# COMET: CONTINUAL MODEL MERGING VIA ADAPTATION TO MISMATCH IN DISTRIBUTIONAL TAILS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Continual model merging requires preserving past solution expertise (stability) while retaining the ability to adapt (plasticity), without access to past data. Recent approaches cast this problem as recursive posterior inference, representing accumulated expertise as a model prior and incorporating new knowledge through the target task likelihood. The resulting posterior often assumes a Gaussian structure, with task-specific curvature statistics accumulated in its precision matrix. Our analysis reveals a previously overlooked geometric interpretation of this formulation. Incorporating knowledge from a new model distribution can be viewed as updating the mode of the existing model distribution along a set of orthogonal latent directions in the inner-product space of its precision matrix. Each direction is associated with a unit scale of update that acts as a base speed controlling how fast new knowledge should be incorporated along that direction. This reveals a novel geometric characterization of stability and plasticity that is largely independent of the model architecture. Importantly, the appropriate unit scale along each direction should adapt to the relative tail behaviors of the existing and incoming model distributions as their disagreement determines how strongly newly acquired knowledge should alter the existing solution. Gaussian structures, however, ignore such tail behavior and consequently impose fixed unit scales of update that are empirically suboptimal. Replacing the Gaussian structure with an energy based distributional structure whose tail decay rate is adaptive restores this missing dependence. Extensive experiments with billion-parameter foundation models show that our approach consistently outperforms state-of-the-art methods.

## 1 INTRODUCTION

Foundation models and their fine-tuned versions for various downstream tasks are becoming increasingly available thanks to the rising open-source ecosystem (Wolf et al., 2019). Due to an abundance of these fine-tuned models, it is desirable to combine their specialized abilities to create multi-task models. Although a multi-task model can be trained on multiple datasets, doing so on demand is prohibitively expensive and inflexible as each new multi-task demand requires accessing the original datasets again, which is often impractical in continual adaptation scenarios (Shi et al., 2025). A better alternative is model merging (Wortsman et al., 2022; Ilharco et al., 2022; Matena & Raffel, 2022) which is a data-free paradigm that aims to integrate existing model artifacts in their parameter space without requiring access to any task-specific datasets. Recent advances in model merging have shown that the performance of the merged model is often comparable to its re-trained multi-task counterpart on various languages (Yang et al., 2024; Tam et al., 2024), vision (Chen et al., 2025; Wang et al., 2024) and multimodal (Wei et al., 2026; Sung et al., 2023) benchmarks.

A key challenge in model merging is task interference where aggregating model parameters can cause knowledge from different tasks to conflict, degrading task-specific expertise and reducing overall performance. Existing approaches have proposed various mitigation techniques such as weight-space interpolation methods (Wortsman et al., 2022; Ilharco et al., 2022; Matena & Raffel, 2022), sub-space geometric strategies (Tang et al., 2023; Stoica et al., 2024; Wei et al., 2025) and probabilistic methods (Daheim et al., 2023; Bui et al., 2026). However, most existing approaches assume that all task-specific models are simultaneously available for merging. This assumption however might not hold in multiple practical scenarios as models and tasks arrive sequentially rather than simultaneously. For example, on device AI systems living on resource limited edge devices



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.

often have to incorporate newly available task specific models on the fly while preserving competence on previously learned tasks. Repeatedly storing all task specific models is also impractical, as memory overhead grows with the number of tasks and specialists. This motivates the development of continual model merging, which maintains a single compact model that is sequentially updated as new task checkpoints arrive (Yang et al., 2026; Tang et al., 2025).

**Research Gap.** Most existing model merging methods such as WEIGHT AVERAGING (Wortsman et al., 2022), TASK ARITHMETIC (Ilharco et al., 2022) or continual Fisher-based averaging (Marouf et al., 2024; Li et al., 2025) can be re-interpreted as solutions to a Bayesian inference task over a product-of-experts (POE) representation on the model space (Bui et al., 2026). Under this interpretation, these approaches can be reconfigured as recursive posterior inference which is naturally suitable for continual merging. However, these approaches implicitly impose a Gaussian structure on the parameter space during merging (Bui et al., 2026) which, as exposed in our theoretical analysis (cite), fixes the unit scales along each update direction which is supposed to be adaptive to mismatch in the tail behaviors of the existing and incoming model distributions. Fixing them impairs the adaptability to new tail behaviors of incoming model distributions and consequently results in suboptimal performance as demonstrated in our empirical studies (cite).

**Solution Approach.** To bridge this gap, we develop COMET, a continual model merging framework that replaces the fixed Gaussian merging structure with a flexible energy based distributional formulation whose tail decay can adapt across model distributions. This design allows disagreement between the existing and incoming models to modulate the unit scale of update along each latent direction, providing a principled mechanism for balancing plasticity and stability during sequential merging. Our analysis also reveals a natural remedy to this limitation. Replacing the fixed Gaussian structure with a more general energy based distributional formulation whose tail decay can adapt across model distributions. Building on this insight, we develop COMET, which allows disagreement between the existing and incoming models to modulate the unit scale of update along each latent direction, thereby providing a principled mechanism for balancing plasticity and stability during continual merging. Under the resulting merging structure, the consolidated model is characterized by the MAP solution of the merged model distribution. To compute this solution efficiently, we exploit the Kronecker structure of the parameter space curvature geometry to derive a scalable fixed point solver, avoiding the expensive optimization required by existing geometric approaches.

**Contributions.** Our solution approach is conceived based on the following technical contributions:

**1.** We develop a geometric analysis of continual model merging that decomposes knowledge incorporation into updates along orthogonal latent directions in the inner product space induced by the precision matrix of the existing model distribution. This reveals a novel characterization of plasticity and stability through these latent directions and their unit scales of update, which depend on differences in the tail behaviors of the existing and incoming model distributions (Section 2).

**2.** We develop an energy-based distributional formulation guided by the above analysis, allowing distributional tail mismatch to adaptively modulate the direction-wise unit scales of update during merging. We also derive an iteratively reweighted fixed point (IRFP) algorithm that exploits the Kronecker structure of task specific curvature to efficiently compute the MAP solution of the merged model distribution with favorable scaling to large foundation models (Section 3).

**3.** We evaluate our framework, termed ***Co**ntinual **Me**rging via **T**ail Adaptation* (COMET) across diverse vision and language benchmark datasets and model architectures, including large scale billion parameter FMs. Our results consistently improve over state-of-the-art (SOTA) baselines, providing empirical support for our theoretical analysis and demonstrating its practical scalability (Section 4).

## 2 GEOMETRIC INTERPRETATION OF CONTINUAL MODEL MERGING

**Gaussian View.** Most model merging methods can be re-interpreted as computing a product of Gaussian experts where the existing ($p_o$) and new model distributions ($p_n$) are represented as:

$$p_o(\boldsymbol{\theta}) \propto \exp\left(-\frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{m}_o)^\top \boldsymbol{\Lambda}_o(\boldsymbol{\theta}-\boldsymbol{m}_o)\right) ; p_n(\boldsymbol{\theta}) \propto \exp\left(-\frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{m}_n)^\top \boldsymbol{\Lambda}_n(\boldsymbol{\theta}-\boldsymbol{m}_n)\right) \quad (1)$$

where $\boldsymbol{\Lambda}_o$ and $\boldsymbol{\Lambda}_n$ are the precision matrix (i.e., inverse covariance) of the existing and new model distributions. Likewise, $\boldsymbol{m}_o$ and $\boldsymbol{m}_n$ are the MAP solutions of the existing and new model distri-

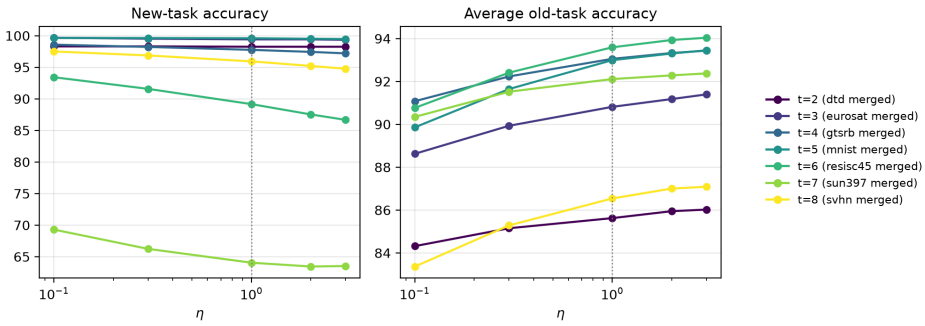


Figure 1: Accuracy curves with respect to different values of $\eta$ on the 8-task ViT-B/16 stream. $\eta = 1$ corresponds to the Gaussian case. Each curve correspond to a merge step $t$. *Left:* new task acuuracy after merge. *Right:* mean accuracy on the previosuly merged tasks.

butions. The merge model $p_m(\boldsymbol{\theta})$ is then given in closed-form via a standard result of Gaussian merging $p_m(\boldsymbol{\theta}) \propto \exp(-(1/2)(\boldsymbol{\theta} - \boldsymbol{m}_m)^\top \boldsymbol{\Lambda}_m(\boldsymbol{\theta} - \boldsymbol{m}_m))$ where:

$$\boldsymbol{\Lambda}_m \;=\; \boldsymbol{\Lambda}_o \,+\, \boldsymbol{\Lambda}_n \quad \text{and} \quad \boldsymbol{m}_m \;=\; (\boldsymbol{\Lambda}_o \,+\, \boldsymbol{\Lambda}_n)^{-1}\,(\boldsymbol{\Lambda}_o \boldsymbol{m}_o \,+\, \boldsymbol{\Lambda}_n \boldsymbol{m}_n)\;. \tag{2}$$

As such, $\boldsymbol{m}_m$ is the MAP solution of the merged model distribution. For large foundation models (FMs), $\boldsymbol{\theta}$ is often used as a low-complexity adaptation unit of a frozen FM and merging thus happens instead of the space of adapters (Stoica et al., 2024; Bui et al., 2026; Wei et al., 2025).

**Generalized View.** The above is a special case of a more generic class of energy-based models:

$$p_o(\boldsymbol{\theta}) \;\propto\; \exp\left(-\frac{1}{2}h\big(s_o(\boldsymbol{\theta})\big)\right) \quad \text{and} \quad p_n(\boldsymbol{\theta}) \;\propto\; \exp\left(-\frac{1}{2}h\big(s_n(\boldsymbol{\theta})\big)\right)\;, \tag{3}$$

where $h(s)$ is a monotonic and differentiable function on $[0, +\infty]$ with $h(0) = 0$ and $h'(0) = 1$ while $s_o(\boldsymbol{\theta}) = (\boldsymbol{\theta} - \boldsymbol{m}_o)^\top \boldsymbol{\Lambda}_o(\boldsymbol{\theta} - \boldsymbol{m}_o)$ and $s_n(\boldsymbol{\theta}) = (\boldsymbol{\theta} - \boldsymbol{m}_n)^\top \boldsymbol{\Lambda}_n(\boldsymbol{\theta} - \boldsymbol{m}_n)$. Under this setup, the Gaussian scheme is recovered via setting $h(s) = s$. The MAP solution of the merged model distribution can then be characterized as solution to the following optimization task:

$$\boldsymbol{\theta}_m \;=\; \arg\min_{\boldsymbol{\theta}} \Big(h(s_o(\boldsymbol{\theta})) + h(s_n(\boldsymbol{\theta}))\Big) \;\Rightarrow\; \boldsymbol{\omega}_o^m \boldsymbol{\Lambda}_o\big(\boldsymbol{\theta}_m - \boldsymbol{m}_o\big) + \boldsymbol{\omega}_n^m \boldsymbol{\Lambda}_n\big(\boldsymbol{\theta}_m - \boldsymbol{m}_n\big) = 0 \tag{4}$$

where $\boldsymbol{\omega}_o^m = \boldsymbol{\omega}(s_o(\boldsymbol{\theta}_m))$ and $\boldsymbol{\omega}_n^m = \boldsymbol{\omega}(s_n(\boldsymbol{\theta}_m))$ with $\boldsymbol{\omega}(s) \triangleq h'(s)$.

