Meta-Merging by Checkpoint Nowcasting

Albert M. Orozco Camacho [1 2] **Boris Knyazev** [3 4 2] **Eugene Belilovsky** [1 2] **Guy Wolf** [4 2]

Abstract

Model merging—the direct combination of parameters from independently fine-tuned networks—offers a way to compose task-specific capabilities without retraining or ensemble inference. Existing merge methods are often built from hand-crafted arithmetic or sparsification heuristics, leaving open whether general learned weight-space operators can be repurposed for merging directly. We study this question with NiNo, a pre-trained checkpoint-nowcasting meta-network originally designed to predict near-future training states from short checkpoint histories. We show that pre-trained NiNo can be reused as a data-free pairwise meta-merge operator for independently fine-tuned models. On an 8-task CLIP ViT-B/16 benchmark, NiNo is competitive with strong arithmetic baselines and consistently lands in the same functional region as weight averaging, Task Arithmetic, and TIES. Moreover, NiNo is best on HumanEval in a Qwen3 language extension among the compared merge methods, while extending meta-merge beyond pairs remains an open challenge. These results position learned checkpoint nowcasting as a practical starting point for data-free model merging and motivate future weight-space learners trained for merging explicitly.

1. Introduction

Large-scale foundation models now dominate vision and language pipelines, yet each new downstream task still demands an expensive fine-tune and extra GPU memory at deployment. Model merging tackles this bottleneck: instead of storing K separate checkpoints, it combines their parameters into a single network that behaves as if it had been fine-tuned on all K tasks. Merging is attractive for on-device inference, continual learning, and federated settings where data cannot be shared.

Recent work has shown paths toward making model parameters compatible with each other. This is a challenging open problem, to which clever algorithms have been proposed to tackle specific settings. Most successful techniques share a two-step recipe:

1. **Layer-wise alignment**: via pruning low-magnitude deltas (Davari & Belilovsky, 2024), or apply sparse binary masks (Yadav et al., 2023) so that corresponding layers become "compatible".
2. **Arithmetic aggregation**: Average or linearly interpolate the aligned weights, occasionally with a small post-hoc re-fine-tune ("stitching").

Knyazev et al. (2025) introduced NiNo, a meta-network algorithm capable of nowcasting parameters to speed up training optimization. Given a series of past checkpoints from a single training run as input, a graph neural network-based model predicts future model parameters. NiNo is *meta-trained* on a dataset of optimization trajectories; since it takes models as inputs and returns the updated model (with the same order of neurons as in the inputs), an unexplored question is: can a pre-trained meta-network, trained to extrapolate future parameters, also serve as a generic operator for merging independently fine-tuned models? We refer to this reuse as *meta-merge*: applying a learned update rule, trained on optimization trajectories, directly to task-specific checkpoints at test time.

This paper studies that question in the pairwise setting. Starting from the open-source CLIP ViT-B/16 visual encoder (Radford et al., 2021; Dosovitskiy et al., 2020), we fine-tune the same pre-trained checkpoint on eight vision tasks and treat each pair of task-specific checkpoints as a pseudo-trajectory rooted at their common initialization. Because the pre-trained NiNo model we use expects a five-checkpoint context, we choose a pairwise input sequence and later validate that choice by ablation. We then aggregate the resulting predicted checkpoints into a default merged model. We compare this merging-by-nowcasting recipe against weight averaging, Task Arithmetic (TA) (Ilharco et al., 2023), TIES (Yadav et al., 2023), and Breadcrumbs (Davari & Belilovsky, 2024) on all 28 source-task

[1]Concordia University [2]Mila – Québec AI Institute [3]Samsung AI Lab, Montréal [4]Université de Montréal. Correspondence to: Albert M. Orozco Camacho <alorozco53@mila.quebec>, Boris Knyazev <b.knyazev@samsung.com>.

