Preference-Aligned LoRA Merging: Preserving Subspace Coverage and Addressing Directional Anisotropy

Supplementary Material

A. Proof of Proposition 1

Proposition 1 (Anisotropy Bounds)**.** *Let $\sigma_{\max}(\boldsymbol{J})$ and $\sigma_{\min}(\boldsymbol{J})$ denote the largest and smallest singular values of $\boldsymbol{J}$. Then, for any coefficient vector $\boldsymbol{\phi}$,*

$$\sigma_{\min}(\boldsymbol{J})\,\|\boldsymbol{\phi}\|_2 \;\le\; \|\boldsymbol{J}\boldsymbol{\phi}\|_2 \;=\; \|\Delta\boldsymbol{f}\|_2 \;\le\; \sigma_{\max}(\boldsymbol{J})\,\|\boldsymbol{\phi}\|_2. \tag{5}$$

When $\kappa(\boldsymbol{J}) = \sigma_{\max}(\boldsymbol{J})/\sigma_{\min}(\boldsymbol{J})$ is large, the map $\boldsymbol{\phi} \mapsto \Delta\boldsymbol{f}$ is anisotropic*, and equal-norm LoRA-direction updates need not yield proportional task-loss changes.*

Proof. Let $\boldsymbol{J} \in \mathbb{R}^{N\times K}$ be the restricted task-loss Jacobian with entries $\boldsymbol{J}_{i,k} = \langle \nabla f_i(\boldsymbol{W}), \boldsymbol{S}_k\rangle$, and take its SVD $\boldsymbol{J} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{U} \in \mathbb{R}^{N\times q}$ and $\boldsymbol{V} \in \mathbb{R}^{K\times q}$ have orthonormal columns ($\boldsymbol{U}^\top\boldsymbol{U} = \boldsymbol{V}^\top\boldsymbol{V} = \boldsymbol{I}_q$), $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1,\dots,\sigma_q)$ with $\sigma_1 \ge \cdots \ge \sigma_q > 0$, and $q = \mathrm{rank}(\boldsymbol{J})$. For any coefficient vector $\boldsymbol{\phi} \in \mathbb{R}^K$, define $\tilde{\boldsymbol{\phi}} = \boldsymbol{V}^\top\boldsymbol{\phi} \in \mathbb{R}^q$. By orthogonal invariance of the Euclidean norm,

$$\|\boldsymbol{J}\boldsymbol{\phi}\|_2 = \|\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top\boldsymbol{\phi}\|_2 = \|\boldsymbol{\Sigma}\boldsymbol{V}^\top\boldsymbol{\phi}\|_2 = \|\boldsymbol{\Sigma}\tilde{\boldsymbol{\phi}}\|_2. \tag{17}$$

The second equality holds since multiplication by $\boldsymbol{U}$ preserves the ℓ_2 norm: for any $y \in \mathbb{R}^q$, $\|\boldsymbol{U}y\|_2 = \|y\|_2$. Also, because $\boldsymbol{V}$ has orthonormal columns, $\|\boldsymbol{V}^\top\boldsymbol{\phi}\|_2 \le \|\boldsymbol{\phi}\|_2$.

Since $\boldsymbol{\Sigma}$ is diagonal,

$$\|\boldsymbol{J}\boldsymbol{\phi}\|_2^2 = \|\boldsymbol{\Sigma}\tilde{\boldsymbol{\phi}}\|_2^2 = \sum_{i=1}^{q} \sigma_i^2\,\tilde{\phi}_i^2. \tag{18}$$

For the upper bound, use $\sigma_i \le \sigma_{\max}(\boldsymbol{J})$ and $\|\tilde{\boldsymbol{\phi}}\|_2 \le \|\boldsymbol{\phi}\|_2$ to obtain

$$\|\boldsymbol{J}\boldsymbol{\phi}\|_2^2 = \sum_{i=1}^{q} \sigma_i^2\,\tilde{\phi}_i^2 \le \sigma_{\max}(\boldsymbol{J})^2 \sum_{i=1}^{q} \tilde{\phi}_i^2 \le \sigma_{\max}(\boldsymbol{J})^2\|\boldsymbol{\phi}\|_2^2, \tag{19}$$

hence $\|\boldsymbol{J}\boldsymbol{\phi}\|_2 \le \sigma_{\max}(\boldsymbol{J})\,\|\boldsymbol{\phi}\|_2$.

For the lower bound, decompose $\boldsymbol{\phi} = \boldsymbol{\phi}_{\mathrm{R}} + \boldsymbol{\phi}_{\mathrm{N}}$ with $\boldsymbol{\phi}_{\mathrm{R}} := \boldsymbol{V}\boldsymbol{V}^\top\boldsymbol{\phi} \in \mathrm{range}(\boldsymbol{V})$ and $\boldsymbol{\phi}_{\mathrm{N}} := (\boldsymbol{I} - \boldsymbol{V}\boldsymbol{V}^\top)\boldsymbol{\phi} \in \ker(\boldsymbol{J})$. Equivalently, $\tilde{\boldsymbol{\phi}} = (\tilde{\phi}_1,\dots,\tilde{\phi}_q)$ are the coordinates of $\boldsymbol{\phi}_{\mathrm{R}}$ in the $\boldsymbol{V}$-basis. Then

$$\|\boldsymbol{J}\boldsymbol{\phi}\|_2^2 = \sum_{i=1}^{q} \sigma_i^2\,\tilde{\phi}_i^2 \ge \sigma_{\min}(\boldsymbol{J})^2 \sum_{i=1}^{q} \tilde{\phi}_i^2 = \sigma_{\min}(\boldsymbol{J})^2\|\boldsymbol{\phi}_{\mathrm{R}}\|_2^2, \tag{20}$$

which implies $\|\boldsymbol{J}\boldsymbol{\phi}\|_2 \ge \sigma_{\min}(\boldsymbol{J})\,\|\boldsymbol{\phi}_{\mathrm{R}}\|_2$. If one restricts to the admissible subspace that excludes the nullspace (e.g., the LoRA span intersected with $\ker(\boldsymbol{J})^\perp$), then $\boldsymbol{\phi}_{\mathrm{N}} = 0$ and $\|\boldsymbol{\phi}_{\mathrm{R}}\|_2 = \|\boldsymbol{\phi}\|_2$, yielding

$$\sigma_{\min}(\boldsymbol{J})\,\|\boldsymbol{\phi}\|_2 \;\le\; \|\boldsymbol{J}\boldsymbol{\phi}\|_2 \;\le\; \sigma_{\max}(\boldsymbol{J})\,\|\boldsymbol{\phi}\|_2.$$

(Equivalently, interpret $\sigma_{\min}(\boldsymbol{J})$ as the smallest *nonzero* singular value on the feasible subspace.) □

B. Additional Related Work

Pre-Merging and Additional Model Merging. Pre-merging studies [81, 111, 113, 121] clarify and formalize the linear-composition behavior that *Task Arithmetic* relies on: they analyze when parameter-space combinations approximate joint training via partial or tangent linearization [39, 53, 68, 81] and connect these to NTK-style local linearization [34]. Building on this foundation, recent formulations explore optimal-transport fusion for Transformers [33], cycle-consistent multi-model merging [11], deriving layer-wise coefficients from model-internal statistics [31], permutation with least-squares alignment [66], constructing merging recipes from collections of fine-tuned models with diverse hyperparameters [93] with early evidence on CLIP that averaging robustness-oriented fine-tunes can improve zero-shot robustness [94], evolutionary search for recipe discovery [2], sparsity- and magnitude-aware sampling to reduce interference [15], and dynamic Fisher weighting guided by Bayesian optimization [46]. AIM [67] uses the information from the activation space of LLMs for merging. Another line of work leverages intermediate activations during inference or training and uses feature responses in specific layers to guide the merging process [52, 67, 95].

Additional LoRA-targeted Merging. Within *LoRA-targeted* methods, a training-free framework decouples direction and scale and orthogonalizes adapter directions before merging [116]. Zhang et al. [112] compose PEFT modules via weight-space arithmetic, enabling flexible transfer without extra training. Zhang and Zhou [111] enforce orthogonal LoRA subspaces pre-finetuning, reducing interference and preserving merge performance. Liu et al. [52] adjusts task and layer-wise merging coefficients using activation/gradient sensitivity and cross-task transferability. Several methods directly exploit low-rank structure for merging [23, 69, 70]. In particular, Panariello et al. [69] propose a core space that can be efficiently combined with existing merging baselines.

Model Merging in LLMs and VLMs. More recent work on model merging directly targets LLMs and VLMs and proposes strategies tailored to these architectures [6, 18, 58, 90, 109, 118, 119]. In particular, RobustMerge [109] introduces a merging method for MLLMs that exploits directional robustness in a low-rank space. Zhang et al. [110] leverage the rotation symmetry of self-attention layers, which substantially enlarges the equivalence set of Transformer models compared to permutation-based symmetries.