**Tail Behavior.** The derivative $\boldsymbol{\omega}(s) = h'(s)$ can be used to characterize the tail behavior of the model distribution as it effectively characterizes the rate at which the log density of the model distribution decreases per unit of energy $s$:

$$\frac{\partial}{\partial s}\big(-\log p(\boldsymbol{\theta})\big) \;=\; \frac{\partial}{\partial s}\left(\frac{1}{2}h(s(\boldsymbol{\theta}))\right) \;=\; \frac{1}{2}\boldsymbol{\omega}(s(\boldsymbol{\theta}))\;, \tag{5}$$

where $p(\boldsymbol{\theta})$ and $s(\boldsymbol{\theta})$ can be replaced with $p_o/p_n$ and $s_o/s_n$ respectively.

**Plasticity-Stability Characterization.** Let $\boldsymbol{r} = \boldsymbol{m}_n - \boldsymbol{m}_o$ denote the displacement from the existing merged solution to the incoming task-specific solution. Rearranging Eq. 4 gives

$$\boldsymbol{\theta}_m - \boldsymbol{m}_o \;=\; \boldsymbol{B}\boldsymbol{r} \quad \text{with} \quad \boldsymbol{B} \;\triangleq\; (\boldsymbol{\omega}_o^m \boldsymbol{\Lambda}_o \,+\, \boldsymbol{\omega}_n^m \boldsymbol{\Lambda}_n)^{-1}\,\boldsymbol{\omega}_n^m \boldsymbol{\Lambda}_n\;, \tag{6}$$

$$\boldsymbol{\theta}_m - \boldsymbol{m}_n \;=\; -\boldsymbol{C}\boldsymbol{r} \quad \text{with} \quad \boldsymbol{C} \;\triangleq\; (\boldsymbol{\omega}_o^m \boldsymbol{\Lambda}_o \,+\, \boldsymbol{\omega}_n^m \boldsymbol{\Lambda}_n)^{-1}\,\boldsymbol{\omega}_o^m \boldsymbol{\Lambda}_o\;. \tag{7}$$

where, intuitively, $\boldsymbol{B}$ characterizes how much of the incoming displacement is incorporated as new knowledge (i.e., plasticity) whereas $\boldsymbol{C}$ characterizes how much of that displacement remains between the merged and incoming solutions to preserve old knowledge (i.e., stability). The tension between plasticity and stability is also concisely characterized via noting that $\boldsymbol{B} + \boldsymbol{C} = \boldsymbol{I}$. We provide a visual description of the plasticity-stability characterization in Fig. 2.

**Geometric Interpretation of Continual Merging.** The geometric structure of continual merging can now be concisely characterized via mapping the plasticity and stability operators, $\boldsymbol{B}$ and $\boldsymbol{C}$,

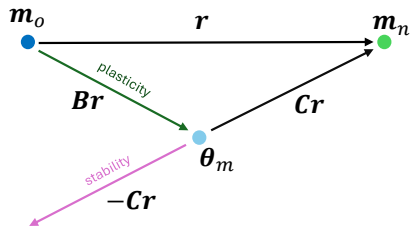


Figure 2: Visual description of plasticity-stability in continual merging via the matrices $\boldsymbol{B}$ and $\boldsymbol{C}$ in Eq. (6). $\boldsymbol{r}$ is the displacement between the existing and the new model. The direction $\boldsymbol{Br}$ is the portion incorporated as new knowledge (plasticity) while $-\boldsymbol{Cr}$ quantifies knowledge retention (stability).

| Method | Accuracy (%) |
|---|---|
| Vicuna-7B | 20.02 |
| WizardMath | 54.74 |
| Llama-2-Coder | 7.43 |
| Task Arithmetic | 43.59 |
| OPCM | 40.94 |
| DOP | 41.08 |
| COMET (Ours) | **47.16** |

Table 1: Accuracy (%) on GSM8K after sequentially merging three 7B-parameter models (Vicuna-7B → WizardMath → Llama-2-Coder). COMET achieves the highest performance, demonstrating scalability to larger scale models.

onto a unified coordinate system whose canonical directions are orthogonal under the inner product induced by the existing model distribution's precision matrix $\boldsymbol{\Lambda}_o$. This decomposition resolves the overall plasticity and stability effects into independent contributions along individual orthogonal update directions. The resulting representation exposes how these direction-wise effects depend on the mismatch in tail behavior between the existing and incoming model distributions.

**Geometric Base.** Let $\boldsymbol{\Lambda}_o = \boldsymbol{L}\boldsymbol{L}^\top$ denote its Cholesky factorization and $\boldsymbol{L}^{-1}\boldsymbol{\Lambda}_n\boldsymbol{L}^{-\top} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top$ denote its eigen-decomposition where $\boldsymbol{\Sigma} = \mathrm{diag}[\sigma_1, \sigma_2, \ldots, \sigma_\kappa]$ and $\boldsymbol{U} = [\boldsymbol{u}_1\ \boldsymbol{u}_2\ \ldots\ \boldsymbol{u}_\kappa]$. Next, let $\boldsymbol{E} = [\boldsymbol{e}_1\ \boldsymbol{e}_2\ \ldots\ \boldsymbol{e}_\kappa]$ where $\boldsymbol{e}_i = \boldsymbol{L}^{-1}\boldsymbol{u}_i$. It follows that $\boldsymbol{e}_a$ and $\boldsymbol{e}_b$ are orthogonal in the inner-product space of $\boldsymbol{\Lambda}_o$ since $\boldsymbol{e}_a^\top\boldsymbol{\Lambda}_o\boldsymbol{e}_b = \mathbb{I}(a = b)$. Moreover, $\boldsymbol{\Lambda}_n\boldsymbol{e}_i = \sigma_i\boldsymbol{\Lambda}_o\boldsymbol{e}_i$. Thus, each $\boldsymbol{e}_i$ is a generalized eigenvector of the pair $(\boldsymbol{\Lambda}_o, \boldsymbol{\Lambda}_n)$ with $\sigma_i$ specifies the relative precision of the incoming model distribution against the existing model distribution along that direction.

**Direction-wise Unit Scale of Update.** The above geometric construction diagonalizes the plasticity and stability operators, $\boldsymbol{B}$ and $\boldsymbol{C}$, thereby decomposing their effects into independent contributions along the orthogonal directions $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_\kappa$. This is formalized via the following lemma:

**Lemma 1** (Tail-Adaptive Unit Scale)**.** *Let $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{E}$, and $\boldsymbol{\Sigma} = \mathrm{diag}[\sigma_1, \sigma_2, \ldots, \sigma_\kappa]$ as defined above. Let $\eta = \boldsymbol{\omega}(s_o(\boldsymbol{\theta}_m))/\boldsymbol{\omega}(s_n(\boldsymbol{\theta}_m))$ which characterizes the ratio between the rates of log density decrease against energy unit ($s$) of the existing and new model distributions. This essentially captures their tail mismatch. We have:*

$$\boldsymbol{B}\boldsymbol{e}_i = \lambda_i\boldsymbol{e}_i \quad \textit{and} \quad \boldsymbol{C}\boldsymbol{e}_i = (1-\lambda_i)\boldsymbol{e}_i \quad \textit{where} \quad \lambda_i = \sigma_i/(\sigma_i + \eta)\,. \tag{8}$$

*A detailed derivation is deferred to Appendix A.1.*

The above result reveals a plasticity/stability decomposition across the orthogonal directions $\boldsymbol{e}_i$:

$$\boldsymbol{r} \triangleq \boldsymbol{m}_n - \boldsymbol{m}_o = \sum_{i=1}^{\kappa} c_i\boldsymbol{e}_i \;\Rightarrow\; \boldsymbol{B}\boldsymbol{r} = \sum_{i=1}^{\kappa} c_i\lambda_i\boldsymbol{e}_i \text{ and } \boldsymbol{C}\boldsymbol{r} = \sum_{i=1}^{\kappa} c_i(1-\lambda_i)\boldsymbol{e}_i\,. \tag{9}$$

Eq. 9 succinctly characterizes the decomposed effect of plasticity/stability across each learning direction. Intuitively, $c_i$ denote the amount of update in direction $\boldsymbol{e}_i$ that $\boldsymbol{r}$ requests. This is to be regulated with the direction-wise plasticity rate $\lambda_i = \sigma_i/(\sigma_i + \eta)$ which is a function of the tail mismatch ratio. Likewise, the direction-wise stability also regulates the preservation of existing knowledge along the direction $\boldsymbol{e}_i$ with stability rate $1 - \lambda_i$.

**Gaussian Deficiency.** The above rates effectively characterize the per-direction unit scales of plasticity/stability which are guided by the tail mismatch parameter $\eta = \boldsymbol{\omega}(s_o(\boldsymbol{\theta}_m))/\boldsymbol{\omega}(s_n(\boldsymbol{\theta}_m))$. In the Gaussian design, however, $h(s) = s$ and so $\boldsymbol{\omega}(s) = h'(s) = 1$ and as such, $\eta = 1$.

The plasticity/stability rate thus become oblivious of the tail mismatch between $s_o(\boldsymbol{\theta}_m)$ and $s_n(\boldsymbol{\theta}_m)$, consequently lacking adaptivity when the incoming model distribution is substantially different than the existing model distribution. As shown in Fig. 1 which plots the post-merging performance on

new (i.e., empirical plasticity) and existing tasks (i.e., empirical stability) against a wide spectrum of $\eta$. The plots show that $\eta = 1$ is not optimal for both plasticity and stability. Moreover, Fig. 3 and 4 also demonstrate a degradation in plasticity for Gaussian-based merging methods.

## 3 Continual Merging via Adaptation to Tail Behaviors

To fix the above deficiency, we will develop (i) a new energy-based distributional model that allows the direction-wise unit scales of update to adapt to distributional tail mismatch (Section 3.1); and (ii) an efficient iteratively reweighted fixed point algorithm for computing the resulting MAP merge. The solver exploits the Kronecker structure of task-specific curvature (Section 3.2), enabling scalable continual merging without access to task data (Section 3.3).

### 3.1 Energy-based Model Distribution with Adaptive Tail Decaying Rate

The above analysis traces the limitation of Gaussian merging to a more fundamental property of the Gaussian distribution. Its negative log density grows linearly with Mahalanobis energy, thus imposing a constant rate of tail decay regardless of how far the center of the new model distribution lies from that of the existing model distribution. Consequently, the relative influence of the existing and new model distributions cannot adapt to their differing tail behaviors and end up fixing the resulting direction-wise unit scales of update. To address this limitation, we introduce a more general energy-based distributional structure whose tail decay rate adapts to its Mahalanobis energy.

For a location parameter $\boldsymbol{m}$, precision matrix $\boldsymbol{\Lambda}$, and decay parameter $\nu > 0$, we define the following energy-based model distribution with adaptive tail decaying rate:

$$p_\nu(\boldsymbol{\theta}; \boldsymbol{m}, \boldsymbol{\Lambda}) \quad \propto \quad \left(1 + \nu^{-1} s(\boldsymbol{\theta})\right)^{-\nu} \;=\; \exp\left(-\frac{1}{2}\left(2\nu \log\left(1 + \frac{s(\boldsymbol{\theta})}{\nu}\right)\right)\right) , \tag{10}$$

where $s(\boldsymbol{\theta}) = (\boldsymbol{\theta} - \boldsymbol{m})^\top \boldsymbol{\Lambda}(\boldsymbol{\theta} - \boldsymbol{m})$. This belongs to the same distributional family in Eq. 3 with influence function $h_\nu(s) = 2\nu \log(1 + s/\nu)$. Its derivative is $\boldsymbol{\omega} = 2\nu/(\nu + s)$ which characterizes the rate at which the log density of $p_\nu$ decreases per unit of Mahalonobis energy $s$ – see Eq. 5. The corresponding MAP solution of the merged model under this energy-based model distribution is obtained by

$$\boldsymbol{\theta}_m \quad = \quad \arg\min_{\boldsymbol{\theta}} \; \nu_o \log\left(1 + \frac{s_o(\boldsymbol{\theta})}{\nu_o}\right) \;+\; \nu_n \log\left(1 + \frac{s_n(\boldsymbol{\theta})}{\nu_n}\right) . \tag{11}$$

Given the above, at each merging step, let $p(\boldsymbol{\theta}; \boldsymbol{m}_o, \boldsymbol{\Lambda}_o)$ and $p(\boldsymbol{\theta}; \boldsymbol{m}_n, \boldsymbol{\Lambda}_n)$ denote the generic distributions of the existing and new models. These distributions are described by the location parameters $(\boldsymbol{m}_o, \boldsymbol{m}_n)$ and scale parameters, which are precision matrices $(\boldsymbol{\Lambda}_o, \boldsymbol{\Lambda}_n)$. The location parameters $\boldsymbol{m}_o$ and $\boldsymbol{m}_n$ are chosen to be the existing model and the new model, respectively, while the precision matrices encode the curvature geometries around them.