LatinX in AI Workshop @ ICML 2026, Seoul, South Korea. 2026.

pairs, using both top-1 accuracy and final-block Centered Kernel Alignment (CKA) (Kornblith et al., 2019; Davari et al., 2022).

Across the 28-pair CLIP grid, the default NiNo merge is competitive with the strongest arithmetic baselines: TIES remains best on average, while NiNo `avg 1:5` stays close in top-1 accuracy and consistently lands in the same activation-space neighborhood as weight averaging, TA, and TIES. The horizon trajectory is also informative: activation-space similarity peaks around k=3, while the horizon average `1:5` slightly improves accuracy and therefore serves as our default merged checkpoint. A compact Qwen3 extension at 0.6B shows a similar but weaker transfer pattern, and exploratory 4-task and 8-task compositions suggest that pairwise meta-merge is currently more robust than many-way scaling.

Contributions

- We propose merging-by-nowcasting as a data-free pairwise meta-merge operator for independently fine-tuned checkpoints.
- We characterize NiNo's horizon trajectory in both accuracy and activation space, showing that its default horizon average lands near a strong arithmetic merge cluster.
- We test this meta-merge picture beyond the main pairwise vision setting via a Qwen3 language extension and exploratory many-way compositions.

Taken together, the experiments uncover an unexpected capability of NiNo: a model trained to nowcast optimization trajectories can be repurposed as a practical pairwise meta-merge operator for independently fine-tuned checkpoints. This meta-merge perspective connects trajectory extrapolation and model merging, and suggests that learned update rules may offer an alternative route to composing task-specific models without task data or hand-crafted sparsity heuristics (Johnston et al., 1999; Hunter, 1986; Jolicoeur-Martineau et al., 2023; Izmailov et al., 2018; Morales-Brotons et al., 2024; Knyazev et al., 2025).

2. Related Work

Weight-space model merging. Model merging typically operates directly in parameter space, which makes it a practical way to combine task-specific models without retraining. Model Soups (Wortsman et al., 2022) showed that simply averaging models fine-tuned from a shared initialization can improve robustness and accuracy at no inference cost. Task Arithmetic (Ilharco et al., 2023) and TIES-Merging (Yadav et al., 2023) are the canonical task-vector recipes, while Model Breadcrumbs (Davari & Belilovsky, 2024) shows that magnitude-based sparsification can reduce interference; a broader averaging lineage—stochastic and exponential weight averaging and their relatives (Izmailov et al., 2018; Johnston et al., 1999; Hunter, 1986; Jolicoeur-Martineau et al., 2023; Morales-Brotons et al., 2024)— treats merging as a robustness-improving mean over checkpoints. Interpolation variants such as SLERP (Jang et al., 2024) and WiSE-FT (Wortsman et al., 2021) are effective in specific regimes but remain fundamentally raw-parameter operations, and their quality is sensitive to merge coefficients, sign conflicts, and checkpoint selection. These arithmetic and sparsified recipes are exactly the baselines we compare against.

Beyond linear merging. A second line of work makes merging structure-aware. SVD-based methods such as KnOTS (Stoica et al., 2024) and Task Singular Vectors (Gargiulo et al., 2025) project layer-wise updates into low-rank subspaces to disentangle and reduce task interference. These approaches are elegant and often data-free, but they remain tied to *linear* decompositions of the task-vector updates, which motivates asking whether a *learned*, nonlinear operator can capture cross-task structure that fixed linear projections miss.

Learned weight-space operators. Weight-space learning has recently emerged as a practical tool for meta-modeling neural parameters (Han et al., 2026). Architectures such as SANE (Schürholt et al., 2024) and NiNo (Knyazev et al., 2025) learn to represent and predict model weights directly; NiNo in particular is a graph-based *nowcaster* originally trained to extrapolate future checkpoints from a short optimization history, in order to accelerate training. Our work departs from this family in how the learner is used: rather than training a new merge-specific model, we repurpose the pre-trained NiNo nowcaster as a data-free merge operator, applying a single forward pass to a pseudo-trajectory assembled from independently fine-tuned checkpoints. This connects optimization-trajectory learning to model merging and, to our knowledge, is the first use of a checkpoint nowcaster as a merge operator. It also contrasts with latent-space merging, e.g. LS-Merge (Soro et al., 2026), which learns a dedicated latent space and interpolates models inside it: we instead operate on raw checkpoints through a forward nowcasting pass, with no learned latent and no merge-specific objective.