Model Merging for Multi-Task and Continual Learning. Another line of work explicitly connects model merging with conventional multi-task learning [45, 76, 91, 103, 104], viewing merging as a mechanism for parameter sharing and representation consolidation across tasks. A complementary set of studies aims to localize task-specific information in the parameters or to quantify interference within linear layers [9, 14, 80, 88], while Marczak et al. [60] further decompose the parameter space into shared and task-specific subspaces. MuDSC [98] also explores heterogeneous settings with diverse dense prediction tasks for evaluating merging performance. However, it is not directly comparable to our setting because it allows branch-like architectures such as ZipIt [78]. Model merging has also been studied in continual learning scenarios [20, 71, 83, 87, 89, 105], where merging is used to mitigate catastrophic forgetting and to accumulate knowledge across tasks over time.

LoRA Composition. Dynamic routing or mixture-of-experts compositions learn to combine multiple LoRAs at inference time [47, 48, 82, 96], and few-shot dynamic composition improves cross-task transfer [28, 30]. In vision, multi-LoRA merging for multi-task recognition demonstrates modularity [41]. In diffusion, multi-LoRA composition and subject-style concept mixing further highlight modularity [22, 75, 106, 117, 120, 122]. These methods dynamically assign or route adapters per input or task in an MoE-like fashion, which alters the inference-time architecture and falls outside our scope.

C. Experimental Settings

C.1. Datasets

C.1.1. Vision Benchmarks

Following Stoica et al. [79], we adopt a ViT-B/32 backbone pre-trained on ImageNet-21k [16], fine-tuned with LoRA rank 16. Evaluation is conducted on eight standard image-classification datasets. Brief summaries are provided below.

SUN397 [97]. Large-scale scene recognition dataset with 397 categories and 108,754 images, each class having at least 100 samples.

Cars [43]. Fine-grained car recognition covering 196 categories with 16,185 images, evenly split into train and test.

RESISC45 [8]. Remote sensing benchmark with 45 scene classes and 31,500 images, about 700 per class.

EuroSAT [27]. Satellite image classification dataset of 27,000 images in 10 land-use categories across diverse regions.

SVHN [108]. Street View House Numbers dataset with 10 digit classes, 73,257 train images, 26,032 test images, and 500k+ extra samples.

GTSRB [77]. Traffic sign recognition dataset with 43 categories and over 50,000 labeled images.

MNIST [44]. Classic handwritten digit dataset of 70,000 grayscale images evenly distributed across 10 classes (60k train / 10k test).

DTD [10]. Texture classification dataset with 47 attributes and 5,640 images, about 120 per class.

C.1.2. Language Benchmarks

In addition to the main benchmarks used in the paper, we evaluate our method on six classification datasets to assess its applicability to large language models. These datasets are used in conjunction with the LLaMA-3 8B model. Below, we briefly summarize each dataset.

QNLI [85]. A question-answering dataset reformulated as sentence-pair classification. Each example is a (question, sentence) pair, and the label indicates whether the sentence contains the answer to the question (binary).

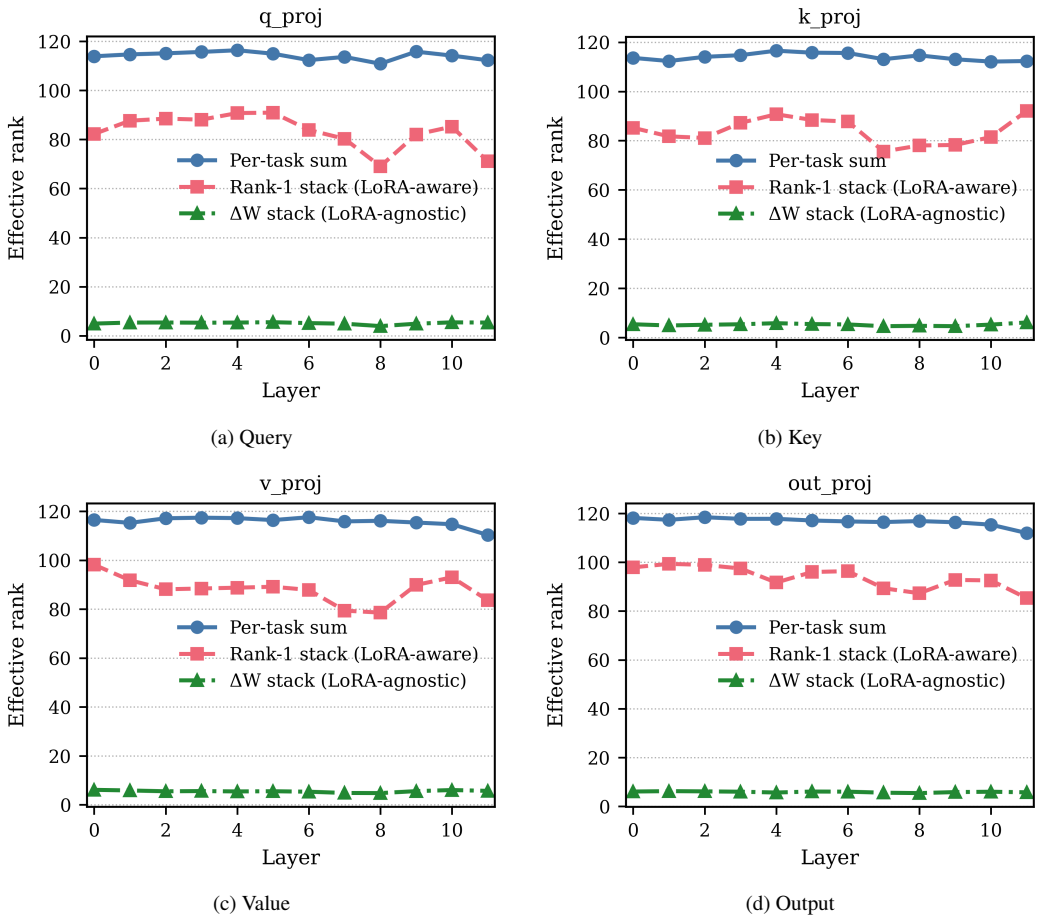
MNLI [92]. A large, crowdsourced collection of sentence pairs annotated for natural language inference. Given a premise and a hypothesis, the task is to predict one of three labels: *entailment*, *contradiction*, or *neutral* (three-class).

SNLI [4]. A corpus of $\sim$570k human-written sentence pairs labeled for *entailment*, *contradiction*, or *neutral*, supporting the standard NLI task (three-class).

RTE [3, 12, 24, 26]. A suite of textual entailment datasets compiled from the RTE1, RTE2, RTE3, and RTE5 challenges. Examples are drawn from news and Wikipedia. In the GLUE formulation, all RTE sets are converted to binary classification by collapsing *neutral* and *contradiction* into *not-entailment* for consistency.

SICK [61]. A dataset of around 10k sentence pairs built from image and video captions. Each pair is labeled for semantic relatedness (1-5 scale) and for textual entailment with three classes: *entailment*, *contradiction*, and *neutral*. It is widely used for evaluating compositional semantics, entailment, and similarity in a unified setting.

SciTail [42]. An entailment dataset derived from multiple-choice science exams and web sentences. Each (premise, hypothesis) pair is labeled as *entails* or *neutral* (binary). The dataset contains 27,026 examples (10,101 *entails* and 16,925 *neutral*).



(a) Query (b) Key

(c) Value (d) Output

Figure 5. **Effective Rank** across layers for each module. Higher curves indicate broader subspace coverage. The gap between $\Delta \boldsymbol{W}$ *stack* and *Rank-1 stack* reflects merge-induced collapse.

C.2. Training Details

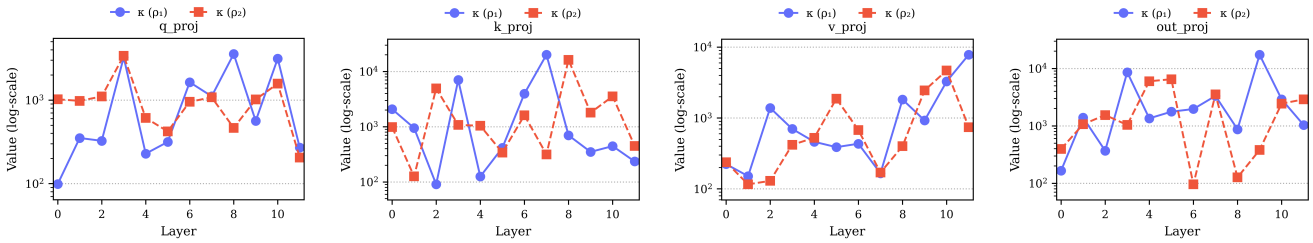
For the vision tasks, we fine-tuned CLIP [72] with a ViT-B/32 backbone. All models were optimized using AdamW [55] with a learning rate of 3e-4 and a cosine learning rate scheduler [54] with a weight decay of 1e-4.

