INCREMENTAL SPECTRAL MERGING WITH PRE-TRAINED MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite much recent progress, continual learning remains unsolved. Every update to a model's weights risks erasing what the model already knows. Model merging is a promising way to address this challenge. Treating each task update as a separate model, merging combines multiple models into one without retraining and without storing data or extra models. Even simple operators such as averaging or maximum-magnitude selection work well in practice. However, there is little theory explaining why the merged model performs well in continual learning. In this paper, we offer a new perspective on model merging for continual learning. Instead of merging all parameters of the low-rank modules used for fine-tuning, we merge only the dominant singular directions of each update, scaled by a damping factor. Our theoretical analysis bounds the loss of the merged model by two terms: the change of the classification head and the weight step between consecutive modules. Spectral merging sets the second term through its damping factor, and realigning the classifier after the merge keeps the first small. Our experiments show that keeping the dominant low-rank information is key to mitigating forgetting. Across four class-incremental benchmarks, our method ranks first in average final accuracy and in robustness to input corruptions, ahead of both prior continual learning methods and published merging rules.

1 INTRODUCTION

A deep learning model that keeps improving needs the ability to learn continually: to acquire new knowledge without forgetting what it has already learned (McCloskey & Cohen, 1989; Parisi et al., 2019; De Lange et al., 2022). Recently, continual fine-tuning of a pre-trained model has become one of the dominant approaches (Wang et al., 2022b; Zhou et al., 2024b). This direction follows the scaling trend in deep learning, where a large model provides robust feature representations (Brown et al., 2020; Dosovitskiy et al., 2021). Unlike a model trained from scratch, whose parameters are all updated on each task, a pre-trained model is usually fine-tuned through much smaller modules, known as parameter-efficient fine-tuning (PEFT) (Houlsby et al., 2019), such as LoRA (Hu et al., 2022).

In class-incremental learning, the incoming knowledge is divided into separate sessions or tasks with disjoint sets of classes. The samples within a task are assumed to be independent and identically distributed, while the data distribution shifts from one task to the next (van de Ven et al., 2022). Under this setting, each task is given its own PEFT module (Liang & Li, 2024; Zhou et al., 2024c). A natural question is how to combine these modules into a single model, so that it acts as one model at inference time. Moreover, since the number of tasks is never known in advance, the model should keep only a limited number of modules. Sequential merging is a promising answer to both concerns. First proposed for combining multiple task models (Wortsman et al., 2022; Matena & Raffel, 2022; Ilharco et al., 2023), model merging studies how to fuse many task models into one model that performs well on every task. Recently, even a simple operator such as maximum-magnitude selection was shown to reduce forgetting in continual learning (Marczak et al., 2024).

Most merging rules, however, were designed for a different problem: combining a fixed set of experts, each fine-tuned independently from the same pre-trained model (Ilharco et al., 2023; Yadav et al., 2023; Yu et al., 2024b; Jang et al., 2024; Stoica et al., 2025). They merge in one shot, so every task vector must be available at merge time, and most of them act entry by entry. Applied to a task

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.

sequence, each rule reduces to a choice of how far to move toward the new module or which of its entries to keep. Both choices are heuristic. Selecting entries by magnitude treats every weight on its own and ignores the low-rank structure of the update, how to set the merging coefficients is still an open question (Yadav et al., 2023; Yang et al., 2024), and little analysis explains why the merged model keeps what it has learned.

We propose *Incremental Spectral Merging* (ISM), a merging rule made for a task sequence rather than for a fixed set of experts. Instead of selecting entries, it selects directions. After each task, we take the update of the newly trained LoRA module relative to the deployed module, keep only its dominant singular directions, which hold most of the energy of the update, and add them to the deployed module with a damping factor. Because the merge moves the feature space, ISM then transports the stored statistics of the old classes to the new space and realigns the classifier on them. We show that the loss of the merged model is bounded by the change of the classification head and by the weight step between consecutive modules, a distance the damping factor sets: the damped merge keeps the second term small, and the realigned head keeps the first small. Our experiments show that keeping only the dominant singular directions of the update mitigates forgetting while retaining the information added by the new task. In summary, our contributions are:

- **Spectral merging.** An incremental merging rule for sequentially fine-tuned LoRA modules that keeps only the dominant singular directions of each update. It merges two modules at a time in closed form, so the memory cost stays constant as tasks arrive and no data is needed at merge time.
- **A theoretical analysis.** We bound the loss of the merged model by the change of the classification head and the weight step between consecutive modules, which explains why a small, damped step, followed by a realigned head, preserves the model's knowledge.
- **Extensive experiments.** On four class-incremental benchmarks (CIFAR-100, CUB-200, ImageNet-R, and Stanford Cars), ISM achieves the highest average final accuracy and average incremental accuracy of all compared rehearsal-free methods, and the best results on ImageNet-R and Stanford Cars, with a single merged module and constant memory. In a controlled comparison where only the merge rule changes, it outperforms every published merging rule in final accuracy on every benchmark. It is also the most robust to input corruptions, ahead of both prior continual learning methods and other merging rules.

2 Related Work

Continual learning with pre-trained models. Rehearsal-free class-incremental learning now starts from a frozen pre-trained model and adapts it with small modules, usually one per task, so the modules must act as one model at test time. One option is to keep every module and route among them: MoE-Adapters (Yu et al., 2024a) trains a router per task over a pool of LoRA experts and selects one by an autoencoder score; CL-LoRA (He et al., 2025) runs a shared adapter and every task-specific adapter at test time; DIA (Li et al., 2025a) mixes the outputs of all task adapters at each patch token. Another option is to fold the updates into one model while keeping per-task pieces: O-LoRA (Wang et al., 2023) trains each new LoRA orthogonal to the earlier ones and sums them; SD-LoRA (Wu et al., 2025) freezes one direction per task and re-learns only their scalar magnitudes; Share (Kaushik et al., 2026) keeps a shared SVD basis with a per-task coefficient block chosen by task identity. In all of these, memory or inference cost grows with the number of tasks, or a selection step or the task identity is needed at test time. We deploy a single LoRA module of fixed size and classify in one forward pass without task identity.

Model merging for continual learning. Model merging combines several fine-tuned models into one without retraining. It builds on weight averaging (Wortsman et al., 2022; Matena & Raffel, 2022) and task vectors (Ilharco et al., 2023); later rules reduce interference by selecting entries by magnitude, sign, or random drop (Yadav et al., 2023; Yu et al., 2024b; Deep et al., 2024), by interpolating toward the initialisation (Jang et al., 2024), by learning coefficients on unlabeled test data (Yang et al., 2024), or by aligning LoRA updates in a shared SVD basis first (Stoica et al., 2025). These rules are one-shot. Several of them are the baselines of our merge-rule comparison (Table 6). Merging inside a task sequence removes this need. MagMax (Marczak et al., 2024) fine-tunes the whole encoder task after task and keeps, per entry, the largest-magnitude value across task vectors.

The solution of sequential training is linearly connected to the multitask solution when both start from the same initialisation (Mirzadeh et al., 2021), and Hess et al. (2025) show that merging sequentially trained models works better than merging models trained in parallel; both support training each module from the previous one, as we do. ACMap (Fukuda et al., 2025) averages task adapters with weight $1/t$ and shifts stored prototypes by a centroid difference; GLAM (Testa et al., 2026) prunes each task LoRA by entry magnitude and averages it into one of a few group adapters, whose ranks add up at inference. Others set a scalar coefficient: DESIRE (Guo et al., 2024) mixes the previous and current LoRA with per-layer weights fitted on unlabeled test data; P&M (Qiu et al., 2025) derives the coefficient from the stored optimum and Fisher of every past task; BECAME (Li et al., 2025c) merges a stable and a plastic solution of the same task in the task-incremental setting. Closest to ours, SLAO (Qiao & Mahdavi, 2026) keeps a single LoRA and only two modules, merges the $\boldsymbol{B}$ factor with a fixed $1/\sqrt{t}$ weight, overwrites $\boldsymbol{A}$, and bounds forgetting through the neural tangent kernel, on language tasks. All of these rules decide how far to move along the update or which entries to keep; none decides which directions of the update to keep. Our rule makes that choice from the singular values of the update and is followed by a correction of the old-class statistics that the merge shifts.

