Localising Failure between Representation and Readout: A Fresh-Head Probe for Parameter-Space Model Merging

Anonymous authors
Paper under double-blind review

Abstract

Parameter-space merging has become increasingly sophisticated about how independently fine-tuned models should be combined, but less explicit about what the resulting post-merge score is allowed to diagnose. The native end-to-end accuracy of a merged model is a valid verdict on the system delivered by the merge; it is not, by itself, evidence that failure entered through the merged representation rather than through the readout the merge also delivered. We formalise this distinction with a fresh-head probe: the merged backbone is held fixed, the union-label readout is re-estimated under matched supervision, and the native–fresh-head gap identifies the readout-recoverable component of the native shortfall. In a controlled CIFAR-100 diagnostic regime with Task Arithmetic and TIES-Merging, this component constitutes a substantial share of the native shortfall across the tested regimes, including structured Ward decompositions and random class-to-task partitions on both ViT-B/16 and ResNet-50. A separate geometry control shows that centroid-routed modular composition has a complementary boundary: it outperforms naive ensembling under structured Ward geometry on both backbones, but its advantage disappears or reverses under random class partitions. These results show that model-merging evaluation needs not only better merge operators, but a stricter evidential contract: post-merge scores should be read as delivered-model verdicts, not self-localising diagnoses of representation failure.

1 Introduction

The native end-to-end number is the object most model-merging papers ask the reader to trust. This trust has been earned for good reasons. Wortsman et al. (2022) showed that averaging fine-tuned models can improve accuracy and robustness when solutions occupy a shared low-error basin; Matena & Raffel (2022) gave model merging a posterior-weighted interpretation through Fisher information; Jin et al. (2023) recast merging as a layer-wise regression problem against candidate-model predictions; and Ilharco et al. (2023) showed that task-specific fine-tuning updates can be manipulated as composable task vectors. Yadav et al. (2023) then sharpened the central obstacle by identifying parameter interference, redundant updates, and sign conflicts as concrete failure modes in weight-space composition. This literature makes parameter-space merging attractive precisely because it promises a single multi-task model without joint retraining over the union data. Its usual experimental object is therefore the native merged model: combine the weights, run the result end-to-end over the target label space, and report the resulting accuracy as the behavioural verdict on the merge.

That verdict is useful, but it is doing more work than one scalar can safely carry. The same native score is often treated both as a deployment verdict on the merged model and as indirect evidence about what the merge did to the representation. Those are different claims. TIES-Merging, for example, directly attacks interference inside the parameter update before aggregation (Yadav et al., 2023); if the resulting native model still performs poorly, it is tempting to conclude that representation-level merging has failed. Yet the end-to-end score is produced by a path with two separable parts: the merged backbone that supplies features, and the readout that maps those features to logits over the union label space. A bad native score can therefore reflect damaged representations, an incompatible readout, or a mismatch between partially usable features

and the classifier delivered by the merge. The same score can be a valid verdict on the delivered merged model and an invalid localisation of why that model failed.

This distinction is not a cosmetic evaluation concern; it is the difference between ranking a system and diagnosing it. Alain & Bengio (2016) introduced linear probes as a way to ask whether representations contain linearly accessible information without changing the representation itself, while Belinkov (2022) later emphasised that probes are diagnostic instruments whose interpretation depends on the question they are designed to answer. Recent model-merging work has also begun to identify classifier or encoder–head misalignment as a concrete failure mode in classification settings (Kong et al., 2024). The remaining problem is narrower and, for model-merging evaluation, more operational: when a native merged model fails, how much of the reported shortfall is recoverable by replacing the delivered readout while leaving the merged backbone fixed?

We answer this question with a fresh-head probe. For each merged model, we freeze the backbone exactly as the merging procedure produces it and train only a new union-label linear classifier on the union data: same frozen backbone, new readout. The native fused readout remains the matched comparator, because it is part of the model actually delivered by Task Arithmetic or TIES-Merging. The resulting native–fresh-head delta is therefore not a claim about the absolute quality of the backbone. Because the fresh head uses union-label supervision, the probe is not a data-free merge procedure; it is a controlled evaluation counterfactual for interpreting the score of the model that the merge delivered. It identifies the component of the native end-to-end shortfall recoverable by replacing only the readout. This is the evaluation-contract distinction: the native score evaluates the delivered merged model, while the fresh-head score tests what the delivered backbone can still support under a repaired linear interface.

The recovery delta is not a small calibration effect. Across both ViT-B/16 and ResNet-50, replacing only the readout recovers a substantial part of the native shortfall for Task Arithmetic and TIES-Merging. The native score therefore cannot be read as a verdict on the merged representation alone: in this regime, a large part of what appears as model-merging collapse is recoverable at the interface between representation and readout. This is not an absolution of the backbone; the merged features may still be degraded, but the native readout makes that degradation appear more decisive than the fresh-head probe supports. A modular-composition contrast supplies the boundary condition: under structured Ward decomposition (Ward Jr., 1963), centroid-routed composition outperforms naive ensembling, whereas under random class-to-task splits that advantage disappears. Together, the results show that usable composition depends jointly on representation support, readout compatibility, and task geometry.

We make three scoped contributions.

First, we show that native parameter-space merging scores are locus-ambiguous under disjoint-support composition. The same end-to-end number serves as a valid verdict on the delivered merged model, but not as a localisation of whether failure entered through the representation, the readout, or their interface.

Second, we introduce a fresh-head probe that separates native delivered-model performance from recoverable representation support. By freezing the merged backbone and replacing only the union-label readout, the probe identifies the component of the native shortfall recoverable under readout replacement with matched supervision, while explicitly not claiming that the backbone is undamaged.

Third, we show that modular composition's advantage is geometry-dependent in the tested regime. Centroid-routed modular composition outperforms naive ensembling under structured Ward decomposition, but the advantage vanishes under random class-to-task splits, bounding the regime in which this form of composition is behaviourally useful.

All three contributions are evaluated on CIFAR-100 under both Ward and random class-to-task decompositions, across two backbone families and two parameter-space merging methods; the limitations of this controlled regime are discussed in Section 6.

2 Related Work

Weight-space compatibility and validity conditions. Parameter-space merging rests on an increasingly explicit account of when separately trained or fine-tuned parameter states can be combined while preserving useful behaviour. Wortsman et al. (2022) ground the practical case: when fine-tuned solutions occupy a shared low-error basin, averaging their weights can improve accuracy and robustness without inference-time cost. Matena & Raffel (2022) formalise the merging operation through a Fisher-weighted average, giving the procedure a posterior/Laplace interpretation rather than a purely heuristic one. Frankle et al. (2020) provide a related connectivity premise: networks trained from a shared initialisation can occupy modes connected by low-loss linear paths, helping explain why averaging nearby fine-tuned solutions can sometimes be meaningful. Ainsworth et al. (2023) sharpen the geometric obstacle: permutation symmetries between independently trained networks can place functionally similar solutions in different coordinate systems, so re-basing across those symmetries becomes a precondition for weight-space alignment to be meaningful. Stoica et al. (2024) extend this alignment logic to networks trained from different initialisations by matching activations rather than only weights, broadening merging beyond the shared-initialisation regime that the connectivity premise most directly supports. Jin et al. (2023) recast merging as layer-wise regression, deriving a closed-form merge that preserves candidate-model behaviour on calibration inputs. These works converge on a shared move: merging is no longer treated as a naive averaging trick, but as an operation whose validity conditions, including basin compatibility, statistical weighting, geometric alignment, and prediction matching, are themselves objects of study. Their success, however, is still typically judged through the delivered merged model's native end-to-end score. The literature has clarified conditions under which weight-space composition may be valid; it has not made the native score self-localising about whether failure, when it occurs, lies in the merged representation or at the readout interface.