Following prior practice (Ritter et al., 2018), the existing and new precision matrices $\boldsymbol{\Lambda}_o$ and $\boldsymbol{\Lambda}_n$ are computed from the Hessian of the task loss evaluated at the corresponding location parameters (Li et al., 2025). As computing the Hessian matrix is prohibitively expensive, it is often approximated via the empirical Fisher information matrix (FIM):

$$\boldsymbol{\Lambda}_o \;\simeq\; \boldsymbol{F}_o(\boldsymbol{\theta})\big|_{\boldsymbol{\theta}=\boldsymbol{m}_o} \;=\; \sum_{i=1}^{|D_o|} \left(\nabla_{\boldsymbol{\theta}} L_o(\boldsymbol{\theta}; \boldsymbol{x}_i, y_i)\big|_{\boldsymbol{\theta}=\boldsymbol{m}_o}\right)\left(\nabla_{\boldsymbol{\theta}} L_o(\boldsymbol{\theta}; \boldsymbol{x}_i, y_i)\big|_{\boldsymbol{\theta}=\boldsymbol{m}_o}^\top\right) , \tag{12}$$

where $L_o$ and $D_o$ denote the corresponding training loss and training data. Likewise, $\boldsymbol{\Lambda}_n \simeq \boldsymbol{F}_n(\boldsymbol{m}_n)$ where $\boldsymbol{F}_n(\boldsymbol{m}_n)$ is defined similar to $\boldsymbol{F}_o(\boldsymbol{m}_o)$ in Eq. 12.

This follows prior practice in various model merging techniques (Matena & Raffel, 2022; Li et al., 2025; Marouf et al., 2024). Computing FIM, however, requires access to the training data $D_o$ and the loss function $L_o$ used to train the corresponding task-specific model. This is not possible in continual merging scenarios where only model checkpoints are available.

Alternatively, computing and storing such curvature matrices are on the other hand infeasible since they scale quadratically with the number of parameters. To sidestep this issue, we instead construct a data-free surrogate from the task vectors which can exploit a decomposable Kronecker structure to avoid having to materialize the entire Fisher matrix, as detailed in Section 3.2.

### 3.2 DATA-FREE APPROXIMATION OF THE CURVATURE MATRIX

Following prior practice (Wortsman et al., 2022; Matena & Raffel, 2022; Ilharco et al., 2022; Stoica et al., 2024; Wei et al., 2025), the merge operation is performed independently for each weight matrix $\boldsymbol{W} = \boldsymbol{W}_o/\boldsymbol{W}_n$ while fixing the rest. This requires forming the FIM $\boldsymbol{F}(\boldsymbol{W}) = \boldsymbol{F}(\boldsymbol{W}_o)/\boldsymbol{F}(\boldsymbol{W}_n)$ for each weight matrix rather than for the entire model $\boldsymbol{m} = \boldsymbol{m}_o/\boldsymbol{m}_n$.

As each weight matrix is associated with a linear block in the model $\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x}$, $\partial L/\partial \boldsymbol{W} = \partial L/\partial \boldsymbol{z} \cdot \partial \boldsymbol{z}/\partial \boldsymbol{W} = \boldsymbol{g}\boldsymbol{x}^\top$ where $\boldsymbol{g} = \partial L/\partial \boldsymbol{z}$. As such $\mathrm{vec}(\partial L/\partial \boldsymbol{W}) = \boldsymbol{x} \otimes \boldsymbol{g}$ where $\otimes$ denotes the Kronecker product. Using Kronecker Approximate Curvature (K-FAC) (Martens et al., 2018):

$$\boldsymbol{F}(\boldsymbol{W}) \;=\; \mathbb{E}\left[(\boldsymbol{x}\boldsymbol{x}^\top) \otimes (\boldsymbol{g}\boldsymbol{g}^\top)\right] \;\simeq\; \mathbb{E}\big[(\boldsymbol{x}\boldsymbol{x}^\top)\big] \otimes \mathbb{E}\big[(\boldsymbol{g}\boldsymbol{g}^\top)\big] \;=\; \boldsymbol{A}(\boldsymbol{x}) \otimes \boldsymbol{G}(\boldsymbol{g})\,, \tag{13}$$

where the second equality follows from the Kronecker identity while $\boldsymbol{A}(\boldsymbol{x}) = \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top]$ and $\boldsymbol{G}(\boldsymbol{g}) = \mathbb{E}[\boldsymbol{g}\boldsymbol{g}^\top]$ which correspond to the activation factor and the gradient factor.

As both factors in K-FAC approximation still require access to data $\boldsymbol{x}$ and gradients $\boldsymbol{g}$ to compute, we estimate them from the task displacement $\boldsymbol{\Delta}_n$ to make the computation data-free. Following (Wei et al., 2025), we can view $\boldsymbol{\Delta}_n$ as the accumulated gradient of the new task's training process and can be written as a sum of outer products between the data and the gradient

$$\boldsymbol{\Delta}_n \;=\; \boldsymbol{W}_n - \boldsymbol{W}_0 \;=\; -\eta \sum_{k=1}^{|D|} \boldsymbol{g}_k \boldsymbol{x}_k^\top\,. \tag{14}$$

where $\eta$ is the learning rate used to train the task model. As such, the row space of $\boldsymbol{\Delta}_n$ is spanned by the task inputs and its column space by the output gradients. This means

$$\boldsymbol{A}_n \;\triangleq\; \boldsymbol{\Delta}_n^\top \boldsymbol{\Delta}_n \in \mathbb{R}^{N\times N}, \qquad \boldsymbol{G}_n \;\triangleq\; \frac{\boldsymbol{\Delta}_n \boldsymbol{\Delta}_n^\top}{\|\boldsymbol{\Delta}_n\|_F^2} \in \mathbb{R}^{M\times M}\,, \tag{15}$$

can be used as the data-free proxy for $\mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top]$ and for $\mathbb{E}[\boldsymbol{g}\boldsymbol{g}^\top]$, respectively. The normalization of $\boldsymbol{G}_n$ by $\|\boldsymbol{\Delta}_n\|_F^2$ yields a unit-Frobenius gradient factor that preserves the dominant gradient subspace while keeping the relative weighting of the Kronecker factors consistent across tasks. As a result, we can compute the new-task precision by $\boldsymbol{\Lambda}_n = \boldsymbol{A}_n \otimes \boldsymbol{G}_n$. On the other hand, the existing precision $\boldsymbol{\Lambda}_o = \boldsymbol{A}_o \otimes \boldsymbol{G}_o$ is maintained recursively by exponentially averaging the two factors,

$$\boldsymbol{A}_o \;\leftarrow\; \boldsymbol{A}_o \;+\; \boldsymbol{A}_n, \quad \boldsymbol{G}_o \;\leftarrow\; \frac{1}{c}\Big(\boldsymbol{G}_o \;+\; \boldsymbol{G}_n\Big), \quad c \;\leftarrow\; c+1, \tag{16}$$

with base conditions $\boldsymbol{A}_o = \boldsymbol{G}_o = \boldsymbol{0}, c = 0$ and $c$ denotes the counter variable for number of tasks seen at the current time. The division by $c$ is to prevent the magnitude of the gradient factor from growing with the number of tasks. In contrast, $\boldsymbol{A}_o$ is accumulated additively so that it retains the cumulative curvature scale contributed by the observed task displacements.

### 3.3 LAYER-WISE TAIL ADAPTIVE MAP SOLVER

To compute the MAP solution of the merged model under the energy-based formulation in Eq. (11), we note that the loss can be rewritten entirely in terms of the smaller activation and gradient factors. This leads to a matrix-space optimization problem whose fixed points meet a non-linear matrix equation (Section 3.3.1). As the energy-based responsibilities render this equation non-linear, we adopt an iteratively reweighted fixed-point (IRFP) scheme that freezes the responsibilities at the current estimate and solves the resulting linear generalized Sylvester equation (Duan, 2015) (Section 3.3.2). Finally, a change of coordinates based on the Cholesky factors of the existing curvature diagonalizes the problem, reducing each linear solve to a set of independent scalar updates and thereby avoiding explicit construction of the full precision matrix (Section 3.3.3).

#### 3.3.1 SOLUTION CHARACTERIZATION

Using the data-free Kronecker precisions above, the quadratic forms in Eq. 11 can be written in matrix form. Concretely, when merging a weight matrix $\boldsymbol{W}_o$ of the existing model $\boldsymbol{m}_o$, the existing and new Mahalanobis energies at a weight matrix $\boldsymbol{W}$ are given by

$$s_o(\boldsymbol{W}) \;=\; \mathrm{vec}(\boldsymbol{W} - \boldsymbol{W}_o)^\top \boldsymbol{\Lambda}_o \mathrm{vec}(\boldsymbol{W} - \boldsymbol{W}_o) \;=\; \mathrm{tr}\left[\boldsymbol{G}_o(\boldsymbol{W} - \boldsymbol{W}_o)\boldsymbol{A}_o(\boldsymbol{W} - \boldsymbol{W}_o)^\top\right],$$

and likewise $s_n(\boldsymbol{W}) = \operatorname{tr}\left[\boldsymbol{G}_n(\boldsymbol{W}-\boldsymbol{W}_n)\boldsymbol{A}_n(\boldsymbol{W}-\boldsymbol{W}_n)^\top\right]$, where $\boldsymbol{A}_n \otimes \boldsymbol{G}_n$ is the new task precision. Plugging these into Eq. 11 leads to the following matrix-valued optimization problem

$$\boldsymbol{W}_m \;=\; \arg\min_{\boldsymbol{W}} \tfrac{1}{2} h_{\nu_o}\big(s_{\mathrm{o}}(\boldsymbol{W})\big) + \tfrac{1}{2} h_{\nu_n}\big(s_{\mathrm{n}}(\boldsymbol{W})\big) \;\triangleq\; \arg\min_{\boldsymbol{W}} g(\boldsymbol{W})\,, \tag{17}$$

whose fixed points are characterized by the following result.

**Proposition 1** (Stationary solution of 17)**.** *Every stationary point $\boldsymbol{W}_m$ of the layer-wise robust MAP loss 17 satisfies the following matrix equation*

$$\omega_{\mathrm{o}}(\boldsymbol{W}_m)\,\boldsymbol{G}_o\,\boldsymbol{W}_m\,\boldsymbol{A}_o \;+\; \omega_n(\boldsymbol{W}_m)\,\boldsymbol{G}_t\,\boldsymbol{W}_m\,\boldsymbol{A}_t = \omega_{\mathrm{o}}(\boldsymbol{W}_m)\,\boldsymbol{G}_n\,\boldsymbol{W}_o\,\boldsymbol{A}_n + \omega_{\mathrm{n}}(\boldsymbol{W}_m)\,\boldsymbol{G}_n\,\boldsymbol{W}_n\,\boldsymbol{A}_n.$$

We provide the proof for this Proposition in Appendix A.2.

### 3.3.2 Fixed-Point Iterative Solver

The tail adaptive formulation introduces nonlinearities into the fixed-point condition 18 through the responsibilities $\omega_o(\boldsymbol{W})$ and $\omega_n(\boldsymbol{W})$. To solve for the fixed-point solution, we adopt an iterative approach which evaluates the responsibilities using the current estimate of the merged model, fix these weights, and solve the resulting generalized Sylvester equation (Duan, 2015).

