REFORGE: REFINING MERGED MODELS WITH ANCHOR-REGULARIZED REGRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Model merging aims to combine multiple task-specific expert models into a single model without joint retraining, offering a practical alternative to multi-task learning when data access or computational budget is limited. Existing model merging methods rarely exploit strong merged models as priors for further improvement. To address this limitation, we propose **ReForge**, a bilevel optimization framework that formulates module-wise refinement as Bayesian linear regression with an anchor-centered prior. The inner level yields a closed-form MAP estimate from unlabeled calibration activations. The outer level uses Bayesian optimization to jointly select heterogeneous regularization strengths and assembly scales using held-out validation data. Furthermore, we develop a data-free variant of ReForge that replaces activation statistics with task-vector Grams, eliminating the need for calibration examples. Across extensive benchmarks, including up to 20-task merging in vision and 5-task merging in language, ReForge consistently outperforms all evaluated *plug-and-play* anchor baselines (e.g., TA, WUDI-Merging, and TSV). On 20-task ViT-B/32, ReForge improves the strongest evaluated baseline, ISO-CTS, from 77.6% to 82.8% in the data-assisted setting and to 81.5% in the data-free setting. On eight-task ViT-L/14, the data-assisted variant achieves 95.1% mean accuracy, compared with 95.8% for the individual task experts. Our source code is available at `https://anonymous.4open.science/r/reforge-6CE5`.

1 INTRODUCTION

Adapting foundation models to downstream tasks via fine-tuning has become standard practice, but maintaining a separate expert model for each task introduces substantial storage, deployment, and operational overhead. While multi-task learning offers a unified alternative (Ruder, 2017), it necessitates joint data access and incurs prohibitive training costs, making it often infeasible under data silos, privacy constraints, or limited computing budgets. Model merging (Matena & Raffel, 2022; Ilharco et al., 2023; Yadav et al., 2023) offers a practical solution. By directly combining multiple expert models into a single architecture, it enables unified inference without revisiting the original training data or incurring the joint retraining costs. This paradigm has become increasingly relevant with the flourishing ecosystem of open-source models on platforms such as Hugging Face (Hugging Face, 2026), which offers an abundant supply of task-specific experts for weight-space integration.

A central challenge in model merging is how to effectively combine expert models when auxiliary data is limited or unavailable. Existing methods fall into two broad regimes based on their use of auxiliary data. *Data-assisted* methods, such as Fisher Merging (Matena & Raffel, 2022) and RegMean (Jin et al., 2023), rely on a small calibration set to estimate empirical statistics for merging. *Data-free* methods, including Task Arithmetic (TA) (Ilharco et al., 2023), TIES (Yadav et al., 2023), WUDI-Merging (Cheng et al., 2025), TSV (Gargiulo et al., 2025), and ISO-CTS (Marczak et al., 2025), avoid auxiliary data for estimating merged parameters, though some still use a held-out validation set for lightweight hyperparameter selection. These methods produce strong merged models that capture useful information from multiple experts. However, these merged models are rarely used as priors to guide further refinement.

To address this limitation, we propose ReForge, a *plug-and-play* bilevel optimization framework that is built on two key ideas. First, ReForge refines an existing merged model by treating it as an

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

anchor and incorporating activation evidence from task experts. This yields an anchor-regularized regression problem with an efficient closed-form solution, using only unlabeled calibration examples. Second, ReForge performs globally coordinated architecture-aware Bayesian optimization to jointly select heterogeneous regularization strengths and assembly scales across different modules of the network. Moreover, we use task-vector Grams as weight-based surrogates for activation statistics, enabling us to derive a data-free variant of ReForge while preserving the closed-form solution.

Empirically, ReForge is effective across both vision and language model merging. Across four backbones from two model families and seven *plug-and-play* anchor models, ReForge consistently improves over every evaluated anchor in both data-assisted and data-free settings, with relative gains of up to 29.8% over weaker merging baselines while still improving the strongest ones. In particular, on the standard ViT-L/14 benchmark for 8-task merging, a single merged model reaches 95.1%, closely matching the average performance of eight task-specific experts (95.8%). Extensive ablation studies are conducted to further validate the main components and design choices of ReForge.

2 Related Work

Training-free model merging. Existing methods generally fall into two broad regimes based on their reliance on auxiliary data: data-assisted and data-free. *Data-assisted* methods leverage a small calibration set or activation statistics to guide the merge. For example, Fisher Merging (Matena & Raffel, 2022) performs Fisher-weighted averaging of task-specific models, while RegMean (Jin et al., 2023) casts linear-layer merging as regression with a closed-form solution. *Data-free* methods eliminate the need for calibration examples when constructing merged weights and rely on the weights of the expert models. In this regime, Task Arithmetic (TA) (Ilharco et al., 2023) provides a baseline for combining task vectors. Subsequent works mitigate interference among models through selective parameter combination, including TIES (Yadav et al., 2023), DARE (Yu et al., 2024), PCB-Merging (Du et al., 2024), and Localize-and-Stitch (He et al., 2025a). TSV (Gargiulo et al., 2025) and ISO-CTS (Marczak et al., 2025) leverage structured subspaces to isolate task-specific features or align task-relevant subspaces. More recently, methods such as WUDI-Merging (Cheng et al., 2025) and DOGE (Wei et al., 2025) optimize explicit data-free merging objectives. Our ReForge is complementary to these approaches: it leverages existing merged solutions as anchors, admits an efficient closed-form solution for module-level refinement, and exploits bilevel optimization for hyperparameter search across different module groups.

Merging coefficient learning and hyperparameter search. Learning how to combine experts offers an alternative to fixed merging rules. AdaMerging (Yang et al., 2024) optimizes task-wise or layer-wise coefficients through entropy minimization on unlabeled test data, while aT-LAS (Zhang et al., 2024) learns block-wise task-vector scalings under supervised or unsupervised objectives. Alongside these gradient-based formulations, Bayesian optimization enables coefficient search driven directly by evaluation metrics. SIP-BMM (Chen et al., 2026) guides layer-wise coefficient search with structural importance information to construct a capability–efficiency Pareto set, whereas DF-Merge (Lee et al., 2025) couples model-level scaling with Fisher information recomputed at each candidate. ReForge instead uses Bayesian optimization to refine an existing merged anchor. It jointly selects heterogeneous regularization strengths and assembly scales in an architecture-aware manner, using validation performance without backpropagation through the merged network.

3 Problem Definition

Notation. Let $\theta_{\mathrm{pre}} \in \mathbb{R}^d$ denote the parameters of a pretrained base model, and $\{\theta^{(t)}\}_{t=1}^{T}$ represent the parameters of T distinct models fine-tuned from θ_{pre} on different downstream tasks. All models share the same architecture and operate within the same parameter space $\mathbb{R}^d$.

Task Vectors. Following the framework of Task Arithmetic (TA) (Ilharco et al., 2023), we use task vectors to represent task-specific parameter offsets. Specifically, the task vector for the t-th model is defined as $\tau^{(t)} = \theta^{(t)} - \theta_{\mathrm{pre}}$, for $t \in \{1, \cdots, T\}$.

Objective. Given a pretrained model θ_{pre} and task vectors $\{\tau^{(t)}\}_{t=1}^{T}$, our goal is to design an aggregation strategy $\mathcal{A}$ that combines T task vectors into a single merged model, parameterized by:

$$\theta_{\text{merged}} = \theta_{\text{pre}} + \mathcal{A}\big(\tau^{(1)}, \cdots, \tau^{(T)}\big), \tag{1}$$

such that the merged model preserves the task-specific capabilities of all the fine-tuned models as much as possible.

4 METHODOLOGY

4.1 MODEL MERGING AS BAYESIAN LINEAR REGRESSION

Module-wise Decomposition. While Eq. (1) defines the merging objective at the full parameter space $\mathbb{R}^d$, deep neural networks are practically composed of multiple distinct modules. For computational tractability, we decompose the full parameter space $\mathbb{R}^d$ into module-wise parameter partitions, indexed by $m \in \{1, \cdots, M\}$. Following the approach of WUDI-Merging (Cheng et al., 2025), our aggregation strategy focuses specifically on 2D weight matrices. Below, we focus on a single module and omit its index for brevity.

Let $\mathbf{W}_{\text{pre}}, \mathbf{W}^{(t)} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ denote the pretrained and expert weights, respectively. The corresponding expert task vector is $\mathbf{U}^{(t)} = \mathbf{W}^{(t)} - \mathbf{W}_{\text{pre}} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$. The merged module weights are expressed in terms of a merged task vector $\mathbf{U}$ as

$$\mathbf{W}_{\text{merged}} = \mathbf{W}_{\text{pre}} + s\mathbf{U}, \tag{2}$$

where s is selected on the validation set.

Activation-based Regression. We frame the estimation of the merged module-wise task vector $\mathbf{U}$ as an activation-based regression problem. In the data-assisted setting, we collect N representative activations for the t-th task, denoted as $\{\mathbf{x}_n^{(t)}\}_{n=1}^{N}$, where $\mathbf{x}_n^{(t)} \in \mathbb{R}^{d_{\text{in}}}$. They are collected by passing the task-specific unlabeled calibration data through the corresponding fine-tuned model. To preserve the task-specific capabilities, we aim to align the residual output induced by the merged task vector $\mathbf{U}$ with that induced by the original expert task vectors $\{\mathbf{U}^{(t)}\}_{t=1}^{T}$. Specifically, we define the residual output as $\mathbf{y}_n^{(t)} = \mathbf{U}^{(t)}\mathbf{x}_n^{(t)} = (\mathbf{W}^{(t)} - \mathbf{W}_{\text{pre}})\mathbf{x}_n^{(t)}$. Before concatenation, we rescale each task's activation–residual pairs by $c_t^{-1/2}$, where $c_t = \sum_{n=1}^{N} \|\mathbf{x}_n^{(t)}\|_2^2 > 0$, giving each task a unit-trace activation Gram. We then concatenate the rescaled vectors across tasks to form $\mathbf{X}$ and $\mathbf{Y}$:

$$\mathbf{X} = \left[c_1^{-1/2}\mathbf{x}_1^{(1)}, \ldots, c_T^{-1/2}\mathbf{x}_N^{(T)}\right], \quad \mathbf{Y} = \left[c_1^{-1/2}\mathbf{y}_1^{(1)}, \ldots, c_T^{-1/2}\mathbf{y}_N^{(T)}\right]. \tag{3}$$

We assume a linear observation model where the residual outputs $\mathbf{Y}$ are generated by the merged module-wise task vector $\mathbf{U}$ acting on $\mathbf{X}$, corrupted by Gaussian noise $\mathbf{E}$:

$$\mathbf{Y} = \mathbf{U}\mathbf{X} + \mathbf{E}, \tag{4}$$

where the noise matrix satisfies $\mathbf{E}_{:,j} \sim \mathcal{N}(0, \beta^{-1}\mathbf{I})$ independently for each column index j. Under this linear observation model, the likelihood function of observing $\mathbf{Y}$ given $\mathbf{U}$ can be expressed as:

$$p(\mathbf{Y} \mid \mathbf{U}, \mathbf{X}, \beta) \propto \exp\left(-\frac{\beta}{2}\|\mathbf{Y} - \mathbf{U}\mathbf{X}\|_F^2\right). \tag{5}$$

Anchor Refinement with Activation Evidence. Purely data-driven estimation of $\mathbf{U}$ on limited calibration data is not only prone to overfitting but also leaves valuable inductive bias of anchor models (e.g., existing merging solutions) unused. To regularize the solution and inject such priors, we introduce a module-wise *anchor* task vector, defined as $\mathbf{U}^{(0)} = \mathbf{W}_{\text{anchor}} - \mathbf{W}_{\text{pre}}$, where $\mathbf{W}_{\text{anchor}}$ denotes the module weights obtained from an existing merging solution (e.g., TA (Ilharco et al., 2023), TIES (Yadav et al., 2023), or TSV (Gargiulo et al., 2025)). We then impose an element-wise independent Gaussian prior on $\mathbf{U}$ centered at anchor task vector $\mathbf{U}^{(0)}$:

$$p(\mathbf{U}|\mathbf{U}^{(0)}, \alpha) \propto \exp\left(-\frac{\alpha}{2}\|\mathbf{U} - \mathbf{U}^{(0)}\|_F^2\right), \tag{6}$$

where α controls the precision of this anchor-induced prior. Given these formulations, the *maximum a posteriori* (MAP) estimate of $\mathbf{U}$ is obtained by maximizing the posterior $p(\mathbf{U}|\mathbf{Y}, \mathbf{X}) \propto$

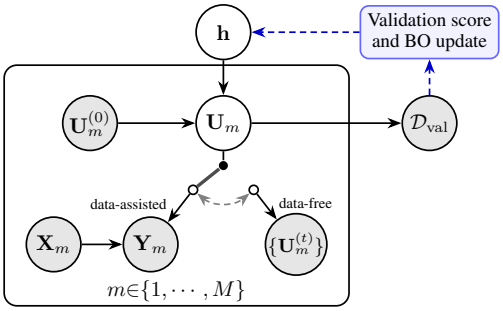

Figure 1: **Inner MAP and outer BO.** ReForge computes closed-form updates from activations or task-vector Gram surrogates. Models assembled with selected scales are evaluated on held-out validation data, guiding joint BO over $\mathbf{h}$.

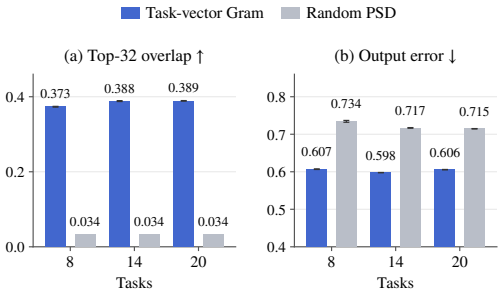

Figure 2: **Task-vector Gram diagnostics.** ViT-B/32 with 8/14/20 tasks. (a) Top-32 subspace overlap with activation Grams (↑). (b) Relative output error against activation-based refinement (↓; ISO-CTS anchor).