For sequence classification, we fine-tuned LLaMA-3 [19] with 8B parameters, following the setup of [79]. Optimization used AdamW with a learning rate of 3e-5 and a linear scheduler. A linear classification head with three outputs was added for natural language inference (entailment, contradiction, neutral), and in binary settings predictions for the unused class were disregarded.

To enable efficient adaptation, we applied LoRA [29] to both CLIP and LLaMA. Specifically, low-rank adapters were attached to the query, key, value, and output projection matrices of each transformer block. Unless otherwise stated, the default LoRA hyperparameters were: rank = 16, α = 16, and dropout = 0.1.

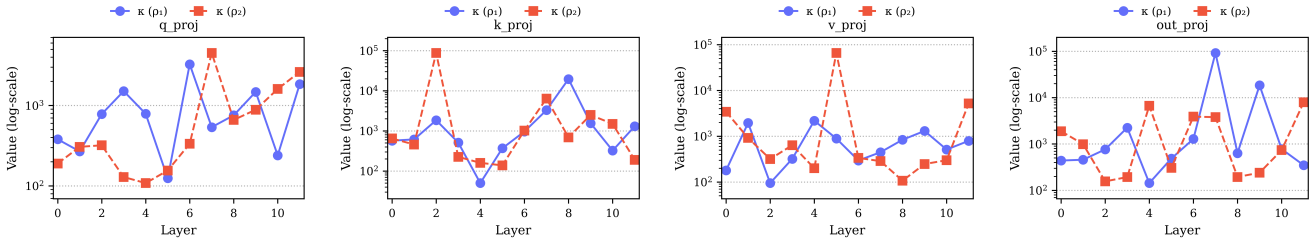
C.3. Implementation Details

In this paper, we compare our approach against ten baselines, following the experimental protocols reported in their original works. For TA [32], we tune the scaling coefficient of task vectors across the range of $[0.1, 0.2, \ldots 1.0]$. For TIES [99] and DARE-TIES [107], both the scaling coefficient λ and pruning coefficient η were selected via grid search. Specifically, λ was varied over $[0.8, 0.9, \ldots, 1.8]$, while η was explored over $[0.1, 0.2, \ldots, 0.9]$. For AdaMerging [101], all coefficients $\{\lambda_k^l\}_{n=1,l=1}^{N,L}$ (where N is the number of tasks and L the number of layers) are initialized to 0.3 before optimization with the entropy surrogate. For KnOTS [79], we evaluate two variants: KnOTS-TIES and KnOTS-DARE-TIES, which combine KnOTS with TIES [99] and DARE [107], respectively. For LoRA-LEGO [114], we reimplement the algorithm following the original paper. We apply k-means clustering to LoRA Minimal Semantic Units (MSUs), where each MSU $u = [a, b]$ consists of a row a from the LoRA down-projection matrix A and the corresponding



(a) q_proj (b) k_proj (c) v_proj (d) out_proj

Figure 6. **Condition Number Anisotropy κ (RAW Basis).** Layer-wise condition number $\kappa(\boldsymbol{\rho})$ per module under the non-orthogonal LoRA basis (RAW). Larger κ indicates stronger *within-preference* directional concentration of loss sensitivity. (Here, $\boldsymbol{\rho}_1$ uniformly weights all tasks, whereas $\boldsymbol{\rho}_2$ assigns all weight to a single task (one-hot).)



(a) q_proj (b) k_proj (c) v_proj (d) out_proj

Figure 7. **Condition Number Anisotropy κ (SVD Basis).** Layer-wise condition number $\kappa(\boldsymbol{\rho})$ per module under the orthogonal LoRA basis (SVD). Larger κ indicates stronger *within-preference* directional concentration of loss sensitivity. (Here, $\boldsymbol{\rho}_1$ uniformly weights all tasks, whereas $\boldsymbol{\rho}_2$ assigns all weight to a single task (one-hot).)

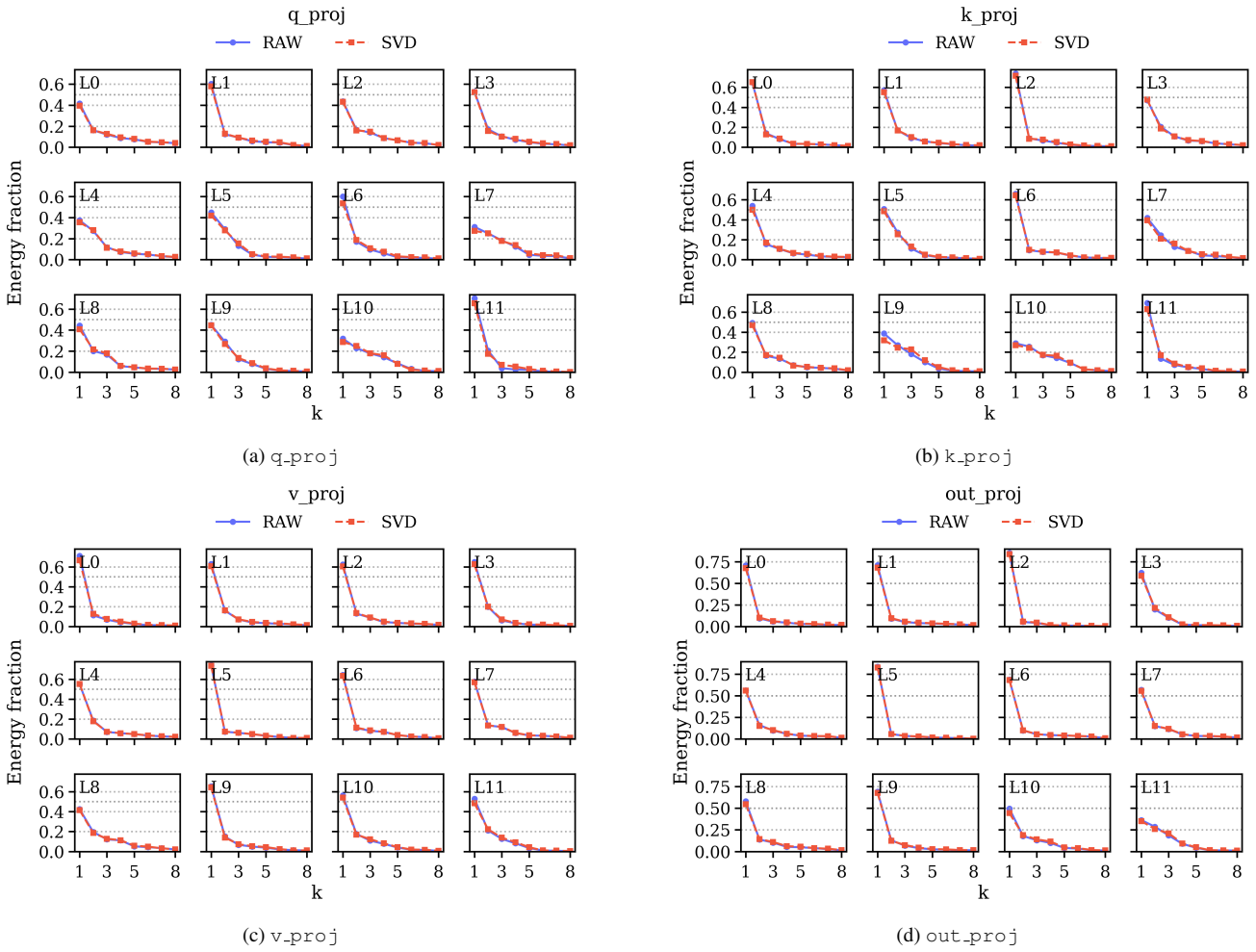
column b from the up-projection matrix B. We experiment with cluster sizes $k \in \{8, 16, 32, 64, 128\}$, and reproduce the dual scaling strategies: parameter reweighting and output reweighting. Parameter reweighting rescales each cluster centroid μ to match the average norm of its members, $\mu' = \frac{\frac{1}{p}\sum_{i=1}^{p}\|s_i\|}{\|\mu\|}\mu$, compensating for the reduced norm after merging, while output reweighting scales the merged LoRA by $\frac{\sqrt{r}}{\sqrt{k}}$ (with r the original LoRA rank and k the merged rank) to stabilize variance. We evaluate every k and both scaling strategies, reporting the best performance. We further include the LoRA-aware RobustMerge [109] and strong vanilla-merging methods EMR-Merging [31], FR-Merging [115], and Iso-CTS [60], and report their results in Sec. F.

C.4. Per-Task and Joint-Task Evaluation

In addition to the standard per-task benchmark, we also adopt the "joint-task" evaluation introduced by Stoica et al. [79]. Unlike the per-task regime, where each dataset is treated independently, the joint-task aggregates the inputs and labels from all eight vision benchmarks into a single evaluation pool. After combining all label sets, duplicate classes are removed (for example, MNIST [44] and SVHN [108] share the same digit categories), resulting in 748 unique labels across the combined benchmark. This setup is particularly demanding because models must not only generalize within a task, but also discriminate among labels that appear across different datasets. In some cases, labels are semantically close or hierarchical, such as "islet" in SUN397 [97] and "island" in RESISC45 [8], which increases the difficulty of classification. To account for such ambiguity, evaluation is reported using Hits@k metrics, where Hits@1 corresponds to top-1 accuracy and higher k values allow partial credit when the correct label appears among the top-k predictions. Joint-task protocol serves as an important benchmark for assessing the generality of merged models, as it directly tests whether a single model can operate across the diverse label space of multiple datasets.