**Corollary 1.** *Let $\boldsymbol{W}_m^{(k)}$ denote the current estimate of the merged model and*

$$\omega_o^{(k)} \;=\; \omega_o(\boldsymbol{W}_m^{(k)}), \quad \omega_n^{(k)} \;=\; \omega_n(\boldsymbol{W}_m^{(k)}).$$

*and the next iterate $\boldsymbol{W}_m^{(k+1)}$ is obtained by via solving Eq. 18, which is a generalized Sylvester equation with these fixed responsibilities. Moreover, this iterative procedure monotonically decreases the MAP loss in Eq. 17, i.e.,*

$$0 \;\le\; g(\boldsymbol{W}_m^{(k+1)}) \;\le\; g(\boldsymbol{W}_m^{(k)})\,. \tag{18}$$

The generalized Sylvester equation 18 couples two different curvature structures: the existing factors $(\boldsymbol{G}_o, \boldsymbol{A}_o)$, which encode the geometry accumulated from previous tasks, and the new-task factors $(\boldsymbol{G}_n, \boldsymbol{A}_n)$, which characterize the geometry of the incoming task. The geometric interpretation for the matrix-valued case can be developed analogously to the vector case in Section 2. In particular, under the Kronecker-factored representation, the Cholesky factor of the full precision matrix $\boldsymbol{\Lambda}_o$ factorizes into the Cholesky factors of its constituent Kronecker factors:

$$\boldsymbol{\Lambda}_o = \boldsymbol{L}\boldsymbol{L}^\top = (\boldsymbol{L}_A \otimes \boldsymbol{L}_G)(\boldsymbol{L}_A \otimes \boldsymbol{L}_G)^\top = (\boldsymbol{L}_A\boldsymbol{L}_A^\top) \otimes (\boldsymbol{L}_G\boldsymbol{L}_G^\top) = \boldsymbol{A}_o \otimes \boldsymbol{G}_o. \tag{19}$$

As such, the geometric analysis developed for the entire precision matrix can be carried out equivalently on its much smaller Kronecker factors. In particular, the corresponding eigenspace transformations decompose into operations on $\boldsymbol{A}_o$ and $\boldsymbol{G}_o$, avoiding explicit construction of the full precision matrix. We provide the detailed derivation in Appendix B.

### 3.3.3 Coordinate Change for Scalable Iterations

The Cholesky factors $\boldsymbol{L}_A$ and $\boldsymbol{L}_G$ of $\boldsymbol{A}_o = \boldsymbol{L}_A\boldsymbol{L}_A^\top$ and $\boldsymbol{G}_o = \boldsymbol{L}_G\boldsymbol{L}_G^\top$ provide a change of coordinates such that we can express the new-task curvature in the same coordinate system via $(\boldsymbol{L}_G^{-1}\boldsymbol{G}_n\boldsymbol{L}_G^{-\top}, \boldsymbol{L}_A^{-1}\boldsymbol{A}_n\boldsymbol{L}_A^{-\top})$. Under this coordinate change, the transformed new-task factors describe how strongly the new task constrains each direction relative to the existing model.

We can find directions in which the two curvature structures can be compared independently by diagonalizing the transformed factors. This allows decomposing the solution update into efficient independent scalar updates, as shown in the Theorem 1 below.

**Theorem 1** (Closed-Form Fixed-Point Map)**.** *Equation (18), whose responsibilities held fixed, admits an efficient closed-form solution. Let $\boldsymbol{G}_o = \boldsymbol{L}_G\boldsymbol{L}_G^\top$ and $\boldsymbol{A}_o = \boldsymbol{L}_A\boldsymbol{L}_A^\top$ be Cholesky factorizations of the existing Kronecker factors, and let*

$$\boldsymbol{L}_G^{-1}\boldsymbol{G}_n\boldsymbol{L}_G^{-\top} = \boldsymbol{U}\,\mathrm{diag}(\boldsymbol{p})\,\boldsymbol{U}^\top, \qquad \boldsymbol{L}_A^{-1}\boldsymbol{A}_n\boldsymbol{L}_A^{-\top} = \boldsymbol{V}\,\mathrm{diag}(\boldsymbol{q})\,\boldsymbol{V}^\top$$

*be the eigen-decompositions of the new-task Kronecker factors under the change of coordinate. Then, let* $\mathcal{T}(\boldsymbol{X}) \triangleq \boldsymbol{U}^\top \boldsymbol{L}_G^\top \boldsymbol{X}\, \boldsymbol{L}_A \boldsymbol{V}$ *define the invertible mapping, it follows that*

$$s_o(\boldsymbol{W}) \;=\; \sum_{i,j} \big([\mathcal{T}(\boldsymbol{W}) - \mathcal{T}(\boldsymbol{W}_o)]_{i,j}\big)^2, \quad s_n(\boldsymbol{W}) \;=\; \sum_{i,j} p_i q_j \big([\mathcal{T}(\boldsymbol{W}) - \mathcal{T}(\boldsymbol{W}_n)]_{i,j}\big)^2, \tag{20}$$

*and (18) decouples into the independent entry-wise characterization*

$$[\mathcal{T}(\boldsymbol{W}_m)]_{ij} \quad = \quad \frac{\omega_{\mathrm{o}}(\boldsymbol{W}_m)\,[\mathcal{T}(\boldsymbol{W}_o)]_{ij} + \omega_n(\boldsymbol{W}_m)\,p_i q_j\,[\mathcal{T}(\boldsymbol{W}_n)]_{ij}}{\omega_o(\boldsymbol{W}_m) + \omega_n(\boldsymbol{W}_m)\,p_i q_j}\,, \tag{21}$$

*The solution of the Sylvester equation (18) is recovered by the inverse mapping*

$$\boldsymbol{W}_m \quad = \quad \mathcal{T}^{-1}(\mathcal{T}(\boldsymbol{W}_m)) \quad = \quad \boldsymbol{L}_G^{-\top} \boldsymbol{U}\, \mathcal{T}(\boldsymbol{W}_m)\, \boldsymbol{V}^\top \boldsymbol{L}_A^{-1}. \tag{22}$$

The proof for Theorem 1 is provided in Appendix A.4.

Theorem 1 enables an efficient and scalable implementation of the merging procedure in Alg. 1, whose complexity analysis is provided in Appendix D. By transforming the problem into the shared eigenspace, the matrix-space optimization decomposes into independent scalar updates, thus avoiding explicit operations on the full curvature matrix.

#### 3.3.4 Hyperparameter Scheduling

**$\nu$-scheduling.** In our implementation, the decay parameters $\nu_o$ of the existing expert and $\nu_n$ of the new-task expert are adaptive to the geometry of the tasks. At the beginning of each task, we set them as the scaled Mahalanobis distance between the existing weight and the new-task matrix,

$$\nu_n \;=\; \kappa\, s_n(\boldsymbol{W}_o), \qquad \nu_o \;=\; \kappa\, s_o(\boldsymbol{W}_n). \tag{23}$$

These values are computed per task and held fixed for the fixed-point iteration. The scalar $\kappa > 0$ controls how aggressively an expert is allowed to be down-weighted.

## 4 Experiments and Results

In this section, we show the effectiveness of our COMET framework to tackle the continual merging problem. We describe the experimental setup in Section 4.1 and provide the main results in Section 4.2. We further provide ablation studies and analyses in Section G. Our implementation is provided in the anonymous repository.

### 4.1 Experimental Setup

**Models and Datasets.** Following the setup in OPCM (Tang et al., 2025), we fine-tune three CLIP-based Vision Transformers of different scales (ViT-B/32, ViT-B/16, and ViT-L/14) (Radford et al., 2021) on 20 vision datasets, producing a set of task-specific models. Merging performance is assessed under three progressively larger task collections: a base set of 8 tasks, an extended set of 14 tasks, and the complete set of 20 tasks. For natural-language tasks we sequentially merge LoRA-adapted Flan-T5-base and Flan-T5-large models (checkpoints released by (Tang et al., 2023)) across the eight GLUE benchmarks (Wang et al., 2018). Finally, for large-language-model merging we sequentially combine three 7B-parameter models: Vicuna-7B, WizardMath (Luo et al., 2025), and Llama-2-Coder (Roziere et al., 2023) which are all derived from the shared Llama-2-7B backbone (Touvron et al., 2023). The resulting merged model is evaluated on the GSM8K mathematical-reasoning suite (Luo et al., 2025). Details of models and datasets are provided in Appendix E.

**Baselines.** We compare COMET with several leading continual model-merging approaches: Weight Averaging (Wortsman et al., 2022), Task Arithmetic (Ilharco et al., 2022), OPCM (Tang et al., 2025), and DOP (Yang et al., 2026). A detailed description is given in Appendix F.

Table 2: Performance of continual merging methods across different merging scenarios. The best results are **bold** and the second best ones are underlined.

| | Method | ViT-B/32 | | | ViT-B/16 | | | ViT-L/14 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | 8 tasks | 14 tasks | 20 tasks | 8 tasks | 14 tasks | 20 tasks | 8 tasks | 14 tasks | 20 tasks |
| | **Continual Model Merging Methods** | | | | | | | | | |
| **ACC (↑)** | Averaging (SWA) | $66.3 \pm 0.0$ | $65.4 \pm 0.0$ | $61.1 \pm 0.0$ | $72.3 \pm 0.0$ | $69.7 \pm 0.0$ | $64.8 \pm 0.0$ | $80.0 \pm 0.0$ | $77.5 \pm 0.0$ | $71.1 \pm 0.0$ |
| | C. Task Arithmetic | $67.5 \pm 0.0$ | $66.5 \pm 0.0$ | $60.6 \pm 0.0$ | $77.1 \pm 0.0$ | $70.9 \pm 0.0$ | $64.2 \pm 0.0$ | $82.1 \pm 0.0$ | $77.9 \pm 0.0$ | $70.3 \pm 0.0$ |
| | C. TIES-Merging | $49.0 \pm 10.2$ | $66.2 \pm 0.6$ | $59.9 \pm 0.7$ | $66.8 \pm 3.7$ | $70.5 \pm 0.8$ | $63.0 \pm 1.6$ | $64.3 \pm 7.0$ | $78.0 \pm 0.6$ | $68.3 \pm 0.9$ |
| | OPCM | $75.5 \pm 0.5$ | $71.9 \pm 0.3$ | $65.7 \pm 0.2$ | $81.8 \pm 0.3$ | $77.1 \pm 0.5$ | $70.3 \pm 0.2$ | $87.0 \pm 0.4$ | $83.5 \pm 0.2$ | $76.0 \pm 0.2$ |
| | DOP | $\underline{78.3 \pm 1.6}$ | $\underline{73.0 \pm 1.2}$ | $\mathbf{68.5 \pm 1.6}$ | $\underline{84.5 \pm 0.8}$ | $\underline{76.2 \pm 1.2}$ | $\underline{72.2 \pm 1.8}$ | $\mathbf{88.3 \pm 0.7}$ | $\mathbf{85.5 \pm 0.8}$ | $\underline{80.7 \pm 0.9}$ |
| | COMET (Ours) | $\mathbf{82.2 \pm 0.8}$ | $\mathbf{74.1 \pm 0.7}$ | $\underline{66.7 \pm 1.6}$ | $\mathbf{85.1 \pm 0.4}$ | $\mathbf{80.3 \pm 0.6}$ | $\mathbf{72.8 \pm 1.5}$ | $\underline{88.1 \pm 0.3}$ | $\underline{85.0 \pm 0.4}$ | $\mathbf{81.7 \pm 0.8}$ |
| **BWT (↓)** | Averaging (SWA) | $-11.5 \pm 2.2$ | $-8.0 \pm 1.3$ | $-7.1 \pm 2.1$ | $-9.7 \pm 1.5$ | $-7.1 \pm 1.4$ | $-7.3 \pm 1.7$ | $-7.3 \pm 1.4$ | $-5.8 \pm 1.0$ | $-6.4 \pm 1.5$ |
| | C. Task Arithmetic | $-9.6 \pm 1.5$ | $-1.3 \pm 0.3$ | $-3.4 \pm 0.4$ | $-4.2 \pm 1.0$ | $-1.3 \pm 0.4$ | $-3.6 \pm 0.4$ | $-7.1 \pm 0.8$ | $-1.8 \pm 0.3$ | $-3.3 \pm 0.3$ |
| | C. TIES-Merging | $-15.3 \pm 8.0$ | $1.9 \pm 0.6$ | $-1.5 \pm 0.7$ | $-5.5 \pm 0.4$ | $1.4 \pm 0.7$ | $-1.5 \pm 1.2$ | $-13.0 \pm 5.7$ | $1.1 \pm 0.4$ | $-2.9 \pm 1.0$ |
| | OPCM | $-6.3 \pm 1.1$ | $-6.0 \pm 1.0$ | $-7.8 \pm 1.5$ | $-4.8 \pm 0.7$ | $-5.1 \pm 1.4$ | $-6.3 \pm 2.2$ | $-2.6 \pm 1.0$ | $-4.3 \pm 0.7$ | $-6.5 \pm 1.8$ |
| | DOP | $-6.1 \pm 2.3$ | $-8.8 \pm 2.1$ | $-12.5 \pm 2.8$ | $-3.9 \pm 1.0$ | $-7.5 \pm 2.2$ | $-11.4 \pm 2.8$ | $-3.2 \pm 1.2$ | $-4.6 \pm 1.0$ | $-8.4 \pm 1.4$ |
| | COMET (Ours) | $-6.7 \pm 1.4$ | $-7.6 \pm 2.3$ | $-9.5 \pm 1.4$ | $-4.2 \pm 0.4$ | $-5.3 \pm 1.6$ | $-8.7 \pm 1.6$ | $-4.1 \pm 0.9$ | $-4.0 \pm 0.8$ | $-5.0 \pm 0.8$ |

### 4.2 Main results

**Vision results.** Table 2 reports the average accuracy and backward transfer when sequentially merging CLIP ViT models of three scales on collections of vision tasks. Across all three architectures COMET consistently outperforms task arithmetic, averaging, TIES-Merging and OPCM. Notably, COMET always achieves the best or second best performance among recent developed continual merging methods, OPCM and DOP on all settings considered. Specifically, on the 8-task suite, COMET improves accuracy by $3.9$ points over DOP and by $6.7$ points over OPCM. The advantage remains visible at 14-tasks, up to $4.1$ points over DOP on ViT-B/16, and at the full 20-task stream, which outperforms OPCM by up to $5.7$ points on ViT-L/14. Backward transfer stays competitive with the geometry-based methods, indicating that the adaptive tail formulation improves adaptation to new tasks without severely sacrificing the retention of earlier tasks.