$p(\mathbf{Y}|\mathbf{U}, \mathbf{X}, \beta)\, p(\mathbf{U}|\mathbf{U}^{(0)}, \alpha)$. Taking the negative logarithm transforms posterior maximization into a minimization problem. Let the regularization weight $\lambda = \frac{\alpha}{\beta} > 0$, the MAP objective simplifies to:

$$\mathbf{U}_{\text{MAP}} = \arg\min_{\mathbf{U}} \left(\|\mathbf{Y} - \mathbf{U}\mathbf{X}\|_F^2 + \lambda \|\mathbf{U} - \mathbf{U}^{(0)}\|_F^2 \right), \tag{7}$$

where λ balances the empirical fit on the task-specific activations (the first term) with the prior knowledge encapsulated by the anchor model (the second term). Setting the derivative of the objective with respect to $\mathbf{U}$ to zero yields the closed-form solution:

$$\mathbf{U}_{\text{MAP}} = \left(\mathbf{Y}\mathbf{X}^\top + \lambda \mathbf{U}^{(0)} \right) \left(\mathbf{X}\mathbf{X}^\top + \lambda \mathbf{I} \right)^{-1}, \tag{8}$$

where $\mathbf{I} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$ is the identity matrix. Given a $\lambda > 0$, this closed-form solution allows us to compute the optimal merged task vectors for all M modules efficiently. For $\lambda > 0$, the MAP solution is unique and equivalent to anchor-centered ridge regression. The Gaussian posterior also provides a covariance matrix shared by all rows of $\mathbf{U}$:

$$\mathbf{\Sigma}_{\text{row}} = \beta^{-1} (\mathbf{X}\mathbf{X}^\top + \lambda \mathbf{I})^{-1}.$$

We can sample module weights from this posterior and average predictions from the resulting models to improve uncertainty calibration (Section 5.4).

4.2 Joint Hyperparameter Selection via Bayesian Optimization

Eq. (8) solves the regression problem for each of the M modules independently. However, neural network modules are highly coupled, so their regularization strengths and assembly scales must be selected jointly according to the end-to-end performance of the full network. We use a held-out validation set to select these hyperparameters. Let $\mathbf{h} = (\boldsymbol{\lambda}, \mathbf{s}) \in \Omega$ denote a candidate configuration, with $\lambda_m > 0$ and s_m assigned to module m. We evaluate each candidate using the average validation accuracy across T tasks:

$$\text{Score}(\mathbf{h}, \mathcal{D}_{\text{val}}) = \frac{1}{T} \sum_{t=1}^{T} \text{Acc}_t \left(\theta_{\text{merged}}(\mathbf{h}), \mathcal{D}_{\text{val}}^{(t)} \right). \tag{9}$$

Here, $\theta_{\text{merged}}(\mathbf{h})$ is constructed from the refined modules as

$$\mathbf{W}_m(\mathbf{h}) = \mathbf{W}_{\text{pre},m} + s_m \mathbf{U}_m^\star(\lambda_m). \tag{10}$$

This gives the bilevel optimization problem

$$\mathbf{h}^\star \in \arg\max_{\mathbf{h} \in \Omega} \text{Score}(\mathbf{h}, \mathcal{D}_{\text{val}}), \tag{11}$$

$$\text{subject to} \quad \mathbf{U}_m^\star(\lambda_m) = \arg\min_{\mathbf{U}_m} \left[\|\mathbf{Y}_m - \mathbf{U}_m \mathbf{X}_m\|_F^2 + \lambda_m \|\mathbf{U}_m - \mathbf{U}_m^{(0)}\|_F^2 \right], \tag{12}$$

Algorithm 1 ReForge-DA refinement

Require: Anchor task vectors $\mathcal{U}^{(0)} = \{\mathbf{U}_m^{(0)}\}_{m=1}^M$, data $\{\mathbf{X}_m, \mathbf{Y}_m\}_{m=1}^M$, val set $\mathcal{D}_{\text{val}}$, #Iters K
Ensure: Merged model θ_{merged}

```
Initialize evaluation history H ← ∅ and GP surrogate M
for k = 1, 2, ..., K do
    # Outer Loop: Surrogate-Guided Proposal
    (λ^(k), s^(k)) ← arg max_{λ,s} EI((λ, s); M)          ▷ Maximize acquisition function
    # Inner Loop: Closed-form MAP solution
    for each module m ∈ {1, ..., M} do
        U*_m ← (Y_m X_m^⊤ + λ_m^(k) U_m^(0))(X_m X_m^⊤ + λ_m^(k) I)^{-1}   ▷ Solve inner MAP via Eq. (8)
        W_m ← W_{pre,m} + s_m^(k) U*_m
    end for
    Assemble full merged model θ_merged^(k) ← {W_m}_{m=1}^M
    # Outer Loop: Validation evaluation
    f^(k) ← Score((λ^(k), s^(k)), D_val)
    H ← H ∪ {((λ^(k), s^(k)), f^(k))}; Update surrogate M
end for
(λ*, s*) ← arg max_{((λ,s),f)∈H} f                        ▷ Extract best observed configuration
return θ_merged(λ*, s*)
```

where Ω denotes the search space of the regularization strengths and assembly scales in $\mathbf{h}$. The inner problem is the module-wise MAP estimation problem in Eq. (7), with the closed-form solution in Eq. (8). The remaining task is to optimize the outer objective in Eq. (11). We treat this objective as a black-box function of $\mathbf{h}$: each evaluation requires assembling a full merged model and measuring its validation performance.

For efficient outer-loop optimization, we adopt Bayesian optimization with a Gaussian process ($\mathcal{GP}$) surrogate (Frazier, 2018). The outer loop proposes hyperparameter combinations by maximizing Expected Improvement (EI), while the inner loop computes the closed-form module updates used to construct each candidate model. The resulting validation score updates the surrogate and guides subsequent proposals. Figure 1 illustrates this interaction, and Algorithm 1 details the procedure.

Searching separate hyperparameters for all M modules creates a high-dimensional search space ($M = 96$ for ViT-L/14 and $M = 224$ for Llama-3.1-8B). To make the search tractable while retaining module-specific control, we use block-wise parameter tying. We partition consecutive Transformer layers into B sequential blocks. Within each block, modules with the same functional role share a regularization strength, yielding four groups: attention-in (Q/K/V), attention-out, MLP-in, and MLP-out. Together with one block-specific assembly scale s (Eq. (2)), this gives five search variables per block and $5B$ variables in total.

4.3 From Data-Assisted to Data-Free

Task-vector Gram surrogate. The closed-form solution in Eq. (8) requires activation statistics estimated from unlabeled calibration data. When such data are unavailable, a weight-based surrogate is needed. ACE-Merging (Xu et al., 2026, Theorem 1) establishes a proportional relationship between the expected gradient Gram and input second-order statistics under its stated assumptions. Motivated by this connection, we replace each normalized activation Gram with

$$\mathbf{B}_t = \frac{(\mathbf{U}^{(t)})^\top \mathbf{U}^{(t)}}{\|\mathbf{U}^{(t)}\|_F^2}, \qquad \mathbf{U}^{(t)} \neq \mathbf{0}. \tag{13}$$

The denominator normalizes each task-vector Gram to unit trace, removing differences in overall scale across tasks. We use the normalized Gram $\mathbf{B}_t$ to approximate activation statistics without calibration data.

Data-free refinement. Replacing each normalized activation Gram in Eq. (8) with $\mathbf{B}_t$, and using the relation $\mathbf{Y}^{(t)} = \mathbf{U}^{(t)}\mathbf{X}^{(t)}$, yields the data-free closed-form update:

$$\mathbf{U}_{\text{DF}}^*(\tilde{\lambda}) = \left(\sum_{t=1}^T \mathbf{U}^{(t)}\mathbf{B}_t + \tilde{\lambda}\mathbf{U}^{(0)}\right)\left(\sum_{t=1}^T \mathbf{B}_t + \tilde{\lambda}\mathbf{I}\right)^{-1}. \tag{14}$$

Here, $\tilde{\lambda} > 0$ controls anchor regularization in the data-free setting. This update requires only the expert task vectors and the anchor. The task-vector fitting objective underlying Eq. (14) coincides with the interference objective of WUDI-Merging (Cheng et al., 2025, Eq. 21). ReForge-DF regularizes the solution toward an externally constructed anchor task vector $\mathbf{U}^{(0)}$ and admits a closed-form solution.

Empirical evidence. We examine whether task-vector Grams capture useful activation structure and support local refinement (Figure 2). Compared with random PSD controls, task-vector Grams show greater overlap with the dominant activation subspace (Figure 2(a)) and yield module-level outputs closer to those obtained with activation Grams (Figure 2(b)). These findings support their use as surrogates for activation statistics in refinement. Experimental details and additional results are provided in Appendix A.

Refinement with few calibration examples. When a small calibration set is available, we could supplement the weight-based surrogate with activation statistics. Let $\mathbf{X}_{t,n}$ collect the module's input activations from n unlabeled calibration examples for task t. We define the trace-normalized activation Gram as

$$\widehat{\mathbf{G}}_{t,n}^{\mathrm{act}} = \frac{\mathbf{X}_{t,n}\mathbf{X}_{t,n}^{\top}}{\mathrm{tr}(\mathbf{X}_{t,n}\mathbf{X}_{t,n}^{\top})},$$

and combine it with the task-vector Gram $\mathbf{B}_t$:

$$\mathbf{G}_{t,n}^{\mathrm{mix}}(\rho) = (1-\rho)\mathbf{B}_t + \rho\widehat{\mathbf{G}}_{t,n}^{\mathrm{act}}, \qquad 0 \le \rho \le 1. \tag{15}$$

Here, ρ controls their relative contributions: $\rho = 0$ uses only task-vector statistics, whereas $\rho = 1$ uses only activation statistics. Replacing $\mathbf{B}_t$ in Eq. (14) with this mixed Gram gives the corresponding refinement update. Experimental results are reported in Section 5.4.

5 Experiments

We evaluate ReForge-DA and ReForge-DF, the data-assisted and data-free variants of ReForge, on vision and language benchmarks with four backbones and seven anchor choices. Experiments run on eight NVIDIA RTX 6000 Ada 48GB GPUs.

5.1 Experimental Settings

Datasets and Models. For *vision* tasks, we follow the scalability evaluation settings of Gargiulo et al. (2025) and Marczak et al. (2025), covering 8-, 14-, and 20-task merging scenarios with ViT-B/32 and ViT-L/14 backbones. For *language* tasks, following He et al. (2025b), we evaluate ReForge for 5-task merging based on Llama-3.2-3B and Llama-3.1-8B. We use separate validation and test sets for hyperparameter selection and final evaluation, respectively. Full benchmark descriptions, metrics, and asset/license info are provided in Appendix F.

Baselines. We use six merging methods as both baselines and sources of anchor models: TA (Ilharco et al., 2023), TIES (Yadav et al., 2023), RegMean (Jin et al., 2023), TSV (Gargiulo et al., 2025), WUDI-Merging (Cheng et al., 2025), and ISO-CTS (Marczak et al., 2025). We additionally use the pretrained backbone as an anchor, corresponding to a zero-centered prior with $\mathbf{U}^{(0)} = \mathbf{0}$, and report individual expert performance for reference.

Experimental details. Following RegMean (Jin et al., 2023), ReForge-DA collects local activation data $\mathcal{D}_{\text{act}} = \{\mathbf{X}_m, \mathbf{Y}_m\}_{m=1}^{M}$ using 128 and 1,000 samples per task for ViT and Llama, respectively. We partition the networks into $B = 3$ (ViT) and $B = 1$ (Llama) sequential blocks. Each block has 5 hyperparameters to tune: one scale $s \in [1.0, 1.3]$ (Eq. 2) and four group-wise regularization strengths (attention-in/out and MLP-in/out). We search the regularization strengths in logarithmic coordinates over $[10^{-4}, 1]$ for ViT and $[10^{-3}, 100]$ for Llama. We use $K = 200$ BO trials for ViT and $K = 100$ for Llama to search the 15-dimensional and five-dimensional hyperparameter spaces, respectively. Additional complexity analysis and wall-clock runtime breakdowns are provided in Appendix E.

