103] X. Zhai, J. Puigcerver, A. Kolesnikov, P. Ruysen, C. Riquelme, M. Lucic, J. Djolonga, A. S. Pinto, M. Neumann, A. Dosovitskiy, L. Beyer, O. Bachem, M. Tschannen, M. Michalski, O. Bousquet, S. Gelly, and N. Houlsby, “The visual task adaptation benchmark,” 2019.
- [104] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- [105] L. Matthey, I. Higgins, D. Hassabis, and A. Lerchner, “dsprites: Disentanglement testing sprites dataset.” <https://github.com/deepmind/dsprites-dataset/>, 2017.
- [106] Y. LeCun, F. J. Huang, and L. Bottou, “Learning methods for generic object recognition with invariance to pose and lighting,” *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. II–104 Vol.2, 2004.
- [107] G. Luo, M. Huang, Y. Zhou, X. Sun, G. Jiang, Z. Wang, and R. Ji, “Towards efficient visual adaption via structural re-parameterization,” *arXiv preprint arXiv:2302.08106*, 2023.
- [108] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, “The caltech-ucsd birds-200-2011 dataset,” 2011.
- [109] G. Van Horn, S. Branson, R. Farrell, S. Haber, J. Barry, P. Ipeirotis, P. Perona, and S. Belongie, “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.
- [110] M.-E. Nilsback and A. Zisserman, “Automated flower classification over a large number of classes,” in *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729, IEEE, 2008.
- [111] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-F. Li, “Novel dataset for fine-grained image categorization: Stanford dogs,” in *Proc. CVPR workshop on fine-grained visual categorization (FGVC)*, vol. 2, 2011.

- [112] T. Gebru, J. Krause, Y. Wang, D. Chen, J. Deng, and L. Fei-Fei, “Fine-grained car detection for visual census estimation,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.
- [113] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, “Scene parsing through ade20k dataset,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 633–641, 2017.
- [114] N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman, “Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 22500–22510, 2023.
- [115] A. Williams, N. Nangia, and S. R. Bowman, “A broad-coverage challenge corpus for sentence understanding through inference,” in *Proceedings of NAACL-HLT*, 2018.
- [116] H. J. Levesque, E. Davis, and L. Morgenstern, “The Winograd schema challenge,” in *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, vol. 46, p. 47, 2011.
- [117] A. Warstadt, A. Singh, and S. R. Bowman, “Neural network acceptability judgments,” *arXiv preprint 1805.12471*, 2018.
- [118] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, “Recursive deep models for semantic compositionality over a sentiment treebank,” in *Proceedings of EMNLP*, pp. 1631–1642, 2013.
- [119] W. B. Dolan and C. Brockett, “Automatically constructing a corpus of sentential paraphrases,” in *Proceedings of the International Workshop on Paraphrasing*, 2005.
- [120] E. Agirre, L. M’arquez, and R. Wicentowski, eds., *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*. Prague, Czech Republic: Association for Computational Linguistics, June 2007.
- [121] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “SQuAD: 100,000+ questions for machine comprehension of text,” in *Proceedings of EMNLP*, pp. 2383–2392, Association for Computational Linguistics, 2016.
- [122] I. Dagan, O. Glickman, and B. Magnini, “The PASCAL recognising textual entailment challenge,” in *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising textual entailment*, pp. 177–190, Springer, 2006.
- [123] R. Bar Haim, I. Dagan, B. Dolan, L. Ferro, D. Giampiccolo, B. Magnini, and I. Szpektor, “The second PASCAL recognising textual entailment challenge,” 2006.
- [124] D. Giampiccolo, B. Magnini, I. Dagan, and B. Dolan, “The third PASCAL recognizing textual entailment challenge,” in *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pp. 1–9, Association for Computational Linguistics, 2007.

- [125] L. Bentivogli, I. Dagan, H. T. Dang, D. Giampiccolo, and B. Magnini, “The fifth PASCAL recognizing textual entailment challenge,” 2009.
- [126] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.
- [127] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (Y. W. Teh and M. Titterton, eds.), vol. 9 of *Proceedings of Machine Learning Research*, (Chia Laguna Resort, Sardinia, Italy), pp. 249–256, PMLR, 13–15 May 2010.
- [128] M. Contributors, “MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark.” <https://github.com/open-mmlab/mms Segmentation>, 2020.
- [129] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, *et al.*, “Huggingface’s transformers: State-of-the-art natural language processing,” *arXiv preprint arXiv:1910.03771*, 2019.
- [130] I. Loshchilov and F. Hutter, “Fixing weight decay regularization in adam,” 2018.
- [131] R. Van Gastel, “Finetuning dinov2 and dinov3 with lora for image segmentation,” 2024.

Appendix A

Supplementary Results

A.1 Detailed Tabulated Results for Pooled & Demographic Adaptation

Table A.1: Performance of models pretrained on CXR and adapted to the CXR dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.

Model	Adaptation		AUC				Δ AUC	FPR Gap No Finding
	Method	Strategy	Aggregate	Asian	Black	White		
MAE (CXR)	E2VPT	D	81.6 $\pm$ 0.06	80.0 $\pm$ 0.21	81.5 $\pm$ 0.17	81.5 $\pm$ 0.06	14.2 $\pm$ 0.44	1.6 $\pm$ 0.24
		P	81.0 $\pm$ 0.05	80.4 $\pm$ 0.21	82.3 $\pm$ 0.16	80.6 $\pm$ 0.06	14.6 $\pm$ 0.45	2.0 $\pm$ 0.21
	Full	D	82.2 $\pm$ 0.06	80.7 $\pm$ 0.23	81.9 $\pm$ 0.17	82.2 $\pm$ 0.06	14.3 $\pm$ 0.45	1.5 $\pm$ 0.23
		P	81.1 $\pm$ 0.06	81.5 $\pm$ 0.21	82.1 $\pm$ 0.19	80.6 $\pm$ 0.06	14.5 $\pm$ 0.37	1.5 $\pm$ 0.18
	GVPT	D	81.5 $\pm$ 0.06	79.9 $\pm$ 0.20	81.3 $\pm$ 0.17	81.5 $\pm$ 0.06	16.8 $\pm$ 0.42	1.5 $\pm$ 0.24
		P	81.7 $\pm$ 0.06	81.2 $\pm$ 0.21	82.8 $\pm$ 0.17	81.3 $\pm$ 0.06	12.7 $\pm$ 0.41	1.6 $\pm$ 0.20
	Linear	D	78.9 $\pm$ 0.06	78.8 $\pm$ 0.21	79.7 $\pm$ 0.18	79.0 $\pm$ 0.06	11.3 $\pm$ 0.34	1.0 $\pm$ 0.23
		P	79.3 $\pm$ 0.06	79.3 $\pm$ 0.21	80.4 $\pm$ 0.17	78.8 $\pm$ 0.06	15.0 $\pm$ 0.41	1.6 $\pm$ 0.20
	VPT	D	80.7 $\pm$ 0.05	79.6 $\pm$ 0.20	81.3 $\pm$ 0.16	80.5 $\pm$ 0.06	14.6 $\pm$ 0.44	1.8 $\pm$ 0.26
		P	81.2 $\pm$ 0.05	80.7 $\pm$ 0.20	82.5 $\pm$ 0.16	80.7 $\pm$ 0.06	14.2 $\pm$ 0.42	1.7 $\pm$ 0.20
MoCoV3 (CXR)	E2VPT	D	80.5 $\pm$ 0.06	79.1 $\pm$ 0.20	80.5 $\pm$ 0.17	80.5 $\pm$ 0.07	9.3 $\pm$ 0.40	1.5 $\pm$ 0.24
		P	80.7 $\pm$ 0.06	80.0 $\pm$ 0.21	81.4 $\pm$ 0.18	80.3 $\pm$ 0.07	13.0 $\pm$ 0.44	1.3 $\pm$ 0.21
	Full	D	80.6 $\pm$ 0.06	79.8 $\pm$ 0.24	81.0 $\pm$ 0.19	80.4 $\pm$ 0.06	15.2 $\pm$ 0.44	1.1 $\pm$ 0.21
		P	81.6 $\pm$ 0.06	82.0 $\pm$ 0.21	82.5 $\pm$ 0.18	81.1 $\pm$ 0.07	12.2 $\pm$ 0.42	1.4 $\pm$ 0.16
	GVPT	D	80.0 $\pm$ 0.06	78.8 $\pm$ 0.22	80.4 $\pm$ 0.18	79.9 $\pm$ 0.06	17.2 $\pm$ 0.32	1.6 $\pm$ 0.26
		P	80.2 $\pm$ 0.06	79.5 $\pm$ 0.22	81.2 $\pm$ 0.18	79.8 $\pm$ 0.06	13.8 $\pm$ 0.46	1.7 $\pm$ 0.23
	Linear	D	79.1 $\pm$ 0.06	77.8 $\pm$ 0.22	79.9 $\pm$ 0.18	78.8 $\pm$ 0.07	15.8 $\pm$ 0.44	2.0 $\pm$ 0.28
		P	79.2 $\pm$ 0.06	78.8 $\pm$ 0.22	80.2 $\pm$ 0.17	78.8 $\pm$ 0.07	14.1 $\pm$ 0.45	1.5 $\pm$ 0.22
	VPT	D	81.2 $\pm$ 0.06	79.1 $\pm$ 0.20	80.7 $\pm$ 0.17	81.2 $\pm$ 0.06	7.8 $\pm$ 0.32	2.1 $\pm$ 0.23
		P	81.5 $\pm$ 0.06	81.1 $\pm$ 0.19	82.5 $\pm$ 0.17	81.1 $\pm$ 0.07	12.5 $\pm$ 0.44	1.3 $\pm$ 0.18