**Natural language results.** Table 4 reports the final average GLUE score and backward transfer obtained by sequentially merging LoRA-adapted Flan-T5 models across all eight GLUE tasks. On Flan-T5-base, COMET improves upon OPCM and DOP while maintaining comparable backward transfer. On the larger Flan-T5-large backbone the same experimental protocol yields close performance scores for all methods due to stronger model capacity. Regardless, COMET still attains better accuracy than the baselines. These results indicate that the tail adaptive formulation continues to provide consistent gains when the underlying model capacity is increased.

**LLM results.** Table 1 evaluates sequential merging of three 7B-parameter models, which are fine-tuned from a shared Llama-2-7B backbone, on the GSM8K mathematical reasoning benchmark. After the full three-model stream, COMET reaches $47.16\%$ accuracy, outperforming Task Arithmetic ($43.59\%$), OPCM ($40.94\%$) and DOP ($41.08\%$). The results show that COMET remains effective at the 7B scale, where independently adapted checkpoints exhibit substantial parameter displacement and the Gaussian assumptions underlying simpler averaging methods become particularly restrictive.

**Other results.** Runtime, stability / plasticity analysis and Kronecker factors accumulation effects of COMET are in Appendix G. NLP results and additional vision results are in Appendix H.

## 5 Conclusion

This paper studied continual model merging from a distributional and geometric perspective, showing that sequential knowledge incorporation can be decomposed into updates along orthogonal latent directions whose unit scales of update regulate the plasticity and stability rates. This analysis reveals a fundamental limitation of existing model merging techniques that can be reinterpreted under a Gaussian merging view. The Gaussian structure imposes a constant tail-decay rate, preventing the direction-wise plasticity and stability rates from adapting to mismatch between the existing and incoming model distributions. Motivated by this observation, we introduced COMET, an energy-based continual merging framework whose tail-decay rate adapts to distributional disagreement, allowing the relative influence of existing and incoming knowledge to adjust during merging. Across vision and language benchmarks, including billion-parameter foundation models, COMET consistently improves over competitive continual merging baselines while retaining computational efficiency.

### AI USE STATEMENT

In this work, we used generative AI tools for grammar checking and proof checking/verification. We have reviewed carefully every output generated by AI.

## REFERENCES


Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European conference on computer vision*, pp. 446–461. Springer, 2014.

Long Minh Bui, Tuan Anh Le Van, Tung Phi Duc, Phi Le Nguyen, Jana Doppa, and Trong Nghia Hoang. Model merging as probabilistic inference in fine-tuning parameter space. *arXiv preprint arXiv:2607.01689*, 2026.

Shiqi Chen, Jinghan Zhang, Tongyao Zhu, Wei Liu, Siyang Gao, Miao Xiong, Manling Li, and Junxian He. Bring reason to vision: Understanding perception and reasoning through model merging. *arXiv preprint arXiv:2505.05464*, 2025.

Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017.

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 *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223. JMLR Workshop and Conference Proceedings, 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 *2017 international joint conference on neural networks (IJCNN)*, pp. 2921–2926. IEEE, 2017.

Nico Daheim, Thomas Möllenhoff, Edoardo Maria Ponti, Iryna Gurevych, and Mohammad Emtiyaz Khan. Model merging by uncertainty-based gradient matching. *arXiv preprint arXiv:2310.12808*, 2023.

Guang-Ren Duan. *Generalized Sylvester equations: unified parametric solutions*. Crc Press, 2015.

Sebastian Dziadzio, Vishaal Udandarao, Karsten Roth, Ameya Prabhu, Zeynep Akata, Samuel Albanie, and Matthias Bethge. How to merge your multimodal models over time? In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20479–20491. IEEE, 2025.

Antonio Andrea Gargiulo, Donato Crisostomi, Maria Sofia Bucarelli, Simone Scardapane, Fabrizio Silvestri, and Emanuele Rodola. Task singular vectors: Reducing task interference in model merging. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 18695–18705, 2025.

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. In *International conference on neural information processing*, pp. 117–124. Springer, 2013.

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*, 12(7):2217–2226, 2019.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. *arXiv preprint arXiv:2212.09849*, 2022.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pp. 554–561, 2013.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images.(2009), 2009.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Yu-Ang Lee, Ching-Yun Ko, Tejaswini Pedapati, I-Hsin Chung, Mi-Yen Yeh, and Pin-Yu Chen. Star: Spectral truncation and rescale for model merging. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 496–505, 2025.

Mei Li, Yuxiang Lu, Qinyan Dai, Suizhi Huang, Yue Ding, and Hongtao Lu. Became: Bayesian continual learning with adaptive model merging. *arXiv preprint arXiv:2504.02666*, 2025.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jian-Guang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, Yansong Tang, et al. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. In *International Conference on Learning Representations*, volume 2025, pp. 49573–49609, 2025.

Imad Eddine Marouf, Subhankar Roy, Enzo Tartaglione, and Stéphane Lathuilière. Weighted ensemble models are strong continual learners. In *European Conference on Computer Vision*, pp. 306–324. Springer, 2024.

James Martens, Jimmy Ba, and Matt Johnson. Kronecker-factored curvature approximations for recurrent neural networks. In *International Conference on Learning Representations*, 2018.

Michael S Matena and Colin A Raffel. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716, 2022.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 7. Granada, 2011.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729. IEEE, 2008.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3498–3505. IEEE, 2012.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *International conference on learning representations*, 2018.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, Zifeng Wang, Sayna Ebrahimi, and Hao Wang. Continual learning of large language models: A comprehensive survey. *ACM Computing Surveys*, 58(5):1–42, 2025.

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 *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *The 2011 international joint conference on neural networks*, pp. 1453–1460. IEEE, 2011.

George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. Model merging with svd to tie the knots. *arXiv preprint arXiv:2410.19735*, 2024.

Yi-Lin Sung, Linjie Li, Kevin Lin, Zhe Gan, Mohit Bansal, and Lijuan Wang. An empirical study of multimodal model merging. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 1563–1575, 2023.

Derek Tam, Margaret Li, Prateek Yadav, Rickard Brüel Gabrielsson, Jiacheng Zhu, Kristjan Greenewald, Mikhail Yurochkin, Mohit Bansal, Colin Raffel, and Leshem Choshen. Llm merging: Building llms efficiently through merging. In *NeurIPS 2024 Competition Track*, 2024.

Anke Tang, Li Shen, Yong Luo, Liang Ding, Han Hu, Bo Du, and Dacheng Tao. Concrete subspace learning based interference elimination for multi-task model fusion. *arXiv preprint arXiv:2312.06173*, 2023.

Anke Tang, Enneng Yang, Li Shen, Yong Luo, Han Hu, Bo Du, and Dacheng Tao. Merging models on the fly without retraining: A sequential approach to scalable continual model merging. *arXiv preprint arXiv:2501.09522*, 2025.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Bastiaan S Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. Rotation equivariant cnns for digital pathology. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 210–218. Springer, 2018.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP*, pp. 353–355, 2018.

Haoxiang Wang, Pavan Kumar Anasosalu Vasu, Fartash Faghri, Raviteja Vemulapalli, Mehrdad Farajtabar, Sachin Mehta, Mohammad Rastegari, Oncel Tuzel, and Hadi Pouransari. Sam-clip: Merging vision foundation models towards semantic and spatial understanding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 3635–3647. IEEE, 2024.

Yongxian Wei, Anke Tang, Li Shen, Zixuan Hu, Chun Yuan, and Xiaochun Cao. Modeling multi-task model merging as adaptive projective gradient descent. *arXiv preprint arXiv:2501.01230*, 2025.

Yongxian Wei, Runxi Cheng, Weike Jin, Enneng Yang, Li Shen, Lu Hou, Sinan Du, Chun Yuan, Xiaochun Cao, and Dacheng Tao. Optmerge: Unifying multimodal llm capabilities and modalities via model merging. In *International Conference on Learning Representations*, volume 2026, pp. 61571–61595, 2026.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. Pmlr, 2022.

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, Krista A Ehinger, James Hays, Antonio Torralba, and Aude Oliva. Sun database: Exploring a large collection of scene categories. *International Journal of Computer Vision*, 119 (1):3–22, 2016.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36:7093–7115, 2023.

Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. Model merging in llms, mllms, and beyond: Methods, theories, applications, and opportunities. *ACM Computing Surveys*, 2024.

Enneng Yang, Anke Tang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, and Jie Zhang. Continual model merging without data: Dual projections for balancing stability and plasticity. *Advances in Neural Information Processing Systems*, 38:39275–39305, 2026.

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 *Forty-first International Conference on Machine Learning*, 2024.

# A PROOFS

## A.1 PROOF OF LEMMA 1

*Proof.* We have from the construction in Section 2, $\boldsymbol{\Lambda}_o = \boldsymbol{L}\boldsymbol{L}^\top$ and $\boldsymbol{L}^{-1}\boldsymbol{\Lambda}_n\boldsymbol{L}^\top = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top$. We choose the canonical basis $\boldsymbol{e}_i = \boldsymbol{L}^{-\top}\boldsymbol{u}_i$.

Note that , $\boldsymbol{\Lambda}_o\boldsymbol{e}_i = \boldsymbol{L}\boldsymbol{u}_i$ and thus $\boldsymbol{\Lambda}_n\boldsymbol{e}_i = \sigma_i\boldsymbol{L}\boldsymbol{u}_i = \sigma_i\boldsymbol{\Lambda}_o\boldsymbol{e}_i$. We can then express

$$\begin{aligned}\boldsymbol{B}\boldsymbol{e}_i &= \left(\omega_o^m\boldsymbol{\Lambda}_o + \omega_n^m\boldsymbol{\Lambda}_n\right)^{-1}\omega_n^m\boldsymbol{\Lambda}_n\boldsymbol{e}_i \;=\; \left(\omega_o^m\boldsymbol{\Lambda}_o + \omega_n^m\boldsymbol{\Lambda}_n\right)^{-1}\omega_n^m\sigma_i\boldsymbol{\Lambda}_o\boldsymbol{e}_i \\ &= (\omega_n^m\sigma_i)\left(\omega_o^m\boldsymbol{\Lambda}_o + \omega_n^m\boldsymbol{\Lambda}_n\right)^{-1}\boldsymbol{\Lambda}_o\boldsymbol{e}_i. \end{aligned} \tag{24}$$

Note that, $(\omega_o^m\boldsymbol{\Lambda}_o + \omega_n^m\boldsymbol{\Lambda}_n)\boldsymbol{e}_i = \omega_o^m\boldsymbol{\Lambda}_o\boldsymbol{e}_i + \omega_n^m\sigma_i\boldsymbol{\Lambda}_o\boldsymbol{e}_i = (\omega_o^m + \omega_n^m\sigma_i)\boldsymbol{\Lambda}_o\boldsymbol{e}_i$. Multiply both sides by the inverse matrix, we have

$$\boldsymbol{e}_i = (\omega_o^m + \omega_n^m\sigma_i)(\omega_o^m\boldsymbol{\Lambda}_o + \omega_n^m\boldsymbol{\Lambda}_n)^{-1}\boldsymbol{\Lambda}_o\boldsymbol{e}_i,$$

thus,

$$(\omega_o^m\boldsymbol{\Lambda}_o + \omega_n^m\boldsymbol{\Lambda}_n)^{-1}\boldsymbol{\Lambda}_o\boldsymbol{e}_i = \frac{1}{\omega_o^m + \omega_n^m\sigma_i}\boldsymbol{e}_i. \tag{25}$$