Spectral merging. A recent line of work merges in the singular basis of each layer rather than entry by entry. TSV-M (Gargiulo et al., 2025) keeps a fixed number of singular components per task and whitens the joint basis; Iso-C (Marczak et al., 2025) shows that the alignment between task and merged singular subspaces predicts merge quality and flattens the spectrum of the summed task vectors; STAR (Lee et al., 2025) truncates each task vector by a fraction of its nuclear norm and rescales the rest; SVC (Li et al., 2026b) shows that summing task vectors over-counts shared directions and rescales the singular values of the merged result. All four merge every task vector at once, so adding a task means redoing the merge. OPCM (Tang et al., 2025) and NUFILT (Qiu et al., 2026) merge independently trained experts one at a time with constant memory by projecting each new task vector away from the leading directions of the accumulated update, but they assume experts trained from the same base rather than a class-incremental stream with a growing classifier. Our merge is also incremental and spectral, but it keeps the leading directions of the new update instead of removing the part of it that overlaps the accumulated update. We take the update of the new module relative to the deployed module, retain the smallest set of its leading singular components that holds a fixed fraction of its energy, and add this part scaled by one damping factor. Our analysis (Section 3.5) ties the loss of the merged module to the spectral distance between consecutive modules.

3 Incremental Spectral Merging (ISM)

This section presents ISM. Section 3.1 states the problem and the constraints on memory, data, and inference. Section 3.2 introduces the merge: the module of each task is fine-tuned from the previous one and folded into the deployed module by a damped, spectrally truncated update. Sections 3.3 and 3.4 handle what the merge changes downstream, the statistics of the old classes and the classifier. Section 3.5 gives the bound that motivates the design.

3.1 Problem setup

We study class-incremental learning over a sequence of T tasks. Task t provides a training set $\mathbb{D}_t = \{(\boldsymbol{x}_i, y_i)\}$ whose labels belong to a class set $\mathbb{Y}_t$, and the class sets of different tasks are disjoint. After task t, the model must classify inputs from all classes seen so far, $\mathbb{Y}_{1:t} = \bigcup_{s \le t} \mathbb{Y}_s$, without being told which task an input comes from.

The model is a pre-trained vision transformer whose weights are frozen. It is adapted with a LoRA module (Hu et al., 2022): in each adapted layer, a low-rank product $\boldsymbol{BA}$ is added to the frozen weight. We write $\boldsymbol{W}_t$ for the LoRA module obtained after training on task t, $\boldsymbol{W}_0$ for the module before any task, and $f(\cdot\,; \boldsymbol{W})$ for the feature extractor equipped with module $\boldsymbol{W}$, which maps an input to a feature in $\mathbb{R}^d$. A linear head h_t maps features to logits over $\mathbb{Y}_{1:t}$. The deployed model after task t is the pair $(\boldsymbol{M}_t, h_t)$, where $\boldsymbol{M}_t$ is a single merged module with the same size as one LoRA module. Throughout, we require that (i) memory does not grow with T: only the previous merged module and the current trained module are kept; (ii) no data from past tasks is stored; and (iii) inference is a single forward pass through one model.

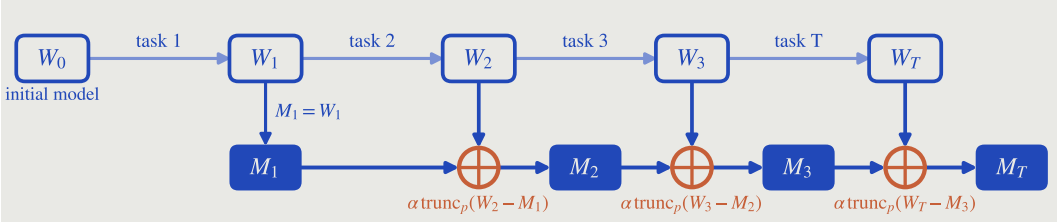


Figure 1: Incremental spectral merging (ISM). Top: the LoRA module is fine-tuned task after task, each time starting from the previous trained module. Bottom: the deployed module $\boldsymbol{M}_t$ is updated by merging only the previous merged module and the new trained module. Each junction adds the damped, energy-truncated update $\alpha\,\mathrm{trunc}_p(\boldsymbol{W}_t - \boldsymbol{M}_{t-1})$. Shown at the weight level.

Each task is processed in four steps (Figure 1 and Algorithm 1). We first fine-tune the LoRA module on the new task (Section 3.2), then merge it into the deployed module with spectral merging (Section 3.2). Because the merge changes the feature space, we compensate the drift of old-class statistics (Section 3.3) and finally align the classifier to the merged extractor (Section 3.4).

3.2 Spectral merging

Sequential fine-tuning. For task t we train one LoRA module with plain cross-entropy on $\mathbb{D}_t$ only, together with the head entries of the new classes. The module is initialised from the module trained on the previous task, $\boldsymbol{W}_{t-1}$, rather than from a fresh initialisation. This choice matters for merging: modules trained one after another from the same starting point stay close to each other in weight space (Mirzadeh et al., 2021), and merging such modules preserves shared knowledge better than merging modules trained independently (Hess et al., 2025).

Incremental merge. Let $\boldsymbol{M}_{t-1}$ be the deployed module before task t and $\boldsymbol{W}_t$ the module trained on task t, and define the update $\boldsymbol{\Delta} = \boldsymbol{W}_t - \boldsymbol{M}_{t-1}$. To obtain a spectral-aware merging, we first compute the singular value decomposition of the update $\boldsymbol{\Delta} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top$, where $\boldsymbol{S}$ is the diagonal matrix of the singular values in decreasing order, $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$.

Instead of merging the full update $\boldsymbol{\Delta}$, we retain only its leading singular components. Given an energy fraction $p \in (0, 1]$, let k be the smallest index whose leading singular values capture a fraction p of the total energy of the update,

$$k = \min\Big\{k' : \sum_{i \leq k'} \sigma_i^2 \geq p \sum_i \sigma_i^2\Big\}, \qquad \boldsymbol{\Delta}_k = \boldsymbol{U}_k \boldsymbol{S}_k \boldsymbol{V}_k^\top, \tag{1}$$

where $\boldsymbol{U}_k$, $\boldsymbol{S}_k$, and $\boldsymbol{V}_k$ contain the first k singular vectors and singular values; we write $\mathrm{trunc}_p(\boldsymbol{\Delta}) = \boldsymbol{\Delta}_k$ for this truncation. The merged module is then defined as

$$\boldsymbol{M}_t = \boldsymbol{M}_{t-1} + \alpha \boldsymbol{\Delta}_k, \qquad 0 < \alpha \leq 1, \tag{2}$$

with $\boldsymbol{M}_1 = \boldsymbol{W}_1$ for the first task. This merge incorporates the dominant low-rank information from $\boldsymbol{W}_t$ while controlling the largest directional deviation from $\boldsymbol{M}_{t-1}$. Since $\|\boldsymbol{M}_t - \boldsymbol{M}_{t-1}\|_2 = \|\alpha \boldsymbol{\Delta}_k\|_2 = \alpha \|\boldsymbol{\Delta}_k\|_2 = \alpha\sigma_1$, the damping α scales every kept direction, so it sets the spectral deviation $\alpha\sigma_1$. The energy fraction p decides how many directions are kept: it removes the small ones, which reduces how much the module moves in total.

3.3 Drift compensation

Merging replaces the feature extractor $f(\cdot\,; \boldsymbol{M}_{t-1})$ by $f(\cdot\,; \boldsymbol{M}_t)$. The classifier, however, relies on per-class feature statistics of the old classes, which were estimated under earlier extractors and no longer describe where those classes lie. Since no old data is stored, the statistics cannot be recomputed. Following drift compensation methods (Yu et al., 2020; Gomez-Villa et al., 2024; Rao et al., 2026), we estimate the change of the feature space from the current task, whose data is available under both extractors, and apply it to the old classes (Algorithm 1).