Table A.2: Performance of models pretrained on IN and adapted to the CXR dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.

Model	Adaptation		AUC				Δ AUC	FPR Gap No Finding
	Method	Strategy	Aggregate	Asian	Black	White		
MAE (IN)	E2VPT	D	80.2 $\pm$ 0.07	77.1 $\pm$ 0.22	78.8 $\pm$ 0.18	80.5 $\pm$ 0.06	8.6 $\pm$ 0.42	3.4 $\pm$ 0.24
		P	79.1 $\pm$ 0.07	78.2 $\pm$ 0.21	79.8 $\pm$ 0.19	78.7 $\pm$ 0.07	14.3 $\pm$ 0.43	1.6 $\pm$ 0.25
	Full	D	81.1 $\pm$ 0.06	78.8 $\pm$ 0.23	80.9 $\pm$ 0.19	81.1 $\pm$ 0.06	12.9 $\pm$ 0.43	2.3 $\pm$ 0.24
		P	83.1 $\pm$ 0.06	83.0 $\pm$ 0.22	84.3 $\pm$ 0.16	82.7 $\pm$ 0.06	12.0 $\pm$ 0.37	1.6 $\pm$ 0.16
	GVPT	D	79.1 $\pm$ 0.06	77.2 $\pm$ 0.21	78.7 $\pm$ 0.19	79.2 $\pm$ 0.06	9.8 $\pm$ 0.42	2.0 $\pm$ 0.23
		P	79.2 $\pm$ 0.06	78.8 $\pm$ 0.20	80.2 $\pm$ 0.16	78.7 $\pm$ 0.06	14.6 $\pm$ 0.42	1.5 $\pm$ 0.20
	Linear	D	72.9 $\pm$ 0.06	73.6 $\pm$ 0.23	74.8 $\pm$ 0.19	72.8 $\pm$ 0.07	33.8 $\pm$ 0.44	2.0 $\pm$ 0.21
		P	72.5 $\pm$ 0.07	72.3 $\pm$ 0.22	73.1 $\pm$ 0.20	72.0 $\pm$ 0.07	20.3 $\pm$ 0.39	1.1 $\pm$ 0.22
	VPT	D	78.0 $\pm$ 0.06	75.6 $\pm$ 0.25	77.9 $\pm$ 0.19	77.9 $\pm$ 0.07	14.8 $\pm$ 0.43	2.3 $\pm$ 0.26
		P	79.0 $\pm$ 0.06	78.6 $\pm$ 0.22	79.7 $\pm$ 0.19	78.5 $\pm$ 0.06	15.3 $\pm$ 0.40	1.2 $\pm$ 0.22
MoCoV3 (IN)	E2VPT	D	79.4 $\pm$ 0.06	77.2 $\pm$ 0.21	79.3 $\pm$ 0.19	79.5 $\pm$ 0.07	10.5 $\pm$ 0.41	2.3 $\pm$ 0.25
		P	78.8 $\pm$ 0.07	78.3 $\pm$ 0.19	79.6 $\pm$ 0.20	78.4 $\pm$ 0.07	11.8 $\pm$ 0.39	1.3 $\pm$ 0.23
	Full	D	79.7 $\pm$ 0.06	78.2 $\pm$ 0.22	79.9 $\pm$ 0.19	79.6 $\pm$ 0.06	17.9 $\pm$ 0.38	1.7 $\pm$ 0.27
		P	81.7 $\pm$ 0.06	82.4 $\pm$ 0.16	82.7 $\pm$ 0.17	81.2 $\pm$ 0.06	11.8 $\pm$ 0.38	1.5 $\pm$ 0.16
	GVPT	D	78.7 $\pm$ 0.06	76.9 $\pm$ 0.21	78.9 $\pm$ 0.18	78.6 $\pm$ 0.06	12.3 $\pm$ 0.35	1.9 $\pm$ 0.27
		P	78.6 $\pm$ 0.06	77.9 $\pm$ 0.21	79.6 $\pm$ 0.19	78.1 $\pm$ 0.07	14.2 $\pm$ 0.39	1.7 $\pm$ 0.24
	Linear	D	74.5 $\pm$ 0.06	74.1 $\pm$ 0.23	75.8 $\pm$ 0.18	74.0 $\pm$ 0.07	18.6 $\pm$ 0.43	1.8 $\pm$ 0.24
		P	74.3 $\pm$ 0.06	74.3 $\pm$ 0.22	75.2 $\pm$ 0.19	73.8 $\pm$ 0.07	18.1 $\pm$ 0.42	1.4 $\pm$ 0.23
	VPT	D	80.0 $\pm$ 0.06	77.0 $\pm$ 0.20	79.1 $\pm$ 0.19	80.3 $\pm$ 0.06	16.0 $\pm$ 0.41	3.3 $\pm$ 0.22
		P	80.6 $\pm$ 0.06	80.1 $\pm$ 0.19	81.5 $\pm$ 0.17	80.1 $\pm$ 0.06	14.3 $\pm$ 0.42	1.4 $\pm$ 0.22
ViT (IN)	E2VPT	D	79.1 $\pm$ 0.06	76.5 $\pm$ 0.23	78.8 $\pm$ 0.17	79.5 $\pm$ 0.06	9.5 $\pm$ 0.33	3.1 $\pm$ 0.24
		P	80.2 $\pm$ 0.06	79.6 $\pm$ 0.21	80.8 $\pm$ 0.16	79.8 $\pm$ 0.06	14.0 $\pm$ 0.37	1.2 $\pm$ 0.21
	Full	D	78.4 $\pm$ 0.07	76.4 $\pm$ 0.21	77.5 $\pm$ 0.20	78.7 $\pm$ 0.07	4.8 $\pm$ 0.35	2.2 $\pm$ 0.21
		P	75.6 $\pm$ 0.07	75.4 $\pm$ 0.23	77.2 $\pm$ 0.21	75.0 $\pm$ 0.08	11.2 $\pm$ 0.44	2.2 $\pm$ 0.25
	GVPT	D	78.5 $\pm$ 0.06	76.2 $\pm$ 0.26	78.4 $\pm$ 0.18	78.7 $\pm$ 0.07	9.0 $\pm$ 0.28	2.5 $\pm$ 0.26
		P	79.0 $\pm$ 0.07	78.7 $\pm$ 0.21	79.8 $\pm$ 0.18	78.6 $\pm$ 0.07	15.3 $\pm$ 0.39	1.2 $\pm$ 0.19
	Linear	D	74.4 $\pm$ 0.06	73.5 $\pm$ 0.26	75.0 $\pm$ 0.20	73.9 $\pm$ 0.07	18.9 $\pm$ 0.40	1.5 $\pm$ 0.30
		P	74.4 $\pm$ 0.06	74.3 $\pm$ 0.25	75.3 $\pm$ 0.20	73.9 $\pm$ 0.07	18.4 $\pm$ 0.42	1.4 $\pm$ 0.24
	VPT	D	77.8 $\pm$ 0.06	75.7 $\pm$ 0.24	77.5 $\pm$ 0.18	77.7 $\pm$ 0.07	13.1 $\pm$ 0.37	2.0 $\pm$ 0.26
		P	78.4 $\pm$ 0.06	77.7 $\pm$ 0.22	79.4 $\pm$ 0.18	77.9 $\pm$ 0.07	15.0 $\pm$ 0.37	1.7 $\pm$ 0.24