5.2 Main Results

Vision Benchmarks (ViT). Table 1 shows that both ReForge-DA and ReForge-DF improve task accuracy over every evaluated anchor across the tested model sizes and task counts. ReForge-DF

Table 1: Model merging on vision tasks with ViT architectures. The blue subscript values indicate the **relative percentage improvement** compared to the corresponding *anchor*. (Indiv.: Individual)

Model	Tasks	Setting	Indiv.	Pretrained	TA	TIES	RegMean	TSV	WUDI	ISO-CTS
ViT-B/32	8	anchor	92.8	47.7	70.4	75.7	82.3	85.9	**87.0**	86.4
		ReForge-DA	-	84.8$_{+77.8}$	85.0$_{+20.7}$	85.7$_{+13.1}$	85.2$_{+3.5}$	88.9$_{+3.5}$	89.4$_{+2.8}$	**90.2**$_{+4.3}$
		ReForge-DF	-	87.9$_{+84.4}$	87.0$_{+23.6}$	87.0$_{+14.9}$	87.9$_{+6.8}$	87.8$_{+2.2}$	87.6$_{+0.7}$	**89.0**$_{+3.0}$
	14	anchor	90.9	56.9	65.2	68.2	76.6	79.9	80.5	**81.5**
		ReForge-DA	-	78.9$_{+38.8}$	79.2$_{+21.5}$	79.7$_{+16.8}$	79.1$_{+3.2}$	84.3$_{+5.5}$	84.6$_{+5.1}$	**85.5**$_{+4.9}$
		ReForge-DF	-	82.7$_{+45.4}$	82.4$_{+26.4}$	82.3$_{+20.6}$	82.9$_{+8.2}$	82.8$_{+3.7}$	81.7$_{+1.5}$	**84.3**$_{+3.5}$
	20	anchor	91.3	55.7	60.4	64.0	72.2	76.9	76.1	**77.6**
		ReForge-DA	-	75.6$_{+35.7}$	75.5$_{+25.0}$	76.3$_{+19.2}$	75.9$_{+5.0}$	82.0$_{+6.7}$	82.2$_{+8.0}$	**82.8**$_{+6.6}$
		ReForge-DF	-	80.0$_{+43.5}$	78.4$_{+29.8}$	79.2$_{+23.7}$	79.7$_{+10.4}$	79.7$_{+3.7}$	78.0$_{+2.6}$	**81.5**$_{+5.0}$
ViT-L/14	8	anchor	95.8	65.1	84.8	87.0	90.0	93.0	94.0	**94.8**
		ReForge-DA	-	92.3$_{+41.7}$	92.8$_{+9.4}$	93.1$_{+7.1}$	92.7$_{+2.9}$	94.4$_{+1.6}$	94.7$_{+0.7}$	**95.1**$_{+0.3}$
		ReForge-DF	-	94.4$_{+44.9}$	94.2$_{+11.0}$	94.2$_{+8.3}$	94.0$_{+4.5}$	94.4$_{+1.5}$	94.3$_{+0.3}$	**95.0**$_{+0.2}$
	14	anchor	94.3	68.5	79.4	80.2	85.5	89.1	90.5	**91.0**
		ReForge-DA	-	88.0$_{+28.5}$	88.3$_{+11.2}$	88.3$_{+10.1}$	88.2$_{+3.2}$	91.6$_{+2.7}$	91.7$_{+1.3}$	**92.2**$_{+1.4}$
		ReForge-DF	-	90.9$_{+32.8}$	90.9$_{+14.5}$	90.8$_{+13.3}$	90.6$_{+6.0}$	91.3$_{+2.4}$	91.0$_{+0.5}$	**92.1**$_{+1.2}$
	20	anchor	94.7	65.4	74.0	76.7	82.8	87.7	88.4	**90.1**
		ReForge-DA	-	86.0$_{+31.5}$	86.2$_{+16.5}$	86.4$_{+12.6}$	86.2$_{+4.1}$	90.9$_{+3.6}$	90.8$_{+2.7}$	**91.5**$_{+1.5}$
		ReForge-DF	-	89.8$_{+37.3}$	89.6$_{+21.1}$	89.5$_{+16.6}$	89.4$_{+7.9}$	90.3$_{+3.0}$	89.7$_{+1.5}$	**91.1**$_{+1.1}$
Avg. Improvement		ReForge-DA	-	+42.3	+17.4	+13.1	+3.7	+3.9	+3.4	+3.2
		ReForge-DF	-	+48.0	+21.1	+16.2	+7.3	+2.7	+1.2	+2.3

Table 2: Model merging on language tasks with Llama architectures. The blue subscript values indicate the **relative percentage improvement** compared to the corresponding *anchor*.

Model	Setting	Indiv.	Pretrained	TA	TIES	RegMean	TSV	WUDI	ISO-CTS
Llama-3.2-3B	anchor	0.499	0.301	0.438	0.435	0.377	**0.471**	0.461	0.441
	ReForge-DA	-	0.342$_{+13.6}$	0.448$_{+2.3}$	0.447$_{+2.8}$	0.406$_{+7.7}$	0.478$_{+1.5}$	**0.479**$_{+3.9}$	0.454$_{+2.9}$
	ReForge-DF	-	0.450$_{+49.3}$	0.455$_{+3.9}$	0.468$_{+7.6}$	0.456$_{+21.0}$	**0.486**$_{+3.2}$	0.478$_{+3.7}$	0.457$_{+3.6}$
Llama-3.1-8B	anchor	0.630	0.385	0.541	0.554	0.526	0.557	**0.560**	0.556
	ReForge-DA	-	0.484$_{+25.7}$	0.568$_{+5.0}$	**0.579**$_{+4.5}$	0.530$_{+0.8}$	0.574$_{+3.1}$	0.564$_{+0.7}$	0.576$_{+3.6}$
	ReForge-DF	-	0.507$_{+31.7}$	0.568$_{+5.0}$	0.558$_{+0.7}$	0.528$_{+0.4}$	**0.573**$_{+2.9}$	0.561$_{+0.2}$	0.564$_{+1.4}$
Avg. Improvement	ReForge-DA	-	+19.6	+3.6	+3.6	+4.2	+2.3	+2.3	+3.3
	ReForge-DF	-	+40.5	+4.4	+4.2	+10.7	+3.0	+1.9	+2.5

achieves average relative improvements of 21.1% over TA and 16.2% over TIES, demonstrating that substantial gains are possible without calibration data. The improvements extend to strong anchors such as WUDI-Merging and ISO-CTS. Although ISO-CTS already performs well on ViT merging benchmarks, both ReForge variants consistently improve its performance, including when merging 20 tasks. On ViT-B/32, ReForge-DA and ReForge-DF raise its accuracy from 77.6% to 82.8% and 81.5%, respectively. Similar gains are observed on ViT-L/14.

ReForge also narrows the gap between merged models and individual task-specific experts. On the ViT-L/14 eight-task benchmark with the ISO-CTS anchor, ReForge-DA reaches 95.1% and ReForge-DF reaches 95.0%, compared with 95.8% for the individual experts. Standard deviations across five seeds and per-task results are provided in Appendices C and G, respectively.

Language Benchmarks (Llama). Table 2 shows that both ReForge variants improve performance over every evaluated anchor on the five-task language benchmarks. The gains extend from weaker anchors to strong merging baselines. On Llama-3.2-3B, ReForge-DF achieves a relative improvement of 49.3% over the pretrained anchor and raises the TSV score from 0.471 to 0.486, achieving the best result among the evaluated 3B merging methods. ReForge-DF also often outperforms ReForge-DA on this backbone.

On Llama-3.1-8B, ReForge-DF improves the TSV score from 0.557 to 0.573, while ReForge-DA with TIES achieves the best result among the evaluated 8B merging methods at 0.579. The relative performance of the two variants therefore depends on the backbone and anchor. Category-level results are provided in Appendix G.

Table 3: Different hyperparameter search methods (*shared-λ*, *random search*, and *BO*) and their impacts on 8-, 14-, and 20-task merging with ViT-B/32 architecture. Results for ReForge-DA and ReForge-DF are reported as mean ± std over five seeds.

Setting	Variant	8 Tasks			14 Tasks			20 Tasks		
		TSV	WUDI	ISO-CTS	TSV	WUDI	ISO-CTS	TSV	WUDI	ISO-CTS
anchor	-	85.9	87.0	86.4	79.9	80.5	81.5	76.9	76.1	77.6
ReForge-DA	shared-λ	$88.1_{\pm 0.01}$	$89.0_{\pm 0.01}$	$89.3_{\pm 0.01}$	$83.2_{\pm 0.00}$	$83.5_{\pm 0.00}$	$83.9_{\pm 0.00}$	$80.1_{\pm 0.01}$	$80.6_{\pm 0.01}$	$80.3_{\pm 0.02}$
	random	$88.6_{\pm 0.09}$	$89.2_{\pm 0.02}$	$89.7_{\pm 0.24}$	$83.7_{\pm 0.11}$	$83.9_{\pm 0.30}$	$84.9_{\pm 0.07}$	$80.6_{\pm 0.18}$	$81.2_{\pm 0.32}$	$81.7_{\pm 0.37}$
	BO	$\mathbf{88.9_{\pm 0.05}}$	$\mathbf{89.4_{\pm 0.02}}$	$\mathbf{90.2_{\pm 0.03}}$	$\mathbf{84.3_{\pm 0.04}}$	$\mathbf{84.6_{\pm 0.04}}$	$\mathbf{85.5_{\pm 0.06}}$	$\mathbf{82.0_{\pm 0.04}}$	$\mathbf{82.2_{\pm 0.06}}$	$\mathbf{82.8_{\pm 0.05}}$
ReForge-DF	shared-λ	$87.1_{\pm 0.00}$	$87.1_{\pm 0.02}$	$88.0_{\pm 0.00}$	$81.5_{\pm 0.00}$	$80.7_{\pm 0.00}$	$82.7_{\pm 0.00}$	$77.9_{\pm 0.00}$	$76.7_{\pm 0.01}$	$78.8_{\pm 0.00}$
	random	$87.4_{\pm 0.20}$	$87.4_{\pm 0.13}$	$88.1_{\pm 0.23}$	$82.2_{\pm 0.24}$	$81.2_{\pm 0.08}$	$83.3_{\pm 0.18}$	$79.0_{\pm 0.18}$	$77.3_{\pm 0.20}$	$80.0_{\pm 0.33}$
	BO	$\mathbf{87.8_{\pm 0.04}}$	$\mathbf{87.6_{\pm 0.04}}$	$\mathbf{89.0_{\pm 0.03}}$	$\mathbf{82.8_{\pm 0.03}}$	$\mathbf{81.7_{\pm 0.06}}$	$\mathbf{84.3_{\pm 0.05}}$	$\mathbf{79.7_{\pm 0.05}}$	$\mathbf{78.0_{\pm 0.02}}$	$\mathbf{81.5_{\pm 0.04}}$

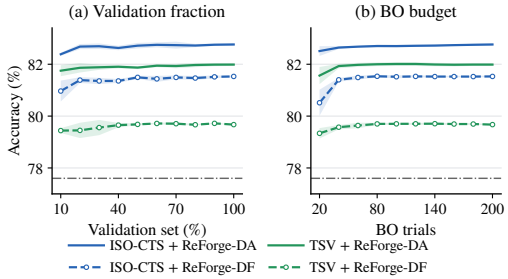

Figure 3: **ReForge ablations on ViT-B/32, 20 tasks.** Test accuracy versus validation fraction (left) and BO trials K (right); mean ± std over five seeds. The gray dot-dashed line is the ISO-CTS baseline.

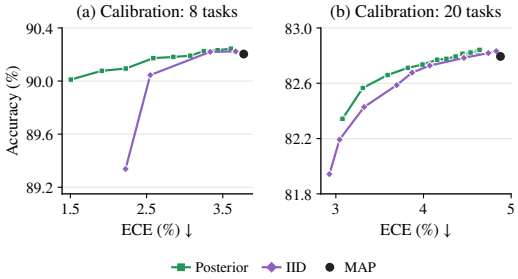

Figure 4: **Posterior calibration on ViT-B/32.** Test-set accuracy–ECE Pareto frontiers for (a) 8 and (b) 20 tasks, using ISO-CTS. Posterior and IID sampling each average ten predictions; black points mark MAP.

5.3 Ablation Study

To investigate the effectiveness of BO-based global coordination, Table 3 compares ReForge (with *BO*) against two hyperparameter-tuning baselines: *shared-λ* and *random search*. While *shared-λ* assigns a single regularization strength λ to all four module groups and optimizes it via an extensive grid search, *random search* relaxes this constraint by allowing module-specific hyperparameters, but tunes them using 200 trials of random search.

Shared-λ improves all evaluated anchors, showing that refinement remains effective with a shared regularization strength. Relaxing the shared constraint to module-specific hyperparameters with *random search* further improves the performance, validating the importance of heterogeneous regularization across different module groups. Finally, ReForge (with *BO*) achieves the best overall results by effectively coordinating the module-specific hyperparameters with guided search. Its advantage is most pronounced in the challenging 20-task data-free setting. On the TSV anchor, ReForge (with *BO*) reaches an accuracy of 79.7%, compared with 79.0% for *random search* and 77.9% for *shared-λ* tuning. This suggests that BO becomes particularly useful when the search space is more complex and the validation signal needs to balance task interference effectively in large-scale merging settings.

We further examine whether ReForge's gains can be achieved solely by applying heterogeneous scaling coefficients to the anchor task vectors across module groups and blocks, without regression refinement. Both ReForge-DA and ReForge-DF outperform anchor rescaling across six ViT-B/32 task–anchor configurations with the same nominal search dimension and validation-evaluation budget (Appendix B). We also compare task-vector Grams with random positive semidefinite (PSD) and isotropic matrices under a common evaluation and search protocol. Across the six configurations, task-vector Grams improve ReForge-DF's mean test accuracy by 0.81–1.40 percentage points over random PSD matrices and by 0.81–1.29 percentage points over isotropic matrices (Appendix A.2).

Table 4: Few-shot Gram mixing on ViT-B/32 (test accuracy, %). Activation-only and mixed variants use one or five unlabeled calibration examples per task, with the mix weight selected by BO on separate validation data. ReForge-DA with 128 examples per task provides a reference. Subscripts indicate sample standard deviations over five seeds with fixed checkpoints and data.

Setting	8 Tasks			14 Tasks			20 Tasks		
	TSV	WUDI	ISO-CTS	TSV	WUDI	ISO-CTS	TSV	WUDI	ISO-CTS
anchor	85.9	87.0	86.4	79.9	80.5	81.5	76.9	76.1	77.6
ReForge-DF	$87.8_{\pm 0.04}$	$87.6_{\pm 0.04}$	$89.0_{\pm 0.03}$	$82.8_{\pm 0.03}$	$81.7_{\pm 0.06}$	$84.3_{\pm 0.05}$	$79.7_{\pm 0.05}$	$78.0_{\pm 0.02}$	$81.5_{\pm 0.04}$
1-shot	$87.8_{\pm 0.06}$	$\mathbf{89.0_{\pm 0.02}}$	$88.7_{\pm 0.03}$	$83.2_{\pm 0.15}$	$\mathbf{84.1_{\pm 0.06}}$	$84.2_{\pm 0.07}$	$80.6_{\pm 0.06}$	$\mathbf{81.1_{\pm 0.04}}$	$80.5_{\pm 0.06}$
1-shot mix	$\mathbf{88.9_{\pm 0.05}}$	$\mathbf{89.0_{\pm 0.08}}$	$\mathbf{89.7_{\pm 0.05}}$	$\mathbf{84.3_{\pm 0.07}}$	$\mathbf{84.1_{\pm 0.05}}$	$\mathbf{85.7_{\pm 0.04}}$	$\mathbf{81.5_{\pm 0.15}}$	$\mathbf{81.1_{\pm 0.03}}$	$\mathbf{82.8_{\pm 0.03}}$
5-shot	$88.5_{\pm 0.11}$	$\mathbf{89.3_{\pm 0.06}}$	$89.5_{\pm 0.03}$	$84.1_{\pm 0.04}$	$\mathbf{84.6_{\pm 0.07}}$	$85.3_{\pm 0.04}$	$81.7_{\pm 0.02}$	$\mathbf{81.9_{\pm 0.04}}$	$82.2_{\pm 0.02}$
5-shot mix	$\mathbf{89.4_{\pm 0.11}}$	$\mathbf{89.3_{\pm 0.07}}$	$\mathbf{90.2_{\pm 0.02}}$	$\mathbf{84.9_{\pm 0.11}}$	$\mathbf{84.6_{\pm 0.08}}$	$\mathbf{86.1_{\pm 0.06}}$	$\mathbf{82.4_{\pm 0.03}}$	$\mathbf{81.9_{\pm 0.03}}$	$\mathbf{83.7_{\pm 0.02}}$
128-sample ReForge-DA	$88.9_{\pm 0.05}$	$89.4_{\pm 0.02}$	$90.2_{\pm 0.03}$	$84.3_{\pm 0.04}$	$84.6_{\pm 0.04}$	$85.5_{\pm 0.06}$	$82.0_{\pm 0.04}$	$82.2_{\pm 0.06}$	$82.8_{\pm 0.05}$

5.4 Additional Analyses

Sensitivity to the Fraction of Validation Set. Figure 3 (left) reports the final test performance as the fraction of validation set used for BO evolves. Across both TSV and ISO-CTS anchors, in data-assisted and data-free settings, ReForge consistently outperforms the strongest anchor baseline (ISO-CTS) even when a small fraction of the validation set is used. The performance curves are relatively stable from 10% to 100% of the validation set, indicating that the outer-loop optimization is not overly sensitive to the validation-set size. This suggests that ReForge can use a small validation subset to reduce the optimization overhead while preserving most of the gains from global hyperparameter search.