D. Additional Analysis: Subspace Coverage

This appendix presents the exact formulations used in the main paper's subspace-coverage analysis. We form all stacks by vectorizing matrices and placing one vector per row. Let $\boldsymbol{W}_0 \in \mathbb{R}^{d\times m}$ be the pre-trained weight. For task $i \in \{1, \ldots, N\}$ with LoRA rank r_i, write $\Delta\boldsymbol{W}_i = \sum_{j=1}^{r_i} \boldsymbol{b}_{ij}\boldsymbol{a}_{ij}^{\top}$ with $\boldsymbol{b}_{ij} \in \mathbb{R}^d$ and $\boldsymbol{a}_{ij} \in \mathbb{R}^m$. Let $\mathrm{vec}(\cdot) : \mathbb{R}^{d\times m} \to \mathbb{R}^{dm}$ denote matrix vectorization. For any matrix $\boldsymbol{X}$, let $\{\sigma_k\}_{k=1}^{R_{\boldsymbol{X}}}$ be its nonzero singular values with



(a) `q_proj`

(b) `k_proj`

(c) `v_proj`

(d) `out_proj`

Figure 8. **Layer-wise Directional-Sensitivity Spectra (RAW vs. SVD).** For each module and layer, we form the task-direction sensitivity matrix with entries $J_{i,k} = \langle \nabla f_i(W),\, S_k \rangle$ and plot the scree curve of its singular values as energy fractions $\sigma_k^2 / \sum_j \sigma_j^2$ versus component index k. "RAW" stacks rank-1 LoRA factors directly, while "SVD" uses a shared orthonormal basis. The two spectra are overlaid in each panel. A larger leading mass indicates stronger concentration of sensitivity into a few modes, whereas flatter spectra indicate a more distributed use of LoRA directions.

$R_{\boldsymbol{X}} = \mathrm{rank}(\boldsymbol{X})$ and define

$$p_k = \frac{\sigma_k^2}{\sum_{j=1}^{R_{\boldsymbol{X}}} \sigma_j^2}, \qquad \mathrm{erank}(\boldsymbol{X}) = \exp\Big(-\sum_{k=1}^{R_{\boldsymbol{X}}} p_k \log p_k \Big).$$

The following entries are used in our analysis. For each entity, we define the stacked matrix and report its subspace coverage via effective rank.

- **Per-task sum** — compute coverage per task and then aggregate across tasks:

$$\boldsymbol{X}_i = \begin{bmatrix} \mathrm{vec}\big(\boldsymbol{b}_{i1}\boldsymbol{a}_{i1}^\top\big) \\ \vdots \\ \mathrm{vec}\big(\boldsymbol{b}_{ir_i}\boldsymbol{a}_{ir_i}^\top\big) \end{bmatrix} \in \mathbb{R}^{r_i \times dm},$$

$$\mathrm{PerTaskSum} = \sum_{i=1}^{N} \mathrm{erank}(\boldsymbol{X}_i).$$

- **$\Delta \boldsymbol{W}$ stack (LoRA-agnostic)** — coverage of task updates when the LoRA rank-1 structure is ignored. We stack the task updates and use $\mathrm{erank}(\boldsymbol{X}_{\mathrm{agnostic}})$ with

$$\boldsymbol{X}_{\mathrm{agnostic}} = \begin{bmatrix} \mathrm{vec}\big(\Delta \boldsymbol{W}_1\big) \\ \vdots \\ \mathrm{vec}\big(\Delta \boldsymbol{W}_N\big) \end{bmatrix} \in \mathbb{R}^{N \times dm},$$

- **Rank-1 stack (LoRA-aware)** — coverage when rank-1 directions from all adapters are retained. We stack all rank-1 directions and use $\mathrm{erank}(\boldsymbol{X}_{\mathrm{aware}})$ with

$$\boldsymbol{X}_{\mathrm{aware}} = \begin{bmatrix} \mathrm{vec}\big(\boldsymbol{b}_{11}\boldsymbol{a}_{11}^\top\big) \\ \vdots \\ \mathrm{vec}\big(\boldsymbol{b}_{Nr_N}\boldsymbol{a}_{Nr_N}^\top\big) \end{bmatrix} \in \mathbb{R}^{(\sum_i r_i) \times dm},$$

Other modules. Supplementary results in Fig. 5 show the same qualitative pattern across additional modules (e.g., Attention and MLP layers). The ordering *Per-task sum* $\geq$ *Rank-1 stack (LoRA-aware)* $\geq$ *ΔW stack (LoRA-agnostic)* consistently holds, and the LoRA-aware stack retains roughly $60\% \sim 70\%$ of the per-task sum. The gap between LoRA-aware and LoRA-agnostic reflects subspace collapse under interpolation-based merging.

E. Additional Analysis: Anisotropy

Within-Preference Anisotropy. We measure how unevenly a single preference $\boldsymbol{\rho} \in \Delta_{N-1}$ concentrates loss sensitivity onto LoRA directions. Let $g(\boldsymbol{\rho}; \boldsymbol{W}) = \sum_{i=1}^{N} \rho_i \nabla f_i(\boldsymbol{W})$ be the scalarized gradient at parameters $\boldsymbol{W}$ obtained by *averaging* task LoRA updates and then scaling by 0.3 (as Task Arithmetic [32]). Given a basis of LoRA directions $\{\boldsymbol{S}_k\}_{k=1}^{K}$ (either the RAW stack of rank-1 factors or the shared SVD-orthonormal basis), define the directional sensitivities with *condition-number*. Larger κ indicates stronger *within-preference* concentration of sensitivity onto a few directions (greater anisotropy), while smaller κ indicates a more balanced use of the LoRA subspace. We report layer-wise $\kappa(\boldsymbol{\rho}; \boldsymbol{W})$ for each module (`q_proj`, `k_proj`, `v_proj`, `out_proj`) under both RAW and SVD bases in Fig. 6 and Fig. 7. Here, $\boldsymbol{\rho}_1$ uniformly weights all tasks, whereas $\boldsymbol{\rho}_2$ assigns all weight to a single task (one-hot).

Figure 8 compares, for each module and layer, the singular-value energy distribution of the task-direction sensitivity matrix under the RAW (rank-1 factor stack) and SVD (shared orthonormal) bases. Each curve shows the energy fraction $\sigma_k^2 / \sum_j \sigma_j^2$ versus component index k. A larger leading mass indicates stronger anisotropy with sensitivity concentrated in a few directions, while flatter curves indicate a more balanced use of the LoRA subspace.

Directional-Sensitivity Misalignment in the SVD Basis. Following Sec. 3.3, we conduct the same analysis with directional sensitivities projected onto the rank-1 LoRA directions obtained by SVD orthogonalization. We then compute a misalignment index $\xi(\boldsymbol{\rho}_1, \boldsymbol{\rho}_2)$ between the uniform and one-hot preferences across the LoRA layers. As shown in Fig. 9, the resulting heatmap closely mirrors the raw-basis result in Fig. 2: substantial layer- and module-wise misalignment persists, indicating preference-dependent sensitive directions that remain even with SVD basis.

F. Additional Results

F.1. Extended Analysis of Baseline Comparisons

In this section, we provide a more detailed analysis of the baseline comparisons presented in the main paper. We elaborate on the per-task performance metrics across both vision and natural language inference (NLI) domains, offering an extended examination of the behavior of recent vanilla and LoRA-aware merging methods.

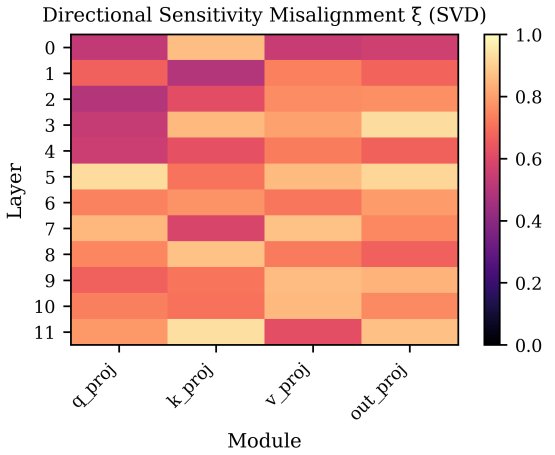


Figure 9. **Directional-sensitivity misalignment $\xi(\boldsymbol{\rho}_1, \boldsymbol{\rho}_2)$** in the SVD basis.