3. Methodology

We compare standard weight-space merge operators with a learned pairwise meta-merge alternative. Section 3.1 defines the arithmetic baselines used throughout the paper, while Section 3.2 describes how we repurpose pre-trained NiNo as a data-free pairwise meta-merge operator for fine-tuned

checkpoints.

3.1. Weight-Space Merge Baselines

Problem set-up. Let $W_0 \in \mathbb{R}^P$ denote the parameters of a pre-trained foundation model and let $\{W_t\}_{t=1}^T$ be the checkpoints obtained after fine-tuning W_0 on T disjoint tasks. A *task vector* is the weight difference $\Delta_t = W_t - W_0$. Weight-space merging seeks a single parameter set $\widehat{W}$ that approximates the performance of $\{W_t\}$ *without* storing multiple models or resorting to ensemble inference. Most methods apply a layer-wise *alignment* operator $\mathcal{A}(\cdot)$—permutations, masks, sign filtering—followed by an arithmetic aggregation:

$$\widehat{W} \;=\; W_0 \;+\; \sum_{t=1}^{T} \alpha_t \, \mathcal{A}(\Delta_t), \tag{1}$$

where the coefficients α_t (scalars or per-layer vectors) are chosen by grid-search on a small validation set. This contrasts with ensembling, whose cost grows linearly in T at inference time.

Weight averaging and Task Arithmetic. Plain weight averaging applies Eq. equation 1 with $\mathcal{A}(\Delta_t) = \Delta_t$ and uniform coefficients, yielding

$$\widehat{W}_{\text{avg}} = \frac{1}{T}\sum_{t=1}^{T} W_t = W_0 + \frac{1}{T}\sum_{t=1}^{T} \Delta_t.$$

Task Arithmetic (TA) (Ilharco et al., 2023) uses the same direction but introduces a global scaling coefficient λ:

$$\widehat{W}_{\text{TA}} = W_0 + \lambda \, \frac{1}{T}\sum_{t=1}^{T} \Delta_t.$$

This simple linear recipe is often surprisingly effective when the tasks are sufficiently "orthogonal," a *linear* combination of task vectors already yields a competent merged model, but it becomes increasingly sensitive to interference as more task vectors are added.

TIES and Breadcrumbs. Trim, Elect, Sign & Merge (TIES-Merging) (Yadav et al., 2023) mitigates interference by *masking* low-magnitude weights in each Δ_t, resolving sign conflicts among the surviving entries, and only then averaging the resulting task vectors. Breadcrumbs (Davari & Belilovsky, 2024) instead removes both very small and very large magnitudes within each layer, leaving a sparse, layer-balanced update that is added back to W_0 with strength α. Both methods aim to reduce task interference without retraining, but do so through different sparsification heuristics.

3.2. NiNo as a Meta-Merge Operator

NiNo (Knyazev et al., 2025) is a pre-trained meta-network for checkpoint nowcasting: given a short context of training checkpoints, it predicts future checkpoints along that trajectory. In our setting, we treat independently fine-tuned models as a *pseudo-trajectory* rooted at the common pre-trained checkpoint W_0, and use NiNo directly as a pairwise meta-merge operator without additional training.