Sensitivity to BO Budget K. Figure 3 (right) reports the evolution of final test performance as the number of BO trials (K) increases. ReForge consistently outperforms the strongest anchor baseline (ISO-CTS) under a small BO budget (K). The performance curves reach a plateau after roughly 40–60 trials, while increasing the budget further yields only a marginal return. These results indicate that BO is efficient in global hyperparameter search, reaching stable results with a modest budget. Additional runtime–performance comparisons against other methods are reported in Appendix E.3.

Few-shot calibration and Gram mixing. On ViT-B/32, Gram mixing matches or improves upon activation-only mean accuracy in all nine task-count–anchor configurations with either one or five calibration examples per task (Table 4). For ISO-CTS on 20 tasks, one-shot mixing raises accuracy from 80.5% to 82.8%, matching the 128-sample ReForge-DA result. Across the nine configurations, one- and five-shot mixing match or exceed the 128-sample ReForge-DA mean in four and seven cases, respectively.

Posterior sampling and calibration. We evaluate whether posterior sampling improves the uncertainty calibration of ReForge-DA on the ViT-B/32 8- and 20-task benchmarks with the ISO-CTS anchor. Starting from the same MAP solution, we compare sampling from the posterior with adding IID Gaussian noise. For each method, we generate ten sampled models and average their predicted class probabilities. We vary the sampling parameters and plot the resulting accuracy–calibration Pareto frontiers on the test sets in Figure 4. Posterior sampling achieves a better trade-off between accuracy and calibration. Appendix D provides the experimental details.

6 Conclusion

This paper introduces **ReForge**, a *plug-and-play* framework for refining existing merged models. ReForge incorporates expert-derived information through module-wise Bayesian linear regression with an anchor-centered prior and a closed-form posterior. Bayesian optimization jointly selects heterogeneous regularization strengths and assembly scales according to validation performance. We also derive a data-free variant of ReForge using task-vector Grams as surrogates for activation statistics, while preserving the closed-form solution. Across vision and language benchmarks, ReForge consistently improves diverse anchor baselines, scales to challenging 20-task settings, and closely approaches the performance of individual fine-tuned experts on high-capacity ViT architectures. Our results show that combining informative anchors with global hyperparameter coordination provides a practical path towards refining merged models when calibration data are limited or unavailable.

AI USE STATEMENT

Generative AI tools assisted with manuscript drafting and editing, as well as LaTeX formatting checks. The authors are responsible for the final content of this paper.

REFERENCES

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101: Mining discriminative components with random forests. In *ECCV*, 2014.

Kesheng Chen, Yamin Hu, Zhenqian Zhu, Yiya Diao, and Wenjian Luo. SIP-BMM: Constructing capability-efficiency pareto set of LLMs via bayesian model merging with structural importance prior. *arXiv preprint arXiv:2512.09972v4*, 2026. URL https://arxiv.org/abs/2512.09972v4.

Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 2017.

Runxi Cheng, Feng Xiong, Yongxian Wei, Wanyun Zhu, and Chun Yuan. Whoever started the interference should end it: Guiding data-free model merging via task vectors. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 10121–10143. PMLR, 2025. URL https://proceedings.mlr.press/v267/cheng25h.html.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *CVPR*, 2014.

Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS*, 2011.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre van Schaik. Emnist: Extending mnist to handwritten letters. In *IJCNN*, 2017.

Guodong Du, Junlin Lee, Jing Li, Runhua Jiang, Yifei Guo, Shuyang Yu, Hanting Liu, Sim Kuan Goh, Ho-Kin Tang, Daojing He, and Min Zhang. Parameter competition balancing for model merging. *Advances in Neural Information Processing Systems*, 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/99fc8bc48b917c301a80cb74d91c0c06-Abstract-Conference.html.

Peter I Frazier. Bayesian optimization. In *Recent advances in optimization and modeling of contemporary problems*, pp. 255–278. Informs, 2018.

Antonio Andrea Gargiulo, Donato Crisostomi, Maria Sofia Bucarelli, Simone Scardapane, Fabrizio Silvestri, and Emanuele Rodolà. Task singular vectors: Reducing task interference in model merging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18695–18705, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Gargiulo_Task_Singular_Vectors_Reducing_Task_Interference_in_Model_Merging_CVPR_2025_paper.html.

Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 4 edition, 2013.

Ian J. Goodfellow, Dumitru Erhan, Pierre Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, et al. Challenges in representation learning: A report on three machine learning contests. *arXiv preprint arXiv:1307.0414*, 2013.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017.

Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms. *arXiv preprint arXiv:2406.18495*, 2024.

Yifei He, Yuzheng Hu, Yong Lin, Tong Zhang, and Han Zhao. Localize-and-stitch: Efficient model merging via sparse task arithmetic. *Transactions on Machine Learning Research*, 2025a. URL https://openreview.net/forum?id=9CWU8Oi86d. Accepted to TMLR.

Yifei He, Siqi Zeng, Yuzheng Hu, Rui Yang, Tong Zhang, and Han Zhao. Mergebench: A benchmark for merging domain-specialized llms. *arXiv preprint arXiv:2505.10833*, 2025b. URL https://arxiv.org/abs/2505.10833.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2019.

Hugging Face. The hugging face hub. https://huggingface.co, 2026. Accessed: 2026-05-04.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=6t0Kwf8-jrj.

Liwei Jiang, Kavel Rao, Seungju Han, Allyson Ettinger, Faeze Brahman, Sachin Kumar, Niloofar Mireshghallah, Ximing Lu, Maarten Sap, Yejin Choi, et al. Wildteaming at scale: From in-the-wild jailbreaks to (adversarially) safer language models. *Advances in Neural Information Processing Systems*, 37:47094–47165, 2024.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=FCnohuR6AnM.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV Workshops*, 2013.

Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being bayesian, even just a bit, fixes overconfidence in relu networks. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5436–5446. PMLR, 2020.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Viet Lai, Chien Nguyen, Nghia Ngo, Thuat Nguyen, Franck Dernoncourt, Ryan Rossi, and Thien Nguyen. Okapi: Instruction-tuned large language models in multiple languages with reinforcement learning from human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 318–327, 2023.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tülu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

Sanwoo Lee, Jiahao Liu, Qifan Wang, Jingang Wang, Xunliang Cai, and Yunfang Wu. Dynamic fisher-weighted model merging via Bayesian optimization. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4923–4935, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.naacl-long.254. URL https://aclanthology.org/2025.naacl-long.254/.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. Hugging Face repository, 2024.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatGPT really correct? rigorous evaluation of large language models for code generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Daniel Marczak, Simone Magistri, Sebastian Cygert, Bartłomiej Twardowski, Andrew D. Bagdanov, and Joost van de Weijer. No task left behind: Isotropic model merging with common and task-specific subspaces. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 43177–43199. PMLR, 2025. URL https://proceedings.mlr.press/v267/marczak25a.html.

Michael S. Matena and Colin A. Raffel. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/70c26937fbf3d4600b69a129031b66ec-Abstract-Conference.html.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, et al. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshops*, 2011.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, 2008.

Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *CVPR*, 2012.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Paul Röttger, Hannah Rose Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. Xstest: A test suite for identifying exaggerated safety behaviours in large language models. *arXiv preprint arXiv:2308.01263*, 2023.

Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017. URL https://arxiv.org/abs/1706.05098.

Shivalika Singh, Freddie Vargus, Daniel Dsouza, Börje F Karlsson, Abinaya Mahendiran, Wei-Yin Ko, Herumb Shandilya, Jay Patel, Deividas Mataciunas, Laura OMahony, et al. Aya dataset: An open-access collection for multilingual instruction tuning. *arXiv preprint arXiv:2402.06619*, 2024.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, 2013.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: A multi-class classification competition. In *IJCNN*, 2011.

Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving. *Advances in Neural Information Processing Systems*, 37:7821–7846, 2024.

Bastiaan S. Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. Rotation equivariant cnns for digital pathology. In *MICCAI*, 2018.

Yongxian Wei, Anke Tang, Li Shen, Zixuan Hu, Chun Yuan, and Xiaochun Cao. Modeling multi-task model merging as adaptive projective gradient descent. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 66178–66193. PMLR, 2025. URL https://openreview.net/forum?id=EqoKRSR5Pa.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. Magicoder: Empowering code generation with oss-instruct. *arXiv preprint arXiv:2312.02120*, 2023.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Exploring a large collection of scene categories. In *IJCV*, 2016.

Bo Xu, Haotian Wu, Hehai Lin, Weiquan Huang, Beier Zhu, Yao Shu, and Chengwei Qin. ACE-Merging: Data-free model merging with adaptive covariance estimation. *arXiv preprint arXiv:2603.02945*, 2026. URL https://arxiv.org/abs/2603.02945.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A. Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=xtaX3WyCj1.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. AdaMerging: Adaptive model merging for multi-task learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/62868cc2fc1eb5cdf321d05b4b88510c-Abstract-Conference.html.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57755–57775. PMLR, 2024. URL https://proceedings.mlr.press/v235/yu24p.html.

Frederic Z. Zhang, Paul Albert, Cristian Rodriguez-Opazo, Anton van den Hengel, and Ehsan Abbasnejad. Knowledge composition using task vectors with learned anisotropic scaling. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://papers.nips.cc/paper_files/paper/2024/hash/7c7baa87e763a7e2fa2527e7bf105508-Abstract-Conference.html.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

A Empirical Diagnostics of the Task-Vector Gram Surrogate

A.1 Local Subspace and Output Diagnostics

We compare task-vector Grams with Gaussian random-PSD and isotropic controls on ViT-B/32, using 8, 14, and 20 tasks with ISO-CTS and Task Arithmetic anchors. We select the ridge strength and correction scale using calibration activations, then evaluate local-output agreement on held-out activations.

Data and evaluation protocol. For each task, we exclude the fixed BO-validation examples and shuffle the remaining training images to form a calibration set of 128 images and a held-out set of another 128 images. We repeat this procedure with five seeds. Calibration and held-out sets are disjoint within each split, although images may recur across splits. Model checkpoints remain fixed, and each task's splits are shared across anchors and task-count settings.

We collect input activations from all 48 linear modules, including the attention output projections. Each held-out image contributes 50 token activations, giving 6,400 activation vectors per task and module. Calibration activations are used for parameter selection; held-out activations are reserved for evaluating local-output agreement.

Gram construction and local regression. Suppress the module index and write $\mathbf{V}_t = \mathbf{U}^{(t)}$ and $\mathbf{U}_0 = \mathbf{U}^{(0)}$. With activations stored column-wise as in the main text, define

$$\mathbf{H}_t^{\mathrm{act}} = \mathbf{X}_t^{\mathrm{cal}}(\mathbf{X}_t^{\mathrm{cal}})^\top, \quad \mathbf{H}_t^{\mathrm{tv}} = \mathbf{V}_t^\top \mathbf{V}_t, \quad \widetilde{\mathbf{H}}_t^q = \frac{\mathbf{H}_t^q}{\mathrm{tr}(\mathbf{H}_t^q)}, \quad q \in \{\mathrm{act}, \mathrm{tv}, \mathrm{rnd}, \mathrm{iso}\}. \tag{16}$$

The random control uses $\mathbf{H}_t^{\mathrm{rnd}} = \mathbf{R}_t^\top \mathbf{R}_t$, with independent standard Gaussian entries in $\mathbf{R}_t \in \mathbb{R}^{d_{\mathrm{out}} \times d_{\mathrm{in}}}$. Ten fixed random draws are generated per task and module; their realizations are shared across anchors. The isotropic control uses $\widetilde{\mathbf{H}}_t^{\mathrm{iso}} = \mathbf{I}/d_{\mathrm{in}}$ for each task. Trace normalization is applied separately to each task Gram, before summing:

$$\mathbf{G}_q = \sum_t \widetilde{\mathbf{H}}_t^q, \qquad \mathbf{C}_q = \sum_t \mathbf{V}_t \widetilde{\mathbf{H}}_t^q, \qquad \mathbf{Z}_q(\lambda) = (\mathbf{C}_q - \mathbf{U}_0 \mathbf{G}_q)(\mathbf{G}_q + \lambda \mathbf{I})^{-1}. \tag{17}$$

Here $\mathbf{Z}_q$ is the *anchor-centered correction* of the corresponding ridge solution.