Task vectors and interference control. A second branch of the merging literature takes the fine-tuning delta as the object to manipulate directly. Ilharco et al. (2023) introduce task vectors: differences between fine-tuned and pretrained weights that can be added, scaled, or negated to compose or remove task-specific behaviour. Ortiz-Jiménez et al. (2023) subsequently characterise the regime in which task arithmetic is well-behaved by linearising fine-tuning trajectories in the tangent space, supplying explicit applicability conditions for the operation rather than a purely empirical justification. Yadav et al. (2023) identify redundant updates and sign conflicts as concrete sources of interference between such deltas, and resolve them through a trim/elect/merge procedure before aggregation. Subsequent work refines this delta-manipulation view along three axes: random dropping and rescaling of delta parameters through DARE (Yu et al., 2024), sparse-mask construction that removes outliers and negligible perturbations (Davari & Belilovsky, 2024), and adaptive learning of task-wise or layer-wise merge coefficients without access to the original training data (Yang et al., 2024). Across this branch, the merge operation has itself become a designed object: interference, redundancy, sparsity, and weighting are no longer incidental by-products of aggregation, but design variables controlled before or during the merge. Yet the output of each procedure remains a delivered merged model whose quality is typically reported as a single end-to-end accuracy over the union label space. Better merge operators do not, by themselves, sharpen what their native scores diagnose: an interference-aware merge that underperforms can fail at the readout interface as well as in the representation, and a single end-to-end number does not distinguish the two. Model merging has become increasingly sophisticated about how parameters should be combined, but much less explicit about what the native post-merge score is allowed to diagnose.

Readout alignment and the score's estimand. The closest neighbouring work makes the present diagnostic problem harder to ignore. Kong et al. (2024) show that, in merged classification models, degradation can arise from misalignment between the merged encoder or backbone outputs and the classifier head, and they propose alignment-based remedies that re-fit or geometrically align the classifier to the merged representation. This is not a remote analogy to our setting; it is evidence that the classifier–backbone interface can be a load-bearing source of the reported score. Its implication, however, is broader than repair. Once classifier-side alignment can materially change the measured performance of a merged model, the native end-to-end number cannot be treated as a transparent estimate of representation damage. It is an estimate of delivered-model utility under the readout that the merge happened to deliver. A parallel observation in

fine-tuning evaluation underscores why head–feature interactions matter: Kumar et al. (2022) show that full fine-tuning can distort pretrained features under distribution shift, so end-to-end accuracy alone can obscure which part of an adapted model is responsible for the observed behaviour. The merging setting differs, but the methodological lesson is the same: a single score can compress quantities that need to be separated.

This distinction separates three quantities that are often collapsed in discussion of merging results: the native score of the delivered model, the representational support available in the frozen merged backbone, and the shortfall recoverable by replacing or aligning the readout. Alignment methods target the third quantity as a remedy. The fresh-head probe uses it as an identification contrast. By holding the merged backbone fixed and changing only the union-label readout, the probe asks how much of the native shortfall was readout-recoverable in the first place. The novelty claimed here is therefore not that classifier heads can misalign with merged backbones. It is that the native score has two different evidential roles: a valid deployment verdict on the delivered merged model, and an invalid localisation of representation failure. Those roles must be separated before a failed merge can be mechanistically interpreted.

Probes, classifier-side diagnostics, and what a fresh head can claim. The probing literature supplies the right discipline, but not the whole interpretation. Alain & Bengio (2016) introduce linear classifier probes as a way to ask what a frozen representation makes linearly accessible without modifying the representation itself. Belinkov (2022) emphasises the corresponding limit: a probe is a diagnostic instrument whose meaning depends on the counterfactual it implements, not a free-standing causal explanation of model behaviour. Hewitt & Liang (2019) sharpen this discipline operationally by introducing control-task baselines that distinguish probe expressivity from representation content; the methodological consequence is that any claim a probe makes is a claim about a specific counterfactual, not about the representation in isolation. The fresh-head probe uses this logic, but its counterfactual is narrower than a generic representation probe. It does not ask whether an arbitrary classifier can extract information from a hidden state; it asks whether the same merged backbone, evaluated over the same union label space with the same readout family, supports substantially better performance when the delivered fused readout is replaced by a newly estimated one. A parallel lesson appears in class-incremental learning, where Wu et al. (2019) show that a small linear correction to the final classifier can materially change end-to-end accuracy without retraining the feature extractor. The legitimate object here is therefore not a certification of intact representation, nor a deployable model-merging recipe. It is the native–fresh-head delta: the component of the native shortfall recoverable by replacing only the readout while holding fixed the merged backbone delivered by the merging procedure.

Modular composition and the geometry of routing. A contrasting tradition avoids parameter-space fusion entirely: it preserves separate components and combines their outputs at inference time through routing or conditional activation. Recent work blurs this boundary: Li et al. (2022) train separate experts on disjoint data and then average their weights, treating fusion and modular specialisation as composable rather than mutually exclusive; this is one of the closest published analogues to the disjoint-support setting we evaluate here under both fusion and routing. Shazeer et al. (2017) make sparse expert routing a scalable conditional-computation mechanism, showing that learned gates can dispatch inputs across many specialised experts without activating the full parameter set. Lepikhin et al. (2021) and Fedus et al. (2022) carry this design into modern scaling regimes, where conditional activation of a small subset of experts becomes a major route to compute-efficient model capacity. Across this lineage, the success condition shifts from "do the merged weights preserve useful behaviour?" to "does the routing representation select a useful component or expert subset for the input?". This is a change of object, not merely a change of scale. Performance under modular composition therefore depends jointly on what each preserved component supports and on whether the routing mechanism can separate those components under the geometry imposed by the task. The present paper does not propose a new modular architecture. It uses modular composition as a contrasting regime and asks when that regime delivers a behavioural advantage over naive ensembling. Under structured Ward decomposition, the modular composer outperforms naive ensembling; under random class-to-task splits, that advantage disappears, indicating that composition's behavioural utility is conditional on support in the routing feature space rather than a universal property.

3 Method and Identification Design

3.1 Benchmark, decompositions, and references

All experiments use CIFAR-100 (Krizhevsky, 2009) as the parent dataset. The primary regime is a disjoint-support Ward decomposition: the 100 classes are partitioned into five variable-sized tasks by hierarchical clustering of class-mean features (Ward Jr., 1963); the random-split controls use balanced 5×20 class partitions. Full partition details are reported in Appendix A. The Ward decomposition is the primary structured regime for Claims 1–3; two random class-to-task partitions provide geometry controls and extend the readout-recovery and modular-composition tests beyond the Ward partition.

Two backbone families are evaluated under the Ward protocol: ViT-B/16 (`timm/vit_base_patch16_224.augreg2_in21k_ft_in1k`) and ResNet-50 (`timm/resnet50.a1_in1k`). For each backbone and Ward task, we train a separate per-task expert from the same pretrained initialisation, repeated across five model seeds. The random-split geometry controls use three model seeds for each of two random partitions. Expert-training hyperparameters are reported in the appendix; within each backbone family, the same training recipe is used across experts so that merging comparisons are controlled by backbone and task decomposition rather than by per-method training differences.

A joint-training reference is computed for each backbone family on the union 100-class task, sharing the pretrained initialisation and evaluation protocol used for the merged models. This reference is not a merging baseline. It calibrates the native-to-joint gap used in Section 4 to report how much of the supervised headroom is recovered by replacing only the readout. Where random partitions are used, the joint-training reference is computed per split and recovered fractions are reported with split-specific denominators rather than pooled across partitions, following the variance-accounting practice appropriate for nested experimental designs (Bouthillier et al., 2021). Full joint-training hyperparameters are reported in the appendix.

3.2 Parameter-space merging methods

Two parameter-space merging methods are evaluated. **Task Arithmetic** (Ilharco et al., 2023) forms task vectors as the elementwise difference between fine-tuned and pretrained weights and combines them additively with a global scaling coefficient α_{TA}, producing a merged model

$$\theta_{\mathrm{pre}} + \alpha_{\mathrm{TA}} \sum_t (\theta_t - \theta_{\mathrm{pre}}),$$

where θ_t is the per-task expert. The sweep grid for α_{TA} is fixed in advance. For each reported run, the validation-selected coefficient is chosen on the corresponding held-out validation slice before test evaluation; selected values are reported in Appendix D. **TIES-Merging** (Yadav et al., 2023) resolves interference between task vectors before aggregation: it trims each delta to its top-magnitude entries, elects a sign per parameter from the trimmed deltas, and aggregates only entries that agree with the elected sign, scaled by a separate global coefficient α_{TIES}. The density and α_{TIES} are selected from fixed validation grids for each reported run before test evaluation; validation-selected values are reported in Appendix D following standard hyperparameter-disclosure practice (Dodge et al., 2019).