Substitute this back to $\boldsymbol{B}\boldsymbol{e}_i$ yields the eigen-equation of $\boldsymbol{B}$

$$\boldsymbol{B}\boldsymbol{e}_i \;=\; \frac{\omega_n^m\sigma_i}{\omega_o^m + \omega_n^m\sigma_i}\boldsymbol{e}_i. \tag{26}$$

The eigenvalues of $B$ are

$$\lambda_i \;=\; \frac{\omega_n^m\sigma_i}{\omega_o^m + \omega_n^m\sigma_i} \;=\; \frac{\sigma_i}{\eta + \sigma_i}. \tag{27}$$

As $\boldsymbol{C} = \boldsymbol{I} - \boldsymbol{B}$, we also have $\boldsymbol{C}\boldsymbol{e}_i = (1 - \lambda_i)\boldsymbol{e}_i$. □

## A.2 PROOF OF PROPOSITION 1

*Proof.* We take the gradient of $g(\boldsymbol{W})$ w.r.t. $\boldsymbol{W}$. By the chain rule, we have

$$\begin{aligned}\nabla_{\boldsymbol{W}} g(\boldsymbol{W}) &= \frac{1}{2}\omega_o(\boldsymbol{W})\cdot\nabla_{\boldsymbol{W}} s_o(\boldsymbol{W}) + \frac{1}{2}\omega_n(\boldsymbol{W})\cdot\nabla_{\boldsymbol{W}} s_n(\boldsymbol{W}) \\ &= \omega_o(\boldsymbol{W})\cdot\boldsymbol{G}_o(\boldsymbol{W} - \boldsymbol{W}_o)\boldsymbol{A}_o + \omega_n(\boldsymbol{W})\cdot\boldsymbol{G}(\boldsymbol{\Delta}_n)(\boldsymbol{W} - \boldsymbol{W}_n)\boldsymbol{A}(\boldsymbol{\Delta}_n)\end{aligned}$$

where the second equality follows from the gradient of the trace operator and the fact that $\boldsymbol{G}_o, \boldsymbol{A}_o, \boldsymbol{G}(\boldsymbol{\Delta}_n), \boldsymbol{A}(\boldsymbol{\Delta}_n)$ are all symmetric. Set $\nabla_{\boldsymbol{W}} g(\boldsymbol{W})$ to 0 and we obtain (18). □

## A.3 PROOF OF COROLLARY 1

*Proof.* Since $h_\nu(s) = 2\nu\log(1 + s/\nu)$ is concave in $s$, we have the tangent upper bound at every $s$, for some constant $s^{(k)}$,

$$h_\nu(s) \;\leq\; h_\nu(s^{(k)}) + h'_\nu(s^{(k)})(s - s^k)) \;= h'_\nu(s^{(k)})s + const. \tag{28}$$

Substituting the two residuals $s_o$ and $s_n$ yields the following

$$g(\boldsymbol{W}) \;\leq\; Q(\boldsymbol{W}, \boldsymbol{W}_m^{(k)}) \;\triangleq\; \omega_o(\boldsymbol{W}_m^{(k)})\, s_o(\boldsymbol{W}) + \omega_n(\boldsymbol{W}_m^{(k)})\, s_n(\boldsymbol{W}) + const. \tag{29}$$

$Q$ is referred to as the *majorizer* of $g$ with equality at $\boldsymbol{W} = \boldsymbol{W}_m^{(k)}$. Thus, it remains to minimize $Q(\cdot; \boldsymbol{W}_m^{(k)})$.

The implicit characterization of the solution for the Sylvester equation implies an iterative algorithm to find the solution via

$$\boldsymbol{W}_m^{(k+1)} \;=\; \arg\min_{\boldsymbol{W}} Q(\boldsymbol{W}, \boldsymbol{W}_m^{(k)}). \tag{30}$$

Thus, we have that

$$g(\boldsymbol{W}_m^{(k+1)}) \;\leq\; Q(\boldsymbol{W}, \boldsymbol{W}_m^{(k+1)}) \;\leq\; Q(\boldsymbol{W}, \boldsymbol{W}_m^{(k)}) \;=\; g(\boldsymbol{W}_m^{(k)}) \tag{31}$$

□

### A.4 Proof of Theorem 1

*Proof.* For notational ease, let $\boldsymbol{D}_o = \boldsymbol{W} - \boldsymbol{W}_o$. Thus, $\mathcal{T}(\boldsymbol{D}_o) = \mathcal{T}(\boldsymbol{W}) - \mathcal{T}(\boldsymbol{W}_o)$. The residuals are then

$$\begin{aligned} s_o(\boldsymbol{W}) &= \operatorname{tr}[L_G L_G^\top \boldsymbol{D}_o L_A L_A^\top \boldsymbol{D}_o^\top] \;=\; \|L_G^\top \boldsymbol{D}_o L_A\|_F^2 \\ &= \|U^\top L_G^\top \boldsymbol{D}_o L_A V\|_F^2 \;=\; \|\mathcal{T}(\boldsymbol{D}_o)\|_F^2 = \sum_{i,j} (\mathcal{T}(\boldsymbol{W}) - \mathcal{T}(\boldsymbol{W}_o))_{i,j}^2, \end{aligned} \tag{32}$$

where the third equality follows from property orthogonal matrices. To compute $s_n$, observe that

$$\boldsymbol{G}_n \;=\; L_G U \operatorname{diag}(p)\, U^\top L_G^\top, \qquad \boldsymbol{A}_n \;=\; L_A V \operatorname{diag}(q)\, V^\top L_A^\top. \tag{33}$$

Thus, with $\boldsymbol{D}_n = \boldsymbol{W} - \boldsymbol{W}_n$

$$\begin{aligned} s_n(\boldsymbol{W}) &= \operatorname{tr}[L_G U\ \operatorname{diag}(p)\ U^\top L_G^\top \boldsymbol{D}_n L_A V \operatorname{diag}(q)\, V^\top L_A^\top \boldsymbol{D}_n^\top L_G U] \\ &= \operatorname{tr}[\operatorname{diag}(p)\, \mathcal{T}(\boldsymbol{D}_n) \operatorname{diag}(q)\, \mathcal{T}(\boldsymbol{D}_n)^\top] \;=\; \sum_{i,j} p_i q_j [\mathcal{T}(\boldsymbol{W}) - \mathcal{T}(\boldsymbol{W}_n)]_{i,j}^2. \end{aligned} \tag{34}$$

For fixed $w_o, w_n$ the objective $Q(\mathcal{T}(\boldsymbol{W}); \mathcal{T}(\boldsymbol{W}_m^{(k)}))$ is therefore a sum of independent scalar quadratics

$$\frac{1}{2}\omega_o [\mathcal{T}(\boldsymbol{W}) - \mathcal{T}(\boldsymbol{W}_o)]_{i,j}^2 + \frac{1}{2}\omega_n p_i q_j [\mathcal{T}(\boldsymbol{W}) - \mathcal{T}(\boldsymbol{W}_n)]_{i,j}^2.$$

This sum is minimized entry-wise by the update $\mathcal{T}(\boldsymbol{W}_m)_{i,j}$ in (21). Since $\mathcal{T}$ is invertible, $\mathcal{T}^{-1}(\boldsymbol{W}_m) = L_G^{-\top} U\, \mathcal{T}(\boldsymbol{W}_m)\, V^\top L_A^{-1}$ minimizes $Q$ over $\boldsymbol{W}$. □

## B Geometric Analysis via Kronecker factorizations.

Given a layer weight $\boldsymbol{W} \in \mathbb{R}^{M \times N}$, its $MN \times MN$ Kronecker-structured precisions matrices are

$$\boldsymbol{\Lambda}_o \;=\; \boldsymbol{A}_o \otimes \boldsymbol{G}_o, \quad \boldsymbol{\Lambda}_n \;=\; \boldsymbol{A}_n \otimes \boldsymbol{G}_n. \tag{35}$$

The Cholesky factor of the full precision matrix $\boldsymbol{\Lambda}_o$ factorizes into the Cholesky factors of its constituent Kronecker factors:

$$\boldsymbol{\Lambda}_o = \boldsymbol{L}\boldsymbol{L}^\top = (\boldsymbol{L}_A \otimes \boldsymbol{L}_G)(\boldsymbol{L}_A \otimes \boldsymbol{L}_G)^\top = (\boldsymbol{L}_A \boldsymbol{L}_A^\top) \otimes (\boldsymbol{L}_G \boldsymbol{L}_G^\top) = \boldsymbol{A}_o \otimes \boldsymbol{G}_o, \tag{36}$$

Likewise $\boldsymbol{\Lambda}_n = \boldsymbol{A}_n \otimes \boldsymbol{G}_n$ also shares this structure. So the eigenproblem $\boldsymbol{L}^{-\top} \boldsymbol{\Lambda}_n \boldsymbol{L}^{-1} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top$ factors into the two smaller eigenproblems

$$\boldsymbol{L}_G^{-1} \boldsymbol{G}_n \boldsymbol{L}_G^{-\top} \;=\; \boldsymbol{U} \operatorname{diag}(\boldsymbol{p}) \boldsymbol{U}^\top, \quad \boldsymbol{L}_A^{-1} \boldsymbol{A}_n \boldsymbol{L}_A^{-\top} \;=\; \boldsymbol{V} \operatorname{diag}(\boldsymbol{q}) \boldsymbol{V}^\top. \tag{37}$$

The eigenvalues $\sigma_{ij}$ and eigenbasis $\boldsymbol{e}_{ij}$ are then computed via

$$\sigma_{ij} \;=\; \boldsymbol{p}_i \boldsymbol{q}_j, \quad \boldsymbol{e}_{ij} \;=\; \left(\boldsymbol{L}_A^{-\top} \boldsymbol{v}_j\right) \otimes \left(\boldsymbol{L}_G^{-\top} \boldsymbol{u}_i\right), \tag{38}$$

where $\boldsymbol{v}_j = \boldsymbol{V}_{:,j}$ and $\boldsymbol{u}_i = \boldsymbol{U}_{:,i}$. In the vector case, we have the discrepancy $\boldsymbol{r} = \sum_i c_i \boldsymbol{e}_i$ in the $\boldsymbol{\Lambda}_o$-orthonormal basis, thus $c_i = \boldsymbol{e}_i^\top \boldsymbol{\Lambda}_o \boldsymbol{r}$. Using (38) and $\boldsymbol{\Lambda}_o = \boldsymbol{L}\boldsymbol{L}^\top$, we have

$$\boldsymbol{e}_{ij}^\top \boldsymbol{\Lambda}_o \;=\; (\boldsymbol{v}_j \otimes \boldsymbol{u}_i)^\top \boldsymbol{L}^{-1} \boldsymbol{\Lambda}_o \;= (\boldsymbol{v}_j \otimes \boldsymbol{u}_i)^\top \boldsymbol{L}^\top. \tag{39}$$

Let $\boldsymbol{R} = \boldsymbol{W}_n - \boldsymbol{W}_o$ be the discrepancy matrix, we can find its coordinates $c_{ij}$ of in the canonical basis $\{\boldsymbol{e}_{ij}\}$ as

$$\begin{aligned} c_{ij} &= (\boldsymbol{v}_j \otimes \boldsymbol{u}_i)^\top \boldsymbol{L}^\top \operatorname{vec}(\boldsymbol{R}) = (\boldsymbol{v}_j \otimes \boldsymbol{u}_i)^\top \operatorname{vec}(\boldsymbol{L}_G^\top \boldsymbol{R} \boldsymbol{L}_A) \\ &= \boldsymbol{u}_i^\top (\boldsymbol{L}_G^\top \boldsymbol{R} \boldsymbol{L}_A) \boldsymbol{v}_j = \left(\boldsymbol{U}^\top \boldsymbol{L}_G^\top \boldsymbol{R} \boldsymbol{L}_A \boldsymbol{V}\right)_{ij} = \mathcal{T}(\boldsymbol{R})_{ij} \end{aligned} \tag{40}$$

## C Fixed Point Procedure

In this section, we provide the algorithm for COMET in Algorithm 1. In which, the iterative fixed-point solver procedure is shown in Algorithm 2. Their complexity analysis is provided in Appendix D.