Local parameter selection. For each module, anchor, and split, select a shared ridge strength and a nonnegative correction scale from calibration data:

$$(\lambda_*, a_*) \in \arg\min_{\lambda \in \Lambda,\ a \geq 0} \frac{\|\mathbf{Z}_{\mathrm{act}}(\lambda) - a\mathbf{Z}_{\mathrm{tv}}(\lambda)\|_F}{\|\mathbf{Z}_{\mathrm{act}}(\lambda)\|_F}, \qquad \Lambda = \{10^{-6+8j/256} : j = 0, \ldots, 256\}. \tag{18}$$

For each λ, the optimal scale is $a(\lambda) = \max\{0, \langle \mathbf{Z}_{\mathrm{tv}}, \mathbf{Z}_{\mathrm{act}} \rangle_F / \|\mathbf{Z}_{\mathrm{tv}}\|_F^2\}$. Both the random-PSD and isotropic output controls use the task-vector-fitted λ_* and a_*.

Dominant eigenspace overlap. Let $\mathbf{Q}_q$ contain the top $k = 32$ orthonormal eigenvectors of $\mathbf{G}_q$. We report

$$\mathrm{Overlap}_{32}(q) = \frac{1}{32} \|\mathbf{Q}_{\mathrm{act}}^\top \mathbf{Q}_q\|_F^2. \tag{19}$$

This statistic does not depend on the anchor or fitted parameters. The isotropic Gram has no unique top-32 eigenspace because all eigenvalues are equal; its overlap is therefore reported as a random-basis Monte Carlo reference. For each module, we take the median overlap of ten Haar-random 32-dimensional subspaces with a fixed reference subspace, then average across modules. We report the mean and sample standard deviation over five Monte Carlo repetitions.

Ridge-filtered operator error. Define the ridge-filtered operator $\mathbf{F}_\lambda(\mathbf{G}) = \mathbf{G}(\mathbf{G} + \lambda \mathbf{I})^{-1}$. For each candidate, fit its nonnegative Gram scale $c_q = \max\{0, \langle \mathbf{G}_{\mathrm{act}}, \mathbf{G}_q \rangle_F / \|\mathbf{G}_q\|_F^2\}$ and compute

$$E_{\mathrm{op}}(q) = \frac{\|\mathbf{F}_{\lambda_*}(\mathbf{G}_{\mathrm{act}}) - \mathbf{F}_{\lambda_*}(c_q \mathbf{G}_q)\|_F}{\|\mathbf{F}_{\lambda_*}(\mathbf{G}_{\mathrm{act}})\|_F}. \tag{20}$$

The ridge strength is fixed by Eq. (18); each surrogate, including each random draw, receives its own prescribed c_q. The Gram scale c_q differs from the correction scale a_*. The calculation uses float64 eigendecompositions and spectral filtering.

Held-out local-output error. Regression Grams are trace-normalized per task; held-out Grams retain their original scale for all three surrogate evaluations. Let $\mathbf{H}_{\text{hold}} = \sum_t \mathbf{X}_t^{\text{hold}} (\mathbf{X}_t^{\text{hold}})^\top$ be the sum of *unnormalized* held-out Grams and $\mathbf{D}_q = \mathbf{Z}_{\text{act}}(\lambda_*) - a_* \mathbf{Z}_q(\lambda_*)$. The output diagnostic is

$$E_{\text{out}}(q) = \left[\frac{\text{tr}(\mathbf{D}_q \mathbf{H}_{\text{hold}} \mathbf{D}_q^\top)}{\text{tr}(\mathbf{Z}_{\text{act}}(\lambda_*) \mathbf{H}_{\text{hold}} \mathbf{Z}_{\text{act}}(\lambda_*)^\top)} \right]^{1/2}. \tag{21}$$

This measures disagreement between local *anchor-centered residual outputs*.

Summary statistics. For each data split, we average each metric over the 48 modules. For the random-PSD control, we first take the median over ten random draws within each module. We then report the mean and sample standard deviation across the five data splits. The isotropic subspace-overlap reference is summarized separately, with its standard deviation reflecting random-basis sampling.

Table 5: Task-vector Gram diagnostics with random-PSD and isotropic controls. Values are means with subscript sample standard deviations over five seeds, averaging 48 modules per seed. Random PSD uses the median of ten draws within each module. † marks the isotropic random-basis reference, whose SD reflects Monte Carlo sampling rather than data splits.

Tasks	Anchor	Gram	Top-32 overlap ↑	Ridge-filtered operator error ↓	Held-out local output error ↓
8	ISO-CTS	Task vector	$0.3733_{\pm 0.0011}$	$0.4367_{\pm 0.0048}$	$0.6069_{\pm 0.0009}$
		Random PSD	$0.0338_{\pm <0.0001}$	$0.5986_{\pm 0.0042}$	$0.7344_{\pm 0.0025}$
		Isotropic	$0.0339^{\dagger}_{\pm <0.0001}$	$0.6111_{\pm 0.0038}$	$0.7318_{\pm 0.0025}$
8	TA	Task vector	$0.3733_{\pm 0.0011}$	$0.7307_{\pm 0.0023}$	$0.8164_{\pm 0.0009}$
		Random PSD	$0.0338_{\pm <0.0001}$	$0.9111_{\pm 0.0016}$	$1.0040_{\pm 0.0001}$
		Isotropic	$0.0339^{\dagger}_{\pm <0.0001}$	$0.9185_{\pm 0.0014}$	$1.0040_{\pm 0.0001}$
14	ISO-CTS	Task vector	$0.3884_{\pm 0.0013}$	$0.4224_{\pm 0.0006}$	$0.5977_{\pm 0.0004}$
		Random PSD	$0.0340_{\pm <0.0001}$	$0.6003_{\pm 0.0008}$	$0.7169_{\pm 0.0009}$
		Isotropic	$0.0339^{\dagger}_{\pm <0.0001}$	$0.6082_{\pm 0.0008}$	$0.7153_{\pm 0.0009}$
14	TA	Task vector	$0.3884_{\pm 0.0013}$	$0.7187_{\pm 0.0017}$	$0.8025_{\pm 0.0008}$
		Random PSD	$0.0339_{\pm <0.0001}$	$0.8805_{\pm 0.0024}$	$1.0032_{\pm <0.0001}$
		Isotropic	$0.0339^{\dagger}_{\pm <0.0001}$	$0.8850_{\pm 0.0019}$	$1.0032_{\pm <0.0001}$
20	ISO-CTS	Task vector	$0.3890_{\pm 0.0008}$	$0.4242_{\pm 0.0009}$	$0.6056_{\pm 0.0006}$
		Random PSD	$0.0338_{\pm <0.0001}$	$0.6036_{\pm 0.0012}$	$0.7146_{\pm 0.0006}$
		Isotropic	$0.0339^{\dagger}_{\pm <0.0001}$	$0.6084_{\pm 0.0012}$	$0.7137_{\pm 0.0006}$
20	TA	Task vector	$0.3890_{\pm 0.0008}$	$0.6928_{\pm 0.0012}$	$0.7703_{\pm 0.0006}$
		Random PSD	$0.0338_{\pm <0.0001}$	$0.8954_{\pm 0.0013}$	$0.9989_{\pm 0.0001}$
		Isotropic	$0.0339^{\dagger}_{\pm <0.0001}$	$0.8979_{\pm 0.0013}$	$0.9989_{\pm 0.0001}$

The task-vector surrogate has higher subspace overlap than the random-PSD control and lower operator and output errors than both controls in all six task–anchor settings (Table 5). Its overlap also exceeds the isotropic random-basis reference. The random-PSD and isotropic controls yield similar output errors when evaluated with the task-vector-fitted parameters.

A.2 Task-Vector Grams versus Uninformative Controls

Setup. We assess whether task-specific information in the task-vector Grams contributes to the final performance of ReForge-DF. On ViT-B/32 with TSV and ISO-CTS anchors, we compare the task-vector Gram with two substitutes: a randomly generated positive semidefinite matrix (Random PSD), and the isotropic matrix $\mathbf{I}/d_{\text{in}}$, where d_{in} is the module input dimension. The isotropic control weights all input directions equally, while the random PSD control introduces structure without using task-specific weight information. The comparisons use the same checkpoints, validation and test data, normalization convention, search space, and evaluation budget. Results summarize five seeds.

Table 6: Gram replacement ablation on ViT-B/32 under a common evaluation and search protocol. Test accuracy (%) is reported as mean with subscript $\pm$ standard deviation across five seeds. TV Gram denotes the task-vector Gram used by ReForge-DF; the isotropic substitute is $\mathbf{I}/d_{\text{in}}$. The last two columns report differences between accuracy means in percentage points. Means are rounded to one decimal place; standard deviations and differences are reported to two decimal places. Differences are computed before rounding.

Tasks	Anchor	TV Gram	Random PSD	Isotropic	TV − PSD	TV − Iso.
8	TSV	$\mathbf{87.8}_{\pm 0.04}$	$86.4_{\pm 0.17}$	$86.5_{\pm 0.15}$	1.40	1.29
8	ISO-CTS	$\mathbf{89.0}_{\pm 0.03}$	$87.8_{\pm 0.06}$	$87.9_{\pm 0.04}$	1.16	1.11
14	TSV	$\mathbf{82.8}_{\pm 0.03}$	$81.5_{\pm 0.04}$	$81.6_{\pm 0.03}$	1.29	1.23
14	ISO-CTS	$\mathbf{84.3}_{\pm 0.05}$	$83.4_{\pm 0.02}$	$83.4_{\pm 0.04}$	0.87	0.87
20	TSV	$\mathbf{79.7}_{\pm 0.05}$	$78.9_{\pm 0.02}$	$78.9_{\pm 0.03}$	0.81	0.81
20	ISO-CTS	$\mathbf{81.5}_{\pm 0.04}$	$80.5_{\pm 0.03}$	$80.5_{\pm 0.02}$	1.03	1.02

Results. Table 6 compares task-vector Grams with random PSD and isotropic substitutes under a common evaluation and search protocol. Across all six settings, task-vector Grams improve mean test accuracy by 0.81–1.40 percentage points over random PSD matrices and by 0.81–1.29 points over isotropic matrices. The two controls differ by at most 0.11 points, with no consistent advantage from random anisotropy. These results show that task-specific structure in the Gram matrices contributes to ReForge-DF refinement beyond the use of an arbitrary PSD weighting matrix.

B Anchor Rescaling versus Regression Refinement

Anchor-rescaling control. We test whether rescaling an existing anchor alone can explain ReForge's gains. For each target module, the control rescales the anchor task vector instead of computing a regression update. On ViT-B/32, we report TSV and ISO-CTS anchors for 8-, 14-, and 20-task merging. The 12 Transformer layers are partitioned into three consecutive blocks (layers 0–3, 4–7, and 8–11), each containing four module groups: attention-in, attention-out, MLP-in, and MLP-out. For each target module m in block b and group g, the control uses

$$\mathbf{W}_{\text{ctrl},m} = \mathbf{W}_{\text{pre},m} + s_b a_{b,g} \mathbf{U}_m^{(0)}, \qquad s_b \in [1, 1.3], \quad a_{b,g} \in [0, 2]. \tag{22}$$

The control has 15 search parameters: three block-wise scales s_b and twelve group-wise scales $a_{b,g}$. The group factors $a_{b,g}$ apply only to the target modules. For other parameters within block b, we scale their offsets from the pretrained model by s_b, giving $\theta_{\text{ctrl}} = \theta_{\text{pre}} + s_b(\theta_{\text{anchor}} - \theta_{\text{pre}})$. Parameters outside the blocks retain their anchor values, following the main method's assembly rule.

We evaluate six task–anchor configurations using five seeds and 200 validation evaluations per run. Three initial configurations set all block scales to one and all group scales to zero, one, or two, respectively; their evaluations count toward the budget. The control matches ReForge's nominal search dimension and evaluation budget, but uses a different parameterization. Each run selects the configuration with the highest mean validation accuracy for test evaluation. We report mean test accuracy and standard deviation across the five seeds, using the corresponding main-experiment results for ReForge.

Results. ReForge-DA and ReForge-DF outperform the rescaling control in all six configurations, with larger gains from the data-assisted variant (Table 7). Gains range from 1.6 to 3.4 percentage points for ReForge-DA and from 0.4 to 1.4 percentage points for ReForge-DF.

Table 7: Anchor rescaling versus regression refinement on ViT-B/32. Test accuracy (%) is reported as mean ± sample standard deviation across five search seeds. Δ_{DF} and Δ_{DA} denote gains over the rescaling control for ReForge-DF and ReForge-DA, respectively, measured in percentage points and computed before rounding.