Table A.3: Performance of models pretrained on CXR and adapted to the CXR-1K dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.

Model	Adaptation		AUC				Δ AUC	FPR Gap No Finding
	Method	Strategy	Aggregate	Asian	Black	White		
MAE (CXR)	E2VPT	D	71.8 $\pm$ 0.07	72.9 $\pm$ 0.23	74.1 $\pm$ 0.17	71.0 $\pm$ 0.09	10.8 $\pm$ 0.43	3.1 $\pm$ 0.22
		P	72.4 $\pm$ 0.06	73.4 $\pm$ 0.24	73.2 $\pm$ 0.22	71.8 $\pm$ 0.07	16.0 $\pm$ 0.46	1.6 $\pm$ 0.19
	Full	D	69.0 $\pm$ 0.07	69.2 $\pm$ 0.30	69.4 $\pm$ 0.24	68.8 $\pm$ 0.08	9.7 $\pm$ 0.31	0.7 $\pm$ 0.21
		P	68.8 $\pm$ 0.07	69.0 $\pm$ 0.23	69.5 $\pm$ 0.26	68.3 $\pm$ 0.08	12.8 $\pm$ 0.49	1.2 $\pm$ 0.26
	GVPT	D	69.5 $\pm$ 0.07	72.2 $\pm$ 0.24	71.7 $\pm$ 0.20	68.9 $\pm$ 0.09	16.8 $\pm$ 0.45	3.3 $\pm$ 0.21
		P	70.5 $\pm$ 0.07	71.3 $\pm$ 0.24	71.1 $\pm$ 0.20	69.9 $\pm$ 0.07	14.9 $\pm$ 0.54	1.4 $\pm$ 0.19
	Linear	D	71.1 $\pm$ 0.07	72.7 $\pm$ 0.25	72.3 $\pm$ 0.20	70.4 $\pm$ 0.09	17.8 $\pm$ 0.39	2.3 $\pm$ 0.21
		P	71.8 $\pm$ 0.07	72.4 $\pm$ 0.26	72.3 $\pm$ 0.23	71.2 $\pm$ 0.07	16.8 $\pm$ 0.44	1.2 $\pm$ 0.21
	VPT	D	71.7 $\pm$ 0.07	73.1 $\pm$ 0.24	73.7 $\pm$ 0.20	70.9 $\pm$ 0.09	16.4 $\pm$ 0.43	2.8 $\pm$ 0.21
		P	72.0 $\pm$ 0.07	72.7 $\pm$ 0.26	72.6 $\pm$ 0.22	71.4 $\pm$ 0.07	14.5 $\pm$ 0.45	1.2 $\pm$ 0.20
MoCoV3 (CXR)	E2VPT	D	70.6 $\pm$ 0.07	72.0 $\pm$ 0.29	72.4 $\pm$ 0.20	70.0 $\pm$ 0.09	18.4 $\pm$ 0.46	2.4 $\pm$ 0.22
		P	73.2 $\pm$ 0.06	73.7 $\pm$ 0.24	74.9 $\pm$ 0.23	72.4 $\pm$ 0.08	16.4 $\pm$ 0.42	2.5 $\pm$ 0.25
	Full	D	67.9 $\pm$ 0.07	66.8 $\pm$ 0.25	68.3 $\pm$ 0.21	67.7 $\pm$ 0.09	13.1 $\pm$ 0.45	1.5 $\pm$ 0.33
		P	67.9 $\pm$ 0.07	67.8 $\pm$ 0.24	69.1 $\pm$ 0.23	67.4 $\pm$ 0.08	14.2 $\pm$ 0.43	1.7 $\pm$ 0.22
	GVPT	D	70.4 $\pm$ 0.08	71.7 $\pm$ 0.27	72.4 $\pm$ 0.19	69.7 $\pm$ 0.10	17.9 $\pm$ 0.41	2.7 $\pm$ 0.22
		P	71.8 $\pm$ 0.06	72.2 $\pm$ 0.28	73.3 $\pm$ 0.25	71.1 $\pm$ 0.07	14.3 $\pm$ 0.44	2.2 $\pm$ 0.23
	Linear	D	70.7 $\pm$ 0.07	71.9 $\pm$ 0.27	72.1 $\pm$ 0.22	70.1 $\pm$ 0.09	20.2 $\pm$ 0.40	2.0 $\pm$ 0.21
		P	72.0 $\pm$ 0.06	72.1 $\pm$ 0.26	73.7 $\pm$ 0.23	71.3 $\pm$ 0.08	13.9 $\pm$ 0.41	2.4 $\pm$ 0.22
	VPT	D	71.4 $\pm$ 0.08	72.2 $\pm$ 0.28	71.9 $\pm$ 0.21	71.0 $\pm$ 0.09	17.9 $\pm$ 0.39	1.2 $\pm$ 0.24
		P	70.1 $\pm$ 0.08	70.4 $\pm$ 0.23	72.2 $\pm$ 0.21	69.3 $\pm$ 0.10	11.9 $\pm$ 0.48	2.9 $\pm$ 0.23

Table A.4: Performance of models pretrained on IN and adapted to the CXR-1K dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.