**Algorithm 1** COMET

**Require:** Pre-trained model $\boldsymbol{\theta}_0$, stream of task checkpoints $\{\boldsymbol{m}_n\}$, scale $\kappa > 0$
**Ensure:** merged model $\boldsymbol{m}_m$
1: $\boldsymbol{m}_m \leftarrow \boldsymbol{\theta}_0,\ \boldsymbol{A}_o \leftarrow \mathbf{0},\ \boldsymbol{G}_o \leftarrow \mathbf{0},\ c \leftarrow 0$
2: **for** $t = 1, \ldots, T$ **do**
3: **for** each linear layer $\ell$ **do**
4: $\boldsymbol{\Delta}_n \leftarrow \boldsymbol{W}_n^{(\ell)} - \boldsymbol{W}_0^{(\ell)}$
5: $\boldsymbol{A}_n \leftarrow \boldsymbol{\Delta}_n^\top \boldsymbol{\Delta}_n,\ \boldsymbol{G}_n \leftarrow \boldsymbol{\Delta}_n \boldsymbol{\Delta}_n^\top / \|\boldsymbol{\Delta}_n\|_F^2$ // New-task factors
6: $\boldsymbol{A}_o \leftarrow \boldsymbol{A}_o + \boldsymbol{A}_n,\ c \leftarrow c+1,\ \tilde{\boldsymbol{G}}_t \leftarrow \frac{\boldsymbol{G}_o + \boldsymbol{G}_n}{c}$ // existing factors
7: // Degrees of freedom
8: $\nu_n \leftarrow \kappa\, s_{\mathrm{n}}\big(\boldsymbol{W}_o^{(\ell)}\big)$
9: $\nu_o \leftarrow \kappa\, s_{\mathrm{o}}\big(\boldsymbol{W}_n^{(\ell)}\big)$
10: $\boldsymbol{W}_m^{(\ell)} \leftarrow \mathrm{IRFP}(\boldsymbol{W}_o^{(\ell)}, \boldsymbol{W}_n^{(\ell)}, \boldsymbol{A}_o, \boldsymbol{G}_o, \boldsymbol{A}_n, \boldsymbol{G}_n, \nu_o, \nu_n)$ (Alg 2) // Iteration
11: **end for**
12: $\boldsymbol{m}_m \leftarrow$ assemble all layers $\boldsymbol{W}_m^{(\ell)}$
13: $\boldsymbol{m}_n \leftarrow \boldsymbol{m}_m,\ \boldsymbol{m}_o \leftarrow \boldsymbol{m}_n$ // Update model
14: **end for**
15: **return** $\boldsymbol{m}_m$

**Algorithm 2** Iteratively reweighted fixed point (IRFP) solver

**Require:** Previous weight $\boldsymbol{W}_o$, task weight $\boldsymbol{W}_n$, factors $\boldsymbol{A}_o, \boldsymbol{G}_o, \boldsymbol{A}_n, \boldsymbol{G}_n$, decay parameters $\nu_o, \nu_n$
**Ensure:** Updated weight $\boldsymbol{W}_m$
1: Initialise $\boldsymbol{W} \leftarrow \boldsymbol{W}_o$
2: $\boldsymbol{G}_o = \boldsymbol{L}_G \boldsymbol{L}_G^\top,\ \boldsymbol{A}_o = \boldsymbol{L}_A \boldsymbol{L}_A^\top$ // Cholesky factors
3: $\boldsymbol{L}_G^{-1} \boldsymbol{G}_n \boldsymbol{L}_G^{-\top} = \boldsymbol{U}\, \mathrm{diag}(\boldsymbol{p})\, \boldsymbol{U}^\top,\ \boldsymbol{L}_A^{-1} \boldsymbol{A}_n \boldsymbol{L}_A^{-\top} = \boldsymbol{V}\, \mathrm{diag}(\boldsymbol{q})\, \boldsymbol{V}^\top$ // Decompose
4: $\mathcal{T}(\boldsymbol{W}_o) \leftarrow \boldsymbol{U}^\top \boldsymbol{L}_G^\top \boldsymbol{W}_o \boldsymbol{L}_A \boldsymbol{V},\ \mathcal{T}(\boldsymbol{W}_n) \leftarrow \boldsymbol{U}^\top \boldsymbol{L}_G^\top \boldsymbol{W}_n \boldsymbol{L}_A \boldsymbol{V}$ // Transform
5: **repeat**
6: $s_o(\boldsymbol{W}), s_n(\boldsymbol{W})$ // Residuals
7: $\omega_o \leftarrow \nu_o / (\nu_o + s_o(\boldsymbol{W})),\ \omega_n \leftarrow \nu_n / (\nu_n + s_n(\boldsymbol{W}))$ // Responsibilities
8: **for** all entries $(i, j)$ **do**
9: $\mathcal{T}(\boldsymbol{W})_{ij} \leftarrow \dfrac{\omega_o\, \mathcal{T}(\boldsymbol{W}_o)_{ij} + \omega_n\, p_i q_j\, \mathcal{T}(\boldsymbol{W}_n)_{ij}}{\omega_o + \omega_n\, p_i q_j}$ // Scalar updates
10: **end for**
11: $\boldsymbol{W} \leftarrow \boldsymbol{L}_G^{-\top} \boldsymbol{U}\, \mathcal{T}(\boldsymbol{W})\, \boldsymbol{V}^\top \boldsymbol{L}_A^{-1}$ // Map back
12: **until** convergence
13: **return** $\boldsymbol{W}_m \leftarrow \boldsymbol{W}$

## D COMPLEXITY ANALYSIS OF COMET

**Fixed-point solver complexity.**

1. Line 2 - Cholesky factorization of $\boldsymbol{G}_o \in \mathbb{R}^{M \times M}$, $\boldsymbol{A}_o \in \mathbb{R}^{N \times N}$: $O(M^3 + N^3)$
2. Line 3 - Compute $\boldsymbol{L}_G^{-1} \boldsymbol{G}_n \boldsymbol{L}_G^{-\top}$ and $\boldsymbol{L}_A^{-1} \boldsymbol{A}_n \boldsymbol{L}_A^{-\top}$: $O(M^3 + N^3)$
3. Line 4 - Compute transformation $\mathcal{T}(\cdot)$: $O(M^2 N + M N^2)$
4. Line 5-11 (repeat $K$ times)
   - Line 6 - $O(MN)$
   - Line 8 - $O(MN)$
   - Line 11 - $O(M^2 N + M N^2)$

Thus, the per-layer complexity is $O\big(M^3 + N^3 + K(M^2 N + M N^2)\big)$.

Given $T$ tasks and $L$ layers inside the model, the total complexity of COMET is thus

$$O\Big(T\sum_{\ell=1}^{L}\big(M_\ell^3+N_\ell^3+K(M_\ell^2N_\ell+M_\ell N_\ell^2)\big)\Big).$$

## E EXPERIMENTAL DETAILS

**Vision tasks.** We fine-tune CLIP-ViT models of three different scales: ViT-B/16, ViT-B/32 and ViT-L/14 Radford et al. (2021) on 20 downstream classification tasks, following Tang et al. (2025). The vision task configurations are: **8 Tasks**: SUN397 Xiao et al. (2016), Cars Krause et al. (2013), RESISC45 Cheng et al. (2017), EUROSAT Helber et al. (2019), SVHN Netzer et al. (2011), GTSRB Stallkamp et al. (2011), MNIST LeCun (1998), DTD. **14 Tasks**: The following 6 tasks are included in addition to the 8-task benchmark: Flowers102 Nilsback & Zisserman (2008), PCAM Veeling et al. (2018), FER2013 Goodfellow et al. (2013), OxfordIIITPet Parkhi et al. (2012), STL10 Coates et al. (2011), CIFAR100 Krizhevsky et al. (2009). **20 Tasks**: Extends the 14-task benchmark with the following 6 tasks are added: CIFAR10 Krizhevsky et al. (2009), Food101 Bossard et al. (2014) [8], FashionMNIST Xiao et al. (2017), EMNIST Cohen et al. (2017), KMNIST Clanuwat et al. (2018), and SST2 Socher et al. (2013).

**Language tasks.** For NLP tasks, we evaluate the methods on the standard 8 tasks from the GLUE benchmark Wang et al. (2018): CoLA, MNLI, MRPC, QNLI, QQP, RTE, SST-2, and STSB. We use the LoRA fine-tuned checkpoints of both Flan-T5-base and Flan-T5-large, which are available via Tang et al. (2023).

**LLM tasks.** We merge three Llama-2-7B (Touvron et al., 2023) fine-tunes into a shared Llama-2-7B backbone one at a time, in the order Vicuna-7B-v1.5 (`lmsys/vicuna-7b-v1.5`), then WizardMath-7B-V1.0 (Luo et al., 2025) (`WizardLM/WizardMath-7B-V1.0`), then Llama-2-Coder-7B (Roziere et al., 2023) (`mrm8488/llama-2-coder-7b`). All checkpoints are the public Hugging Face releases, and the base is `NousResearch/Llama-2-7b-hf`. We evaluate on the 1,319-problem GSM8K test set (Cobbe et al., 2021) using vLLM. Every model gets the same prompt that WizardMath (Luo et al., 2025) was trained with.

**Hyperparameters.** Across all settings we use the same three hyperparameters: the scale multiplier $\kappa = 1$, and $K = 15$ fixed-point iterations. All of our experiments are conducted on a single NVIDIA-RTX A40 GPU with 48GB memory. All results are averaged over ten random task shuffles.

**Evaluation Metrics.** We report two standard metrics for continual model merging. Average Accuracy (ACC $\uparrow$) measures the mean performance of the final merged model across all tasks seen so far; higher values indicate better overall balance between retaining previously acquired knowledge and incorporating new tasks. Backward Transfer (BWT) quantifies the change in performance on earlier tasks after new models have been merged, thereby characterizing stability.

## F CONTINUAL MERGING BASELINES

**Continual Weight Interpolation under Gaussian view.** Both continual weight averaging and task arithmetic can be interpreted within the Gaussian product-of-experts framework. The Gaussian MAP objective admits a closed-form solution

$$\boldsymbol{m}_m \quad = \quad (\boldsymbol{\Lambda}_o+\boldsymbol{\Lambda}_n)^{-1}\Big(\boldsymbol{\Lambda}_o\boldsymbol{m}_o+\boldsymbol{\Lambda}_n\boldsymbol{m}_n\Big). \tag{41}$$

At task $t$, under isotropic Gaussian experts $p_o(\boldsymbol{\theta}) = \mathcal{N}\Big(\boldsymbol{m}_o; \frac{1}{1-t}\boldsymbol{I}\Big)$, $p_n(\boldsymbol{\theta}) = \mathcal{N}\Big(\boldsymbol{m}_n; \boldsymbol{I}\Big)$, the closed-form solution reduces to continual Weight Averaging

$$\boldsymbol{m}_m = \frac{t-1}{t}\boldsymbol{m}_o+\frac{1}{t}\boldsymbol{m}_n. \tag{42}$$

**Orthogonal Projection-based Continual Merging (OPCM) (Tang et al., 2025)** addresses interference between incoming task updates by identifying directions that conflict with previous task information and projecting the new task update onto a compatible subspace. Given pre-trained

Table 3: Effect of Kronecker factors accumulation on the 8-task CLIP ViT-B-32 benchmark. COMET accumulates the consolidated Kronecker factors $\boldsymbol{A}_o, \boldsymbol{G}_o$ across all past merge steps while comet (no acc,) retains only the new-task factors.

| Variant | Final Avg. Accuracy (%) | Backward Transfer (%) |
|---|---|---|
| COMET | **81.61** | **-10.83** |
| COMET (no acc.) | 76.82 | -17.38 |

model $\boldsymbol{m}_0$, a new task vector $\boldsymbol{\tau}_n$, the existing task vector $\boldsymbol{\tau}_o$ and the subspace spanned by previously consolidated knowledge, OPCM constructs a projected update

$$\boldsymbol{m}_m \quad = \quad \boldsymbol{m}_0 \; + \; \frac{\lambda_o \boldsymbol{\tau}_{t-1} + \mathcal{P}^o(\boldsymbol{\tau}_n)}{\lambda_n}, \tag{43}$$

where $\mathcal{P}^o$ is the projection operation to the orthogonal directions of the task vector spanning the existing subspace.

**Dual Orthogonal Projection (DOP) (Yang et al., 2026)** studies stability and plasticity of continual model merging from the lens of orthogonal projection. DOP projects the difference $\boldsymbol{r}$ between the existing model and the new model on respective data spaces and solves a resulting multi-objective optimization problem using gradient descent.