Algorithm 1 Incremental Spectral Merging (ISM)

Require: frozen backbone with initial LoRA module $\boldsymbol{W}_0$; tasks $\mathbb{D}_1, \ldots, \mathbb{D}_T$; energy fraction p; damping α
Ensure: deployed pair $(\boldsymbol{M}_T, h_T)$
$\boldsymbol{W}_1 \leftarrow$ fine-tune $\boldsymbol{W}_0$ on $\mathbb{D}_1$; $\boldsymbol{M}_1 \leftarrow \boldsymbol{W}_1$
$(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \leftarrow$ mean and covariance of $f(\mathbb{D}_1; \boldsymbol{M}_1)$, $c \in \mathbb{Y}_1$
for $t = 2, \ldots, T$ **do**
  $\boldsymbol{W}_t \leftarrow$ fine-tune $\boldsymbol{W}_{t-1}$ on $\mathbb{D}_t$ ▷ Sequential fine-tuning, Section 3.2
  $\boldsymbol{M}_t \leftarrow \boldsymbol{M}_{t-1} + \alpha \operatorname{trunc}_p(\boldsymbol{W}_t - \boldsymbol{M}_{t-1})$ ▷ Spectral merging, Section 3.2
  $\boldsymbol{T}_t \leftarrow$ ridge map on $\mathbb{D}_t$ from $f(\cdot\,; \boldsymbol{M}_{t-1})$ to $f(\cdot\,; \boldsymbol{M}_t)$ ▷ Drift compensation, Section 3.3
  $(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \leftarrow (\boldsymbol{T}_t \boldsymbol{\mu}_c, \boldsymbol{T}_t \boldsymbol{\Sigma}_c \boldsymbol{T}_t^\top)$, $c \in \mathbb{Y}_{1:t-1}$
  $(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \leftarrow$ mean and covariance of $f(\mathbb{D}_t; \boldsymbol{M}_t)$, $c \in \mathbb{Y}_t$ ▷ Classifier alignment, Section 3.4
  $h_t \leftarrow$ LCA on $\mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$, $c \in \mathbb{Y}_{1:t}$
  discard $\boldsymbol{W}_{t-1}$ and $\boldsymbol{M}_{t-1}$; deploy $(\boldsymbol{M}_t, h_t)$
end for

Let $\boldsymbol{F}_{t-1}, \boldsymbol{F}_t \in \mathbb{R}^{n \times d}$ stack the features of the n training inputs of task t under $\boldsymbol{M}_{t-1}$ and $\boldsymbol{M}_t$, one input per row. We fit a linear map $\boldsymbol{T}_t \in \mathbb{R}^{d \times d}$ by ridge regression anchored at the identity,

$$\boldsymbol{T}_t = \arg\min_{\boldsymbol{T}} \; \|\boldsymbol{F}_{t-1}\boldsymbol{T}^\top - \boldsymbol{F}_t\|_F^2 + \lambda \|\boldsymbol{T} - \boldsymbol{I}\|_F^2, \tag{3}$$
$$\boldsymbol{T}_t = \boldsymbol{I} + (\boldsymbol{F}_t - \boldsymbol{F}_{t-1})^\top \boldsymbol{F}_{t-1} \big(\boldsymbol{F}_{t-1}^\top \boldsymbol{F}_{t-1} + \lambda \boldsymbol{I}\big)^{-1}.$$

The identity anchor is deliberate. Anchoring at $\boldsymbol{I}$ keeps every direction in which the current task gives no evidence of drift unchanged, so the extrapolation to old classes is conservative. The mean and covariance of each old class $c \in \mathbb{Y}_{1:t-1}$ are then transported as

$$\boldsymbol{\mu}_c \leftarrow \boldsymbol{T}_t \boldsymbol{\mu}_c, \qquad \boldsymbol{\Sigma}_c \leftarrow \boldsymbol{T}_t \boldsymbol{\Sigma}_c \boldsymbol{T}_t^\top. \tag{4}$$

This differs from Rao et al. (2026), who regularise the map toward zero and additionally distil features during training. We use no distillation; the merge itself keeps consecutive extractors close, and the identity-anchored map corrects what remains.

3.4 Classifier alignment

After drift compensation, every seen class has a Gaussian model of its features under the current extractor: old classes through Eq. 4, new classes from the empirical mean and covariance of $\boldsymbol{F}_t$. We then align the head with local classifier alignment (LCA) (Tran et al., 2026). With the extractor frozen, LCA retrains only the head on pseudo-features $\hat{\boldsymbol{z}} \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$ sampled from these Gaussians, using cross-entropy plus a local term that penalises large loss differences between two draws from the same class. The local term keeps the decision boundaries robust to the mismatch between the Gaussian model and the true feature distribution. We follow the sampling scheme and loss of Tran et al. (2026) without change. The result is the deployed pair $(\boldsymbol{M}_t, h_t)$.

3.5 Theoretical analysis for ISM

ISM is inspired by the following theoretical result, whose proof appears in Appendix A.

Theorem 1. *Let $g_t = h_t \circ f_t$ be the overall model after training on task t, where f_t is the feature extractor and h_t is the classification head. For the neural network f_t, let $f_t^k(\boldsymbol{x})$ be the output of layer k for a given input $\boldsymbol{x}$, and let $\boldsymbol{W}_t^k$ be the weight matrix mapping $f_t^k(\boldsymbol{x})$ to the input of layer $k+1$. For any input distribution P_x and loss function ℓ, the expected loss of g_t is defined as $L(g_t) = \mathbb{E}_{\boldsymbol{x} \sim P_x} \ell(g_t(\boldsymbol{x}))$. Denote by P_t the pushforward of P_x through the backbone f_t and let $C_k = \mathbb{E}_{\boldsymbol{x} \sim P_x} \|f_{t+1}^k(\boldsymbol{x})\|$.*

1. *If ℓ is L_ℓ-Lipschitz and h_{t+1} is L_h-Lipschitz, then*

$$L(g_{t+1}) \le L(g_t) + \mathop{\mathbb{E}}_{\boldsymbol{z} \sim P_t} [\ell(h_{t+1}(\boldsymbol{z})) - \ell(h_t(\boldsymbol{z}))] + L_\ell L_h \mathop{\mathbb{E}}_{\boldsymbol{x} \sim P_x} \|f_{t+1}(\boldsymbol{x}) - f_t(\boldsymbol{x})\|. \tag{5}$$

2. *Suppose further that f_{t+1} keeps the portion of f_t after layer k frozen and that this common portion is $L_{f,k}$-Lipschitz continuous with respect to the input of layer $k+1$. Then*

$$\mathop{\mathbb{E}}_{\boldsymbol{x}\sim P_x}\|f_{t+1}(\boldsymbol{x})-f_t(\boldsymbol{x})\| \le L_{f,k}C_k\|\boldsymbol{W}_{t+1}^k-\boldsymbol{W}_t^k\|+L_{f,k}\|\boldsymbol{W}_t^k\|\mathop{\mathbb{E}}_{\boldsymbol{x}\sim P_x}\|f_{t+1}^k(\boldsymbol{x})-f_t^k(\boldsymbol{x})\|. \quad (6)$$

The assumptions in Theorem 1 are practical for model adaptation methods that update only part of a pretrained backbone while keeping the portion after layer k frozen, with LoRA being one example. The Lipschitz conditions are also reasonable when the loss, classification head, and frozen layers use Lipschitz operations. In particular, under compatible norms, the Lipschitz constants of neural network components can be bounded using the operator norms of their weight matrices and the Lipschitz constants of their activation functions, such as sigmoid and ReLU. The theorem shows that the change in expected loss is bounded in terms of the change in the classification head, the weight update at layer k, and the change in the intermediate representations. Therefore, restricting parameter updates, stabilizing intermediate representations, and controlling operator norms offer a general mechanism for limiting performance degradation on the old tasks.

Connection to ISM: Theorem 1 provides theoretical support for the design of ISM by relating its main components to the terms in the loss bound. After fine-tuning on task $t+1$, spectral merging aims to retain the knowledge acquired from the new task while introducing only limited changes to f_t, thereby controlling $\|\boldsymbol{W}_{t+1}^k - \boldsymbol{W}_t^k\|$. Meanwhile, the drift-compensation step transports the statistics of the old classes to the representation space of f_{t+1}, and the LCA alignment step retrains the classification head on these statistics. Together they aim to keep $\mathbb{E}_{\boldsymbol{z}\sim P_t}[\ell(h_{t+1}(\boldsymbol{z})) - \ell(h_t(\boldsymbol{z}))]$ small, thereby limiting the increase in loss induced by updating the classification head on representations associated with the previous task. Consequently, the main components of ISM target the principal terms in the upper bound on $L(g_{t+1})$, providing a theoretical justification for its ability to preserve performance on previous tasks while adapting to a new task.

4 Experiments

We evaluate ISM on four class-incremental benchmarks against seventeen rehearsal-free methods (Section 4.2) and on two domain-incremental benchmarks (Section 4.3), test its robustness to input corruptions (Section 4.4), and then study the choices of the merge (Section 4.5). The comparison with published merge rules, in accuracy and robustness, and the contribution of each stage of ISM are in Appendices C and D.