Model	Adaptation		AUC				Δ AUC	FPR Gap No Finding
	Method	Strategy	Aggregate	Asian	Black	White		
MAE (IN)	E2VPT	D	66.8 $\pm$ 0.08	67.5 $\pm$ 0.26	70.0 $\pm$ 0.23	66.1 $\pm$ 0.08	21.3 $\pm$ 0.34	3.9 $\pm$ 0.24
		P	68.5 $\pm$ 0.07	67.9 $\pm$ 0.28	70.0 $\pm$ 0.26	67.9 $\pm$ 0.07	13.4 $\pm$ 0.44	2.0 $\pm$ 0.27
	Full	D	65.0 $\pm$ 0.08	64.4 $\pm$ 0.30	66.5 $\pm$ 0.23	64.6 $\pm$ 0.08	20.0 $\pm$ 0.44	2.1 $\pm$ 0.32
		P	64.6 $\pm$ 0.07	64.2 $\pm$ 0.28	65.6 $\pm$ 0.23	64.1 $\pm$ 0.08	14.7 $\pm$ 0.44	1.6 $\pm$ 0.29
	GVPT	D	66.3 $\pm$ 0.07	65.5 $\pm$ 0.30	68.6 $\pm$ 0.21	65.7 $\pm$ 0.07	19.7 $\pm$ 0.32	3.1 $\pm$ 0.29
		P	66.3 $\pm$ 0.07	65.6 $\pm$ 0.26	66.8 $\pm$ 0.21	65.8 $\pm$ 0.08	29.2 $\pm$ 0.42	1.2 $\pm$ 0.27
	Linear	D	65.5 $\pm$ 0.08	66.5 $\pm$ 0.28	67.4 $\pm$ 0.20	64.6 $\pm$ 0.09	13.7 $\pm$ 0.45	2.7 $\pm$ 0.24
		P	68.0 $\pm$ 0.08	67.8 $\pm$ 0.26	68.7 $\pm$ 0.22	67.4 $\pm$ 0.08	24.9 $\pm$ 0.49	1.2 $\pm$ 0.21
	VPT	D	66.4 $\pm$ 0.07	67.2 $\pm$ 0.28	69.9 $\pm$ 0.22	65.3 $\pm$ 0.07	13.8 $\pm$ 0.45	4.7 $\pm$ 0.25
		P	67.8 $\pm$ 0.07	68.1 $\pm$ 0.27	69.4 $\pm$ 0.26	67.1 $\pm$ 0.09	15.3 $\pm$ 0.45	2.3 $\pm$ 0.24
MoCoV3 (IN)	E2VPT	D	66.7 $\pm$ 0.07	65.2 $\pm$ 0.25	68.5 $\pm$ 0.23	66.2 $\pm$ 0.08	6.0 $\pm$ 0.35	3.3 $\pm$ 0.34
		P	69.1 $\pm$ 0.07	68.9 $\pm$ 0.28	71.1 $\pm$ 0.24	68.3 $\pm$ 0.07	15.6 $\pm$ 0.43	2.8 $\pm$ 0.25
	Full	D	64.1 $\pm$ 0.07	63.1 $\pm$ 0.27	66.9 $\pm$ 0.22	63.0 $\pm$ 0.08	16.2 $\pm$ 0.43	3.9 $\pm$ 0.27
		P	64.0 $\pm$ 0.06	63.9 $\pm$ 0.29	64.7 $\pm$ 0.23	63.7 $\pm$ 0.08	9.2 $\pm$ 0.33	1.0 $\pm$ 0.23
	GVPT	D	66.1 $\pm$ 0.07	66.8 $\pm$ 0.26	68.8 $\pm$ 0.20	65.4 $\pm$ 0.08	19.2 $\pm$ 0.35	3.4 $\pm$ 0.24
		P	67.7 $\pm$ 0.07	67.0 $\pm$ 0.28	69.5 $\pm$ 0.24	67.1 $\pm$ 0.08	10.6 $\pm$ 0.45	2.5 $\pm$ 0.32
	Linear	D	67.8 $\pm$ 0.07	67.0 $\pm$ 0.24	69.3 $\pm$ 0.20	67.2 $\pm$ 0.09	21.9 $\pm$ 0.44	2.4 $\pm$ 0.29
		P	69.3 $\pm$ 0.07	68.7 $\pm$ 0.28	70.1 $\pm$ 0.22	68.8 $\pm$ 0.08	17.3 $\pm$ 0.48	1.4 $\pm$ 0.26
	VPT	D	66.2 $\pm$ 0.07	66.8 $\pm$ 0.24	69.3 $\pm$ 0.21	65.1 $\pm$ 0.08	16.2 $\pm$ 0.43	4.3 $\pm$ 0.24
		P	66.9 $\pm$ 0.07	67.1 $\pm$ 0.22	68.8 $\pm$ 0.21	66.2 $\pm$ 0.08	14.1 $\pm$ 0.42	2.5 $\pm$ 0.22
ViT (IN)	E2VPT	D	66.6 $\pm$ 0.08	67.4 $\pm$ 0.27	68.6 $\pm$ 0.21	65.8 $\pm$ 0.08	24.9 $\pm$ 0.38	2.8 $\pm$ 0.26
		P	66.5 $\pm$ 0.07	65.2 $\pm$ 0.26	67.8 $\pm$ 0.23	66.0 $\pm$ 0.08	10.8 $\pm$ 0.43	2.6 $\pm$ 0.36
	Full	D	66.6 $\pm$ 0.07	64.8 $\pm$ 0.27	66.7 $\pm$ 0.22	66.2 $\pm$ 0.08	11.2 $\pm$ 0.47	1.9 $\pm$ 0.32
		P	65.3 $\pm$ 0.08	64.7 $\pm$ 0.26	65.5 $\pm$ 0.25	65.0 $\pm$ 0.09	15.4 $\pm$ 0.39	0.8 $\pm$ 0.28
	GVPT	D	64.9 $\pm$ 0.08	64.9 $\pm$ 0.28	66.8 $\pm$ 0.22	64.4 $\pm$ 0.08	38.2 $\pm$ 0.45	2.4 $\pm$ 0.25
		P	64.7 $\pm$ 0.08	65.3 $\pm$ 0.28	65.9 $\pm$ 0.23	64.1 $\pm$ 0.08	12.8 $\pm$ 0.54	1.8 $\pm$ 0.21
	Linear	D	66.7 $\pm$ 0.07	66.8 $\pm$ 0.27	68.7 $\pm$ 0.18	65.9 $\pm$ 0.09	22.9 $\pm$ 0.52	2.7 $\pm$ 0.21
		P	67.5 $\pm$ 0.07	66.9 $\pm$ 0.28	67.3 $\pm$ 0.20	67.1 $\pm$ 0.07	20.4 $\pm$ 0.47	0.5 $\pm$ 0.20
	VPT	D	66.7 $\pm$ 0.07	66.6 $\pm$ 0.28	69.6 $\pm$ 0.20	65.7 $\pm$ 0.09	12.7 $\pm$ 0.47	3.8 $\pm$ 0.24
		P	64.8 $\pm$ 0.08	64.0 $\pm$ 0.24	65.5 $\pm$ 0.22	64.3 $\pm$ 0.09	14.6 $\pm$ 0.46	1.5 $\pm$ 0.32

Table A.5: Performance of models pretrained on IN and adapted to the CelebA dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.