Unlike WA and TA, which combine models directly in parameter space, OPCM and DOP explicitly manipulate the geometry of task-relevant subspaces to control interference. These methods therefore provide a complementary comparison to COMET. Instead of modifying the probabilistic form of the parameter-space expert, they control continual adaptation through geometric projection.

## G Ablations and Analysis

**Efficiency Analysis.** We report average per-layer and per-task merging wall-clock time (s) for ViT-B/16 and a 7B-parameter LLM setting. At ViT scale, COMET's absolute cost is modest, averaging 0.3s per layer. At the 7B scale, COMET shows promising scalability: its per-layer merging time is lower than the recent OPCM and DOP. Notably, COMET is significantly faster than DOP, which requires many gradient steps to solve the multi-objective optimization problem. The scalability of COMET is due to the one-time factorization of the Kronecker factors, after which each iteration requires only an $O(mn)$ elementwise update in $\mathcal{T}$ space. On the other hand, OPCM requires a full SVD of the $m \times n$ weight matrix while DOP requires gradient-descent optimization to update the model. Thus, COMET's factorized computation scales more favorably for larger models.

**Effect of the Kronecker factors accumulation.** Table 3 compares the performance of COMET against the variant without accumulating the Kronecker factors, i.e. $\boldsymbol{A}_o \leftarrow \boldsymbol{A}_n$ and $\boldsymbol{G}_o \leftarrow \boldsymbol{G}_n$, when merging 8 ViT-B/32 models. Removing the accumulated existing evidence entirely degrades both the final average accuracy and backward transfer relative to full accumulation on the same fixed-order 8-task stream. Without precision accumulation, each merge step only sees the immediate task's curvature, thus the model can no longer reflect evidence from tasks further back, and previously-consolidated directions are left unprotected against being overwritten by the newest task.

**Stability and Plasticity Analysis.** We analyze the performance evolution of different continual merging methods when tasks are sequentially introduced. Fig. 6 shows the performance changes during the merging of 8 Flan-T5-base models. It can be seen that COMET exhibit stronger plasticity and stability than the baselines task arithmetic and OPCM. On vision tasks, Fig 5 shows that COMET maintains the highest average accuracy throughout the 8-task sequence on both ViT-B/16 and ViT-L/14 models, with a slower decline than OPCM and DOP. These results corroborate the advantages of COMET's adaptive tail mechanism in balancing plasticity and stability during continual merging.

## H Additional results

In this section, we provide additional empirical results and analysis. **Plasticity gap analysis.** We define the plasticity gap as the difference between the accuracy task $t$ specific model at that task and

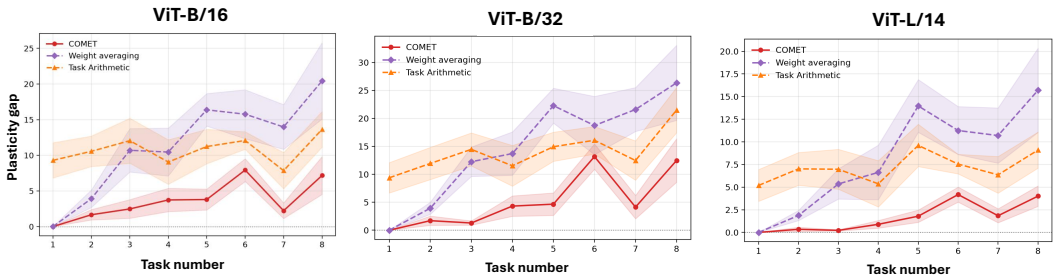


Figure 3: Plasticity gap when merging ViT models of different scales on the 8-task vision benchmarks, averaged over 10 runs for each method.

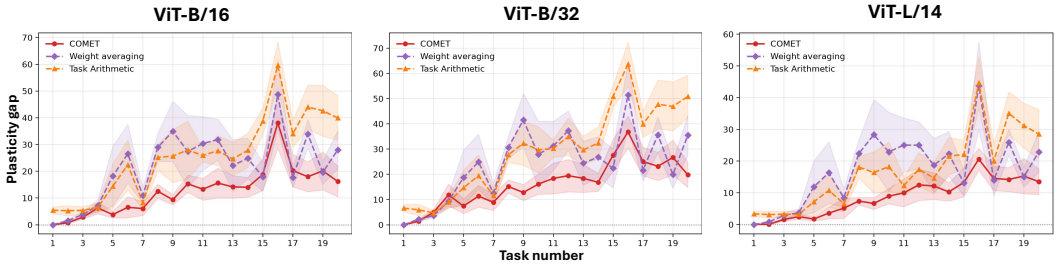


Figure 4: Plasticity gap when merging ViT models of different scales on the 20-task vision benchmarks, averaged over 10 runs for each method.

the accuracy after merging it with the consolidated model. The lower the plasticity gap, the more adaptive the model is. Fig. 3 and 4 show the plasticity gaps of COMET, weight averaging and task arithmetic when merging ViT models on 8-task and 20-task benchmarks. COMET exhibits much lower plasticity gaps than the baselines thanks to its adaptive tail mechanism, which in turn balances plasticity and stability.

**Flan models merging results.** Table 4 show COMET outperforms the baselines on the 8-task GLUE benchmark using both Flan-T5-base and Flan-T5-large model families. Fig. 6 further shows that COMET achieves better stability and plasticity than Task Arithmetic and OPCM, further showing the effectiveness of the tail adaptive mechanism.

**Additional vision results.** We also compare the average accuracy after merging each task for ViT-B/16 and ViT-L/14 on the 8 task suite. Across both models, COMET maintains the highest accuracy over seen tasks as the number of tasks increases. The advantage is particularly pronounced for ViT-B/16, where COMET degrades more gradually than OPCM and DOP. A similar trend is observed for ViT-L/14. Although COMET and DOP become comparable after the last task, COMET's performance remains above DOP throughout the task sequence and outperforms OPCM. Overall, the results indicate that COMET is more robust to the accumulation of multiple task-specific models, exhibiting less degradation in average seen-task accuracy as the number of merged tasks grows.

Table 4: Final average GLUE score and backward transfer (BWT) obtained by continually merging LoRA-adapted Flan-T5-base and Flan-T5-large models across all eight GLUE tasks. Results are averaged over 10 random task orderings (shuffles).

| Method | Flan-T5-base Score / BWT | Flan-T5-large Score / BWT |
|---|---|---|
| Task Arithmetic | 76.50 ± 0.22 / -2.98 ± 0.47 | 85.94 ± 0.02 / -0.39 ± 0.21 |
| OPCM | 77.27 ± 0.10 / -3.99 ± 0.31 | 86.16 ± 0.12 / -0.81 ± 0.53 |
| DOP | 78.09 ±0.41 / -4.62 ± 0.60 | 86.50 ± 0.16 / -0.70 ± 0.25 |
| COMET (Ours) | **78.69** ± **0.52** / -4.56 ± 0.92 | **86.84** ± **0.14**/ -1.20 ± 0.29 |

Table 5: Average merging time per merge and per layer for different methods on LLM (7B) and ViT-B/16 models.

| Method | LLM (7B) | | ViT-B/16 | |
|---|---|---|---|---|
| | Avg per merge | Avg per layer | Avg per merge | Avg per layer |
| Task Arithmetic | 8.4 s | 0.029 s | 0.018 s | 0.00012 s |
| OPCM | 1529.7 s | 5.26 s | 7.66 s | 0.0504 s |
| DOP | 11359.1 s | 39.03 s | 49.9 s | 0.328 s |
| COMET (Ours) | 1166.6 s | 4.01 s | 61.5 s | 0.404 s |

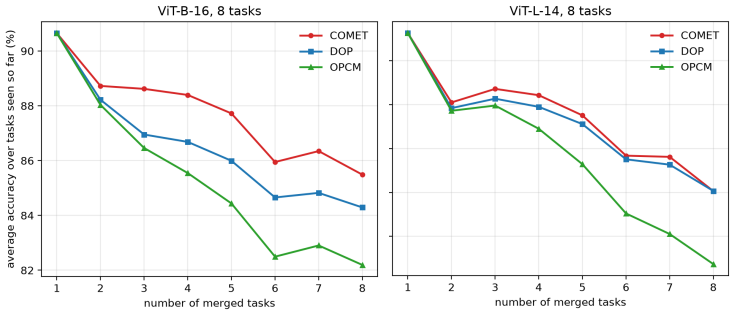


Figure 5: Average accuracy curves for COMET, OPCM and DOP after each task during the continual merging procedure.

# I Related Work

**Model merging** aims to combine multiple task-specific models into one multi-task model in the parameter space. Weight interpolation merging aims to locate optimal merging coefficients to the expert models and linearly combine them to obtain the merged model (Wortsman et al., 2022; Ilharco et al., 2022). Weight Averaging (Wortsman et al., 2022) is the most straightforward but can hinder performance. Task Arithmetic methods (Ilharco et al., 2022; Yu et al., 2024; Yadav et al., 2023) merge models via task vectors and use grid search to find the optimal coefficients to the task vectors. Others merging methods leverage statistical information from the data to guide the merging process. Fisher merging (Matena & Raffel, 2022) weights the contribution of each parameter using the curvature information obtained from task loss gradients. RegMean (Jin et al., 2022) use Gram matrix built from the data to obtain a close-form solution of the merge. However, these methods require access to the underlying data used to train the model, which is not available in the continual merging setting. Subspace-based merging methods identify sparse or low-rank subspaces of the weights to reduce parameter interference during the merging process. DARE (Yu et al., 2024) randomly prunes a fraction of parameters while rescaling the others. KNOTS (Stoica et al., 2024), STAT (Lee et al., 2025) and TSV (Gargiulo et al., 2025) utilize SVD to locate common and specific subspace in weight matrices amenable for merging.

**Continual model merging** considers the setting where the task-specific models arrive sequentially and the merge must be performed on demand. TIME (Dziadzio et al., 2025) conducts an empirical performance analysis of merging methods for multimodal models in the continual setting and show that traditional model merging methods struggle in the streaming configurations. OPCM (Tang et al., 2025) projects the incoming model onto the subspace spanned by the consolidated model to reduce interference. DOP (Yang et al., 2026) exploits the geometry of data, gradients and task vector subspaces and use orthogonal projections to balance stability and plasticity. While these projection-based methods have shown promising performance, they are purely geometric while our COMET provides a principled probabilistic design for continual merging.

**Continual learning with model merging** explores model merging as a mechanism for continual learning, where a single set of weights is maintained and updated as new task checkpoints become available via sequential fine-tuning. CoMA (Marouf et al., 2024) performs sequential weight aver-

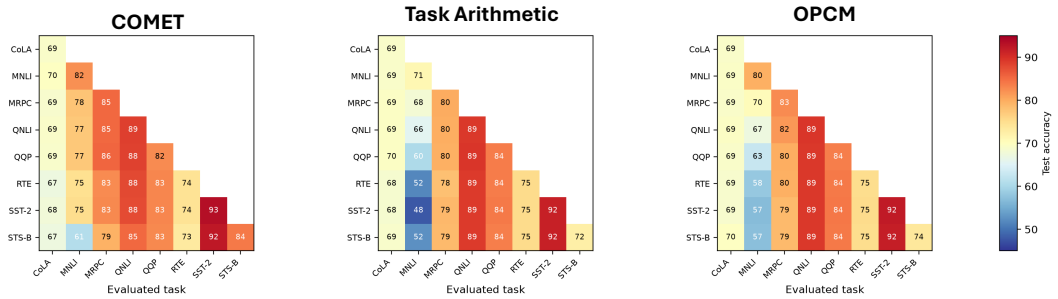


Figure 6: Performance evolution over the sequential merging of eight GLUE tasks using the Flan-T5-base architecture. Each row $t$ reports the test accuracy on previously merged tasks $(1-t)$ after incorporating the $t$-th model, measuring stability, while the diagonal entries represent plasticity, i.e., performance on the newly incorporated task.

aging between the current consolidated model and the newly fine-tuned specialist, while its Fisher-weighted variant CoFiMA replaces uniform averaging by a diagonal Fisher precision, emphasizing parameters that are important for previous tasks. BECAME (Li et al., 2025) formalizes the merge as a Bayesian MAP problem under a Laplace approximation (Ritter et al., 2018) and derives a closed-form adaptive mixing coefficient. However, these methods are developed for data-aware continual adaptation where the data is available for sequential fine-tuning and thus not directly applicable to the data-free continual model merging setting.