Tasks	Anchor	Original	Rescale	ReForge-DF	ReForge-DA	Δ_{DF}	Δ_{DA}
8	TSV	85.9	$86.6_{\pm 0.19}$	$87.8_{\pm 0.04}$	$88.9_{\pm 0.05}$	1.2	2.3
8	ISO-CTS	86.4	$87.6_{\pm 0.04}$	$89.0_{\pm 0.03}$	$90.2_{\pm 0.03}$	1.4	2.6
14	TSV	79.9	$81.8_{\pm 0.10}$	$82.8_{\pm 0.03}$	$84.3_{\pm 0.04}$	1.0	2.5
14	ISO-CTS	81.5	$83.9_{\pm 0.10}$	$84.3_{\pm 0.05}$	$85.5_{\pm 0.06}$	0.4	1.6
20	TSV	76.9	$78.6_{\pm 0.21}$	$79.7_{\pm 0.05}$	$82.0_{\pm 0.04}$	1.1	3.4
20	ISO-CTS	77.6	$80.5_{\pm 0.06}$	$81.5_{\pm 0.04}$	$82.8_{\pm 0.05}$	1.1	2.4

C Vision Results with Standard Deviations

Table 8: Model merging on vision tasks with ViT architectures. Values report mean accuracy over five seeds; subscripts indicate sample standard deviations, omitted from Table 1 for space. (Indiv.: Individual)

Model	Tasks	Setting	Indiv.	Pretrained	TA	TIES	RegMean	TSV	WUDI	ISO-CTS
ViT-B/32	8	anchor	92.8	47.7	70.4	75.7	82.3	85.9	**87.0**	86.4
		ReForge-DA	-	$84.8_{\pm 0.02}$	$85.0_{\pm 0.05}$	$85.7_{\pm 0.03}$	$85.2_{\pm 0.05}$	$88.9_{\pm 0.05}$	$89.4_{\pm 0.02}$	**90.2**$_{\pm 0.03}$
		ReForge-DF	-	$87.9_{\pm 0.06}$	$87.0_{\pm 0.05}$	$87.0_{\pm 0.06}$	$87.9_{\pm 0.09}$	$87.8_{\pm 0.04}$	$87.6_{\pm 0.04}$	**89.0**$_{\pm 0.03}$
	14	anchor	90.9	56.9	65.2	68.2	76.6	79.9	80.5	**81.5**
		ReForge-DA	-	$78.9_{\pm 0.03}$	$79.2_{\pm 0.04}$	$79.7_{\pm 0.04}$	$79.1_{\pm 0.02}$	$84.3_{\pm 0.04}$	$84.6_{\pm 0.04}$	**85.5**$_{\pm 0.06}$
		ReForge-DF	-	$82.7_{\pm 0.06}$	$82.4_{\pm 0.03}$	$82.3_{\pm 0.10}$	$82.9_{\pm 0.09}$	$82.8_{\pm 0.03}$	$81.7_{\pm 0.06}$	**84.3**$_{\pm 0.05}$
	20	anchor	91.3	55.7	60.4	64.0	72.2	76.9	76.1	**77.6**
		ReForge-DA	-	$75.6_{\pm 0.05}$	$75.5_{\pm 0.04}$	$76.3_{\pm 0.03}$	$75.9_{\pm 0.06}$	$82.0_{\pm 0.04}$	$82.2_{\pm 0.06}$	**82.8**$_{\pm 0.05}$
		ReForge-DF	-	$80.0_{\pm 0.06}$	$78.4_{\pm 0.06}$	$79.2_{\pm 0.05}$	$79.7_{\pm 0.03}$	$79.7_{\pm 0.05}$	$78.0_{\pm 0.02}$	**81.5**$_{\pm 0.04}$
ViT-L/14	8	anchor	95.8	65.1	84.8	87.0	90.0	93.0	94.0	**94.8**
		ReForge-DA	-	$92.3_{\pm 0.01}$	$92.8_{\pm 0.05}$	$93.1_{\pm 0.03}$	$92.7_{\pm 0.03}$	$94.4_{\pm 0.03}$	$94.7_{\pm 0.02}$	**95.1**$_{\pm 0.03}$
		ReForge-DF	-	$94.4_{\pm 0.02}$	$94.2_{\pm 0.01}$	$94.2_{\pm 0.03}$	$94.0_{\pm 0.02}$	$94.4_{\pm 0.01}$	$94.3_{\pm 0.04}$	**95.0**$_{\pm 0.02}$
	14	anchor	94.3	68.5	79.4	80.2	85.5	89.1	90.5	**91.0**
		ReForge-DA	-	$88.0_{\pm 0.01}$	$88.3_{\pm 0.03}$	$88.3_{\pm 0.03}$	$88.2_{\pm 0.02}$	$91.6_{\pm 0.01}$	$91.7_{\pm 0.03}$	**92.2**$_{\pm 0.07}$
		ReForge-DF	-	$90.9_{\pm 0.08}$	$90.9_{\pm 0.04}$	$90.8_{\pm 0.02}$	$90.6_{\pm 0.05}$	$91.3_{\pm 0.05}$	$91.0_{\pm 0.06}$	**92.1**$_{\pm 0.06}$
	20	anchor	94.7	65.4	74.0	76.7	82.8	87.7	88.4	**90.1**
		ReForge-DA	-	$86.0_{\pm 0.02}$	$86.2_{\pm 0.03}$	$86.4_{\pm 0.01}$	$86.2_{\pm 0.01}$	$90.9_{\pm 0.03}$	$90.8_{\pm 0.02}$	**91.5**$_{\pm 0.01}$
		ReForge-DF	-	$89.8_{\pm 0.02}$	$89.6_{\pm 0.01}$	$89.5_{\pm 0.01}$	$89.4_{\pm 0.04}$	$90.3_{\pm 0.02}$	$89.7_{\pm 0.02}$	**91.1**$_{\pm 0.11}$

Table 8 provides the comprehensive results corresponding to Table 1, including sample standard deviations across five seeds. Most sample standard deviations lie between 0.01 and 0.06 percentage points. The results vary little across repeated searches with fixed checkpoints and data.

D Sampling-based ReForge for Uncertainty Calibration

Sampling-based ReForge. ReForge uses a MAP point estimate of $\mathbf{U}$ for computational efficiency. The Bayesian linear regression formulation in Section 4.1 also yields a Gaussian posterior with independent rows, each having mean given by the corresponding row of $\mathbf{U}_{\mathrm{MAP}}$ and covariance $\mathbf{\Sigma}_{\mathrm{row}}$:

$$\mathbf{U}_{\mathrm{MAP}} = \left(\mathbf{Y}\mathbf{X}^\top + \lambda \mathbf{U}^{(0)}\right)\left(\mathbf{X}\mathbf{X}^\top + \lambda \mathbf{I}\right)^{-1}, \qquad \mathbf{\Sigma}_{\mathrm{row}} = \beta^{-1}\left(\mathbf{X}\mathbf{X}^\top + \lambda \mathbf{I}\right)^{-1}, \tag{23}$$

Here $\mathbf{I} \in \mathbb{R}^{d_{\mathrm{in}} \times d_{\mathrm{in}}}$ is the identity matrix, $\mathbf{\Sigma}_{\mathrm{row}}$ is the covariance matrix for each row of $\mathbf{U}$, and $\beta > 0$ is the Gaussian noise precision. The matrices $\mathbf{X}$ and $\mathbf{Y}$ follow the task-wise normalization in Section 4.1. Once BO selects the regularization strengths and assembly scales, we keep them fixed and sample multiple merged models for uncertainty calibration (Guo et al., 2017). Varying β with $\alpha_m = \beta \lambda_m^\star$ changes the posterior covariance without changing the MAP estimate.

Specifically, let $\{\theta^{(r)}_{\text{merged}}\}_{r=1}^{S}$ be S merged models obtained by drawing module task vectors from the Gaussian posteriors and assembling them as $\mathbf{W}_m^{(r)} = \mathbf{W}_{\text{pre},m} + s_m^\star \mathbf{U}_m^{(r)}$. Let $\mathbf{z}(\mathbf{x};\theta)$ denote the logit vector of model θ. The ensemble prediction is the average probability across the S models:

$$p(\cdot \mid \mathbf{x}) = \frac{1}{S}\sum_{r=1}^{S} \text{softmax}\Big(\mathbf{z}(\mathbf{x};\theta^{(r)}_{\text{merged}})\Big)\,. \tag{24}$$

Expected Calibration Error. We measure calibration using ECE with ten equal-width confidence bins:

$$\text{ECE} = \sum_{b=1}^{\kappa} \frac{|B_b|}{n} \left|\text{acc}\left(B_b\right) - \text{conf}\left(B_b\right)\right|, \tag{25}$$

Here $\kappa = 10$, B_b contains examples whose predicted confidence falls in bin b, $|B_b|$ is the number of examples in that bin, and n is the total number of examples. The quantities $\text{acc}(B_b)$ and $\text{conf}(B_b)$ denote the bin's empirical accuracy and mean predicted confidence, respectively. Lower ECE indicates closer agreement between binned confidence and empirical accuracy.

Experimental Setup. We evaluate sampling-based ReForge on the ViT-B/32 8-task and 20-task benchmarks in the data-assisted setting, using ISO-CTS as the anchor. Based on preliminary experiments, we sample only the final MLP output matrix and keep the remaining modules at their MAP estimates – similar observations have also been made by (Kristiadi et al., 2020). As a baseline, we also perturb the MAP solution of the last MLP output matrix with IID Gaussian noise (controlled by σ_{iid}). Both methods generate $S = 10$ sampled models and use prediction averaging in Eq. (24) for final classification. Varying β and σ_{iid} changes the trade-off between accuracy and ECE. Figure 4 reports the resulting test-set Pareto frontiers; the displayed configurations are not selected using validation data.

Results and Analysis. Figure 4 shows that sampling-based ReForge offers a better trade-off between accuracy and ECE than perturbing the MAP model with IID Gaussian noise. The separation between the two Pareto frontiers is more pronounced on the 8-task benchmark than on the 20-task benchmark. These results suggest that the posterior covariance structure is useful for improving uncertainty calibration while retaining most of the MAP model's accuracy.

E Computational Costs and Runtime

E.1 Complexity Analysis

We briefly analyze the costs of the closed-form estimators (Eqs. 8, 14) and the BO search (11). Let T be the number of tasks, M be the number of merged 2D modules, and $\mathbf{U}_m \in \mathbb{R}^{d_{\text{out},m}\times d_{\text{in},m}}$. In the data-assisted setting, with N calibration samples per task, $\mathbf{X}_m \in \mathbb{R}^{d_{\text{in},m}\times N_{\text{act}}}$ and $\mathbf{Y}_m \in \mathbb{R}^{d_{\text{out},m}\times N_{\text{act}}}$, where $N_{\text{act}} = NT$. The matrices $\mathbf{X}_m$ and $\mathbf{Y}_m$ include the per-task trace rescaling defined in Section 4.1. We omit the one-time forward cost for collecting activations.

Closed-form estimators. The data-assisted estimator in Eq. (8) first caches $\mathbf{X}_m\mathbf{X}_m^\top$ and $\mathbf{Y}_m\mathbf{X}_m^\top$, which costs $\mathcal{O}\big(N_{\text{act}}d_{\text{in},m}^2 + N_{\text{act}}d_{\text{out},m}d_{\text{in},m}\big)$. Then, given a module-wise λ_m, solving

$$\left(\mathbf{X}_m\mathbf{X}_m^\top + \lambda_m\mathbf{I}\right)\mathbf{U}_m^\top = \left(\mathbf{Y}_m\mathbf{X}_m^\top + \lambda_m\mathbf{U}_m^{(0)}\right)^\top,$$

costs $\mathcal{O}\big(d_{\text{in},m}^3 + d_{\text{out},m}d_{\text{in},m}^2\big)$ by using a dense Cholesky solver (Golub & Van Loan, 2013). Similarly, the data-free estimator in Eq. (14) replaces activation statistics with the sum of per-task trace-normalized task-vector Grams, whose construction costs $\mathcal{O}(Td_{\text{out},m}d_{\text{in},m}^2)$ per module. The resulting linear system has the same per-module solve cost as in the data-assisted setting. Thus, the cost of the closed-form estimators is insignificant on modern high-performance GPUs.

BO search overhead. Algorithm 1 performs K BO trials. The exact $\mathcal{GP}$ update costs $\mathcal{O}(K^3)$ due to the Cholesky factorization of the $\mathcal{GP}$ kernel matrix (Rasmussen & Williams, 2006; Frazier, 2018). In practice, this overhead is very small compared with repeated validation-set evaluations. Hence, the practical runtime of BO is dominated by evaluating candidate merged models on the validation set.

Table 9: **Incremental refinement costs of ReForge given an existing anchor.** The ViT timings are measured on a single NVIDIA RTX 6000 Ada 48GB GPU, while the Llama timings are reported per worker under an 8-way parallel BO execution. *Gram* denotes the one-time Gram-matrix construction cost in the data-assisted setting. *Closed-form* and $\mathcal{GP}$ denote the cumulative costs of closed-form solution and $\mathcal{GP}$ surrogate updates, respectively. *Search* includes Gram construction, closed-form solution, $\mathcal{GP}$ updates, model assembly, checkpoint loading/book-keeping, and rounding. *Val. Cost* reports the time spent on validation-set evaluations.

Model	Tasks	Setting	Gram	Closed-form	$\mathcal{GP}$	Search	Val. Cost	Total
ViT-B/32	8	ReForge-DA	0.4 min	2.5 min	0.4 min	3.3 min	17.3 min	20.6 min
ViT-B/32	8	ReForge-DF	–	2.2 min	0.4 min	2.9 min	17.3 min	20.2 min
ViT-B/32	14	ReForge-DA	0.6 min	4.5 min	0.4 min	6.2 min	22.3 min	28.5 min
ViT-B/32	14	ReForge-DF	–	5.2 min	0.4 min	5.6 min	22.3 min	27.9 min
ViT-B/32	20	ReForge-DA	0.8 min	4.6 min	0.4 min	8.0 min	41.6 min	49.6 min
ViT-B/32	20	ReForge-DF	–	6.9 min	0.4 min	7.3 min	41.6 min	48.8 min
ViT-L/14	8	ReForge-DA	2.8 min	8.2 min	0.4 min	17.1 min	168.4 min	185.5 min
ViT-L/14	8	ReForge-DF	–	13.9 min	0.4 min	14.3 min	168.4 min	182.7 min
ViT-L/14	14	ReForge-DA	5.8 min	12.1 min	0.4 min	18.6 min	292.1 min	310.7 min
ViT-L/14	14	ReForge-DF	–	11.3 min	0.4 min	12.8 min	292.1 min	304.9 min
ViT-L/14	20	ReForge-DA	9.8 min	17.0 min	0.4 min	31.7 min	539.0 min	570.7 min
ViT-L/14	20	ReForge-DF	–	21.5 min	0.4 min	21.9 min	539.0 min	560.9 min
Llama-3.2-3B	5	ReForge-DA	1.0 min	12.9 min	0.1 min	14.5 min	126.9 min	141.4 min
Llama-3.2-3B	5	ReForge-DF	–	13.4 min	0.1 min	13.5 min	126.9 min	140.3 min
Llama-3.1-8B	5	ReForge-DA	2.8 min	38.4 min	0.1 min	43.4 min	190.7 min	234.1 min
Llama-3.1-8B	5	ReForge-DF	–	39.9 min	0.1 min	39.9 min	190.7 min	230.6 min

Table 10: Baseline search costs and incremental ReForge refinement costs in the data-free setting. For ViT-B/32, ReForge uses ISO-CTS as anchor; for Llama-3.2-3B, ReForge uses TSV as anchor. Baselines use their full grid-search budgets. The ViT runtimes are measured on a single NVIDIA RTX 6000 Ada 48GB GPU, and the Llama runtimes report 8-GPU parallel wall-clock time. Scores denote test accuracies for ViT-B/32 and aggregate benchmark scores for Llama-3.2-3B.