Each merged model is evaluated under two readout conditions on identical test partitions. The **native** condition uses the classifier/readout weights delivered by the merging method itself: the per-task heads are combined under the same parameter-space protocol applied to the backbone. The **fresh-head** condition retrains only a single union-label linear classifier on the frozen merged backbone and is defined in Section 3.3. Two further comparators are evaluated on the same data under the same evaluation protocol: a **naive output-averaging ensemble**, which averages logit vectors from all per-task experts at inference time, and a **centroid-routed modular composer**, which selects an expert subset by centroid similarity in the frozen pretrained backbone feature space and combines the selected experts' class-set-masked logits as a softmax-weighted sum.

Fixed	Changed / referenced	Identifies	Does not identify
Merged backbone; union label space; train/test split; readout family	Delivered native readout vs re-estimated fresh head	Readout-recoverable component of the native shortfall	Absolute representational quality of the merged backbone
Native score; fresh-head score; joint-training protocol	Native-to-joint gap used as calibration reference	Fraction of the supervised headroom closed by readout replacement	Whether the unrecovered residual is representation-side, optimisation-side, or data-limited
Fresh-head protocol; merging methods; Ward decomposition	Backbone family: ViT-B/16 vs ResNet-50	Cross-backbone reproducibility of the readout-recovery direction	Universality across architectures, datasets, or non-vision modalities
Dataset; merging/composition methods; evaluation protocol	Task partition: structured Ward decomposition vs random class-to-task split	Whether modular-composition advantage is geometry-conditioned in the tested regime	Whether the readout-locus diagnosis reproduces under non-clustered task splits unless explicitly tested

Table 1: Identification contract for the fresh-head probe and geometry control. The probe identifies readout-recoverable native shortfall under matched evaluation; it does not certify that the merged representation is undamaged.

3.3 Fresh-head probe

The fresh-head probe is a matched-interface intervention on the delivered merged model. It changes only the readout estimation source: the native condition uses the readout delivered by the merge, while the fresh-head condition uses a newly estimated linear readout trained on the same union label space. The merged backbone weights, union label space, train/val/test split, and readout family (a single linear layer of matched output cardinality) are held fixed between the two conditions.

Concretely, for each merged model produced under Section 3.2 (Task Arithmetic or TIES-Merging at its validation-selected hyperparameters), we (i) freeze every backbone parameter, (ii) replace the merged classifier head with a freshly initialised linear layer of the same input dimension and output cardinality as the union label space, and (iii) train only the new head on the union of all task training partitions for 10 epochs with AdamW at learning rate 10^{-3}, weight decay 10^{-4}, and batch size 64. The best head checkpoint by union-validation accuracy is retained and evaluated on the union test set. The native condition is evaluated on the same test data under the same evaluation protocol, with the merged classifier used unchanged.

The native–fresh-head delta therefore isolates the effect of replacing the delivered readout while holding the merged backbone fixed. Section 3.4 states explicitly what this delta does and does not identify.

3.4 Identification contract

The fresh-head probe and the geometry control each isolate a specific quantity under matched conditions; neither identifies a property of the system in general. Table 1 states the contract row by row: what is held fixed, what is changed or used as the comparison reference, what the contrast identifies, and what it does not identify. Claims that match a "Does not identify" entry are not supported by this paper's evidence and are not made.

The four rows correspond to the paper's four scoped contrasts: the matched-interface intervention defining the probe, the joint-training reference that converts the raw native–fresh-head delta into a calibrated fraction of supervised headroom, the cross-backbone replication that tests whether the readout-recovery direction persists across two architectures, and the geometry control that bounds when the modular-composition advantage holds. Each "Does not identify" entry is a deliberate scope commitment, not an omission; Section 6

discusses cases where extending the probe to the corresponding object would require evidence this paper does not produce.

4 Results

All reported uncertainties are sample standard deviations across model seeds; we use them to describe run-to-run variation rather than as formal hypothesis tests.

4.1 Native end-to-end scores are locus-ambiguous

Under the disjoint-support Ward decomposition of CIFAR-100, native parameter-space merging produces large end-to-end shortfalls on the union 100-class task for both backbone families. On ViT-B/16, Task Arithmetic reaches 0.203 ± 0.013 and TIES-Merging reaches 0.131 ± 0.017; on ResNet-50, the corresponding scores are 0.428 ± 0.025 and 0.489 ± 0.010 (five model seeds; sample standard deviation, ddof $= 1$). The joint-training reference under the same union-label protocol reaches 0.885 ± 0.004 on ViT-B/16 and 0.857 ± 0.002 on ResNet-50. These native scores therefore establish the severity of the delivered-model shortfall. They do not, by themselves, establish whether that shortfall entered through the merged backbone, the delivered readout, or the interface between them.

A second observation reinforces this locus ambiguity at the level of method ranking. On ViT-B/16, TIES-Merging underperforms Task Arithmetic under the native readout (0.131 vs. 0.203); on ResNet-50, the ordering reverses (0.489 vs. 0.428). This inversion should not be over-read as a general ranking result over the two methods. It shows something narrower and more diagnostic for this paper: in the tested protocol, the native ranking is backbone-conditioned. The native end-to-end number is therefore not a method-intrinsic severity measure; it reflects an interaction between the merge operator, backbone family, and readout delivered by the merge. The next subsection tests the readout component of this interaction directly by replacing only the delivered readout while holding the merged backbone fixed.

4.2 Most of the native shortfall is readout-recoverable in the tested regimes

Under the matched-interface intervention defined in Section 3.3, freezing the merged backbone and retraining only a union-label linear classifier recovers most of the native shortfall on both backbone families. On ViT-B/16, Task Arithmetic recovers from 0.203 ± 0.013 to 0.878 ± 0.001, and TIES-Merging recovers from 0.131 ± 0.017 to 0.851 ± 0.004 (five model seeds; ddof $= 1$). On ResNet-50, Task Arithmetic recovers from 0.428 ± 0.025 to 0.786 ± 0.004, and TIES-Merging recovers from 0.489 ± 0.010 to 0.791 ± 0.003. Calibrated against the joint-training reference, the fresh-head probe closes approximately 99.0% (Task Arithmetic) and 95.5% (TIES-Merging) of the native-to-joint gap on ViT-B/16, and approximately 83.5% and 82.1% on ResNet-50. These ratios are computed from the reported means. The dominant component of the native shortfall is therefore readout-recoverable in both backbone families under the Ward decomposition. Figure 1 visualises this contrast across both backbone families and both partition regimes.

Two stronger interpretations are unsupported by the probe and are not made. First, the result does not show that the merged backbones are undamaged: it shows that a freshly estimated linear readout, trained under matched union-label supervision, can recover most of the supervised headroom from the frozen merged backbone. Second, the result does not show that readout-interface failure is the entire story. On ResNet-50, approximately 16–18% of the native-to-joint gap remains unrecovered, and the probe does not assign that residual to representation damage, readout optimisation, data limitations, or any other specific source. Per the identification contract in Table 1, rows 1 and 2, the defensible claim is narrower: most of the native shortfall is readout-recoverable, while the unrecovered residual remains unlocalised.

The cross-backbone pattern is nevertheless informative. The recovered fraction is smaller on ResNet-50 than on ViT-B/16, but the direction of the effect is stable: replacing the delivered readout with a freshly estimated one substantially improves end-to-end performance for both merging methods on both backbone families. This rules out a ViT-specific interpretation of the fresh-head recovery direction, while preserving the regime clause that the magnitude of recovery is backbone-conditioned in the tested protocol.