The pre-trained NiNo model used here expects a context of five checkpoints. This fixes the input length, but not the ordering of the task-specific checkpoints. For a pair of task-specific models W_{t_1} and W_{t_2}, we choose the context order

$$C_{01221} = [W_0, W_{t_1}, W_{t_2}, W_{t_2}, W_{t_1}],$$

which we refer to as the `01221` recipe. For each horizon $k \in \{1, \ldots, 5\}$, we denote the corresponding NiNo prediction by

$$\widehat{W}^{(k)}_{NiNo} = \text{NiNo}_k(C_{01221}). \tag{2}$$

Our default pairwise meta-merge is then the full horizon average

$$\widehat{W}_{NiNo} = \frac{1}{5}\sum_{k=1}^{5} \widehat{W}^{(k)}_{NiNo}, \tag{3}$$

which we denote as `avg 1:5`.

All pairwise NiNo results use the chosen `01221+avg 1:5` recipe by default.

3.3. Context-Order and Horizon Ablation

Because the pre-trained NiNo model expects a trajectory-like input, pairwise meta-merge depends on both the 5-state context order and on which predicted horizon(s) are retained. To justify the chosen pairwise recipe, we run an ablation on 10 source-task pairs sampled from the 8-task CLIP pool. For each pair, we evaluate the five single-horizon outputs and the prefix averages 1:2, 1:3, 1:4, and 1:5 across six candidate context orders.

Figure 1 summarizes the selection rule used in the paper. Panel A aggregates all six orders and shows the horizon trend: k=3 is the strongest single horizon, while the full average 1:5 remains competitive because it retains the early-horizon gains. Panel B reports the final 1:5 candidate by context sequence and highlights the chosen order $01221 = [W_0, W_1, W_2, W_2, W_1]$. Panel C repeats the horizon analysis after restricting to this chosen order, confirming that the same trend holds once the ordering is fixed. Error bars are standard errors over the 10 sampled pairs.

Figure 1 therefore supports the same rule used in the main text: k=3 is the strongest single horizon, 1:5 is the best robust default, and 01221 is the strongest tested full-average order.

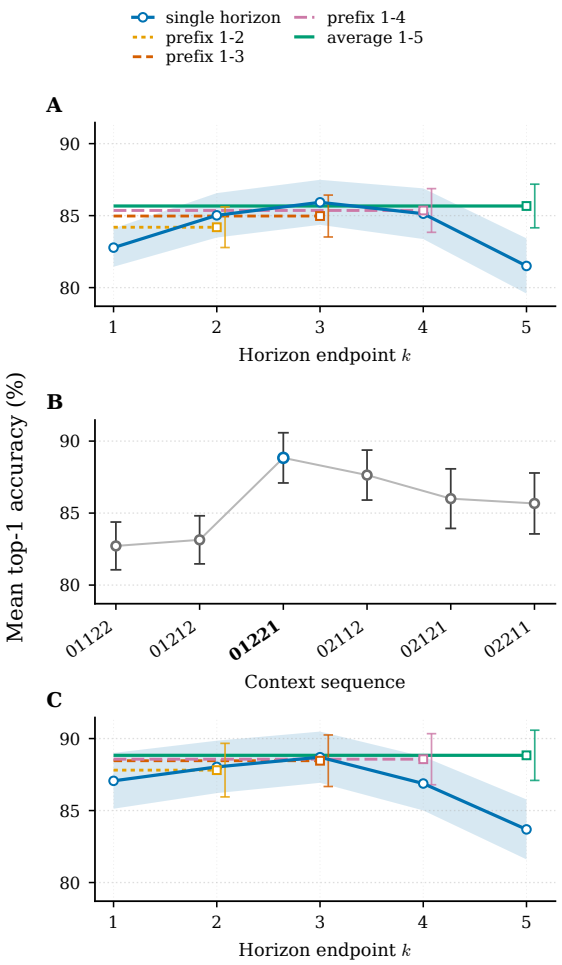

Figure 1. Context-order and horizon ablation for NiNo on two-task merges. Error bars indicate standard error over sampled task pairs.