Model	Tasks	Method	Search Dim.	#Trials	Runtime	Score
ViT-B/32	20	TSV	1	30	8.4 min	76.9
ViT-B/32	20	WUDI-Merging	2	24	116.1 min	76.1
ViT-B/32	20	ISO-CTS	3	225	91.2 min	77.6
ViT-B/32	20	ReForge ($K = 20$)	15	20	5.60 min	80.51
ViT-B/32	20	ReForge ($K = 60$)	15	60	15.55 min	81.5
Llama-3.2-3B	5	TSV	1	30	73.4 min	0.471
Llama-3.2-3B	5	WUDI-Merging	2	24	418.8 min	0.461
Llama-3.2-3B	5	ISO-CTS	3	225	318.0 min	0.441
Llama-3.2-3B	5	ReForge ($K = 20$)	5	20	28.6 min	0.465
Llama-3.2-3B	5	ReForge ($K = 60$)	5	60	84.8 min	0.473

E.2 Runtime Breakdown

Table 9 reports ReForge refinement costs after an anchor is available. The vision experiments with $K = 200$ BO trials are measured on a single NVIDIA RTX 6000 Ada 48GB GPU, with validation tensors cached locally to reduce I/O overhead. The language experiments with $K = 100$ BO trials report per-worker timings under an 8-GPU parallel setup. Across both vision and language benchmarks, the algorithmic overhead of ReForge remains small compared with validation-set forward passes, indicating that the main runtime cost comes from model evaluation rather than from the ReForge update itself.

E.3 Runtime–Performance Trade-offs

Figure 3(b) and Table 10 characterize the additional cost of refining an existing anchor in the data-free setting. On 20-task ViT-B/32, ReForge refines the ISO-CTS anchor from 77.6% to 80.51% with $K = 20$ trials and 5.60 minutes of additional computation; $K = 60$ reaches 81.5% in 15.55 additional minutes.

For Llama-3.2-3B, $K = 20$ costs 28.6 additional minutes and reaches 0.465, below the TSV anchor's 0.471; $K = 60$ costs 84.8 additional minutes and reaches 0.473. The language timings in this table are 8-GPU parallel wall-clock times, whereas Table 9 reports per-worker timings; they should not be directly equated.

F Experimental Protocol and Assets

F.1 Vision Benchmarks

For the *vision* experiments, to ensure a rigorous and fair comparison, we strictly adhere to the benchmarks established by TSV (Gargiulo et al., 2025) and ISO-CTS (Marczak et al., 2025), adopting their exact datasets and identical training, validation, and test splits. Specifically, our evaluation spans three benchmarks with increasing task diversity. The initial 8-task benchmark comprises Stanford Cars (Krause et al., 2013), DTD (Cimpoi et al., 2014), EuroSAT (Helber et al., 2019), GTSRB (Stallkamp et al., 2011), MNIST (LeCun et al., 1998), RESISC45 (Cheng et al., 2017), SUN397 (Xiao et al., 2016), and SVHN (Netzer et al., 2011). The 14-task benchmark extends this setting by adding CIFAR-100 (Krizhevsky & Hinton, 2009), STL-10 (Coates et al., 2011), Flowers102 (Nilsback & Zisserman, 2008), Oxford-IIIT Pets (Parkhi et al., 2012), PCAM (Veeling et al., 2018), and FER2013 (Goodfellow et al., 2013). Finally, the largest benchmark encompasses 20 tasks by further incorporating EMNIST (Cohen et al., 2017), CIFAR-10 (Krizhevsky & Hinton, 2009), Food101 (Bossard et al., 2014), Fashion-MNIST (Xiao et al., 2017), Rendered SST-2 (Socher et al., 2013), and KMNIST (Clanuwat et al., 2018). Table 1 in the main paper reports the performance of the merged models averaged across the 8, 14, and 20 tasks, while the "mean $\pm$ std" results and the per-task breakdown radar charts are provided in Appendices C and G, respectively.

F.2 Language Benchmarks

Table 11: Summary of training and evaluation datasets for natural language processing experiments. Official standard splits are adopted for all benchmarks, with the exception of the GSM8k validation set (500 samples from the training set) and datasets marked with * (custom 1/3-2/3 splits).

Category	Training	Validation	Test	Metric
Instruction	TULU-3 Persona (Lambert et al., 2024)	IFEval* (Zhou et al., 2023)	IFEval* (Zhou et al., 2023)	Prompt Acc.
Mathematics	DART (Tong et al., 2024), Numina Math (Li et al., 2024)	GSM8k (Cobbe et al., 2021)	GSM8k (Cobbe et al., 2021)	EM (8-shot)
Multilingual	Aya (Singh et al., 2024)	M_MMLU (Lai et al., 2023) (fr, es, de, ru)	M_MMLU (Lai et al., 2023) (fr, es, de, ru)	Accuracy
Coding	Magicoder (Wei et al., 2023)	MBPP (Austin et al., 2021)	HumanEval+ (Liu et al., 2023), MBPP+ (Liu et al., 2023)	Pass@1
Safety	WildGuard (Han et al., 2024), Wild Jailbreak (Jiang et al., 2024)	Harm Bench* (Mazeika et al., 2024), XSTest* (Röttger et al., 2023)	Harm Bench* (Mazeika et al., 2024), XSTest* (Röttger et al., 2023)	$1 - \mathrm{ASR}$ / Acc.

For the NLP experiments, Table 11 summarizes the training, validation, and test splits, together with the corresponding evaluation metrics. For the coding category, we use MBPP for validation after

removing 13 examples whose normalized problem statements exactly overlap with examples in the MBPP+ test set. The resulting validation and test sets are disjoint.

Language validation and scoring. For Llama, benchmark scores are expressed on a $[0, 1]$ scale and aggregated into five category scores. Instruction following uses IFEval prompt accuracy, and mathematics uses GSM8k exact match. Multilingual performance is the arithmetic mean of M_MMLU accuracies for French, Spanish, German, and Russian. Coding uses MBPP pass@1 for validation and the arithmetic mean of HumanEval+ and MBPP+ pass@1 for testing. Safety uses the harmonic mean

$$S_{\text{Safety}} = H(1 - \text{ASR}_{\text{HarmBench}}, \text{Acc}_{\text{XSTest}}), \qquad H(a, b) = \frac{2ab}{a + b},$$

with $H(0, 0) = 0$. The final score is the equally weighted arithmetic mean of the five category scores, $S_{\text{overall}} = \frac{1}{5} \sum_{c=1}^{5} S_c$. Validation scores guide BO, while the corresponding held-out test scores are used for final reporting in Table 2.

F.3 Licenses and Asset Usage

We use existing public assets, including pretrained backbones, task-specific checkpoints, benchmark datasets, evaluation suites, and baseline implementations. These assets are described above, and their original creators are credited through the corresponding references.

For vision experiments, we follow public model-merging benchmark settings based on ViT backbones and task-specific expert checkpoints. For language experiments, we use Llama-3.2-3B and Llama-3.1-8B backbones under the corresponding Meta Llama Community License Agreements and acceptable-use policy. We do not redistribute third-party checkpoints or model weights except where permitted by their original licenses.

All datasets and evaluation suites are used only for research, calibration, validation, and evaluation. Their source URLs, versions when applicable, and license or source-term information are provided in the supplementary asset manifest. Safety benchmarks may contain harmful or adversarial prompts and are used only for safety research and evaluation.

Baseline methods and third-party libraries are used according to their original papers, public repositories, and licenses. We retain required copyright notices, citations, and license files where applicable. We do not introduce new scraped datasets or human-subject data. All copyrights remain with the original asset owners, and all external assets are used in compliance with their respective licenses and terms of use.

G Radar Charts: Per-Task Breakdowns

Tables 1 and 2 report the performance of merged models averaged across multiple tasks. However, such summaries may hide task-specific trade-offs. To address this issue, Figures 5–8 provide vision per-task radar charts for all evaluated ViT settings. Similarly, Figures 9–10 provide language per-task radar charts for all evaluated Llama settings.

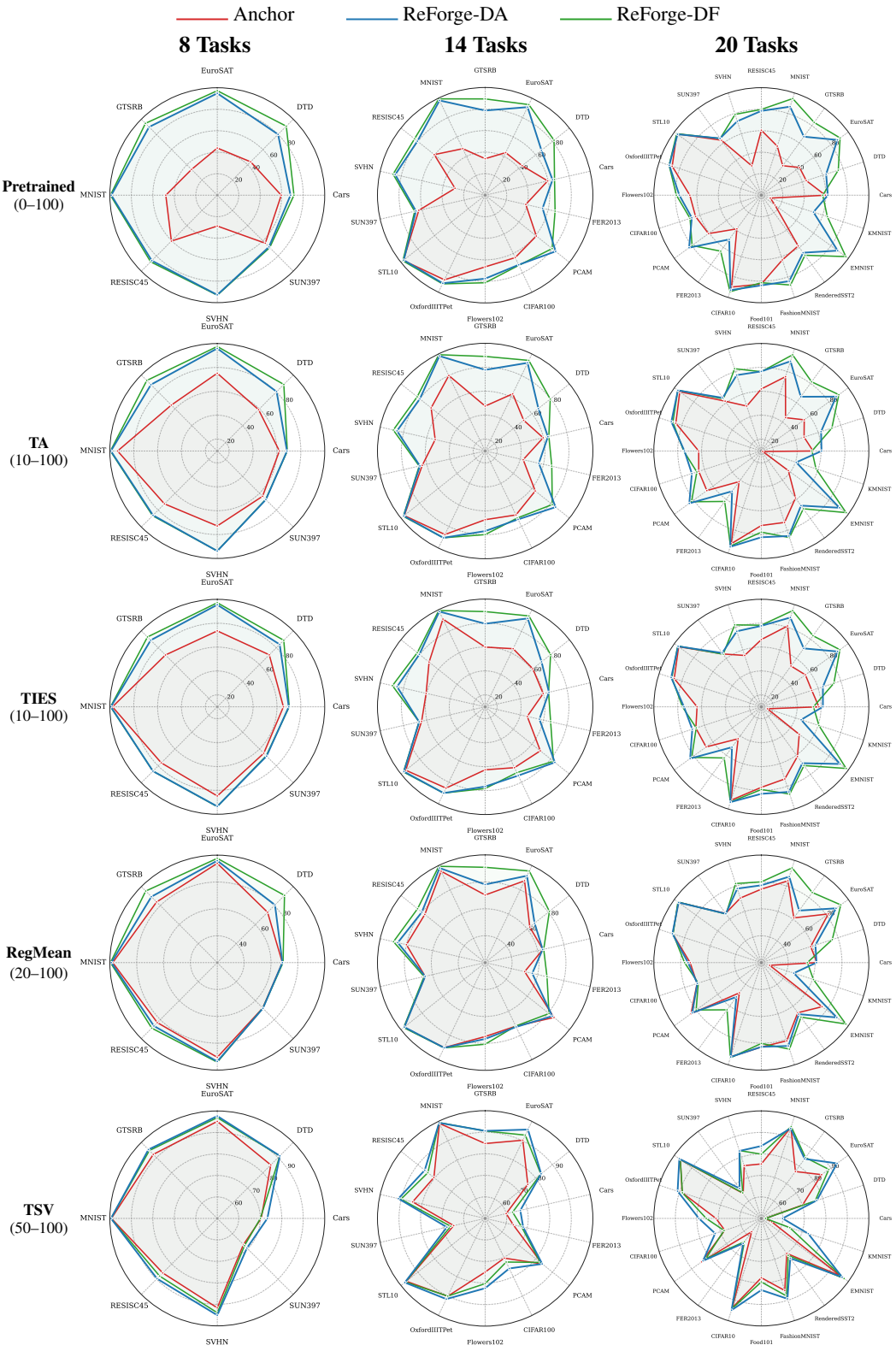

Figure 5: **Radar charts: ViT-B/32 per-task breakdowns** (corresponding to Table 1). The radial axis limits (min–max) are annotated on the left.

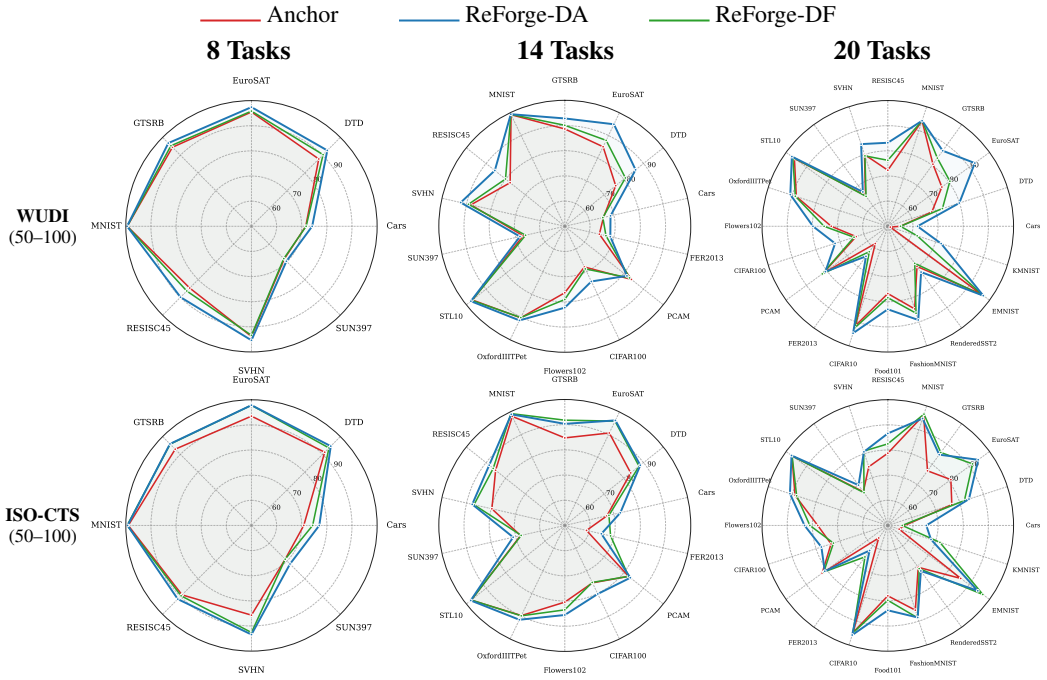

Figure 6: **Radar charts: ViT-B/32 per-task breakdowns** (corresponding to Table 1). Continued from Figure 5 on the remaining anchor models: WUDI and ISO-CTS.