4.1 Experimental setup

Datasets and protocol. We evaluate on four class-incremental benchmarks that are standard for pre-trained vision transformers: CIFAR-100 (Krizhevsky, 2009), CUB-200 (Wah et al., 2011), ImageNet-R (Hendrycks et al., 2021), and Stanford Cars (Krause et al., 2013). Each dataset is split into 10 tasks with disjoint classes: 10 classes per task for CIFAR-100, 20 classes per task for CUB-200 and ImageNet-R, and 16 classes in the first task followed by 20 classes per task for Stanford Cars. The class order is shuffled with the random seed. All images are resized to 224×224; training uses random resized crops and horizontal flips, and testing uses a resize and center crop. No exemplars are stored at any point.

Metrics. Let $a_{i,j}$ be the accuracy on the classes of task j after training on task i. We report the final accuracy $\mathrm{FA} = \frac{1}{T}\sum_{j=1}^{T} a_{T,j}$, that is, the class-balanced accuracy over all classes after the last task, and the average incremental accuracy $\mathrm{AA} = \frac{1}{T}\sum_{i=1}^{T}\frac{1}{i}\sum_{j\le i} a_{i,j}$, which averages the accuracy over all seen classes after every task. Higher is better for both. Our results are the mean and standard deviation over three seeds (1993, 1994, 1995).

Implementation details. The backbone is a ViT-B/16 pre-trained on ImageNet-21K and fine-tuned on ImageNet-1K (Dosovitskiy et al., 2021; Steiner et al., 2022), kept frozen throughout, with one LoRA module attached to every transformer block. Training, merging, drift, and alignment settings, together with the per-dataset hyperparameters, are listed in Appendix B.

Table 1: Class-incremental results on four benchmarks with 10 tasks each (mean ± std over three seeds). FA: final accuracy; AA: average incremental accuracy; Avg: mean over the four benchmarks. Baseline numbers are protocol-matched published results or re-runs (10 tasks, ViT-B/16 pre-trained on ImageNet-21K). Best continual method in bold.

	CIFAR-100		CUB-200		ImageNet-R		Stanford Cars		Avg	
Method	FA	AA	FA	AA	FA	AA	FA	AA	FA	AA
Joint training	93.48±0.04	–	89.88±0.33	–	87.11±0.25	–	89.19±0.29	–	89.92	–
L2P	82.76±1.17	88.48±0.83	62.21±1.92	73.83±1.67	66.49±0.40	72.83±0.56	38.18±2.33	51.79±4.19	62.41	71.73
DualPrompt	85.56±0.33	90.33±0.33	66.00±0.57	77.92±0.50	68.50±0.52	72.59±0.24	40.14±2.36	56.74±1.78	65.05	74.40
CODA-Prompt	87.00±0.31	90.68±1.02	77.23±1.12	81.90±0.85	72.82±0.50	78.13±0.52	44.89±0.61	58.91±0.37	70.49	77.41
SLCA	91.53±0.28	94.09±0.87	84.71±0.40	90.94±0.68	77.00±0.33	81.17±0.64	67.73±0.85	76.93±1.21	80.24	85.78
LAE	85.59±0.46	89.96±0.44	77.48±0.94	85.83±0.68	72.66±0.63	78.91±0.89	52.47±1.46	64.08±1.01	72.05	79.70
RanPAC	91.09±0.25	94.03±0.58	87.88±0.53	92.57±0.55	75.28±0.14	80.66±0.58	61.43±0.84	72.36±1.18	78.92	84.91
APER	87.32±0.28	91.21±1.35	87.04±0.13	92.13±0.27	65.79±0.98	72.42±1.41	37.65±0.09	49.22±1.66	69.45	76.25
CoMA	92.00±0.13	94.12±0.63	85.95±0.29	90.75±0.39	77.47±0.05	81.32±0.17	73.35±0.59	78.55±0.42	82.19	86.19
CoFiMA	**92.77**±0.24	94.89±0.94	87.11±0.56	91.87±0.69	78.25±0.26	81.48±0.56	76.96±0.64	82.65±0.96	83.77	87.72
EASE	86.54±0.29	91.68±0.18	79.90±0.29	86.65±0.63	77.75±0.18	83.87±0.78	35.46±1.50	48.96±0.96	69.91	77.79
SSIAT	91.51±0.20	94.29±0.88	88.32±0.48	**92.95**±0.55	79.61±0.40	83.71±0.42	67.43±1.47	75.53±1.26	81.72	86.62
SLCA++	91.46±0.18	94.20±0.71	86.59±0.29	91.63±0.72	78.09±0.22	82.95±0.78	73.97±0.22	79.49±0.80	82.53	87.07
MOS	90.53±0.35	94.09±0.91	**89.91**±0.25	92.87±0.56	78.10±0.12	83.61±0.40	67.76±1.28	74.38±1.13	81.58	86.24
TUNA	91.79±0.15	94.88±0.06	88.40±0.42	92.01±0.68	79.44±0.38	84.80±0.37	69.46±0.35	76.95±0.39	82.27	87.16
SLDC	91.48±0.24	94.38±0.69	87.15±0.50	92.38±0.57	80.00±0.29	84.01±0.46	80.50±0.30	85.45±0.41	84.78	89.06
LCA	88.25±0.47	92.96±0.18	80.55±0.36	88.08±0.73	80.47±0.31	85.66±0.15	61.52±0.53	73.12±1.36	77.70	84.95
E^2-LoRA	92.13±0.19	**95.01**±0.02	89.77±0.16	92.68±0.48	82.77±0.10	87.18±0.12	75.82±0.28	80.90±0.73	85.12	88.94
ISM (ours)	91.95±0.27	94.82±0.14	88.90±0.46	92.16±0.63	**83.68**±0.15	**87.54**±0.20	**83.72**±0.05	**87.09**±0.47	**87.06**	**90.40**

Baselines. Table 1 compares against seventeen rehearsal-free methods for class-incremental learning on pre-trained vision transformers, all with a ViT-B/16 backbone pre-trained on ImageNet-21K and the same 10-task splits. We report each method's published numbers for this protocol, or a protocol-matched re-run where the original protocol differed. The methods fall into four groups. (i) Prompt-based methods that keep the backbone frozen and learn prompts: L2P (Wang et al., 2022b), DualPrompt (Wang et al., 2022a), and CODA-Prompt (Smith et al., 2023). (ii) Adapter-based methods that add or expand task-specific modules: LAE (Gao et al., 2023), EASE (Zhou et al., 2024c), SSIAT (Tan et al., 2024), MOS (Sun et al., 2025), TUNA (Wang et al., 2025), and E^2-LoRA (Li et al., 2026a). (iii) Methods that fine-tune or freeze one shared backbone and focus on the classifier or the feature statistics: SLCA (Zhang et al., 2023), SLCA++ (Zhang et al., 2024), RanPAC (McDonnell et al., 2023), APER (Zhou et al., 2024a), and SLDC (Rao et al., 2026). (iv) Merging-based methods: CoMA and CoFiMA (Marouf et al., 2024), which average sequentially fine-tuned backbones, and LCA (Tran et al., 2026), which merges LoRA modules incrementally and aligns the classifier with the local loss we adopt in Section 3.4. As an upper bound we also report joint training, which fine-tunes the same LoRA module on all classes at once. For the domain-incremental setting (Section 4.3) we additionally compare with MEMO (Zhou et al., 2023), SimpleCIL (Zhou et al., 2024a), DCE (Li et al., 2025b), and DUCT (Zhou et al., 2025).

4.2 Main results

Table 1 reports FA and AA on the four benchmarks. ISM has the highest average final accuracy and average incremental accuracy, 87.06 and 90.40, ahead of the strongest baseline on each metric by 1.9 points (E^2-LoRA) and 1.3 points (SLDC). The gain comes from the harder benchmarks: on ImageNet-R and Stanford Cars ISM is best on both metrics and leads the strongest baseline by 0.9 and 3.2 points of final accuracy, while on CIFAR-100 and CUB-200, where the recent methods are close to each other, it is within about one point of the best entry in every column. ImageNet-R and Stanford Cars are also where the best baseline trails joint training furthest, by 4.3 and 8.7 points, and ISM closes 0.9 and 3.2 of that gap; on CIFAR-100 and CUB-200 the best baselines are already within a point of joint training.