Model	Adaptation		AUC			Δ AUC
	Method	Strategy	Aggregate	Male	Female	
MAE (IN)	E2VPT	D	93.0 $\pm$ 0.02	92.1 $\pm$ 0.05	92.1 $\pm$ 0.03	0.0 $\pm$ 0.04
		P	93.2 $\pm$ 0.02	92.4 $\pm$ 0.07	92.3 $\pm$ 0.03	0.1 $\pm$ 0.05
	Full	D	92.8 $\pm$ 0.02	91.9 $\pm$ 0.08	91.8 $\pm$ 0.03	0.2 $\pm$ 0.06
		P	93.1 $\pm$ 0.02	92.5 $\pm$ 0.05	92.1 $\pm$ 0.03	0.4 $\pm$ 0.06
	GVPT	D	90.9 $\pm$ 0.02	89.7 $\pm$ 0.06	89.2 $\pm$ 0.04	0.5 $\pm$ 0.07
		P	90.5 $\pm$ 0.02	88.7 $\pm$ 0.06	89.0 $\pm$ 0.04	0.2 $\pm$ 0.07
	Linear	D	87.1 $\pm$ 0.03	85.2 $\pm$ 0.07	85.0 $\pm$ 0.05	0.3 $\pm$ 0.08
		P	86.2 $\pm$ 0.03	83.0 $\pm$ 0.09	84.0 $\pm$ 0.05	1.0 $\pm$ 0.11
	VPT	D	90.6 $\pm$ 0.02	89.6 $\pm$ 0.05	89.0 $\pm$ 0.04	0.6 $\pm$ 0.06
		P	90.4 $\pm$ 0.02	88.5 $\pm$ 0.07	88.8 $\pm$ 0.04	0.3 $\pm$ 0.07
MoCoV3 (IN)	E2VPT	D	92.5 $\pm$ 0.02	91.4 $\pm$ 0.10	91.5 $\pm$ 0.03	0.1 $\pm$ 0.07
		P	91.3 $\pm$ 0.02	89.7 $\pm$ 0.08	90.0 $\pm$ 0.03	0.3 $\pm$ 0.08
	Full	D	92.7 $\pm$ 0.02	91.8 $\pm$ 0.10	91.5 $\pm$ 0.03	0.3 $\pm$ 0.08
		P	92.5 $\pm$ 0.02	91.7 $\pm$ 0.05	91.4 $\pm$ 0.03	0.3 $\pm$ 0.06
	GVPT	D	90.9 $\pm$ 0.02	89.4 $\pm$ 0.11	89.2 $\pm$ 0.04	0.2 $\pm$ 0.08
		P	90.7 $\pm$ 0.02	89.3 $\pm$ 0.07	89.0 $\pm$ 0.04	0.3 $\pm$ 0.07
	Linear	D	88.3 $\pm$ 0.03	86.4 $\pm$ 0.08	86.5 $\pm$ 0.04	0.1 $\pm$ 0.05
		P	88.1 $\pm$ 0.03	85.7 $\pm$ 0.08	86.3 $\pm$ 0.05	0.6 $\pm$ 0.09
	VPT	D	92.2 $\pm$ 0.02	90.8 $\pm$ 0.14	91.0 $\pm$ 0.03	0.2 $\pm$ 0.11
		P	92.4 $\pm$ 0.02	91.3 $\pm$ 0.06	91.3 $\pm$ 0.03	0.0 $\pm$ 0.04
ViT (IN)	E2VPT	D	92.9 $\pm$ 0.02	92.0 $\pm$ 0.05	91.9 $\pm$ 0.03	0.1 $\pm$ 0.05
		P	93.1 $\pm$ 0.02	92.3 $\pm$ 0.06	92.0 $\pm$ 0.03	0.3 $\pm$ 0.06
	Full	D	88.4 $\pm$ 0.03	89.1 $\pm$ 0.16	89.7 $\pm$ 0.04	0.6 $\pm$ 0.17
		P	90.8 $\pm$ 0.02	89.3 $\pm$ 0.16	89.6 $\pm$ 0.04	0.4 $\pm$ 0.15
	GVPT	D	92.5 $\pm$ 0.02	91.5 $\pm$ 0.06	91.3 $\pm$ 0.04	0.2 $\pm$ 0.06
		P	92.5 $\pm$ 0.02	91.6 $\pm$ 0.05	91.3 $\pm$ 0.04	0.3 $\pm$ 0.05
	Linear	D	90.1 $\pm$ 0.02	89.0 $\pm$ 0.06	88.5 $\pm$ 0.04	0.5 $\pm$ 0.07
		P	89.8 $\pm$ 0.03	88.3 $\pm$ 0.07	88.3 $\pm$ 0.04	0.1 $\pm$ 0.05
	VPT	D	92.3 $\pm$ 0.02	91.4 $\pm$ 0.06	91.1 $\pm$ 0.03	0.3 $\pm$ 0.06
		P	92.2 $\pm$ 0.02	91.3 $\pm$ 0.05	91.0 $\pm$ 0.03	0.3 $\pm$ 0.06

Table A.6: Performance of models pretrained on IN and adapted to the CelebA-1K dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.

Model	Adaptation		AUC			Δ AUC
	Method	Strategy	Aggregate	Male	Female	
MAE (IN)	E2VPT	D	86.6 ± 0.04	84.1 ± 0.21	84.1 ± 0.06	0.0 ± 0.14
		P	85.6 ± 0.03	82.9 ± 0.15	83.0 ± 0.06	0.0 ± 0.10
	Full	D	77.2 ± 0.05	80.3 ± 0.16	66.9 ± 0.11	13.5 ± 0.17
		P	70.1 ± 0.06	64.5 ± 0.17	66.8 ± 0.10	2.3 ± 0.17
	GVPT	D	75.5 ± 0.05	83.8 ± 0.21	56.4 ± 0.11	27.4 ± 0.25
		P	81.6 ± 0.04	77.8 ± 0.14	78.0 ± 0.08	0.2 ± 0.11
	Linear	D	80.8 ± 0.04	77.7 ± 0.20	78.7 ± 0.07	1.1 ± 0.19
		P	81.6 ± 0.04	77.7 ± 0.11	78.8 ± 0.07	1.1 ± 0.13
	VPT	D	86.1 ± 0.03	83.4 ± 0.19	82.4 ± 0.07	1.0 ± 0.19
		P	85.5 ± 0.04	82.9 ± 0.14	82.8 ± 0.05	0.1 ± 0.10
MoCoV3 (IN)	E2VPT	D	85.5 ± 0.03	83.3 ± 0.20	82.4 ± 0.06	0.9 ± 0.19
		P	84.5 ± 0.03	82.2 ± 0.13	81.3 ± 0.06	0.9 ± 0.17
	Full	D	84.8 ± 0.04	81.5 ± 0.10	82.9 ± 0.07	1.4 ± 0.12
		P	84.4 ± 0.04	81.9 ± 0.12	81.5 ± 0.06	0.4 ± 0.12
	GVPT	D	83.5 ± 0.04	71.1 ± 0.18	83.6 ± 0.06	12.6 ± 0.22
		P	85.9 ± 0.03	84.4 ± 0.11	83.0 ± 0.05	1.4 ± 0.15
	Linear	D	86.0 ± 0.03	82.8 ± 0.21	83.1 ± 0.06	0.3 ± 0.16
		P	85.3 ± 0.03	82.7 ± 0.08	82.8 ± 0.05	0.1 ± 0.06
	VPT	D	84.3 ± 0.03	71.1 ± 0.22	85.1 ± 0.06	14.0 ± 0.23
		P	87.4 ± 0.03	85.4 ± 0.11	84.9 ± 0.04	0.5 ± 0.13
ViT (IN)	E2VPT	D	87.5 ± 0.03	84.7 ± 0.17	85.2 ± 0.06	0.5 ± 0.16
		P	87.2 ± 0.03	85.4 ± 0.10	84.5 ± 0.06	1.0 ± 0.11
	Full	D	87.7 ± 0.03	85.5 ± 0.10	85.6 ± 0.07	0.1 ± 0.08
		P	86.8 ± 0.03	84.8 ± 0.12	84.2 ± 0.06	0.6 ± 0.11
	GVPT	D	87.0 ± 0.03	84.3 ± 0.16	84.0 ± 0.06	0.3 ± 0.13
		P	85.9 ± 0.03	83.7 ± 0.17	82.8 ± 0.06	0.9 ± 0.18
	Linear	D	85.8 ± 0.03	81.8 ± 0.15	82.9 ± 0.07	1.1 ± 0.13
		P	85.4 ± 0.03	82.7 ± 0.13	82.5 ± 0.06	0.2 ± 0.10
	VPT	D	87.8 ± 0.03	85.9 ± 0.20	84.7 ± 0.06	1.2 ± 0.20
		P	87.7 ± 0.03	86.1 ± 0.12	85.2 ± 0.05	0.9 ± 0.13

Table A.7: Performance of models pretrained on IN and adapted to the FairVLMed10k dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.