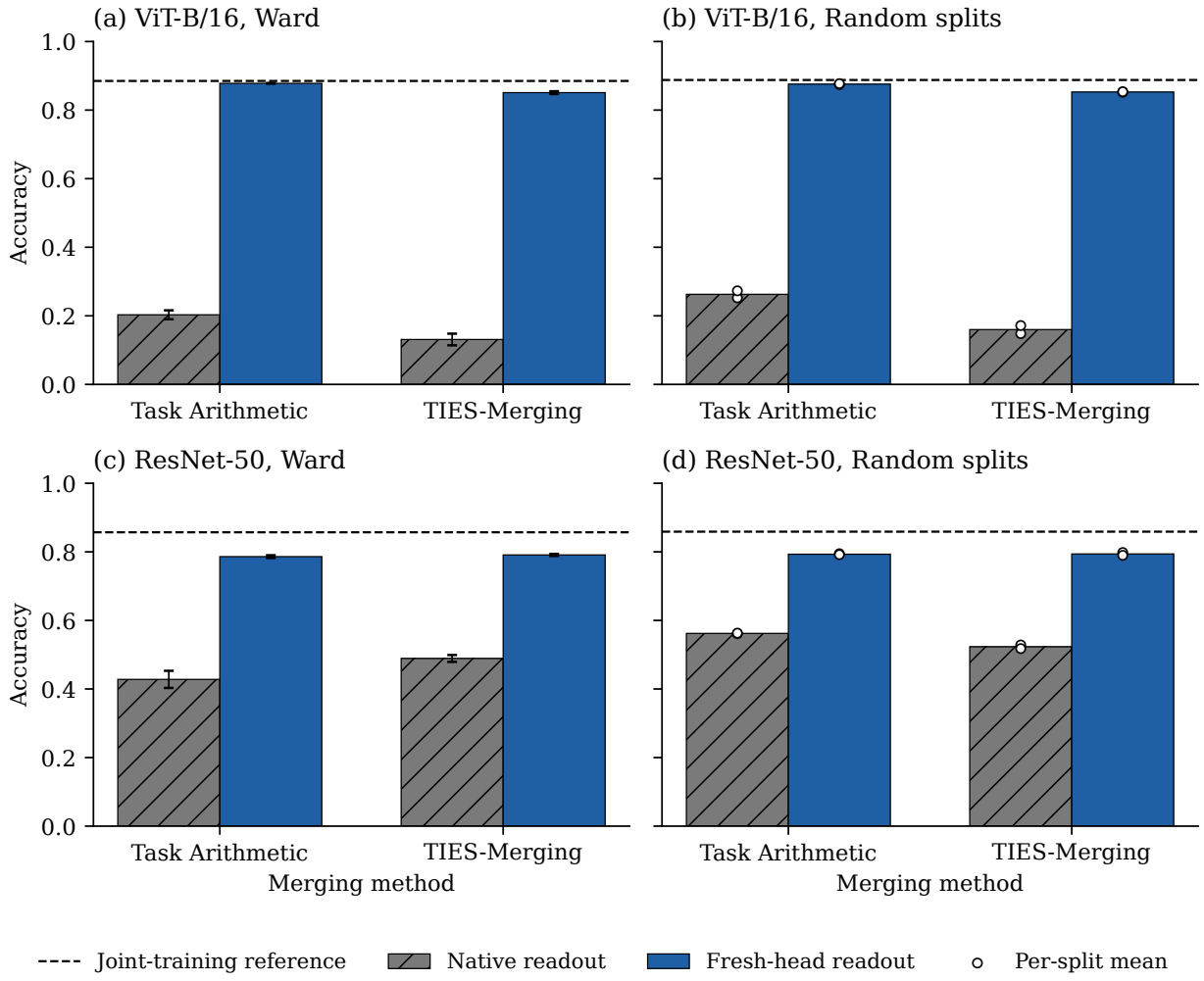

Figure 1: Fresh-head recovery separates delivered-model performance from readout-recoverable shortfall. Each panel reports native and fresh-head accuracy for Task Arithmetic and TIES-Merging under a fixed backbone and task-partition regime; the dashed line gives the corresponding joint-training reference used to calibrate the native-to-joint gap. Across both backbone families and both CIFAR-100 partition regimes, replacing only the delivered readout recovers a substantial share of the native shortfall. The figure visualises the paper's central identification contrast: the native score is a valid delivered-model verdict, but it does not by itself localise whether failure entered through the merged representation or the delivered readout. Ward panels (a, c) report mean ± sample SD over five model seeds; random panels (b, d) plot the average of the two split-level means with per-split means overlaid as faint markers, and use the average of the two split-specific joint-training references for visualisation only.

Within CIFAR-100 on ViT-B/16, the readout-recovery direction is not confined to the structured Ward decomposition: it persists under two random class-to-task partitions. Under random-split seed 100, Task Arithmetic recovers from 0.2524 ± 0.0108 to 0.8746 ± 0.0053, and TIES-Merging recovers from 0.1482 ± 0.0209 to 0.8521 ± 0.0034 (three model seeds; ddof = 1). Under random-split seed 200, the corresponding recoveries are 0.2729 ± 0.0191 to 0.8771 ± 0.0027, and 0.1716 ± 0.0134 to 0.8537 ± 0.0021. Calibrated against split-specific joint-training references, the fresh-head probe closes approximately 97.8% (Task Arithmetic) and 95.1% (TIES-Merging) of the native-to-joint gap on random-split 100, and approximately 98.4% and 95.3% on random-split 200. These fractions are comparable to the Ward-ViT values reported above (99.0% for Task Arithmetic; 95.5% for TIES-Merging). The result extends the readout-recovery evidence beyond the

Backbone	Geometry	Composer	Ensemble	Composer–Ensemble
ViT-B/16	Ward	0.8525 ± 0.0042	0.7626 ± 0.0063	$+0.0899 \pm 0.0068$
ViT-B/16	Random 100	0.6635 ± 0.0104	0.6602 ± 0.0160	$+0.0034 \pm 0.0120$
ViT-B/16	Random 200	0.6655 ± 0.0038	0.6750 ± 0.0025	-0.0095 ± 0.0061
ResNet-50	Ward	0.7785 ± 0.0041	0.7023 ± 0.0064	$+0.0762 \pm 0.0054$
ResNet-50	Random 100	0.5967 ± 0.0019	0.6588 ± 0.0064	-0.0620 ± 0.0046
ResNet-50	Random 200	0.5953 ± 0.0042	0.6615 ± 0.0036	-0.0662 ± 0.0035

Table 2: Geometry boundary for centroid-routed modular composition. Values are mean accuracy ± sample standard deviation over model seeds; deltas are computed per seed before aggregation. Ward rows use five seeds; random rows use three seeds per split.

structured Ward partition within CIFAR-100, while remaining scoped to ViT-B/16, the two tested merging methods, and two random partitions. The corresponding ResNet-50 random-partition result completes the cross-backbone check within the CIFAR-100 random-split regime.

On ResNet-50, applying the same fresh-head probe to the same random partitions gives the corresponding cross-backbone check. Under random-split seed 100, Task Arithmetic recovers from 0.5613 ± 0.0171 to 0.7944 ± 0.0045, and TIES-Merging recovers from 0.5283 ± 0.0069 to 0.7979 ± 0.0026 (three model seeds; ddof = 1). Under random-split seed 200, the corresponding recoveries are 0.5629 ± 0.0051 to 0.7918 ± 0.0039, and 0.5178 ± 0.0237 to 0.7898 ± 0.0181. Calibrated against split-specific ResNet-50 joint-training references, the fresh-head probe closes 78.5% and 77.2% of the native-to-joint gap for Task Arithmetic, and 81.7% and 79.6% for TIES-Merging, across random-split seeds 100 and 200 respectively. These fractions are lower than the corresponding ViT-B/16 random-partition fractions and slightly lower than ResNet-50 Ward, but every tested backbone–geometry–method cell remains above 0.77. Thus, within CIFAR-100, the readout-recoverable component is dominant across both backbone families, both partition regimes, and both merging methods tested. The readout-locus diagnosis is therefore cross-backbone and cross-partition within the tested CIFAR-100 regimes; it remains bounded to CIFAR-100 and to Task Arithmetic and TIES-Merging.

4.3 The modular-composition advantage is geometry-conditioned in the tested regime

Table 2 compares the centroid-routed modular composer against naive output-averaging under structured Ward and random class-to-task decompositions. Under Ward geometry, the composer outperforms the ensemble on both backbones: the per-seed composer–ensemble delta is $+0.0899 \pm 0.0068$ on ViT-B/16 and $+0.0762 \pm 0.0054$ on ResNet-50. The structured-Ward advantage therefore reproduces directionally across the two backbone families tested.