Table 2: Domain-incremental results (mean ± std over three seeds). FA: final accuracy; AA: average incremental accuracy. Best in bold.

	Office-Home		DomainNet	
Method	FA	AA	FA	AA
L2P	80.03±1.29	79.72±4.19	48.72±2.83	50.45±4.10
DualPrompt	80.85±0.14	80.20±3.81	50.46±3.17	52.28±3.35
CODA-Prompt	85.07±0.34	84.70±2.94	59.99±0.88	59.85±4.49
MEMO	63.09±1.80	71.18±2.76	58.41±3.20	61.92±5.39
RanPAC	82.28±0.07	82.30±3.34	54.80±0.36	55.20±3.93
EASE	76.33±2.16	81.16±3.52	43.72±1.70	50.50±2.27
SimpleCIL	75.72±0.00	75.69±5.03	44.08±0.00	42.95±4.84
DCE	84.40±0.20	84.60±3.00	63.50±0.50	64.30±6.00
DUCT	85.42±0.33	81.28±0.31	67.01±1.35	67.16±3.75
E^2-LoRA	88.25±0.25	84.94±0.21	69.63±0.05	68.69±0.04
ISM (ours)	**89.96**±0.13	**88.10**±0.20	**71.05**±0.14	**69.98**±0.07

Table 3: Robustness of ISM and published methods on CIFAR-100 (mean ± std over three seeds). Clean: final accuracy of the evaluated models; CIFAR-100-C: mean accuracy over 19 corruptions at five severities; CIFAR-100-P: mean flip probability over 10 perturbations, lower is better. Best in bold.

Method	Clean ↑	CIFAR-100-C ↑	CIFAR-100-P ↓
ISM (ours)	**91.94**±0.30	**79.75**±0.19	**3.24**±0.02
E^2-LoRA	90.86±0.34	77.34±0.34	3.61±0.02
LCA	88.22±0.50	75.26±0.35	3.95±0.08
SLCA	89.06±0.20	74.20±0.37	4.27±0.05
EASE	85.45±0.29	70.34±0.16	5.74±0.05

4.3 Domain-incremental learning

In domain-incremental learning, the label space stays fixed across tasks while the input distribution shifts. We evaluate ISM on Office-Home (Venkateswara et al., 2017), four domains with 65 classes each, and DomainNet (Peng et al., 2019), six domains with 345 classes each, following the protocol and baseline numbers of Li et al. (2026a): each domain is one task, and accuracy is computed over the shared label space and weighted by the number of test samples per domain. Table 2 shows that ISM obtains the best final and average incremental accuracy on both benchmarks, with margins of 1.7 and 3.2 points on Office-Home and 1.4 and 1.3 points on DomainNet over the strongest baseline.

4.4 Robustness

A merged model is only useful if its accuracy survives distribution shift at test time. We evaluate the CIFAR-100 models on CIFAR-100-C and CIFAR-100-P (Hendrycks & Dietterich, 2019): the 19 corruptions at five severities, scored by mean accuracy, and the 10 perturbation sequences, scored by the mean flip probability of the prediction, where lower is better. Every model is exported after the last task and run through one shared inference harness. Table 3 compares ISM with the baselines of Table 1 that we could re-run under this protocol. ISM is the most robust method on both benchmarks: it is ahead of E^2-LoRA, the next best, by 2.4 points of corrupted accuracy and 0.37 of flip probability, margins several times the standard deviation over seeds. ISM also loses the least accuracy under corruption relative to its clean accuracy, 12.2 points against 13.0 to 15.1 for the baselines. The same comparison across merge rules is in Appendix C.

4.5 Ablations

Truncation variant, energy fraction, and damping. A LoRA update is the product $\boldsymbol{BA}$ of two factors, so the truncation can be applied to the update of $\boldsymbol{A}$ only, of $\boldsymbol{B}$ only, of both factors separately, or of the effective update $\boldsymbol{BA}$, re-factored to rank r. Figure 2 sweeps the variant, the energy fraction p, and the damping α on ImageNet-R over a $4 \times 6 \times 6$ grid with three seeds each; the corner $p = 1$, $\alpha = 1$ deploys the sequentially trained module without any merge, with drift compensation and alignment still applied. Table 4 adds average incremental accuracy, forgetting, and the split of the final accuracy into old and new tasks for the $\boldsymbol{A}$-factor variant.

Damping. A large step forgets: from $\alpha = 0.3$ to $\alpha = 1$, forgetting rises from 3.8 to 6.6 points and the oldest five tasks lose 3.3 points, while the newest task gains nothing (84.8 at $\alpha = 0.3$, 83.8 at $\alpha = 1$). This is what Theorem 1 predicts: the bound grows with the spectral distance $\alpha\sigma_1$. The bound covers only what the module already knows, and on the other side a small step under-learns: at $\alpha = 0.1$ the newest task is 0.7 lower and the average incremental accuracy 0.6 lower than at $\alpha = 0.3$. We use the smallest step at which the newest task is fully learned, $\alpha = 0.3$ on ImageNet-R, which is also the best column for every variant.

Energy fraction. The spectral distance in Theorem 1 is set by the largest kept singular value and does not change with p; the truncation removes the low-energy directions of the update, which lowers the

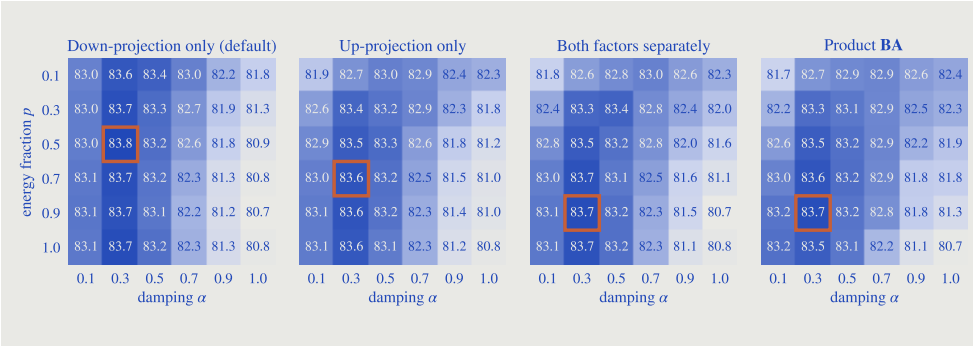


Figure 2: Merge sweep on ImageNet-R (final accuracy, mean over three seeds). One panel per truncation variant; rows are the energy fraction p and columns the damping α. The best cell of each panel is outlined. $p = 1$ applies no truncation, and $p = 1$, $\alpha = 1$ deploys the raw sequential module.

Table 4: Damping and truncation on ImageNet-R ($\boldsymbol{A}$-factor variant, mean over three seeds). Forgetting: mean over the old tasks of the drop from their highest to their final accuracy. Old: final accuracy on tasks 1–9; New: on task 10; Oldest: on tasks 1–5.

p	α	FA	AA	Forgetting	Old	New	Oldest
0.9	0.1	83.14	86.95	3.51	83.03	84.10	83.38
0.9	0.3	83.65	87.56	3.81	83.52	84.75	83.16
0.9	0.5	83.14	87.38	4.57	82.99	84.45	82.37
0.9	0.7	82.21	87.10	5.40	81.94	84.65	81.32
0.9	1.0	80.70	86.32	6.63	80.36	83.79	79.84
0.1	1.0	81.77	86.82	5.53	81.56	83.62	81.21
0.5	1.0	80.94	86.36	6.27	80.66	83.45	80.20
1.0	1.0	80.78	86.29	6.51	80.47	83.52	79.93

deviation of the merged module on typical inputs rather than in the worst case. Consistent with this, the truncation protects the oldest tasks when the step is large: at $\alpha = 1$, keeping a tenth of the energy of each update forgets one point less than keeping all of it and holds the oldest five tasks 1.3 higher, at no cost on the newest task. Once the step is small the truncation has less left to remove: at $\alpha = 0.3$ every p is within 0.25 of the best cell. We use $p = 0.9$, which removes only the low-energy tail of each update.

Truncation variant. The four variants reach the same best accuracy within 0.2. The $\boldsymbol{A}$-factor variant is the least sensitive to p and holds the best cell of the grid, so we use it.