Model	Adaptation		AUC				Δ AUC
	Method	Strategy	Aggregate	Asian	Black	White	
MAE (IN)	E2VPT	D	80.7 $\pm$ 0.47	82.6 $\pm$ 1.54	75.0 $\pm$ 1.31	81.6 $\pm$ 0.51	7.6 $\pm$ 1.63
		P	80.8 $\pm$ 0.44	86.4 $\pm$ 1.30	75.6 $\pm$ 1.24	81.1 $\pm$ 0.49	10.8 $\pm$ 1.78
	Full	D	80.0 $\pm$ 0.46	81.5 $\pm$ 1.56	73.8 $\pm$ 1.43	81.2 $\pm$ 0.49	7.7 $\pm$ 1.64
		P	80.5 $\pm$ 0.45	82.6 $\pm$ 1.49	74.3 $\pm$ 1.28	81.1 $\pm$ 0.50	8.3 $\pm$ 1.64
	GVPT	D	80.2 $\pm$ 0.46	82.0 $\pm$ 1.52	73.4 $\pm$ 1.32	80.6 $\pm$ 0.51	8.6 $\pm$ 1.64
		P	81.2 $\pm$ 0.45	83.9 $\pm$ 1.39	78.0 $\pm$ 1.17	81.5 $\pm$ 0.52	6.0 $\pm$ 1.47
	Linear	D	66.0 $\pm$ 0.53	77.6 $\pm$ 1.65	72.3 $\pm$ 1.32	77.5 $\pm$ 0.56	5.3 $\pm$ 1.57
		P	77.4 $\pm$ 0.47	78.0 $\pm$ 1.64	75.2 $\pm$ 1.24	77.4 $\pm$ 0.56	2.7 $\pm$ 1.28
	VPT	D	79.5 $\pm$ 0.47	79.7 $\pm$ 1.63	76.9 $\pm$ 1.23	81.1 $\pm$ 0.52	4.2 $\pm$ 1.28
		P	79.8 $\pm$ 0.46	84.8 $\pm$ 1.36	74.8 $\pm$ 1.27	80.1 $\pm$ 0.51	10.0 $\pm$ 1.74
MoCoV3 (IN)	E2VPT	D	82.1 $\pm$ 0.43	86.8 $\pm$ 1.33	77.1 $\pm$ 1.20	82.7 $\pm$ 0.48	9.6 $\pm$ 1.64
		P	81.3 $\pm$ 0.44	86.6 $\pm$ 1.29	78.6 $\pm$ 1.14	81.2 $\pm$ 0.52	8.0 $\pm$ 1.51
	Full	D	79.8 $\pm$ 0.46	84.4 $\pm$ 1.44	73.8 $\pm$ 1.31	80.9 $\pm$ 0.49	10.7 $\pm$ 1.76
		P	81.0 $\pm$ 0.46	82.7 $\pm$ 1.49	79.4 $\pm$ 1.16	81.2 $\pm$ 0.50	3.3 $\pm$ 1.23
	GVPT	D	81.6 $\pm$ 0.44	87.0 $\pm$ 1.30	77.0 $\pm$ 1.26	81.8 $\pm$ 0.49	10.1 $\pm$ 1.71
		P	80.5 $\pm$ 0.45	85.1 $\pm$ 1.38	76.5 $\pm$ 1.21	80.5 $\pm$ 0.52	8.5 $\pm$ 1.64
	Linear	D	77.4 $\pm$ 0.50	81.6 $\pm$ 1.56	75.8 $\pm$ 1.22	81.0 $\pm$ 0.51	5.8 $\pm$ 1.48
		P	80.1 $\pm$ 0.46	81.4 $\pm$ 1.59	77.4 $\pm$ 1.16	80.4 $\pm$ 0.52	3.9 $\pm$ 1.37
	VPT	D	81.7 $\pm$ 0.46	85.3 $\pm$ 1.37	75.6 $\pm$ 1.27	82.5 $\pm$ 0.49	9.7 $\pm$ 1.64
		P	81.8 $\pm$ 0.44	86.4 $\pm$ 1.28	77.7 $\pm$ 1.15	82.0 $\pm$ 0.49	8.7 $\pm$ 1.54
ViT (IN)	E2VPT	D	79.7 $\pm$ 0.46	82.2 $\pm$ 1.58	74.7 $\pm$ 1.31	80.4 $\pm$ 0.51	7.5 $\pm$ 1.66
		P	78.5 $\pm$ 0.47	84.9 $\pm$ 1.33	74.5 $\pm$ 1.27	78.7 $\pm$ 0.52	10.5 $\pm$ 1.78
	Full	D	78.3 $\pm$ 0.47	81.1 $\pm$ 1.59	70.0 $\pm$ 1.42	81.2 $\pm$ 0.50	11.2 $\pm$ 1.68
		P	81.3 $\pm$ 0.44	83.3 $\pm$ 1.49	75.9 $\pm$ 1.19	82.2 $\pm$ 0.48	7.4 $\pm$ 1.53
	GVPT	D	79.6 $\pm$ 0.46	80.6 $\pm$ 1.57	74.2 $\pm$ 1.31	79.9 $\pm$ 0.52	6.4 $\pm$ 1.57
		P	76.0 $\pm$ 0.50	82.6 $\pm$ 1.54	70.2 $\pm$ 1.38	76.5 $\pm$ 0.56	12.4 $\pm$ 1.96
	Linear	D	75.3 $\pm$ 0.51	75.6 $\pm$ 1.70	70.6 $\pm$ 1.38	75.7 $\pm$ 0.56	5.2 $\pm$ 1.54
		P	75.0 $\pm$ 0.51	76.4 $\pm$ 1.74	69.4 $\pm$ 1.45	75.6 $\pm$ 0.56	7.0 $\pm$ 1.66
	VPT	D	75.7 $\pm$ 0.49	80.2 $\pm$ 1.58	72.4 $\pm$ 1.32	79.5 $\pm$ 0.52	7.8 $\pm$ 1.60
		P	77.6 $\pm$ 0.47	82.7 $\pm$ 1.60	72.6 $\pm$ 1.30	78.1 $\pm$ 0.53	10.0 $\pm$ 1.87

Table A.8: Performance of models pretrained on IN and adapted to the GF3300 dataset. Values are reported as mean $\pm$ standard deviation estimated from 1000 bootstrap resamples. Adaptation strategies: P = pooled, D = demographic-specific.