Under random class-to-task partitions, the Ward advantage does not persist. On ViT-B/16, the composer–ensemble deltas straddle zero across the two random partitions. On ResNet-50, the sign reverses: the composer underperforms the ensemble by roughly six percentage points on both random partitions. The result is not that centroid routing generally fails, nor that modular composition is universally geometry-dependent. The bounded conclusion is narrower: in the tested CIFAR-100 regimes, centroid-routed composition is useful when the task partition aligns with the structured Ward decomposition, but its advantage disappears or reverses under random class partitions.

This geometry result also separates Claim 3 from the readout-recovery result in Section 4.2. The fresh-head recovery direction persists across the Ward and random partitions on both backbone families, while the modular-composition advantage changes sign with task-partition geometry. Per the identification contract in Table 1, row 4, this boundary is established for the tested CIFAR-100 decompositions and two backbone families; whether the same boundary holds outside CIFAR-100 is not established here.

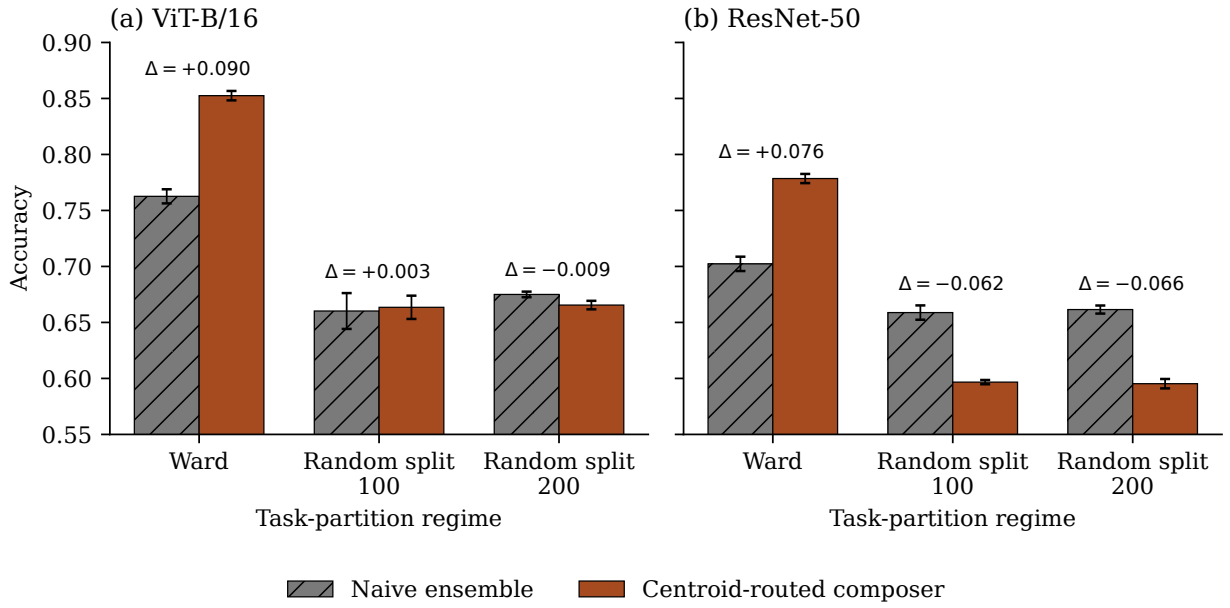

Figure 2: Task-partition geometry bounds the advantage of centroid-routed modular composition. Under the structured Ward decomposition, the centroid-routed composer outperforms naive output-averaging on both ViT-B/16 and ResNet-50. Under random class-to-task partitions, that advantage disappears on ViT-B/16 and reverses on ResNet-50. This contrast is not a claim that centroid routing fails generally; it shows that, in the tested CIFAR-100 regimes, the behavioural value of modular composition depends on whether the geometry of the routing feature space supports the task partition. Bars report mean ± sample SD over model seeds (Ward $N = 5$; random $N = 3$); Δ annotations are per-seed-delta means.

5 Discussion

The central lesson is not that native end-to-end scores are wrong, but that they are overloaded. A native score is a valid verdict on the model a merging procedure actually delivers: it tells us how that merged backbone and its delivered readout perform as a single system. It is not, without an additional intervention, a valid diagnosis of what failed inside that system. The fresh-head results in Section 4.2 make this distinction operational. When replacing only the readout recovers most of the native shortfall under the tested regimes, the representation-damage interpretation of the native score becomes under-identified: the same low number is compatible with damaged features, an incompatible delivered readout, or a mismatch between the two. The contribution is therefore an evidential correction to model-merging evaluation. The paper does not discard the native score; it narrows what that score is allowed to mean.

This distinction separates three objects that are often compressed into a single post-merge score: the merge operator, the delivered readout, and the evidential interpretation of the resulting accuracy. Ilharco et al. (2023) and Yadav et al. (2023) act on the first object by making task-vector composition and interference control more explicit before aggregation. Kong et al. (2024) acts on the second by treating classifier–backbone compatibility as a repairable source of degradation in merged classifiers. These advances make the merged system better specified, but they do not by themselves determine what a low native score is evidence for. A score produced after operator-level merging and readout-level alignment can still serve two roles: it can evaluate the model that was delivered, and it can be used, incorrectly, to infer where that model failed. The fresh-head probe targets this remaining evidential layer. It holds the merge fixed and changes only the readout estimation source, thereby separating delivered-model performance from the readout-recoverable component of the native shortfall.

The evaluation consequence is not simply to add another score, but to separate estimands that the native protocol collapses. For disjoint-support classification merges of the kind tested here, the native score estimates delivered-system utility: the performance of the merged backbone together with the readout produced by the merge. The fresh-head contrast estimates a different object: the component of that native shortfall that disappears when the backbone, label space, data split, and readout family are held fixed, but the readout is re-estimated from union-label supervision. These two quantities answer different questions. The first asks whether the merge, as delivered, works. The second asks how much of its failure survives after the delivered readout is removed as the bottleneck. The fresh-head probe is therefore distinct in scope from softer readout corrections such as temperature scaling (Guo et al., 2017), which rescale delivered logits rather than refit the readout. The probe re-estimates the readout itself under matched union-label supervision, so it isolates a different quantity: not whether the delivered readout is well-tempered, but how much of the native shortfall remains when a freshly estimated linear readout is fit on the frozen merged backbone. This distinction matters because a low native score is otherwise allowed to masquerade as evidence about representation damage, even when most of the shortfall is readout-recoverable. The fresh-head score is therefore not a rescue score and not a deployment recipe; it is a controlled counterfactual that prevents delivered-model failure from being over-read as representation failure.

The two §4 findings both involve task-partition geometry, but they concern different scientific objects. Section 4.2 reports a property of the diagnostic itself: the fresh-head recovery direction holds under both Ward and random class-to-task partitions on both backbone families, so the readout-locus identification is not confined to the structured Ward partition in the tested CIFAR-100 regimes. Section 4.3 reports a property of a method: centroid-routed modular composition outperforms naive output-averaging under Ward on both backbones; under random partitions, the advantage does not persist on ViT-B/16 and reverses on ResNet-50. Diagnostic generalisation and method generalisation are not the same evidence. The first tells us that the identified quantity, the readout-recoverable share of the native shortfall, is robust to task-partition geometry within the tested CIFAR-100 regimes; the second tells us that one specific composition method's behavioural advantage is not. The bounded interpretation is therefore that the readout-locus diagnosis applies across both partitions and both backbone families, while centroid-routed modular composition is geometry-conditioned. They should therefore not be collapsed into a single claim.

6 Limitations

The experiments above make the post-merge score more interpretable, but they do not make it fully diagnostic. Three boundaries remain: the unrecovered residual is unassigned, the empirical regime is controlled but narrow, and the fresh-head probe is an evaluation counterfactual rather than a deployment procedure.

The first boundary is residual attribution. The fresh-head probe identifies one operationally defined component of the native shortfall: the part removed when the delivered readout is replaced under matched supervision. It leaves the remaining residual unassigned. On ResNet-50 under Ward, that residual is approximately 16–18% of the supervised headroom. The probe does not distinguish whether this residual reflects representation damage, readout-optimisation limits, finite-data effects in fitting the fresh head, remaining interface mismatch, or some combination of these factors. This limitation follows from the intervention itself: the probe varies only the readout estimation source while holding the merged backbone fixed. Separating the residual into causal components would require additional interventions that vary the representation, readout capacity, optimisation budget, and supervision separately. The paper's claim therefore stops at the readout-recoverable component of the native shortfall. It does not certify that the merged backbone is undamaged, and it does not assign the unrecovered residual.