With these settings the merge lifts the raw sequential module from 80.8 to 83.7 final accuracy and nearly halves its forgetting, from 6.5 to 3.8. Its share of the total gain of ISM grows with how far the sequential module drifts (Appendix D).

AI USE STATEMENT

In this work, we used generative AI tools (large language model assistants) for four tasks. First, literature analysis: to search for and organise the related work discussed in Section 2, and to check the bibliography entries against their sources. Second, paper drafting and polishing: to refine the grammar, wording, and formatting of text written by the authors. Third, scientific figures: to write the scripts that draw the figures from the logged results. Fourth, code: to help reimplement the baseline methods used for comparison, following their published descriptions and code. We did not use generative AI tools to develop the method or its theoretical analysis, to formulate or prove mathematical claims, to propose hypotheses, to design the experiments, or to generate or process data. All AI-assisted text, figures, and code were reviewed and verified by the authors: every citation was checked against its source, every figure against the logged results, and the baseline implementations

were tested against their published results. The authors take full responsibility for the content of this paper.

Ethics statement

This work studies continual image classification on public benchmark datasets (CIFAR-100, CUB-200, ImageNet-R, Stanford Cars, Office-Home, DomainNet, CIFAR-100-C, and CIFAR-100-P) and a publicly released pre-trained backbone, used under their respective licences. It involves no human subjects, no personal or sensitive data, and no new data collection. The method is a general technique for adapting a pre-trained model to a sequence of tasks. It carries the same societal risks as image classifiers in general, and a model built with it inherits any bias present in the pre-training data and in the task data; we are not aware of a direct harmful application. The authors declare no conflicts of interest and adhere to the ICLR Code of Ethics.

Reproducibility statement

Algorithm 1 gives the complete training pipeline, and Sections 3.2–3.4 define every step with its equations. Section 4.1 states the datasets, task splits, metrics, seeds, and backbone, and Appendix B lists all hyperparameters, including the per-dataset settings in Table 5, together with the hardware and runtime. The proof of Theorem 1 is in Appendix A. The sources of the baseline numbers are stated in Section 4.1, and the robustness protocol in Section 4.4; the merge-rule comparison, the contribution of each stage, and the initialisation study (Appendices C–E) use the same protocol. The code, the configuration of every experiment, and the scripts that produce every table and figure are available at `https://anonymous.4open.science/r/spectral_merging-4A9E`.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3366–3385, 2022.

Pala Tej Deep, Rishabh Bhardwaj, and Soujanya Poria. DELLA-Merging: Reducing interference in model merging through magnitude-based sampling. *arXiv preprint arXiv:2406.11617*, 2024.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

Takuma Fukuda, Hiroshi Kera, and Kazuhiko Kawamoto. Adapter merging with centroid prototype mapping for scalable class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Qiankun Gao, Chen Zhao, Yifan Sun, Teng Xi, Gang Zhang, Bernard Ghanem, and Jian Zhang. A unified continual learning framework with general parameter-efficient tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

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 (CVPR)*, 2025.

Alex Gomez-Villa, Dipam Goswami, Kai Wang, Andrew D. Bagdanov, Bartlomiej Twardowski, and Joost van de Weijer. Exemplar-free continual representation learning via learnable drift compensation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

Haiyang Guo, Fei Zhu, Fanhu Zeng, Bing Liu, and Xu-Yao Zhang. DESIRE: Dynamic knowledge consolidation for rehearsal-free continual learning. *arXiv preprint arXiv:2411.19154*, 2024.

Jiangpeng He, Zhihao Duan, and Fengqing Zhu. CL-LoRA: Continual low-rank adaptation for rehearsal-free class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations (ICLR)*, 2019.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Timm Hess, Gido M. van de Ven, and Tinne Tuytelaars. Forgetting of task-specific knowledge in model merging-based continual learning. *arXiv preprint arXiv:2507.23311*, 2025.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *International Conference on Learning Representations (ICLR)*, 2023.

Dong-Hwan Jang, Sangdoo Yun, and Dongyoon Han. Model stock: All we need is just a few fine-tuned models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

Prakhar Kaushik, Ankit Vaidya, Shravan Chaudhari, Rama Chellappa, and Alan Yuille. Shared LoRA subspaces for almost strict continual learning. *arXiv preprint arXiv:2602.06043*, 2026.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3D object representations for fine-grained categorization. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2013.

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

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 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2025.

Jiashuo Li, Shaokun Wang, Bo Qian, Yuhang He, Xing Wei, Qiang Wang, and Yihong Gong. Dynamic integration of task-specific adapters for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025a.

Lan Li, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Addressing imbalanced domain-incremental learning through dual-balance collaborative experts. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2025b.

Longhua Li, Lei Qi, Qi Tian, and Xin Geng. Energy-structured low-rank adaptation for continual learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2026a.

Mei Li, Yuxiang Lu, Qinyan Dai, Suizhi Huang, Yue Ding, and Hongtao Lu. BECAME: BayEsian continual learning with adaptive model MErging. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2025c.

Yayuan Li, Ze Peng, Jian Zhang, Jintao Guo, Yue Duan, and Yinghuan Shi. When shared knowledge hurts: Spectral over-accumulation in model merging. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2026b.

Yan-Shuo Liang and Wu-Jun Li. InfLoRA: Interference-free low-rank adaptation for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Daniel Marczak, Bartłomiej Twardowski, Tomasz Trzciński, and Sebastian Cygert. MagMax: Leveraging model merging for seamless continual learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

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 International Conference on Machine Learning (ICML)*, 2025.

Imad Eddine Marouf, Subhankar Roy, Enzo Tartaglione, and Stéphane Lathuilière. Weighted ensemble models are strong continual learners. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

Michael Matena and Colin Raffel. Merging models with Fisher-weighted averaging. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165, 1989.

Mark D. McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton van den Hengel. RanPAC: Random projections and pre-trained models for continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Dilan Gorur, Razvan Pascanu, and Hassan Ghasemzadeh. Linear mode connectivity in multitask and continual learning. In *International Conference on Learning Representations (ICLR)*, 2021.

German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Fuli Qiao and Mehrdad Mahdavi. Merge before forget: A single LoRA continual learning via continual merging. In *International Conference on Learning Representations (ICLR)*, 2026.

Haomiao Qiu, Miao Zhang, Ziyue Qiao, and Liqiang Nie. Train with perturbation, infer after merging: A two-stage framework for continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Zihuan Qiu, Lei Wang, Yang Cao, Runtong Zhang, Bing Su, Yi Xu, Fanman Meng, Linfeng Xu, Qingbo Wu, and Hongliang Li. Null-space filtering for data-free continual model merging: Preserving stability, promoting plasticity. In *International Conference on Learning Representations (ICLR)*, 2026.

Xuan Rao, Simian Xu, Zheng Li, Bo Zhao, Derong Liu, Mingming Ha, and Cesare Alippi. Compensating distribution drifts in continual learning with pre-trained vision transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2026.

James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. CODA-Prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your ViT? data, augmentation, and regularization in vision transformers. *Transactions on Machine Learning Research*, 2022.

George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. Model merging with SVD to tie the Knots. In *International Conference on Learning Representations (ICLR)*, 2025.

Hai-Long Sun, Da-Wei Zhou, Hanbin Zhao, Le Gan, De-Chuan Zhan, and Han-Jia Ye. MOS: Model surgery for pre-trained model-based class-incremental learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2025.

Yuwen Tan, Qinhao Zhou, Xiang Xiang, Ke Wang, Yuchuan Wu, and Yongbin Li. Semantically-shifted incremental adapter-tuning is a continual ViTransformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Anke Tang, Enneng Yang, Li Shen, Yong Luo, Han Hu, Lefei Zhang, Bo Du, and Dacheng Tao. Merging on the fly without retraining: A sequential approach to scalable continual model merging. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Irene Testa, Luigi Quarantiello, Eric Nuertey Coleman, Samrat Mukherjee, Julio Hurtado, and Vincenzo Lomonaco. GLAM: Efficient continual learning at scale via grouped LoRA adapter merging. In *Conference on Lifelong Learning Agents (CoLLAs)*, 2026.

Tung Tran, Danilo Vasconcellos Vargas, and Khoat Than. LCA: Local classifier alignment for continual learning. In *International Conference on Learning Representations (ICLR)*, 2026.

Gido M. van de Ven, Tinne Tuytelaars, and Andreas S. Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4:1185–1197, 2022.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP*, 2023.