Model	Adaptation		AUC				Δ AUC
	Method	Strategy	Aggregate	Asian	Black	White	
MAE (IN)	E2VPT	D	86.9 $\pm$ 0.54	89.3 $\pm$ 0.89	83.4 $\pm$ 1.06	87.1 $\pm$ 0.93	5.9 $\pm$ 1.20
		P	86.0 $\pm$ 0.56	89.7 $\pm$ 0.89	81.8 $\pm$ 1.08	85.4 $\pm$ 0.95	7.9 $\pm$ 1.29
	Full	D	80.3 $\pm$ 0.64	89.7 $\pm$ 0.87	79.9 $\pm$ 1.14	87.4 $\pm$ 0.93	9.8 $\pm$ 1.34
		P	85.9 $\pm$ 0.56	89.1 $\pm$ 0.88	80.8 $\pm$ 1.12	86.5 $\pm$ 0.89	8.3 $\pm$ 1.33
	GVPT	D	86.9 $\pm$ 0.54	89.8 $\pm$ 0.87	82.8 $\pm$ 1.07	87.7 $\pm$ 0.88	7.0 $\pm$ 1.25
		P	85.8 $\pm$ 0.55	89.4 $\pm$ 0.90	81.3 $\pm$ 1.12	85.6 $\pm$ 0.97	8.1 $\pm$ 1.33
	Linear	D	85.3 $\pm$ 0.59	87.7 $\pm$ 0.95	81.8 $\pm$ 1.09	85.2 $\pm$ 1.01	5.9 $\pm$ 1.23
		P	85.2 $\pm$ 0.59	87.5 $\pm$ 0.97	82.4 $\pm$ 1.07	85.0 $\pm$ 1.00	5.0 $\pm$ 1.19
	VPT	D	86.8 $\pm$ 0.55	89.4 $\pm$ 0.84	82.2 $\pm$ 1.09	87.1 $\pm$ 0.92	7.2 $\pm$ 1.24
		P	85.1 $\pm$ 0.59	89.0 $\pm$ 0.88	81.1 $\pm$ 1.14	84.3 $\pm$ 1.00	7.9 $\pm$ 1.31
MoCoV3 (IN)	E2VPT	D	84.7 $\pm$ 0.59	89.8 $\pm$ 0.86	82.8 $\pm$ 1.07	88.7 $\pm$ 0.86	7.0 $\pm$ 1.23
		P	86.1 $\pm$ 0.54	89.1 $\pm$ 0.90	80.5 $\pm$ 1.10	87.0 $\pm$ 0.90	8.6 $\pm$ 1.29
	Full	D	85.8 $\pm$ 0.56	88.2 $\pm$ 0.92	81.8 $\pm$ 1.08	87.0 $\pm$ 0.96	6.5 $\pm$ 1.23
		P	84.5 $\pm$ 0.59	87.9 $\pm$ 0.94	80.2 $\pm$ 1.15	84.3 $\pm$ 1.02	7.7 $\pm$ 1.31
	GVPT	D	86.3 $\pm$ 0.55	89.4 $\pm$ 0.85	82.5 $\pm$ 1.06	87.8 $\pm$ 0.87	6.9 $\pm$ 1.21
		P	85.7 $\pm$ 0.55	88.8 $\pm$ 0.92	81.0 $\pm$ 1.09	86.0 $\pm$ 0.94	7.9 $\pm$ 1.31
	Linear	D	85.2 $\pm$ 0.57	87.9 $\pm$ 0.93	80.5 $\pm$ 1.12	86.0 $\pm$ 0.93	7.4 $\pm$ 1.33
		P	85.6 $\pm$ 0.56	88.1 $\pm$ 0.95	81.0 $\pm$ 1.08	86.2 $\pm$ 0.92	7.1 $\pm$ 1.29
	VPT	D	85.6 $\pm$ 0.55	88.8 $\pm$ 0.89	82.3 $\pm$ 1.10	87.3 $\pm$ 0.86	6.5 $\pm$ 1.25
		P	86.1 $\pm$ 0.55	89.1 $\pm$ 0.89	80.5 $\pm$ 1.11	87.3 $\pm$ 0.88	8.6 $\pm$ 1.30
ViT (IN)	E2VPT	D	86.5 $\pm$ 0.54	90.5 $\pm$ 0.83	83.4 $\pm$ 1.06	88.6 $\pm$ 0.87	7.1 $\pm$ 1.24
		P	85.2 $\pm$ 0.57	89.5 $\pm$ 0.87	79.9 $\pm$ 1.10	85.2 $\pm$ 1.04	9.7 $\pm$ 1.32
	Full	D	86.5 $\pm$ 0.54	89.2 $\pm$ 0.91	80.9 $\pm$ 1.12	86.7 $\pm$ 0.96	8.3 $\pm$ 1.33
		P	84.8 $\pm$ 0.59	88.1 $\pm$ 0.99	80.5 $\pm$ 1.09	84.8 $\pm$ 1.01	7.6 $\pm$ 1.35
	GVPT	D	84.2 $\pm$ 0.59	89.1 $\pm$ 0.91	83.4 $\pm$ 1.05	88.3 $\pm$ 0.86	5.7 $\pm$ 1.18
		P	85.9 $\pm$ 0.57	88.7 $\pm$ 0.93	81.2 $\pm$ 1.09	86.6 $\pm$ 0.95	7.5 $\pm$ 1.31
	Linear	D	79.6 $\pm$ 0.66	88.1 $\pm$ 0.94	82.4 $\pm$ 1.09	85.0 $\pm$ 0.95	5.7 $\pm$ 1.24
		P	84.4 $\pm$ 0.58	87.2 $\pm$ 0.98	80.0 $\pm$ 1.15	85.0 $\pm$ 0.95	7.2 $\pm$ 1.36
	VPT	D	85.9 $\pm$ 0.56	89.1 $\pm$ 0.93	81.9 $\pm$ 1.08	87.8 $\pm$ 0.89	7.2 $\pm$ 1.27
		P	85.7 $\pm$ 0.56	89.5 $\pm$ 0.90	80.1 $\pm$ 1.11	86.5 $\pm$ 0.95	9.4 $\pm$ 1.35

A.2 VTAB-1K Detailed Tabulated Results

This section presents a comprehensive summary of our experimental results, complementing the findings reported in the main paper. For the VTAB-1K benchmark [90], we report the average performance over three independent runs for each dataset. Detailed results for the three VTAB-1K categories are provided in Tables table A.9 (Natural), table A.10 (Specialized), and table A.11 (Structured). The results for the majority of methods are reported in their respective papers, with the exception of LoRA [37] and FourierFT [54], which are cited from other sources.

Table A.9: ViT-B/16 top-1 accuracy on VTAB-1K Natural (avg. over 3 runs). Tuned / Total (%) indicates the percentage of tunable parameters. Rank (r), sinusoidal dimension (d), frequency scale (f), and scaling factor (α) are reported per dataset. **Bold** and underline indicate best and second-best performance.

ViT-Base/16 (85.8M)	VTAB-1K– Natural							Mean
	CIFAR100	caltech101	DTD	Flowers102	Pets	SVHN	Sun397	
Full [CVPR22]	68.9	87.7	64.3	97.2	86.9	87.4	38.8	75.9
Linear [CVPR22]	63.4	85.0	63.2	97.0	86.3	36.6	51.0	68.9
<i>Visual Prompt Tuning</i>								
VPT-S [ECCV22]	77.7	86.9	62.6	97.5	87.3	74.5	51.2	76.8
VPT-D [ECCV22]	78.8	90.8	65.8	98.0	88.3	78.1	49.6	78.5
E ² VPT [ICCV23]	78.6	89.4	67.8	98.2	88.5	85.3	52.3	80.0
VFPT [NeurIPS24]	80.7	91.4	69.4	<u>99.3</u>	90.3	85.6	52.7	81.3
<i>Reparameterization-Based Adaptation</i>								
LoRA [ICLR22]	72.5	<u>91.5</u>	71.9	99.1	91.4	89.6	56.0	81.7
SPT-LoRA [ICCV23]	73.5	93.3	72.5	99.3	<u>91.5</u>	87.9	55.5	81.9
FourierFT [ICML24]	<u>79.8</u>	90.4	68.8	98.3	89.9	85.0	53.1	80.8
FouRA [NeurIPS24]	72.2	91.4	70.1	99.1	91.0	87.0	55.7	80.9
LoRA-XS [ECAI25]	69.3	90.3	69.3	99.0	90.9	86.9	55.3	80.1
CDRA-SPT [CVPR25]	73.8	93.3	72.7	99.4	91.6	89.9	55.8	<u>82.4</u>
LoRA-Sine [ICLR25]	73.7	91.9	71.4	99.0	91.4	90.4	<u>56.2</u>	82.0
<i>Ours</i>								
LoRA-SPS (Efficient)	75.4±.27	90.6±.08	72.3±.36	99.0±.02	91.6±.09	<u>90.4±.11</u>	56.9±.21	82.3±.11
- Tuned / Total (%)	.35	.35	.11	.11	.10	.27	.48	.25
- $r/d/f/\alpha$	16/48/1024/0.01	8/96/512/0.01	8/24/1024/0.01	4/12/1024/0.01	16/12/512/0.01	8/96/1024/0.1	32/12/1024/0.1	
LoRA-SPS (Best)	75.4±.27	90.9±.22	<u>72.6±.33</u>	<u>99.3±.06</u>	91.6±.09	92.1±.05	56.9±.21	82.7±.10
- Tuned / Total (%)	.35	.48	.30	.35	.10	1.53	.48	.51
- $r/d/f/\alpha$	16/48/1024/0.01	96/12/128/0.1	32/24/1024/0.01	8/96/1024/0.1	16/12/512/0.01	96/48/512/1.0	32/12/1024/0.1	

Table A.10: ViT-B/16 top-1 accuracy on VTAB-1K Specialized (avg. over 3 runs). Tuned / Total (%) indicates the percentage of tunable parameters. Rank (r), sinusoidal dimension (d), frequency scale (f), and scaling factor (α) are reported per dataset. **Bold** and underline indicate best and second-best performance.