A second scope condition concerns external validity. The paper deliberately works in a controlled regime: one parent dataset, CIFAR-100; two backbone families, ViT-B/16 and ResNet-50; two parameter-space merging methods, Task Arithmetic and TIES-Merging; and one structured Ward decomposition together with two random class-to-task partitions. This restriction is not incidental. The central claim depends on comparing a native score, a fresh-head contrast, a joint-training reference, and a geometry control under matched data, labels, backbones, and task partitions. Broadening the regime without preserving those controls would make the result wider but less interpretable. The cost of this control is external validity. The experiments do

not establish that the same recovered fractions, native-score ambiguities, or modular-composition boundary will hold on larger benchmarks, non-CIFAR vision domains, additional backbone families, non-vision tasks, or large-scale transformer merging regimes of the kind studied by Yadav et al. (2025). In particular, the paper does not test regression or probabilistic approaches such as RegMean and Fisher-weighted averaging, averaging-based methods such as model soups, or adaptive and sparsified delta methods such as AdaMerging, DARE, and Model Breadcrumbs. The paper therefore establishes a controlled regime, not a general law. Evidence from untested settings would be needed to redraw that regime boundary; it is not licensed by the present experiments alone.

The final scope condition concerns the role of the fresh-head probe. The probe is an evaluation counterfactual, not a deployable merge: it measures the component of the native shortfall removed when the delivered readout is replaced under matched supervision. That supervision is part of the intervention. The fresh-head readout is fitted on union-label data, the kind of supervision that data-free or low-data merging protocols typically seek to avoid, so its accuracy is not the deployment number of the merging procedure. Nor is the probe a replacement for operator-level merging methods (Ilharco et al., 2023; Yadav et al., 2023) or classifier–backbone alignment procedures (Kong et al., 2024), which address a different problem: how to produce a better delivered model under their own data-access constraints. The fresh-head probe instead changes the evidence available to the evaluator. It asks how much of the delivered model's shortfall is readout-recoverable once the merged backbone is held fixed; it does not show how to obtain the same recovery without union-label supervision. Treating the fresh-head accuracy as the model that the merging procedure should ship would therefore conflate the counterfactual used to interpret the system with the system being evaluated.

7 Conclusion

A native post-merge score answers one question cleanly and another only ambiguously: it tells us how the delivered merged model performs, but not where a failure in that model entered. The fresh-head probe turns that ambiguity into a controlled contrast by holding the merged backbone fixed and replacing only the delivered readout under matched supervision. Across the tested CIFAR-100 regimes, this contrast shows that much of what appears as native model-merging failure is readout-recoverable across both backbone families and both partition regimes, while the geometry control shows that centroid-routed modular composition is useful only when the task partition supports it. These results do not make the native score obsolete; they make its evidential boundary explicit. The contribution is therefore not a new merge operator, but a stricter interpretation of post-merge evidence: model merging does not only need better ways to combine weights, but better discipline about what its scores are allowed to diagnose.

References

Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. In *International Conference on Learning Representations, ICLR 2023*, 2023.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2016. arXiv:1610.01644.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Nazanin Mohammadi Sepahvand, Edward Raff, Kanika Madan, Vikram Voleti, Samira Ebrahimi Kahou, Vincent Michalski, Tal Arbel, Chris Pal, Gael Varoquaux, and Pascal Vincent. Accounting for variance in machine learning benchmarks. In *Proceedings of Machine Learning and Systems 3, MLSys 2021*, 2021.

MohammadReza Davari and Eugene Belilovsky. Model breadcrumbs: Scaling multi-task model merging with sparse masks. In *European Conference on Computer Vision, ECCV 2024*, 2024.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. Show your work: Improved reporting of experimental results. In *Proceedings of the 2019 Conference on Empirical Methods in*

Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, 2019.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning, ICML 2020*, 2020.

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

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*, 2019.

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*, 2023.

Xisen Jin, Xiang Ren, Daniel Preoţiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. In *International Conference on Learning Representations, ICLR 2023*, 2023.

Fanshuang Kong, Richong Zhang, Zhijie Nie, Hang Zhou, Ziqiao Wang, Qiang Sun, and Chunming Hu. MOMA: Masked orthogonal matrix alignment for zero-additional-parameter model merging, 2024. arXiv:2412.13526 (v3).

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

Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations, ICLR 2022*, 2022.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. GShard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations, ICLR 2021*, 2021.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. Branch-train-merge: Embarrassingly parallel training of expert language models, 2022. arXiv:2208.03306.

Michael S. Matena and Colin Raffel. Merging models with fisher-weighted averaging. In *Advances in Neural Information Processing Systems 35, NeurIPS 2022*, 2022.

Guillermo Ortiz-Jiménez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. In *Advances in Neural Information Processing Systems 36, NeurIPS 2023*, 2023.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations, ICLR 2017*, 2017.

George Stoica, Daniel Bolya, Jakob Brandt Bjorner, Pratik Ramesh, Taylor Hearn, and Judy Hoffman. ZipIt! merging models from different tasks without training. In *International Conference on Learning Representations, ICLR 2024*, 2024.

Joe H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak 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 *International Conference on Machine Learning, ICML 2022*, 2022.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2019*, pp. 374–382, 2019.

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 36, NeurIPS 2023*, 2023.

Prateek Yadav, Tu Vu, Jonathan Lai, Alexandra Chronopoulou, Manaal Faruqui, Mohit Bansal, and Tsendsuren Munkhdalai. What matters for model merging at scale? *Transactions on Machine Learning Research*, 2025.

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*, 2024.

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 *International Conference on Machine Learning, ICML 2024*, 2024.

A Dataset and partitions

A.1 CIFAR-100

All experiments use CIFAR-100 (Krizhevsky, 2009), which contains 60,000 colour images of native resolution 32×32 across 100 classes, with 50,000 training images and 10,000 test images. Validation slices used for merging-method hyperparameter selection and fresh-head checkpoint selection are drawn from the training split as specified in Appendices B–C. Test images are not used for hyperparameter selection.

A.2 Ward decomposition

The Ward partition assigns the 100 classes to five disjoint tasks by hierarchical agglomerative clustering of class-mean features. Per-class features are computed by passing CIFAR-100 training images through the frozen pretrained ViT-B/16 backbone under the evaluation transform and averaging the resulting penultimate-layer features per class. Ward linkage (Ward Jr., 1963) is then applied to the 100 class-mean vectors with target cluster count $K = 5$. The resulting partition has variable task sizes of 5, 26, 20, 24, and 25 classes, summing to 100. The same Ward class-to-task assignment is used for both ViT-B/16 and ResNet-50 experiments, so cross-backbone comparisons under Ward hold the partition fixed. The released manifests contain the full class-to-task assignments.

A.3 Random class-to-task partitions

The geometry-control random partitions are generated deterministically from a partition seed. For each seed in $\{100, 200\}$, the 100 CIFAR-100 class indices are uniformly permuted using `torch.randperm(100)` under a seeded generator and split into five contiguous blocks of 20 indices each. This yields two balanced random partitions, each containing five disjoint 20-class tasks. The same random-partition manifests are used across both backbone families, so cross-backbone comparisons under random geometry hold the class-to-task assignment fixed.

B Training protocols

B.1 Backbones

Two backbone families are used. Both are sourced from the `timm` library and loaded with pretrained weights.

- **ViT-B/16**: `vit_base_patch16_224.augreg2_in21k_ft_in1k`; pretrained on ImageNet-21k and fine-tuned on ImageNet-1k.
- **ResNet-50**: `resnet50.a1_in1k`; pretrained on ImageNet-1k.

Input resolution is 224×224 for both backbones; CIFAR-100 images (native 32×32) are upsampled by torchvision `Resize` (bilinear default) before model input. The training transform composes (in order): resize to 224, random crop to 224 with padding 28, random horizontal flip, RandAugment ($N = 2$, $M = 9$), conversion to tensor, and ImageNet normalisation (mean $(0.485, 0.456, 0.406)$, standard deviation $(0.229, 0.224, 0.225)$). The evaluation transform composes resize to 224, conversion to tensor, and the same ImageNet normalisation. The same transforms are used for both backbone families and across Ward and random partitions.