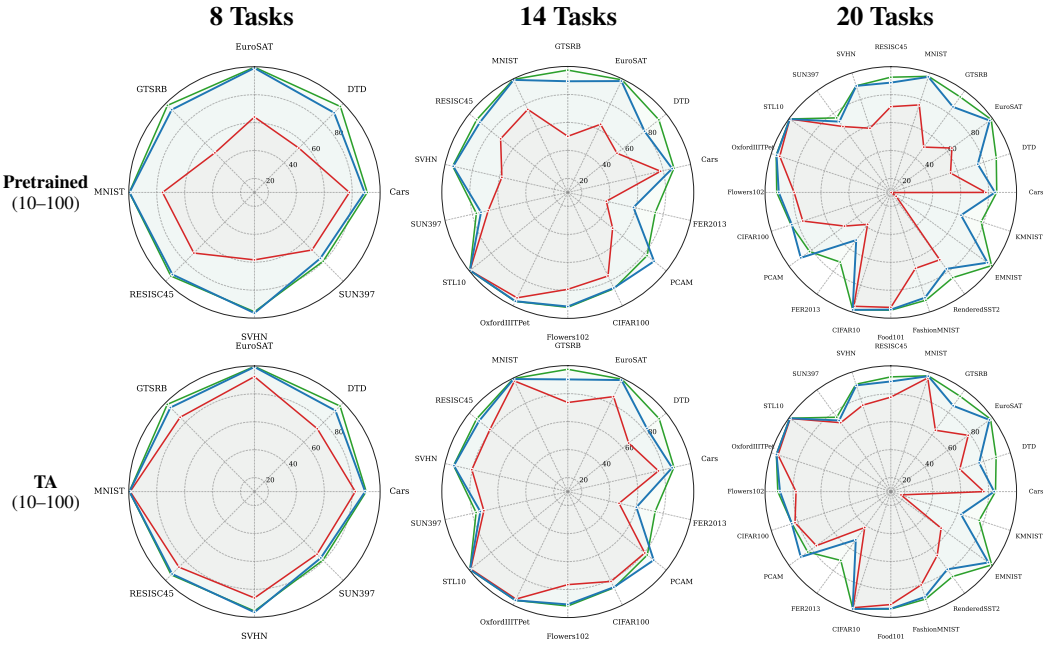

Figure 7: **Radar charts: ViT-L/14 per-task breakdowns** (corresponding to Table 1). Layout, legend, and row-specific axis scaling follow Figure 5.

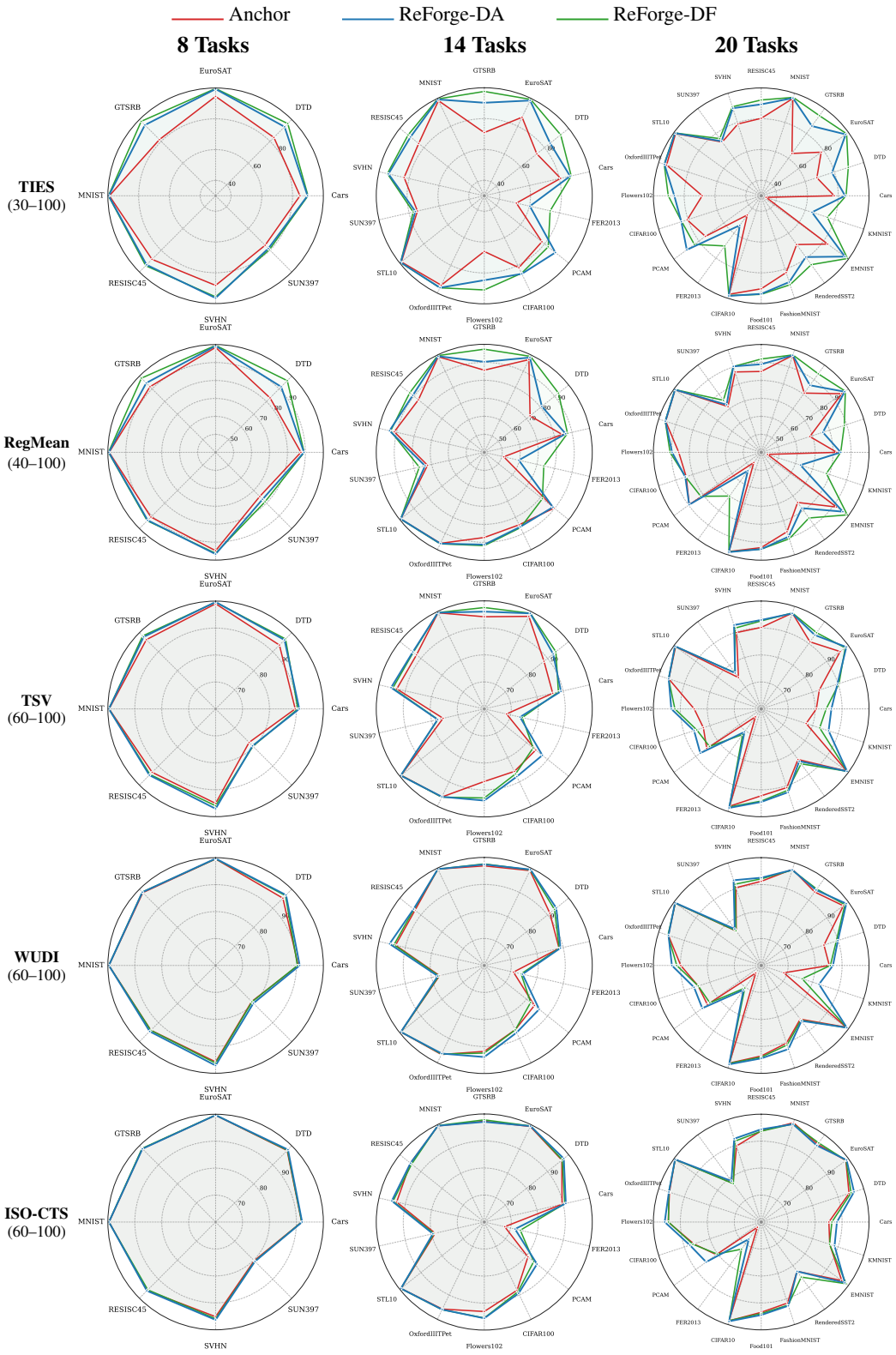

Figure 8: **Radar charts: ViT-L/14 per-task breakdowns** (corresponding to Table 1). Continued from Figure 7 on the remaining anchor models.

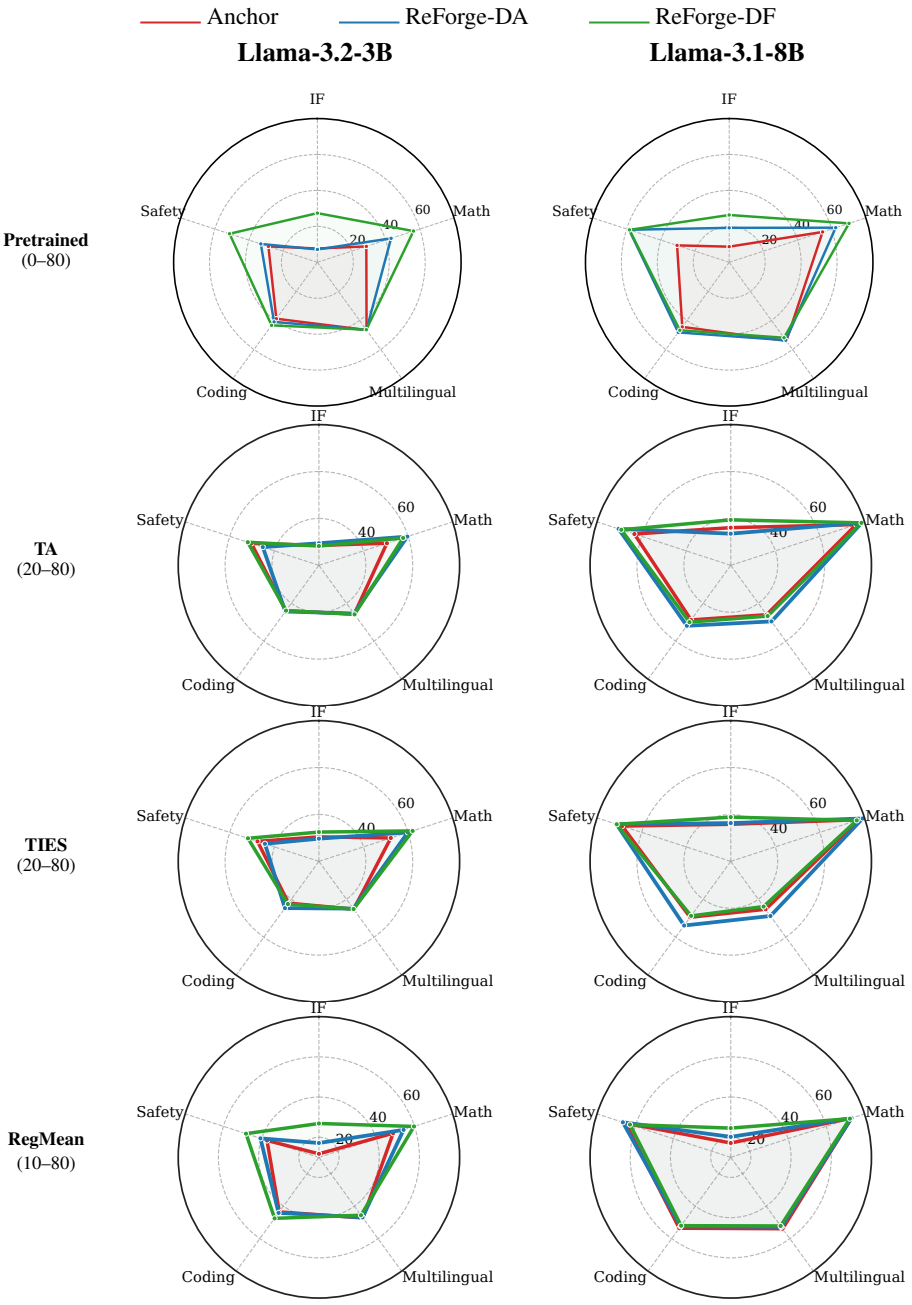

Figure 9: **Radar charts: Llama per-task breakdowns** (corresponding to Table 2). The radial axis limits (min-max) are annotated on the left. "IF" denotes Instruction Following.

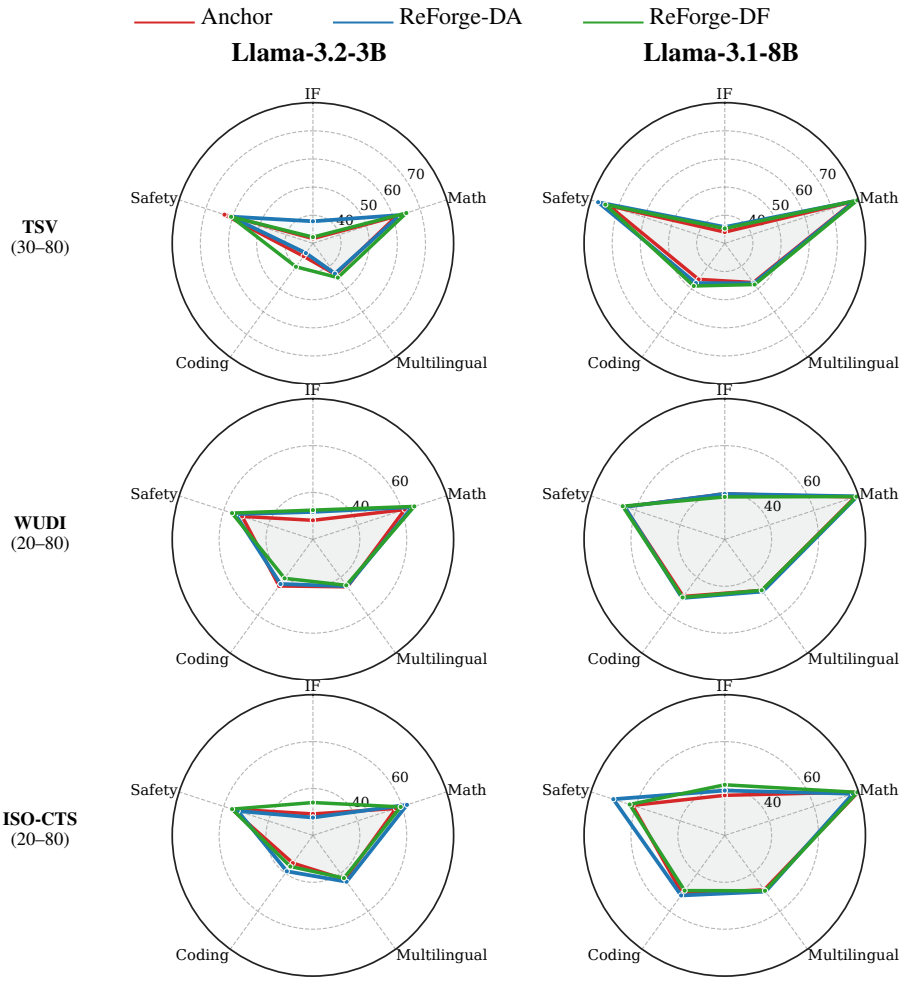

Figure 10: **Radar charts: Llama per-task breakdowns** (corresponding to Table 2). Continued from Figure 9 on the remaining anchor models. "IF" denotes Instruction Following.