ViT-Base/16 (85.8M)	VTAB-1K – Specialized				Mean
	Camelyon	EuroSAT	Resisc45	Retinopathy	
Full [CVPR22]	79.7	95.7	84.2	73.9	83.4
Linear [CVPR22]	78.5	87.5	68.6	74.0	77.2
<i>Visual Prompt Tuning</i>					
VPT-S [ECCV22]	78.2	92.0	75.6	72.9	79.7
VPT-D [ECCV22]	81.8	96.1	83.4	68.4	82.4
E ² VPT [ICCV23]	82.5	96.8	84.8	73.6	84.4
VFPT [NeurIPS24]	83.5	<u>96.5</u>	84.4	75.4	85.0
<i>Reparameterization-Based Adaptation</i>					
LoRA [ICLR22]	<u>87.6</u>	95.3	84.0	75.0	85.5
SPT-LoRA [ICCV23]	85.7	96.2	85.9	75.9	85.9
FourierFT [ICML24]	82.7	95.2	82.6	74.9	83.9
FouRA [NeurIPS24]	87.5	95.1	84.4	75.0	85.5
LoRA-XS [ECAI25]	85.1	94.3	80.2	74.5	83.5
CDRA-SPT [CVPR25]	86.8	96.2	<u>86.2</u>	<u>76.0</u>	<u>86.3</u>
LoRA-Sine [CVPR25]	<u>87.6</u>	95.1	84.5	74.7	85.5
<i>Ours</i>					
LoRA-SPS (Efficient)	87.9 ±.22	95.2±.29	86.0±.30	<u>76.0</u> ±.08	<u>86.3</u> ±.09
- Tuned / Total (%)	.13	.07	.30	.07	.14
- $r/d/f/\alpha$	32/12/1024/0.1	16/12/1024/0.1	32/24/1024/0.1	8/24/1024/0.01	
LoRA-SPS (Best)	87.9 ±.22	95.5±.18	86.9 ±.34	76.2 ±.35	86.6 ±.07
- Tuned / Total (%)	.13	1.03	1.06	.39	.65
- $r/d/f/\alpha$	32/12/1024/0.1	64/48/256/0.1	64/48/1024/0.1	96/12/1024/0.1	

Table A.11: ViT-B/16 top-1 accuracy on VTAB-1K Structured (avg. over 3 runs). Tuned / Total (%) indicates the percentage of tunable parameters. Rank (r), sinusoidal dimension (d), frequency scale (f), and scaling factor (α) are reported per dataset. **Bold** and underline indicate best and second-best performance.

ViT-Base/16 (85.8M)	VTAB-1K – Structured								Mean
	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Ele	
Full [CVPR22]	56.3	58.6	41.7	65.5	57.5	46.7	25.7	29.1	47.6
Linear [CVPR22]	34.3	30.6	33.2	55.4	12.5	20.0	9.6	19.2	26.8
<i>Visual Prompt Tuning</i>									
VPT-S [ECCV22]	50.5	58.6	40.5	67.1	68.7	36.1	20.2	34.1	47.0
VPT-D [ECCV22]	68.5	60.0	46.5	72.8	73.6	47.9	32.9	37.8	55.0
E ² VPT [ICCV23]	71.7	61.2	47.9	75.8	80.8	48.1	31.7	41.9	57.4
VFPT [NeurIPS24]	75.8	63.2	48.3	79.3	81.5	<u>56.0</u>	34.1	43.4	60.2
<i>Reparameterization-Based Adaptation</i>									
LoRA [ICLR22]	83.6	64.3	51.6	80.9	86.0	51.8	<u>36.8</u>	42.3	62.2
SPT-LoRA [ICCV23]	84.4	<u>67.6</u>	52.5	<u>82.0</u>	81.0	51.1	30.2	41.3	61.3
FourierFT [ICML24]	73.5	60.1	43.0	76.9	71.2	47.6	28.1	29.4	53.7
FouRA [NeurIPS24]	83.1	63.9	52.0	81.1	85.6	51.7	<u>36.8</u>	41.7	62.0
LoRA-XS [ECAI25]	76.1	61.2	48.7	77.0	80.5	48.2	31.8	38.6	57.8
CDRA-SPT [CVPR25]	83.0	68.9	51.5	82.1	80.9	51.7	33.0	<u>43.2</u>	61.8
LoRA-Sine [ICLR25]	83.9	64.7	52.9	80.3	85.8	52.9	36.2	43.0	62.5
<i>Ours</i>									
LoRA-SPS (Efficient)	83.1±.22	64.8±.50	<u>55.3±.39</u>	80.8±.63	<u>88.6±.78</u>	56.6±.76	37.3±.19	42.1±.39	<u>63.6±.21</u>
Tuned / Total (%)	.26	.07	.26	.13	.27	.27	.40	.27	.24
- $r/d/f/\alpha$	64/12/256/0.01	16/12/128/0.01	32/24/1024/0.1	8/48/1024/0.1	32/24/1024/0.1	32/24/1024/0.01	96/12/512/1.0	64/12/128/0.1	
LoRA-SPS (Best)	<u>84.0±.38</u>	64.8±.50	55.9±.22	80.8±.63	88.7±.60	56.6±.76	37.3±.19	42.1±.39	63.8±.19
Tuned / Total (%)	2.03	.07	1.03	.13	1.03	.27	.40	.27	.65
- $r/d/f/\alpha$	64/96/256/0.01	16/12/128/0.01	32/96/1024/0.1	8/48/1024/0.1	64/48/512/0.1	32/24/1024/0.01	96/12/512/1.0	64/12/128/0.1	

A.3 FGVC Detailed Tabulated Results

We report extensive results on multiple FGVC datasets, with each top-1 accuracy value representing the average over three independent test runs. A summary of the evaluation results for each dataset is provided in Table table A.12.

Table A.12: ViT-B/16 top-1 accuracy on FGVC (avg. over 3 runs). Tuned / Total (%) indicates the percentage of tunable parameters. Rank (r), sinusoidal dimension (d), frequency scale (f), and scaling factor (α) are reported per dataset. **Bold** and underline indicate best and second-best performance.

ViT-Base/16 (85.8M)	FGVC					Mean
	CUB-2011	NABirds	Ox.Flowers	Stan.Dogs	Stan.Cars	
Full	87.3	82.7	98.8	89.4	84.5	88.5
Linear	85.3	75.9	97.9	86.2	51.3	79.3
<i>Visual-Prompt Tuning</i>						
VPT-S [ECCV22]	86.7	78.8	98.4	90.7	68.7	84.6
VPT-D [ECCV22]	88.5	84.2	99.0	90.2	83.6	89.1
E ² VPT [ICCV23]	89.1	84.6	99.1	90.5	82.8	89.2
VFPT [NeurIPS24]	88.7	84.5	99.1	90.4	83.6	89.2
<i>Reparameterization-Based Adaptation</i>						
LoRA [ICLR22]	85.6	79.8	98.9	87.6	72.0	84.8
FourierFT [ICML24]	88.9	84.5	98.9	89.7	78.3	88.1
FouRA [NeurIPS24]	<u>89.3</u>	86.2	99.3	<u>91.3</u>	85.6	<u>90.4</u>
LoRA-XS [ECAI25]	89.1	85.4	99.3	90.5	82.4	89.3
CDRA-SPT [CVPR25]	88.6	83.4	99.3	89.8	88.2	89.9
LoRA-Sine [ICLR25]	89.0	85.2	<u>99.2</u>	90.3	85.9	89.9
<i>Ours</i>						
LoRA-SPS (Efficient)	89.6 ±.09	85.4±.06	99.3 ±.01	<u>91.3</u> ±.20	86.1±.04	<u>90.4</u> ±.06
- Tuned / Total (%)	.19	.56	.16	.17	.43	.30
- $r/d/f/\alpha$	2/24/1024/0.01	8/24/128/0.01	16/12/256/0.01	4/48/1024/0.01	16/48/512/0.1	
LoRA-SPS (Best)	89.6 ±.09	<u>85.8</u> ±.08	99.3 ±.01	91.4 ±.09	<u>87.6</u> ±.25	90.8 ±.08
- Tuned / Total (%)	.19	1.00	.16	.36	.94	.53
- $r/d/f/\alpha$	2/24/1024/0.01	16/96/256/0.1	16/12/256/0.01	8/96/512/0.01	96/24/256/1.0	