B.2 Expert training

Each per-task expert is fine-tuned from the pretrained backbone on the task's training images.

- **Trainable parameters**: all backbone parameters and a freshly initialised 100-class linear classifier head; no parameters are frozen.
- **Loss**: cross-entropy over the 100-class output.

- **Optimizer**: AdamW.
- **Learning rate**: 1×10^{-4}.
- **Weight decay**: 0.05.
- **Learning-rate schedule**: cosine learning-rate schedule.
- **Epochs**: 30.
- **Batch size**: 64.
- **Augmentation**: training transform above; evaluation uses the evaluation transform above.
- **Validation split**: a 10% validation slice is drawn from the task's training images by the seeded dataset wrapper (`val_fraction=0.1`); the remaining 90% is used for fine-tuning. Test images are not used for model selection.
- **Checkpoint selection**: at each epoch the model is evaluated on the per-task validation slice; the checkpoint with maximum validation accuracy across the 30 epochs is retained as the expert.
- **Seeding**: the model seed is the unit of randomness control. Per-task RNG streams are decoupled within a model seed via `seed_everything(seed + task_id)` prior to each task's training run, so that parameter initialisation and stochastic augmentation differ across tasks within the same model seed.

The same recipe is used for both backbone families and for both Ward and random partitions; only the task partition and the model seed vary.

B.3 Joint-training reference

The joint-training reference is a single classifier trained on the union of all task training data for a given (backbone, partition, model seed) triple. It is not a merging baseline; it provides the partition-matched calibration reference used to express fresh-head recovery as a fraction of the native-to-joint gap in Section 4.

- **Trainable parameters**: full backbone plus a 100-class linear classifier head, initialised from the same pretrained weights as the corresponding expert runs.
- **Loss, optimizer, learning rate, weight decay, schedule, epochs, batch size, augmentation, evaluation transform**: identical to the expert recipe (AdamW, 1×10^{-4}, 0.05, cosine schedule, 30 epochs, batch 64).
- **Validation split**: the union of the per-task validation slices, formed by concatenating each task's validation dataset under the same seeded dataset-wrapper invocation used during expert training. No test images enter the joint-training pipeline.
- **Checkpoint selection**: at each epoch the model is evaluated on the union validation slice; the checkpoint with maximum union-validation accuracy across the 30 epochs is retained.
- **Per-partition computation**: the joint-training reference is computed separately for the Ward partition, for random partition seed 100, and for random partition seed 200. Recovered fractions in Section 4 use the partition-matched joint reference as the denominator.

Recipe parity between expert training and the joint-training reference is intentional: it compares the delivered merged system with a same-recipe single-model reference under the same partition and data protocol.

C Probe and composer protocols

C.1 Fresh-head probe

The fresh-head probe changes only the readout estimation source while holding the merged backbone fixed.

- **Input**: a merged model produced by Task Arithmetic or TIES-Merging at its validation-selected hyperparameters. The merged backbone state is loaded into a fresh model instance; the delivered classifier head is then discarded and replaced by the fresh-head module.
- **Backbone freezing**: all loaded backbone parameters are set to `requires_grad=False`. No backbone parameter is updated during the probe.
- **Classifier head**: the delivered classifier module is replaced with a freshly initialised `nn.Linear` layer whose input dimension equals the backbone feature dimension and whose output dimension is 100. The new head is the only trainable module. It is reinitialised once per (model seed, merging method, partition) run, so no delivered classifier weights are reused.
- **Training data**: the union of all task training images for the corresponding partition, formed by `ConcatDataset` over task training splits. The training transform from Appendix B is applied.
- **Validation data**: the union of all task validation slices for the corresponding partition, formed by `ConcatDataset` over task validation splits. The evaluation transform from Appendix B is applied. Test images are not used during training, validation, or checkpoint selection.
- **Loss**: cross-entropy over the global 100-class output space.
- **Optimizer**: AdamW over the fresh-head parameters only.
- **Learning rate**: 1×10^{-3}.
- **Weight decay**: 1×10^{-4}.
- **Epochs**: 10.
- **Batch size**: 64.
- **Checkpoint selection**: the checkpoint with maximum union-validation accuracy across the 10 epochs is retained.
- **Test evaluation**: the retained checkpoint is evaluated on the task test splits for the corresponding partition; per-task counts are aggregated by sample count into the reported overall accuracy.

The protocol is applied identically across backbone families, partition regimes, merging methods, and model seeds. The 10-epoch budget applies only to the freshly initialised linear head; it is not a matched full-model training budget.

C.2 Centroid-routed modular composer

The composer combines per-task expert logits at inference time without fusing the expert parameters.

- **Routing backbone**: a pretrained backbone of the same family as the experts, loaded without a classifier head and set to `eval()` mode. All routing-backbone parameters have `requires_grad=False`. The routing backbone produces features for centroid construction and inference-time queries; it is not trained.
- **Per-expert centroid**: for each expert, the centroid is the mean routing-backbone feature over that expert's training images, computed once at registration time. Centroids are stored before evaluation.

- **Query features**: for each test input, query features are extracted by the routing backbone in `eval()` mode under `torch.no_grad()`.
- **Routing similarity**: query features and centroids are L2-normalised, and cosine similarity is computed as their inner product, yielding a (B, N) similarity matrix for batch size B and N stored experts.
- **Top-k selection**: the top-k experts by similarity are selected independently for each input. All reported runs use $k = 3$.
- **Routing weights**: the selected similarities are converted to weights by applying $\mathrm{softmax}(s_i/\tau)$ over the selected experts. All reported runs use $\tau = 1.0$.
- **Class-set masks**: each expert carries a binary mask over the global 100-class output space, with entries set to `True` for classes in that expert's training class set and `False` otherwise.
- **Per-input masking**: for each input independently, only the selected experts' logits on their supported class sets are allowed to contribute. Masking is applied per input, not as a batch-wide union over selected experts.
- **Logit aggregation**: for each output class covered by at least one selected expert, the final logit is the routing-weighted sum of the contributing masked logits. Classes covered by no selected expert for that input receive a large negative sentinel, -10^9, before prediction.
- **Disjoint-support simplification**: in the partitions used here, each global class is assigned to exactly one expert. Therefore, for a covered class, the aggregation reduces to the selected expert's masked logit scaled by its routing weight.

The composer has no learned routing parameters. The routing backbone, k, τ, centroids, and class-set masks are fixed before evaluation.

D Numerical audit trail

Backbone	Geometry	Split	Seed	α_{TA}	Validation accuracy
ViT-B/16	Ward	–	0	0.25	0.1996
ViT-B/16	Ward	–	1	0.25	0.2232
ViT-B/16	Ward	–	2	0.25	0.1932
ViT-B/16	Ward	–	3	0.25	0.2048
ViT-B/16	Ward	–	4	0.25	0.2006
ViT-B/16	Random	100	0	0.25	0.2324
ViT-B/16	Random	100	1	0.25	0.2604
ViT-B/16	Random	100	2	0.25	0.2584
ViT-B/16	Random	200	0	0.25	0.2754
ViT-B/16	Random	200	1	0.25	0.2532
ViT-B/16	Random	200	2	0.25	0.2718
ResNet-50	Ward	–	0	0.25	0.4618
ResNet-50	Ward	–	1	0.25	0.4330
ResNet-50	Ward	–	2	0.25	0.4066
ResNet-50	Ward	–	3	0.25	0.4366
ResNet-50	Ward	–	4	0.25	0.4096
ResNet-50	Random	100	0	0.25	0.5798
ResNet-50	Random	100	1	0.25	0.5622
ResNet-50	Random	100	2	0.25	0.5452
ResNet-50	Random	200	0	0.25	0.5696
ResNet-50	Random	200	1	0.25	0.5506
ResNet-50	Random	200	2	0.25	0.5582

Table 3: Task Arithmetic: validation-selected scaling coefficient α_{TA} and corresponding validation accuracy at the selected setting per (backbone, geometry, partition seed, model seed).