Per-Task Evaluation across Vision Tasks. For the eight image-classification benchmarks, Tab. 1 reports both the absolute per-task accuracies of the fine-tuned LoRA checkpoints and the normalized accuracies of the merged models. Strong vanilla merging methods such as Iso-CTS [60] reach up to 73.5% of the fine-tuned performance on average, but still lag behind the fine-tuned baselines and show large variance across datasets. Among LoRA-aware approaches, KnOTS-TIES [79] provides the strongest baseline, whereas RobustMerge [109] yields noticeably weaker average performance and smaller gains, indicating that its robustness benefits are limited in this setting. On top of these baselines, TARA-Variant A and TARA-Variant B further improve the average normalized accuracy to 74.0% and 76.3%, respectively, with consistent gains, showing that our method better preserves per-task performance when merging multiple checkpoints than more recent merging baselines.

Per-Task Evaluation on 6 NLI Tasks with LLMs. For the six NLI benchmarks, Tab. 2 shows a different picture from CLIP-based vision settings. Although Iso-C [60] performs strongly on vision benchmarks, in the NLI setting it produces highly unbalanced results: it attains more than 100% of the fine-tuned accuracy on SCITAIL, but suffers large drops on the other tasks, leading to the lowest average among recent vanilla baselines. EMR-Merging [31] and FR-Merging [115], which are effective in conventional CLIP-based multi-task setups, also fail to transfer cleanly to LLMs and do not achieve competitive averages in this regime. LoRA-aware methods generally improve over vanilla merging, but RobustMerge offers only modest gains over simpler LoRA-aware baselines, with limited improvement in average normalized accuracy. In contrast,

Table 5. To further address concerns regarding latency coming from gradient-based optimization, we experimented our method with LLaVA [50, 51] on merging six VQA experts.

Method	Dataset						
	ScienceQA	VizWiz	IconQA	ImageNet	ChartQA	DocVQA	Avg
	Per-task absolute accuracies (%)						
Finetuned	86.6	69.7	65.2	96.0	39.3	41.2	66.4
	Per-task accuracies of merged models, normalized to finetuned (%)						
TA [32]	81.6	71.3	57.2	43.6	66.6	79.5	66.6
TIES [99]	90.9	69.8	66.5	54.8	79.1	81.9	73.8
KnOTS-TIES [79]	87.4	77.0	60.5	44.8	76.5	85.6	72.0
AdaMerging [101]	87.9	81.0	66.3	47.0	74.5	76.1	72.1
TARA-Variant A	88.3	78.7	69.2	60.5	78.4	80.4	75.9
TARA-Variant B	89.1	79.2	69.5	59.6	78.3	80.0	76.0

Table 6. Time and memory cost analysis. We group baselines into gradient-free and gradient-based methods and compare their time and memory cost on LLaMA and LLaVA. For gradient-free methods, we exclude the time spent on grid search for tuning the scaling factor on the validation set. We report only *time per iteration* for gradient-free methods and all times measured in minutes (min). TIES, KnOTS-TIES, and TARA-Variant B require additional computation to resolve parameter conflicts or perform SVD, so we report both their total time and, in parentheses, the separate contributions from preprocessing and either one validation run (for gradient-free methods) or one optimization run (for gradient-based methods). Memory usage is reported as VRAM in MiB.

	LLaMA-3-8B		LLaVA-1.5-7B	
Method	Time (min)	VRAM (MiB)	Time (min)	VRAM (MiB)
TA [32]	3.6	16,150	10.6	14,814
TIES [99]	8.1 (4.5+3.6)	16,150	18.3 (7.7+10.6)	14,814
KnOTS-TIES [79]	10.8 (7.2+3.6)	16,150	24.3 (13.7+10.6)	14,814
AdaMerging [101]	8.0	19,918	13.9	19,788
TARA-Variant A	8.2	19,922	14.1	19,788
TARA-Variant B	15.2 (6.6+8.6)	20,040	27.7 (14.6+13.1)	20,662

our TARA-Variant A and TARA-Variant B reach 79.7% and 80.3% average normalized accuracy, respectively, outperforming all vanilla and LoRA-aware baselines. These results indicate that methods tailored to CLIP-based models do not automatically carry over to LLMs, while our approach scales robustly from vision to language tasks.

F.2. Evaluation on 6 VLM Tasks

We further evaluate the scalability of our method on vision–language models (VLMs). Specifically, we use LLaVA [50, 51] on six benchmarks: ScienceQA [57], VizWiz [25], IconQA [56], ImageNet [74], ChartQA [62], and DocVQA [64]. We fine-tune each LoRA adapter with rank 16 applied to the query, key, value and output projection modules of the attention layers, and train them for 500 iterations using AdamW [55] with a learning rate of 3×10^{-5} and batch size 1. We compare against vanilla model merging baselines, Task Arithmetic (TA) [32] and TIES [99], as well as the LoRA-aware methods KnOTS-TIES [79] and AdaMerging [101]. On LLaVA, AdaMerging underperforms LoRA-targeted merging methods such as KnOTS-TIES, indicating that its gains do not transfer well to the VLM regime. In contrast, both TARA-Variant A and TARA-Variant B achieve clear and stable improvements over all baselines across the six benchmarks, confirming that our approach also works reliably in multimodal VLM scenarios.

F.3. Ablations and Time/Memory Analysis

Table 6 summarizes the empirical efficiency of each method in terms of time cost and peak VRAM. We apply rank-16 LoRA and merge six LLaMA and six LLaVA models on a single NVIDIA GeForce RTX 3090. The upper panel reports gradient-free methods, and the lower panel reports gradient-based methods. A key observation is that, even when merging six fine-tuned LoRA checkpoints at foundation-model scale, the peak memory usage only increases by about 4–5 GB, so gradient-based merging does not incur prohibitive VRAM overhead.

For time, we report the cost of a single validation evaluation for gradient-free methods and the total optimization time for gradient-based methods. At first glance, gradient-free and gradient-based approaches have similar per-run time, but gradient-free methods usually require multiple validation runs to tune the scaling factor λ, which substantially increases their effective cost. For example, tuning the Task Arithmetic scaling factor in $\boldsymbol{W}_{\text{merge}} = \boldsymbol{W}_0 + \lambda \sum_{i=1}^{N} \Delta \boldsymbol{W}_i$ with a grid from 0.3 to 0.7 in steps of 0.1 en-

tails five trials. If each evaluation takes 10.6 minutes, the total time rises to about 53 minutes.

Gradient-based methods do not require such grid search, so their total time can be more favorable. TARA has runtime comparable to AdaMerging but achieves higher accuracy. Methods such as TIES, KnOTS-TIES, and TARA-Variant B incur additional preprocessing to compute an SVD over all task vectors. For these methods, we therefore report both their total time and, in parentheses, a breakdown into preprocessing time and either one validation run for the gradient-free setting or one optimization run for the gradient-based setting. For six LLaMA-3-8B models, for instance, TARA-Variant B spends 6.6 of its 15.2 minutes on SVD preprocessing and the remaining 8.6 minutes on optimization, which is comparable to Variant A and AdaMerging. Even for large models such as LLaMA-3-8B and LLaVA-1.5-7B, a merge completes within 30 minutes, underscoring the practical applicability of TARA.

To complement the analysis on foundation models, we also evaluate computational efficiency on relatively lightweight vision targets. Table 7 reports the time and memory costs using the ViT-B/32 backbone. Consistent with the trends observed in larger models, the runtimes for learning-free methods (e.g., KnOTS-TIES, LoRA-LEGO) represent only a single validation pass. Because practical deployment requires grid searches to find optimal scaling factors, their total operational time typically exceeds the single-run optimization time of TARA. Regarding memory consumption, while a distinct gap in VRAM usage exists between learning-free and gradient-based methods on the smaller ViT-B/32 architecture, this discrepancy diminishes at the foundation-model scale (as seen with LLaMA-3). Since frozen base weights dominate the overall memory footprint of large models, the relative overhead introduced by gradient computation becomes less pronounced.

Table 7. Computational Cost (ViT-B/32). Times denote a single validation pass (learning-free) vs. optimization (learning-based).

Method	Time (min)	VRAM (MiB)
KnOTS-TIES [79]	2.8	2262
LoRA-LEGO [114]	4.0	1416
AdaMerging [101]	4.1	4344
TARA (Variant A)	4.1	4344
TARA (Variant B)	5.1	5922

F.4. Additional Ablation Studies

Ablation on Hyperparameter α. Figure 10 reports the effect of the STCH scaling hyperparameter α on normalized accuracy for both Variant A and Variant B. Performance is slightly degraded at very small scaling ($\alpha = 0.1$), but quickly stabilizes once α reaches 0.5. For $\alpha \geq 0.5$, both variants show only minor fluctuations and maintain consistently strong performance across all benchmarks, indicating that our method is not sensitive to the exact choice of α in this range. Unless otherwise noted, we therefore fix $\alpha = 1$ in all main experiments.