Yan Wang, Da-Wei Zhou, and Han-Jia Ye. Integrating task-specific and universal adapters for pre-trained model-based class-incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. DualPrompt: Complementary prompting for rehearsal-free continual learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022a.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 139–149, 2022b.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2022.

Yichen Wu, Hongming Piao, Long-Kai Huang, Renzhen Wang, Wanhua Li, Hanspeter Pfister, Deyu Meng, Kede Ma, and Ying Wei. SD-LoRA: Scalable decoupled low-rank adaptation for class incremental learning. In *International Conference on Learning Representations (ICLR)*, 2025.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-Merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. AdaMerging: Adaptive model merging for multi-task learning. In *International Conference on Learning Representations (ICLR)*, 2024.

Jiazuo Yu, Yunzhi Zhuge, Lu Zhang, Ping Hu, Dong Wang, Huchuan Lu, and You He. Boosting continual learning of vision-language models via mixture-of-experts adapters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024a.

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 International Conference on Machine Learning (ICML)*, 2024b.

Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. SLCA: Slow learner with classifier alignment for continual learning on a pre-trained model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. SLCA++: Unleash the power of sequential fine-tuning for continual learning with pre-training. *arXiv preprint arXiv:2408.08295*, 2024.

Da-Wei Zhou, Qi-Wei Wang, Han-Jia Ye, and De-Chuan Zhan. A model or 603 exemplars: Towards memory-efficient class-incremental learning. In *International Conference on Learning Representations (ICLR)*, 2023.

Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. *International Journal of Computer Vision*, 2024a.

Da-Wei Zhou, Hai-Long Sun, Jingyi Ning, Han-Jia Ye, and De-Chuan Zhan. Continual learning with pre-trained models: A survey. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2024b.

Da-Wei Zhou, Hai-Long Sun, Han-Jia Ye, and De-Chuan Zhan. Expandable subspace ensemble for pre-trained model-based class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024c.

Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, Lijun Zhang, and De-Chuan Zhan. Dual consolidation for pre-trained model-based domain-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

A PROOFS

Proof of Theorem 1. For simplicity of presentation, we use $h \circ f(\boldsymbol{x})$ to denote $h(f(\boldsymbol{x}))$.

Observe that

$$\begin{aligned} L(g_{t+1}) &= L(g_{t+1}) - \mathbb{E}_{\boldsymbol{x}\sim P_x}\left[\ell(h_{t+1}\circ f_t(\boldsymbol{x}))\right] + \mathbb{E}_{\boldsymbol{x}\sim P_x}\left[\ell(h_{t+1}\circ f_t(\boldsymbol{x}))\right] \\ &\quad - L(g_t) + L(g_t) \qquad (7) \\ &= \mathbb{E}_{\boldsymbol{x}\sim P_x}\left[\ell(h_{t+1}\circ f_{t+1}(\boldsymbol{x})) - \ell(h_{t+1}\circ f_t(\boldsymbol{x}))\right] \qquad (8) \\ &\quad + \mathbb{E}_{\boldsymbol{x}\sim P_x}\left[\ell(h_{t+1}\circ f_t(\boldsymbol{x})) - \ell(h_t\circ f_t(\boldsymbol{x}))\right] + L(g_t). \qquad (9) \end{aligned}$$

By the definition of P_t,

$$\mathbb{E}_{\boldsymbol{x}\sim P_x}\left[\ell(h_{t+1}\circ f_t(\boldsymbol{x})) - \ell(h_t\circ f_t(\boldsymbol{x}))\right] = \mathbb{E}_{\boldsymbol{z}\sim P_t}\left[\ell(h_{t+1}(\boldsymbol{z})) - \ell(h_t(\boldsymbol{z}))\right]. \tag{10}$$

Furthermore, by the Lipschitz continuity of ℓ and h_{t+1}, for every $\boldsymbol{x}$,

$$\ell(h_{t+1}\circ f_{t+1}(\boldsymbol{x})) - \ell(h_{t+1}\circ f_t(\boldsymbol{x})) \le L_\ell L_h \|f_{t+1}(\boldsymbol{x}) - f_t(\boldsymbol{x})\|. \tag{11}$$

Combining these inequalities with equation 9 proves equation 5.

For the second result, since f_{t+1} keeps the portion of f_t after layer k frozen, the same mapping is applied to the respective inputs of layer $k+1$ in both backbones. Since this common portion is $L_{f,k}$-Lipschitz continuous,

$$\begin{aligned}
\|f_{t+1}(\boldsymbol{x}) - f_t(\boldsymbol{x})\| &\le L_{f,k}\|\boldsymbol{W}^k_{t+1} f^k_{t+1}(\boldsymbol{x}) - \boldsymbol{W}^k_t f^k_t(\boldsymbol{x})\| && (12)\\
&= L_{f,k}\|\boldsymbol{W}^k_{t+1} f^k_{t+1}(\boldsymbol{x}) - \boldsymbol{W}^k_t f^k_{t+1}(\boldsymbol{x}) + \boldsymbol{W}^k_t f^k_{t+1}(\boldsymbol{x}) - \boldsymbol{W}^k_t f^k_t(\boldsymbol{x})\| && (13)\\
&\le L_{f,k}\Big[\|\boldsymbol{W}^k_{t+1} f^k_{t+1}(\boldsymbol{x}) - \boldsymbol{W}^k_t f^k_{t+1}(\boldsymbol{x})\| \\
&\qquad + \|\boldsymbol{W}^k_t f^k_{t+1}(\boldsymbol{x}) - \boldsymbol{W}^k_t f^k_t(\boldsymbol{x})\|\Big] && (14)\\
&\le L_{f,k}\Big[\|\boldsymbol{W}^k_{t+1} - \boldsymbol{W}^k_t\|\,\|f^k_{t+1}(\boldsymbol{x})\| + \|\boldsymbol{W}^k_t\|\,\|f^k_{t+1}(\boldsymbol{x}) - f^k_t(\boldsymbol{x})\|\Big]. && (15)
\end{aligned}$$

Taking expectations over $\boldsymbol{x}\sim P_x$ on both sides and using the definition of C_k proves equation 6. □

B Implementation details

Backbone and LoRA. The frozen backbone is that of Section 4.1. One LoRA module of rank $r = 64$, with LoRA scale parameter 128 (so each low-rank product is scaled by 2) and no dropout, is attached to the query-key-value projection and the two MLP layers of every transformer block, 8.26M trainable parameters in total. Features are the 768-dimensional class token of the backbone. The head is a linear layer without bias; each task appends the output rows of its classes.

Sequential fine-tuning. Each task is trained for 10 epochs with SGD (momentum 0.9), learning rate 10^{-2} for both the LoRA module and the head, a cosine schedule to 10^{-6}, batch size 64, and cross-entropy on the current task only. Weight decay is set per dataset; Table 5 lists every setting that differs across datasets.

Spectral merging. The truncation is applied to the update of the LoRA down-projection factor $\boldsymbol{A}$ of each adapted layer, $\boldsymbol{A}^{\text{merged}}_t = \boldsymbol{A}^{\text{merged}}_{t-1} + \alpha\,\text{trunc}_p(\boldsymbol{A}_t - \boldsymbol{A}^{\text{merged}}_{t-1})$, where trunc_p is the energy truncation of Section 3.2. The up-projection factor $\boldsymbol{B}$ and all other trainable parameters are interpolated with the same α. We use $p = 0.9$ on every dataset and set α per dataset. After the first task the merged module is the trained module, $\boldsymbol{M}_1 = \boldsymbol{W}_1$.

Drift compensation. The map $\boldsymbol{T}_t$ of Eq. 3 is fitted on the features of all training images of task t under $\boldsymbol{M}_{t-1}$ and $\boldsymbol{M}_t$, with a ridge weight λ set per dataset. Both the mean and the full covariance of every old class are transported (Eq. 4). Class covariances are the empirical covariances plus $10^{-4}\boldsymbol{I}$.

Classifier alignment. The head is retrained with LCA (Tran et al., 2026) for 10 epochs of 100 steps. Each step draws $N = 40$ classes (all seen classes while fewer than 40 have been seen) and $K = 64$ pseudo-features per class from the class Gaussians, and uses SGD with learning rate 10^{-2}, momentum 0.9, and weight decay 10^{-4}. The weight of the local term is set per dataset.

Protocol and per-dataset settings. Table 5 lists the settings that differ across datasets; everything else is shared.