Backbone	Geometry	Split	Seed	Density	α_{TIES}	Validation accuracy
ViT-B/16	Ward	–	0	0.5	1.0	0.1092
ViT-B/16	Ward	–	1	0.5	1.0	0.1446
ViT-B/16	Ward	–	2	0.5	1.0	0.1272
ViT-B/16	Ward	–	3	0.5	1.0	0.1388
ViT-B/16	Ward	–	4	0.5	1.0	0.1392
ViT-B/16	Random	100	0	0.5	1.0	0.1254
ViT-B/16	Random	100	1	0.5	1.0	0.1546
ViT-B/16	Random	100	2	0.5	1.0	0.1616
ViT-B/16	Random	200	0	0.5	1.0	0.1616
ViT-B/16	Random	200	1	0.5	1.0	0.1730
ViT-B/16	Random	200	2	0.5	1.0	0.1844
ResNet-50	Ward	–	0	0.1	1.0	0.5152
ResNet-50	Ward	–	1	0.1	1.0	0.4972
ResNet-50	Ward	–	2	0.1	1.0	0.4864
ResNet-50	Ward	–	3	0.1	1.0	0.5054
ResNet-50	Ward	–	4	0.1	1.0	0.4836
ResNet-50	Random	100	0	0.1	1.0	0.5416
ResNet-50	Random	100	1	0.1	1.0	0.5298
ResNet-50	Random	100	2	0.1	1.0	0.5332
ResNet-50	Random	200	0	0.1	1.0	0.5138
ResNet-50	Random	200	1	0.2	0.5	0.4940
ResNet-50	Random	200	2	0.1	1.0	0.5350

Table 4: TIES-Merging: validation-selected density and scaling coefficient α_{TIES} and corresponding validation accuracy at the selected setting per (backbone, geometry, partition seed, model seed). One validation-selected cell, ResNet-50 Random 200 seed 1, differs from the modal per-backbone setting, selecting $(0.2, 0.5)$ rather than $(0.1, 1.0)$.

Backbone	Method	Seed	Native	Fresh-head	Joint reference	Recovered fraction
ViT-B/16	TA	0	0.1963	0.8776	0.8898	0.9824
ViT-B/16	TA	1	0.2191	0.8788	0.8831	0.9935
ViT-B/16	TA	2	0.1851	0.8768	0.8820	0.9925
ViT-B/16	TA	3	0.2099	0.8775	0.8890	0.9831
ViT-B/16	TA	4	0.2061	0.8790	0.8835	0.9934
ViT-B/16	TIES	0	0.1037	0.8503	0.8898	0.9498
ViT-B/16	TIES	1	0.1435	0.8548	0.8831	0.9617
ViT-B/16	TIES	2	0.1240	0.8547	0.8820	0.9640
ViT-B/16	TIES	3	0.1394	0.8472	0.8890	0.9442
ViT-B/16	TIES	4	0.1428	0.8487	0.8835	0.9530
ResNet-50	TA	0	0.4587	0.7868	0.8543	0.8294
ResNet-50	TA	1	0.4403	0.7864	0.8577	0.8292
ResNet-50	TA	2	0.4013	0.7861	0.8570	0.8444
ResNet-50	TA	3	0.4352	0.7919	0.8579	0.8439
ResNet-50	TA	4	0.4033	0.7809	0.8603	0.8263
ResNet-50	TIES	0	0.4968	0.7876	0.8543	0.8134
ResNet-50	TIES	1	0.4944	0.7943	0.8577	0.8255
ResNet-50	TIES	2	0.4768	0.7893	0.8570	0.8219
ResNet-50	TIES	3	0.4972	0.7901	0.8579	0.8120
ResNet-50	TIES	4	0.4790	0.7946	0.8603	0.8277

Table 5: Fresh-head recovery per-seed numerics, Ward partition. The recovered fraction is computed as $(\text{FH} - \text{N})/(\text{J} - \text{N})$ using the Ward joint-training reference for the same backbone and model seed.

Backbone	Split	Seed	Method	Native	Fresh-head	Joint reference	Recovered fraction
ViT-B/16	100	0	TA	0.2400	0.8769	0.8898	0.9801
ViT-B/16	100	1	TA	0.2593	0.8686	0.8887	0.9681
ViT-B/16	100	2	TA	0.2580	0.8784	0.8872	0.9860
ViT-B/16	200	0	TA	0.2843	0.8762	0.8883	0.9800
ViT-B/16	200	1	TA	0.2508	0.8750	0.8850	0.9842
ViT-B/16	200	2	TA	0.2836	0.8801	0.8878	0.9873
ViT-B/16	100	0	TIES	0.1250	0.8532	0.8898	0.9521
ViT-B/16	100	1	TIES	0.1540	0.8548	0.8887	0.9539
ViT-B/16	100	2	TIES	0.1655	0.8482	0.8872	0.9460
ViT-B/16	200	0	TIES	0.1596	0.8560	0.8883	0.9557
ViT-B/16	200	1	TIES	0.1690	0.8531	0.8850	0.9554
ViT-B/16	200	2	TIES	0.1861	0.8520	0.8878	0.9490
ResNet-50	100	0	TA	0.5790	0.7991	0.8598	0.7838
ResNet-50	100	1	TA	0.5601	0.7940	0.8579	0.7854
ResNet-50	100	2	TA	0.5448	0.7902	0.8570	0.7860
ResNet-50	200	0	TA	0.5672	0.7879	0.8566	0.7626
ResNet-50	200	1	TA	0.5573	0.7957	0.8642	0.7768
ResNet-50	200	2	TA	0.5643	0.7917	0.8575	0.7756
ResNet-50	100	0	TIES	0.5321	0.8009	0.8598	0.8203
ResNet-50	100	1	TIES	0.5203	0.7965	0.8579	0.8181
ResNet-50	100	2	TIES	0.5325	0.7964	0.8570	0.8133
ResNet-50	200	0	TIES	0.5111	0.7990	0.8566	0.8333
ResNet-50	200	1	TIES	0.4981	0.7689	0.8642	0.7397
ResNet-50	200	2	TIES	0.5441	0.8015	0.8575	0.8213

Table 6: Fresh-head recovery per-seed numerics, random partitions. The recovered fraction is computed as $(\mathrm{FH}-\mathrm{N})/(\mathrm{J}-\mathrm{N})$ using the partition-matched joint-training reference for the same backbone, partition seed, and model seed.

Backbone	Geometry	Split	Seed	Naive ensemble	Composer	Composer − Ensemble
ViT-B/16	Ward	–	0	0.7718	0.8574	+0.0856
ViT-B/16	Ward	–	1	0.7654	0.8463	+0.0809
ViT-B/16	Ward	–	2	0.7618	0.8527	+0.0909
ViT-B/16	Ward	–	3	0.7563	0.8547	+0.0984
ViT-B/16	Ward	–	4	0.7575	0.8512	+0.0937
ResNet-50	Ward	–	0	0.7134	0.7812	+0.0678
ResNet-50	Ward	–	1	0.6997	0.7786	+0.0789
ResNet-50	Ward	–	2	0.6989	0.7796	+0.0807
ResNet-50	Ward	–	3	0.6977	0.7716	+0.0739
ResNet-50	Ward	–	4	0.7020	0.7816	+0.0796
ViT-B/16	Random	100	0	0.6600	0.6725	+0.0125
ViT-B/16	Random	100	1	0.6443	0.6521	+0.0078
ViT-B/16	Random	100	2	0.6762	0.6660	−0.0102
ViT-B/16	Random	200	0	0.6721	0.6691	−0.0030
ViT-B/16	Random	200	1	0.6761	0.6657	−0.0104
ViT-B/16	Random	200	2	0.6767	0.6616	−0.0151
ResNet-50	Random	100	0	0.6532	0.5950	−0.0582
ResNet-50	Random	100	1	0.6573	0.5965	−0.0608
ResNet-50	Random	100	2	0.6658	0.5987	−0.0671
ResNet-50	Random	200	0	0.6656	0.5977	−0.0679
ResNet-50	Random	200	1	0.6590	0.5905	−0.0685
ResNet-50	Random	200	2	0.6599	0.5977	−0.0622

Table 7: Geometry-boundary per-seed numerics. Deltas are computed as composer accuracy minus naive-ensemble accuracy for the same backbone, partition, and model seed.