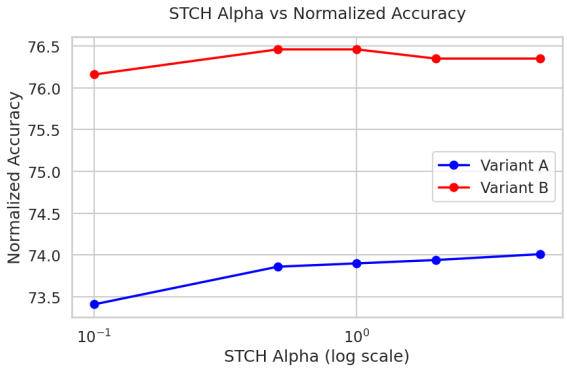


Figure 10. Ablation study of the hyperparameter α. Normalized accuracy is reported for two model variants across $\alpha \in \{0.1, 0.5, 1.0, 2.0, 5.0\}$. Both Variant A and Variant B exhibit stable performance over a wide range of α values, indicating that the method is not sensitive to this hyperparameter. Unless otherwise stated, we use $\alpha = 1$ for all main experiments.

Two-Task Pareto Fronts across Pairs. Figure 11 plots accuracy trade-offs for four task pairs on CLIP ViT-B/32. Sweeping the preference with TARA yields smooth Pareto fronts that are consistently dominant over AdaMerging, that is, TARA attains higher accuracy for the same preference across most of the preference range, especially in the balanced region where both tasks must be retained. Baseline mergers appear as isolated points and frequently lie below or off our front. We attribute this dominance to TARA 's LoRA-aware design: it preserves subspace coverage, retaining useful low-rank directions across tasks, while reweighting anisotropic directions to mitigate interference, keeping the merged model competitive near each task-optimal end without collapsing in the middle.

Robustness under Preference Perturbations. Figure 12 tests robustness with 8 tasks by fixing the two focal-task weights to 0.125 each (total 0.25) and randomly choosing the other six weights so they are nonnegative and sum to 0.75. Each scatter cloud shows 30 such samplings and the ellipses summarize their empirical covariance, with the mean marked by "×". Across all pairs, TARA produces tighter ellipses, indicating lower variance under preference perturbations, while AdaMerging shows larger spread. We also applied both a linear weighted-sum objective and a smooth Tchebycheff objective to TARA and to AdaMerging; the choice of objective yielded nearly identical ellipses and means.

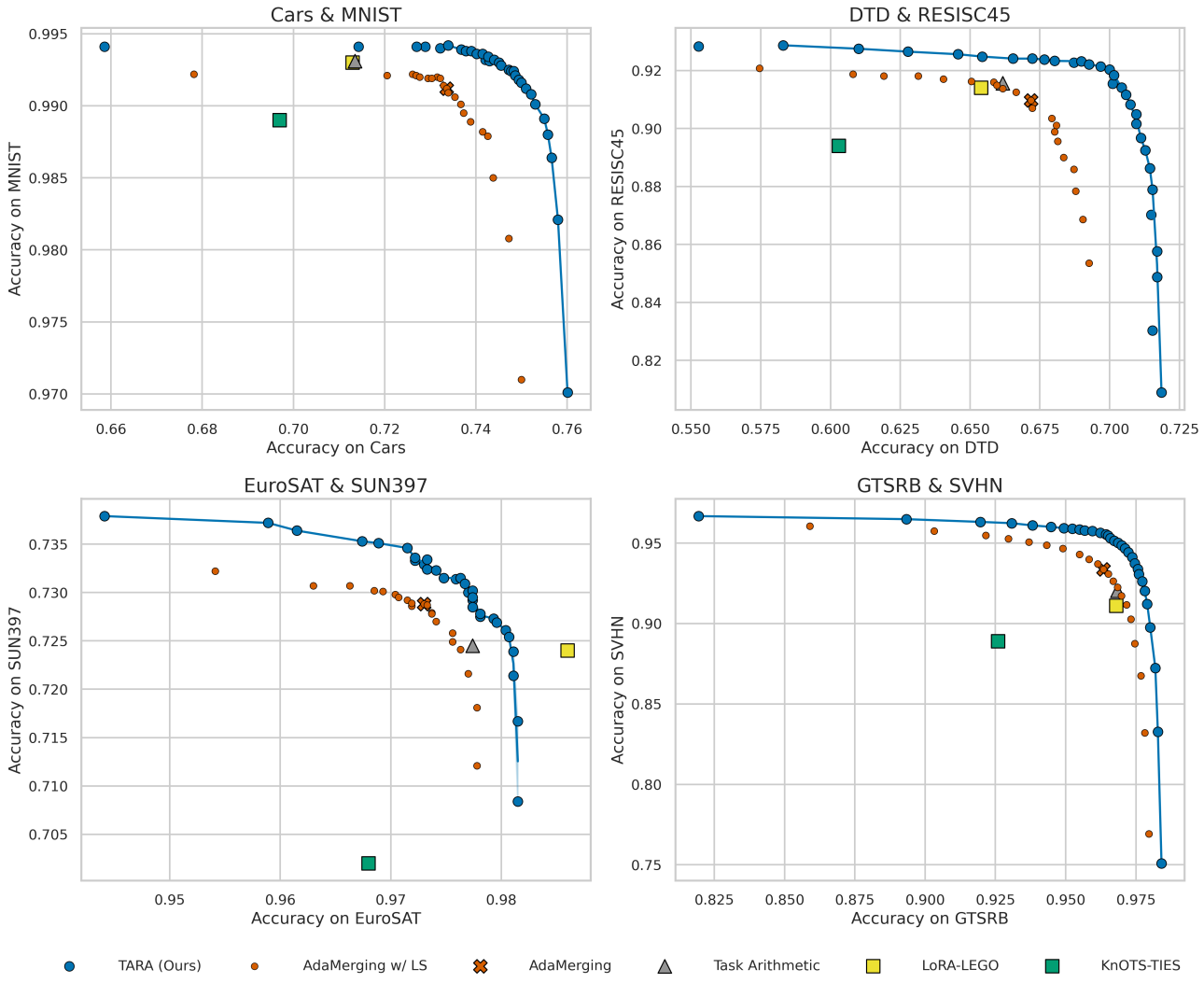


Figure 11. **Two-Task Trade-Offs on CLIP ViT-B/32.** Pairs shown from top-left to bottom-right: Cars-MNIST, DTD-RESISC45, EuroSAT-SUN397, GTSRB-SVHN.

Merging Models with LoRA Rank 16 on our Checkpoints. We complement the main-paper results (which use the KnOTS-released checkpoints [79]) with additional experiments on our own LoRA checkpoints to gauge robustness across sources. Specifically, we fine-tune CLIP ViT-B/32 adapters with a learning rate different from that used by KnOTS, and then evaluate the same merging protocols. Table 9 reports per-task absolute accuracies for the fine-tuned adapters (upper panel) and, for each merger, task-wise accuracies normalized to the corresponding fine-tuned baseline (lower panel). Table 11 reports generalization to unseen tasks on our checkpoints. The results are consistent with the main paper, indicating that the advantages of preserving subspace coverage and addressing anisotropy hold across different fine-tuning learning rates. A comparison of checkpoints is provided in Tab. 10.

Merging Models with LoRA Rank 4. LoRA is known to exhibit weaker cross-task alignment than full-rank fine-tuning, so LoRA-aware merging becomes especially important at low ranks. Using our CLIP ViT-B/32 checkpoints fine-tuned with rank-4 adapters, we evaluate vanilla and LoRA-aware merging in Table 12. Vanilla methods (e.g., TIES) degrade substantially, LoRA-aware baselines (KnOTS-TIES, LoRA-LEGO) recover more accuracy, and TARA achieves the strongest results overall: Variant B (500 iters) attains the best average normalized accuracy at **90.5**% and leads on most datasets. These rank-4 outcomes mirror the trends at higher ranks and underscore the benefit of modeling subspace coverage and anisotropy in low-rank LoRA merging.