Hardware and runtime. All experiments ran on one workstation with a single NVIDIA RTX 4090 (24 GB), an Intel Core i9-12900K, and 125 GB of RAM, under Linux with Python 3.9, PyTorch 2.7.0 (CUDA 11.8), and timm 1.0.24. Backbone training is the only GPU-heavy stage; the per-task modules are cached, so the merge, drift, and alignment ablations re-run only the later stages. With cached modules, one 10-task run takes about 10 minutes per seed; one CIFAR-100-C evaluation takes about 1.5 hours per model and one CIFAR-100-P evaluation about 1.1 hours.

Table 5: Per-dataset settings. All other hyperparameters are shared across datasets.

Dataset	Tasks	Weight decay	Damping α	Ridge λ	Local-term weight
CIFAR-100	10×10	5×10^{-4}	0.15	100	0.2
CUB-200	10×20	10^{-3}	0.10	10	0.1
ImageNet-R	10×20	2×10^{-4}	0.30	100	0.1
Stanford Cars	$16 + 9 \times 20$	10^{-3}	0.20	100	0.2

C MERGE-RULE COMPARISON

Accuracy. To isolate the contribution of spectral merging, we keep every other stage fixed, the same sequentially trained modules, drift compensation, and classifier alignment, and replace only the merge rule. We compare against seven published rules under the same sequential, pairwise protocol: plain averaging, which is task arithmetic with unit coefficient (Ilharco et al., 2023); TIES (Yadav et al., 2023); maximum-magnitude selection, the rule of MagMax (Marczak et al., 2024); DARE (Yu et al., 2024b); DELLA (Deep et al., 2024); Model Stock (Jang et al., 2024); and KnOTS with TIES (Stoica et al., 2025). Table 6 reports the results. ISM has the highest final accuracy on all four benchmarks, by 0.3 points on ImageNet-R and up to 2.5 on CUB-200 over the best published rule, and the highest average incremental accuracy on three; on Stanford Cars DELLA is 0.02 higher and task arithmetic ties. Rules that select entries by sign, magnitude, or random drop (TIES, max-abs, DARE) fall furthest behind on the fine-grained benchmarks.

Table 6: Merge-rule comparison. Only the merge rule changes; all other stages are those of ISM. Each published rule is tuned on ImageNet-R over a small grid and its best setting is reused on the other benchmarks. FA: final accuracy; AA: average incremental accuracy; mean over three seeds. Best in bold.

	CIFAR-100		CUB-200		ImageNet-R		Stanford Cars	
Merge rule	FA	AA	FA	AA	FA	AA	FA	AA
ISM (ours)	**91.95**	**94.82**	**88.90**	**92.16**	**83.68**	**87.54**	**83.72**	87.09
Model Stock	90.55	94.37	86.42	91.04	83.36	87.48	82.65	87.08
Task Arithmetic (average)	90.26	94.26	85.87	90.81	83.32	87.45	82.36	87.09
DELLA	90.30	94.30	85.43	90.65	83.00	87.37	82.29	**87.11**
KnOTS-TIES	89.97	93.98	84.75	90.09	82.97	87.35	81.90	86.77
DARE	89.98	94.19	84.47	90.43	82.48	87.21	81.88	86.97
TIES	89.58	93.90	84.54	89.88	82.67	87.14	81.73	86.77
Max-abs (MagMax)	88.70	93.44	83.33	89.27	80.93	86.48	79.46	85.89

Robustness. We evaluate the CIFAR-100 models of every rule in Table 6 on CIFAR-100-C and CIFAR-100-P under the protocol of Section 4.4. Table 7 reports the results. ISM is the most robust rule on both benchmarks, ahead of the best published rule, Model Stock, by 1.2 points of corrupted accuracy and 0.14 of flip probability, and ahead of plain averaging by 1.5 and 0.19. It is also ahead of every rule on every single corruption and perturbation. The rules keep nearly the same order as on clean data, with ISM first and maximum-magnitude selection last on every axis.

D CONTRIBUTION OF EACH STAGE

Table 8 builds ISM up stage by stage from sequential fine-tuning and reports the Shapley share of merging, defined in the caption. Merging is the first stage and the one that transfers everywhere: on CIFAR-100 and ImageNet-R it adds 2.0 and 2.6 points on its own, and its share of the total gain, 2.7 and 2.9 points, is close to what it gives alone. Its role grows with how far the sequential module drifts. On Stanford Cars, where sequential fine-tuning ends at 59.9 and forgetting is largest, merging alone adds 4.8 points and its share is 6.6, the largest of the four benchmarks. This matches the damping study of Table 4. On CUB-200 merging adds nothing by itself but has a share of 4.4 points: it pays out through the alignment stage, which needs a stable feature space for the class

Table 7: Robustness of the merge rules on CIFAR-100 (mean ± std over three seeds). Clean: final accuracy of the evaluated models; CIFAR-100-C: mean accuracy over 19 corruptions at five severities; CIFAR-100-P: mean flip probability over 10 perturbations, lower is better. Best in bold.

Merge rule	Clean ↑	CIFAR-100-C ↑	CIFAR-100-P ↓
ISM (ours)	**91.94**±0.30	**79.75**±0.19	**3.24**±0.02
Model Stock	90.67±0.31	78.58±0.30	3.38±0.08
Task Arithmetic (average)	90.26±0.36	78.21±0.34	3.43±0.05
DELLA	90.23±0.26	78.14±0.31	3.45±0.04
KnOTS-TIES	89.98±0.33	77.98±0.35	3.44±0.07
DARE	90.16±0.27	77.74±0.21	3.50±0.02
TIES	89.58±0.36	77.68±0.21	3.52±0.06
Max-abs (MagMax)	88.70±0.53	75.97±0.31	3.81±0.15

Gaussians it samples from, and the two stages together add 11.2 points. Alignment is the largest single stage on the fine-grained benchmarks, and drift compensation matters where the backbone drifts far, adding 12.7 points on Stanford Cars and 1.1 to 1.6 elsewhere.

Table 8: Contribution of each stage (final accuracy, mean over three seeds). Each row adds one stage to the previous one; the last row is ISM. The Shapley share is the gain from switching merging on, averaged over whether the other two stages are on or off, with Shapley weights. The full stack reproduces the numbers of Table 1 within 0.1.

	CIFAR-100	CUB-200	ImageNet-R	Stanford Cars
Sequential fine-tuning	86.19	76.17	78.94	59.93
+ spectral merging	88.15	76.17	81.56	64.73
+ classifier alignment	90.37	87.32	82.51	70.98
+ drift compensation (ISM)	91.92	88.94	83.63	83.63
Shapley share of merging	2.71	4.42	2.88	6.62

E SEQUENTIAL VERSUS INDEPENDENT INITIALISATION

ISM initialises the module of task t from the trained module of task $t-1$ (Section 3.2). Table 9 repeats the ImageNet-R experiment with every task trained from the same initial module $\boldsymbol{W}_0$ instead, and then applies each merge rule to the resulting modules as in Appendix C. Forgetting is defined as in Table 4. With independent initialisation the best rule, spectral merging of the effective update, reaches 80.64 final accuracy, 3.0 points below the 83.68 of ISM with sequential initialisation, and every rule forgets more (5.8 to 7.9, and 8.1 with no merge, against 3.7). The ordering of the rules is unchanged: the spectral variants lead, then averaging and TIES, and maximum-magnitude selection falls below no merge at all.

Table 9: Merge rules with independent initialisation on ImageNet-R (mean ± std over three seeds). Every task module starts from the same initial module; all other stages are those of ISM. FA: final accuracy; AA: average incremental accuracy.

Merge rule	FA	AA	Forgetting
Spectral, effective update	80.64±0.20	85.67±0.25	5.77±0.13
Spectral, both factors	80.45±0.05	85.53±0.25	5.97±0.07
Spectral, $\boldsymbol{A}$ factor	80.43±0.13	85.53±0.27	6.05±0.19
Spectral, $\boldsymbol{B}$ factor	80.38±0.15	85.57±0.25	6.09±0.17
Task Arithmetic (average)	79.88±0.24	85.29±0.25	6.75±0.39
TIES	79.72±0.11	85.16±0.33	6.92±0.20
No merge	77.95±0.35	83.95±0.37	8.14±0.48
Max-abs (MagMax)	77.75±0.55	84.69±0.51	7.94±0.14
ISM, sequential initialisation	83.68±0.15	87.54±0.20	3.74±0.39