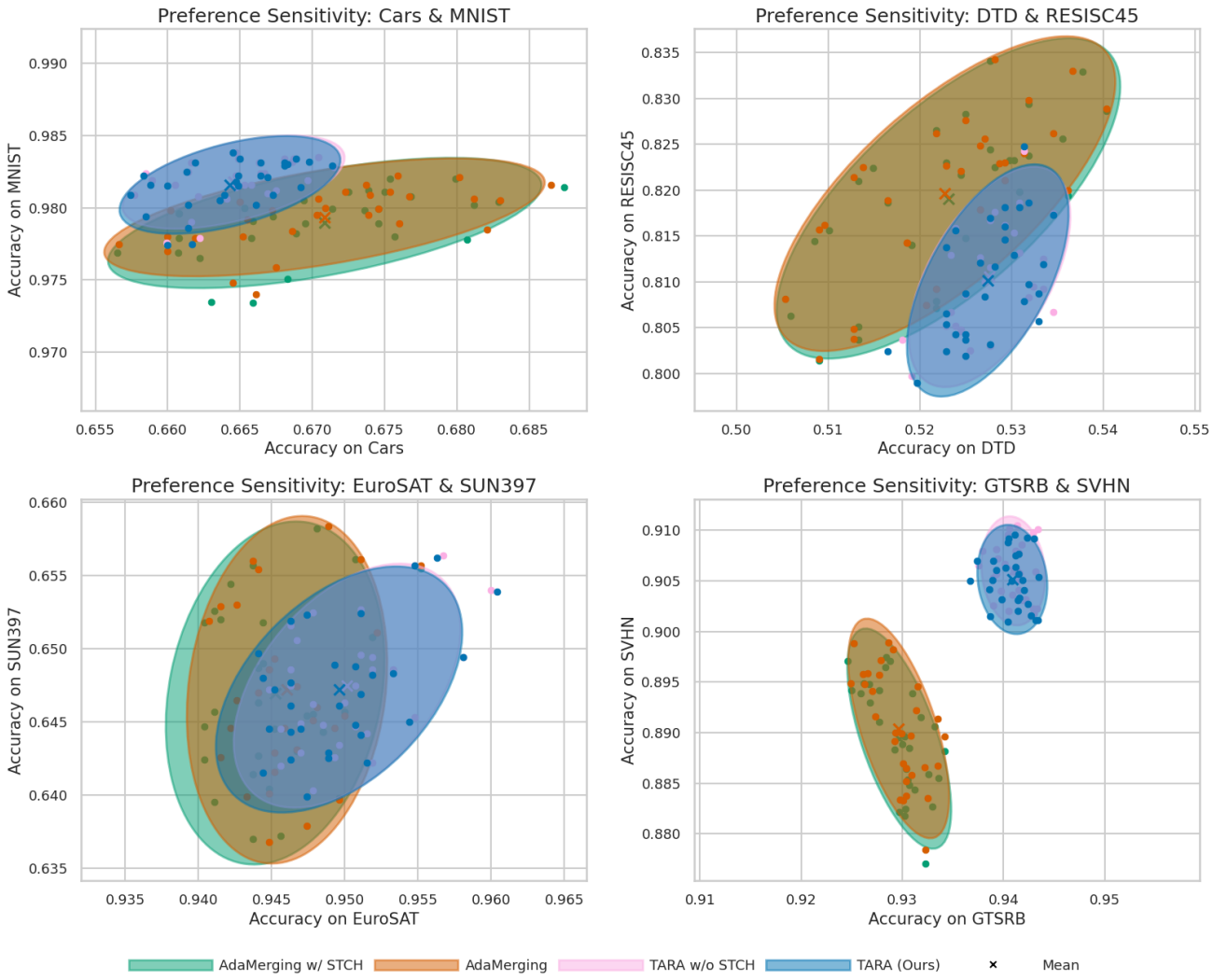


Figure 12. **Preference Sensitivity under Random Global Preferences.** Pairs shown from top-left to bottom-right: Cars-MNIST, DTD-RESISC45, EuroSAT-SUN397, GTSRB-SVHN.

Table 8. **Eight CLIP/ViT-B-32 Models Joint-Task Results.**

Method	Metric	Joint-Task Performances (%)								
		Cars	DTD	EuroSAT	GTSRB	MNIST	RESISC45	SUN397	SVHN	Avg
TA [32]	Hits@1	60.7	40.7	15.3	38.8	31.8	59.7	61.9	29.2	43.5
	Hits@3	84.9	63.7	23.0	66.1	48.4	83.6	83.9	50.1	65.2
	Hits@5	92.0	74.0	31.0	77.9	55.7	90.2	89.9	61.6	74.0
TIES [99]	Hits@1	60.4	39.7	13.0	35.0	33.4	58.6	61.3	32.9	43.6
	Hits@3	84.9	61.9	21.9	63.3	48.2	82.8	83.7	53.6	65.3
	Hits@5	92.1	72.4	29.0	75.3	54.0	89.4	89.9	64.1	73.9
DARE-TIES [107]	Hits@1	60.8	39.3	12.8	33.7	34.3	57.5	60.4	35.5	44.0
	Hits@3	85.3	61.4	18.1	63.6	50.2	82.3	82.6	57.8	66.4
	Hits@5	92.6	73.0	20.9	74.6	55.7	89.0	89.1	69.5	75.1
AdaMerging [101]	Hits@1	58.7	37.9	18.1	36.7	51.6	57.4	63.1	61.6	48.1
	Hits@3	83.1	59.5	56.9	63.6	71.0	81.3	84.8	85.1	73.2
	Hits@5	91.0	70.5	72.7	75.9	80.7	89.1	90.8	92.8	83.0
KnOTS-TIES [79]	Hits@1	61.7	40.5	16.2	44.2	39.1	59.0	60.6	36.7	46.8
	Hits@3	85.8	63.8	22.3	69.0	52.6	83.9	82.8	58.4	68.1
	Hits@5	92.6	74.5	31.1	79.8	58.4	90.4	89.1	68.5	76.3
KnOTS -DARE-TIES [79]	Hits@1	60.4	40.3	15.9	41.9	34.6	58.4	60.4	34.8	45.2
	Hits@3	85.0	63.5	21.1	68.4	50.0	83.8	82.4	56.5	66.9
	Hits@5	92.2	74.5	26.6	78.9	55.9	90.4	88.9	67.4	75.3
LoRA-LEGO [114]	Hits@1	60.3	39.9	15.9	41.6	29.8	57.6	60.1	30.2	43.1
	Hits@3	84.5	61.8	19.5	68.8	47.8	82.8	82.3	51.2	65.0
	Hits@5	91.8	74.0	21.1	78.3	56.9	89.7	88.8	63.5	73.9
TARA-Variant A	Hits@1	60.8	39.5	15.2	39.9	65.2	59.0	63.2	65.7	51.1
	Hits@3	84.8	60.8	58.0	67.0	80.7	82.6	85.2	87.8	75.9
	Hits@5	92.2	71.8	74.9	78.2	86.4	89.9	91.0	95.0	84.9
TARA-Variant B	Hits@1	63.2	40.1	12.1	41.9	64.6	62.1	63.6	46.4	49.3
	Hits@3	87.0	61.3	56.3	67.0	83.5	84.7	85.5	74.2	74.9
	Hits@5	93.4	72.5	76.9	77.2	88.9	90.9	91.3	89.9	85.1

Table 9. Using our checkpoints. Per-task accuracy on eight image-classification benchmarks. We merge eight ViT-B/32 checkpoints, each finetuned with LoRA. The upper panel shows the per-task absolute accuracy of the finetuned baselines; the lower panel reports accuracy of merged models, normalized by their corresponding finetuned baseline (%).

Method	Dataset								
	Cars	DTD	EuroSAT	GTSRB	MNIST	RESISC45	SUN397	SVHN	Avg
	Per-task absolute accuracies (%)								
Finetuned	76.2	72.5	98.6	98.3	99.2	93.1	73.7	96.7	88.5
	Per-task accuracies of merged models, normalized to finetuned (%)								
Vanilla Merging									
TA [32]	85.6	70.8	81.9	89.1	97.9	81.9	89.1	91.9	86.0
TIES [99]	72.2	61.2	64.7	66.6	90.0	70.0	86.1	73.7	73.1
DARE-TIES [107]	73.0	62.7	56.1	63.6	85.7	69.6	86.2	71.5	71.0
AdaMerging [101]	**87.7**	71.2	96.3	94.4	98.7	87.3	86.8	91.9	89.3
LoRA-aware Merging									
SVD [84]	76.0	60.9	74.8	90.4	98.1	74.2	85.0	93.6	81.6
Linear [59]	74.9	67.3	68.4	67.6	91.7	72.9	86.7	75.3	75.6
KnOTS-TIES [79]	85.4	69.3	77.8	76.6	93.3	80.1	89.4	82.0	81.7
KnOTS-DARE-TIES [79]	84.1	68.7	77.9	78.6	94.4	80.1	**89.8**	83.8	82.2
LoRA-LEGO [114]	84.9	71.4	82.7	91.1	98.5	81.2	87.4	93.4	86.3
TARA-Variant A	87.2	72.5	96.5	**95.7**	99.0	87.0	88.0	93.7	89.9
TARA-Variant B	85.8	**72.0**	**96.8**	96.4	**99.1**	**88.0**	87.4	**95.2**	**90.1**

Table 10. Comparison between merging methods between our checkpoints and KnOTS checkpoints.

Method	Dataset								
	Cars	DTD	EuroSAT	GTSRB	MNIST	RESISC45	SUN397	SVHN	Avg
	Per-task absolute accuracies (%)								
Finetuned (KnOTS)	74.0	58.3	99.0	92.7	99.3	88.4	64.5	96.2	84.1
Finetuned (Ours)	76.2	72.5	98.6	98.3	99.2	93.1	73.7	96.7	88.5
	Per-task accuracies of merged models, normalized to finetuned (%)								
TA [32] (KnOTS)	82.0	73.6	48.8	42.1	53.1	71.5	97.5	41.2	63.7
TA [32] (Ours)	83.1	68.7	79.7	93.0	98.8	79.4	85.2	93.8	85.2
TARA-Variant A (KnOTS)	84.5	76.2	68.9	39.4	82.2	72.8	97.5	70.0	73.9
TARA-Variant A (Ours)	86.5	72.0	94.4	95.0	98.8	85.3	87.4	93.5	89.1
TARA-Variant B (KnOTS)	86.2	78.4	76.8	42.9	82.7	75.4	98.6	69.7	76.3
TARA-Variant B (Ours)	86.1	72.7	95.7	95.6	99.0	86.4	88.0	94.2	89.7

Table 11. Using our checkpoints. Generalization results on two unseen tasks when merging ViT-B/32 models trained on six tasks.

	Seen Tasks							Unseen Tasks			All Tasks
Method	Cars	DTD	GTSRB	RESISC45	SUN397	SVHN	**Avg Acc**	EuroSAT	MNIST	**Avg Acc**	**Avg Acc**
	Per-task accuracies of merged models, normalized to finetuned (%)										
TA [32]	85.6	73.0	94.7	82.4	88.4	95.0	86.5	47.3	78.1	62.7	80.6
TIES [99]	76.5	62.5	71.4	73.3	87.5	75.0	74.4	49.0	64.5	56.8	69.9
KnOTS-TIES [79]	85.0	69.4	83.9	81.1	89.4	84.8	82.3	56.6	73.4	65.0	78.0
KnOTS-DARE-TIES [79]	85.7	69.3	85.0	81.4	**90.3**	86.4	83.0	56.7	74.0	65.4	78.6
LoRA-LEGO [114]	85.2	72.9	94.9	82.3	88.3	94.8	86.4	46.9	77.8	62.4	80.4
AdaMerging [101]	87.1	73.8	93.4	**91.2**	88.4	96.2	88.4	52.2	83.1	67.7	83.2
TARA-Variant A	**87.4**	**74.9**	94.9	89.6	89.2	**96.7**	**88.8**	57.4	86.6	72.0	84.6
TARA-Variant B	87.2	74.7	**95.4**	89.4	89.3	**96.7**	**88.8**	**63.4**	**87.7**	**75.5**	**85.5**
Method	Cars	DTD	EuroSAT	GTSRB	MNIST	SUN397	**Avg Acc**	RESISC45	SVHN	**Avg Acc**	**Avg Acc**
	Per-task accuracies of merged models, normalized to finetuned (%)										
TA [32]	87.2	75.6	86.4	95.6	98.4	90.6	89.0	60.4	61.6	61.0	82.0
TIES [99]	74.7	62.8	65.3	74.6	88.3	88.0	75.6	65.1	48.9	57.0	71.0
KnOTS-TIES [79]	86.7	70.8	80.8	83.3	94.1	90.4	84.3	**67.5**	57.2	62.4	78.8
KnOTS-DARE-TIES [79]	87.6	71.0	80.4	85.3	95.1	90.7	85.0	66.7	56.2	61.4	79.1
LoRA-LEGO [114]	87.0	75.3	86.0	95.9	98.5	90.2	88.8	60.0	61.5	60.8	81.8
AdaMerging [101]	89.7	74.7	96.7	95.0	**99.4**	90.4	91.0	60.1	**63.7**	61.9	83.7
TARA-Variant A	**89.9**	75.0	97.1	96.2	99.3	90.9	**91.4**	61.2	63.5	**62.4**	**84.1**
TARA-Variant B	89.7	74.9	**97.4**	**96.4**	99.2	**91.0**	**91.4**	60.7	58.6	59.7	83.5

Table 12. Using our checkpoints with LoRA (Rank-4). Per-task accuracy on eight image-classification benchmarks. We merge eight ViT-B/32 checkpoints, each finetuned with LoRA. The upper panel shows the per-task absolute accuracy of the finetuned baselines; the lower panel reports accuracy of merged models, normalized by their corresponding finetuned baseline (%).

Method	Dataset								
	Cars	DTD	EuroSAT	GTSRB	MNIST	RESISC45	SUN397	SVHN	Avg
	Per-task absolute accuracies (%)								
Finetuned	72.1	70.2	98.4	98.1	99.2	93.3	73.4	96.7	87.7
	Per-task accuracies of merged models, normalized to finetuned (%)								
Vanilla Merging									
TA [32]	87.4	76.2	74.0	78.1	98.1	83.7	89.7	92.3	84.9
TIE [99]	81.3	64.4	46.5	51.9	77.4	69.1	86.8	64.1	67.7
DARE-TIES [107]	81.2	65.4	50.9	48.4	76.3	69.8	86.7	63.5	67.8
AdaMerging [101]	87.9	75.1	94.7	92.0	98.2	86.9	89.5	91.1	89.4
LoRA-aware Merging									
SVD [84]	80.5	62.7	40.6	73.7	94.4	69.3	87.3	92.4	75.1
Linear [59]	44.0	25.6	12.4	37.8	71.5	23.3	44.2	51.7	38.8
KnOTS-TIES [79]	88.2	72.2	72.5	62.3	89.7	79.3	89.5	77.0	78.8
KnOTS-DARE-TIES [79]	87.7	73.0	73.4	65.0	90.4	79.9	89.5	78.8	79.7
LoRA-LEGO [114]	88.2	75.8	74.8	76.9	97.5	83.8	**90.4**	91.5	84.8
TARA-Variant A	88.7	76.4	**94.9**	92.1	98.3	86.8	90.0	92.0	89.9
TARA-Variant B	**89.8**	**76.7**	94.6	**92.8**	**98.4**	**88.0**	90.1	**93.2**	**90.5**

Ablation Study on LoRA Rank. We evaluate the impact of varying LoRA ranks r on the merging performance using the ViT-B/32 backbone. Table 13 presents the absolute and normalized accuracy across different rank configurations. TARA achieves higher accuracy than the baselines across all evaluated ranks. Notably, in the 'Full' setting, where the LoRA rank matches the ViT embedding dimension to emulate full-parameter fine-tuning, TARA does not simply converge to the FFT merging solution, rather, it maintains a distinct performance margin.

Table 13. Absolute (Normalized) Accuracy across LoRA Ranks.

Method	r=4	r=16	r=64	r=256	Full
TA [32]	71.9(82.9)	76.2(86.2)	77.9(86.2)	77.1(85.3)	72.0(80.8)
Adamerging [101]	76.3(87.2)	78.7(88.5)	80.2(88.7)	79.8(88.6)	79.1(88.2)
TARA	77.5(88.8)	80.5(90.6)	81.9(90.5)	80.9(89.7)	80.5(89.5)

Efficiency Analysis with Recent Baselines To assess computational efficiency, we compare TARA against recent gradient-based merging methods, CALM [100] and FW-Merging [5], using the CLIP-ViT-B/32 backbone. Table 14 reports the average accuracy, runtime, and peak memory (VRAM) usage. The results indicate that TARA requires significantly less runtime and memory overhead while maintaining competitive accuracy compared to other gradient-based approaches.

Table 14. Comparison with Recent Baselines.

Method	Adamerging [101]	CALM [100]	FW-Merging [5]	TARA-variant B
Avg. Acc.	78.7	80.0	77.9	80.5
Time (min)	4.1	20.0	13.4	5.1
VRAM (MiB)	4344	8066	7542	5922

Extended Evaluation on Vision Benchmarks. To verify the scalability of our approach, we extend the evaluation to different model scales (ViT-B/32 and ViT-L/14) and varying numbers of tasks (8, 14, and 20) following settings similar to those in FW-Merging [5]. As detailed in Tab. 15, TARA consistently demonstrates comparable or superior absolute and normalized accuracy across these configurations relative to recent baselines. This performance gap is particularly evident in large-scale settings, such as merging 20 tasks with the ViT-L/14 backbone.

Table 15. Absolute (Normalized) accuracy on vision benchmark.

Backbone (# Tasks)	TA [32]	AdaMer. [101]	CALM [100]	FW-Mer. [5]	TARA
ViT-B/32 (8)	76.2(86.2)	78.7(88.5)	80.0(90.6)	77.9(87.6)	80.5(90.6)
ViT-B/32 (14)	73.6(83.6)	75.0(84.8)	74.7(84.6)	75.4(85.2)	75.4(85.2)
ViT-B/32 (20)	64.6(72.7)	66.6(74.8)	66.9(75.2)	68.2(77.2)	68.7(77.2)
ViT-L/14 (8)	88.0(95.8)	88.9(96.7)	83.0(90.3)	87.0(94.7)	88.9(96.7)
ViT-L/14 (14)	85.0(92.7)	85.2(93.0)	80.5(87.7)	85.7(93.6)	85.9(93.7)
ViT-L/14 (20)	80.7(87.2)	81.3(87.8)	78.5(84.5)	77.4(84.1)	81.5(88.0)