Synthetic midpoints as a negative control. A natural alternative to repeating the task-checkpoint slots in the chosen `01221` sequence is to insert synthetic midpoints, $[W_0, \frac{W_0+W_{t_1}}{2}, W_{t_1}, \frac{W_{t_1}+W_{t_2}}{2}, W_{t_2}]$, mirroring `merge_vit.py`'s `--midpoints` flag. The pre-trained NiNo model is unchanged; only the input sequence differs. On the same 28-pair grid, the midpoint recipe is uniformly worse: for $k \in \{2, 3, 4\}$ and avg 1:5 it loses 1.4–2.0 p.p., and the late-horizon k=5 output collapses by 23.2 p.p. – its weight-space cosine to the chosen k=5 output stays high (0.911), but activation CKA drops to 0.791, far outside the 0.92–0.98 range seen across the rest of the horizon sweep. Synthetic intermediates therefore point in roughly the same direction but land outside the useful activation cluster, supporting the `01221` ordering as the operating recipe.

4. Experiments

Experimental set-up. We start from the open-source CLIP ViT-B/16 (Radford et al., 2021; Dosovitskiy et al., 2020) encoder (W_0) and fine-tune it on eight disjoint vision tasks from the benchmark of Ilharco et al. (2023): MNIST (*hand-written digits*), DTD (*textures*), EuroSAT (*satellite imagery*), Cars (*fine-grained vehicles*), RESISC-45 (*remote-sensing scenes*), SVHN (*street numbers*), GTSRB (*traffic signs*), and SUN397 (*scene recognition*). For each source-task pair $\{d_1, d_2\}$ we create two single-task checkpoints W_{d_1}, W_{d_2} and merge them with the five methods under study: plain weight averaging, Task Arithmetic (TA), TIES, Model Breadcrumbs, and NiNo. For NiNo, we use the pairwise recipe chosen in Section 3.2, namely `01221` followed by the full-horizon average `avg 1:5`. This yields all $\binom{8}{2} = 28$ benchmark pairs. Unless otherwise stated, we evaluate each merged model only on its two source datasets and report top-1 accuracy averaged over the 28 source-task pairs. A compact language-domain extension with Qwen3 is presented later in this section; full per-benchmark results are deferred to Appendix A.6.

Similarity Measurement. To quantify representational similarity, we use *linear Centered Kernel Alignment (CKA)* (Kornblith et al., 2019; Davari et al., 2022). For two activation matrices, $X, Y \in \mathbb{R}^{n\times d}$, it is defined as $\mathrm{CKA}(X, Y) = \|X^\top Y\|_F^2 / \left(\|X^\top X\|_F \|Y^\top Y\|_F\right)$, where $\|\cdot\|_F$ denotes the Fröbenius norm. Linear CKA is invariant to orthogonal transformations and isotropic scaling, and takes values in $[0, 1]$, with 1 indicating identical representations. All CKA scores are computed on the last vision-block activations, after feeding the test sets of the corresponding source datasets.

4.1. Pairwise Accuracy and Representational Alignment

To study NiNo's pairwise meta-merge behavior relative to the standard arithmetic and sparsified baselines, we compare both predictive performance and representational similarity on the full 28-pair grid. Table 1 summarizes the main pairwise vision results. TIES is the strongest method on average, reaching 0.908 top-1 accuracy, with TA close behind at 0.905. NiNo's horizon sweep under our chosen `01221` context order shows a clear best aggregate candidate: `avg 1:5` reaches 0.891 top-1 accuracy, narrowly above the best single horizon k=3 at 0.890. Breadcrumbs trails the other methods on this pairwise benchmark.

Figure 2 makes this pair-level geometry concrete for the nearest baseline, weight averaging. Across most task pairs, the CKA panel remains above 0.95, with the weakest alignment concentrated on EuroSAT-involving merges, while the accuracy panel shows only a small residual gap on average. The rightmost column of Table 1 summarizes

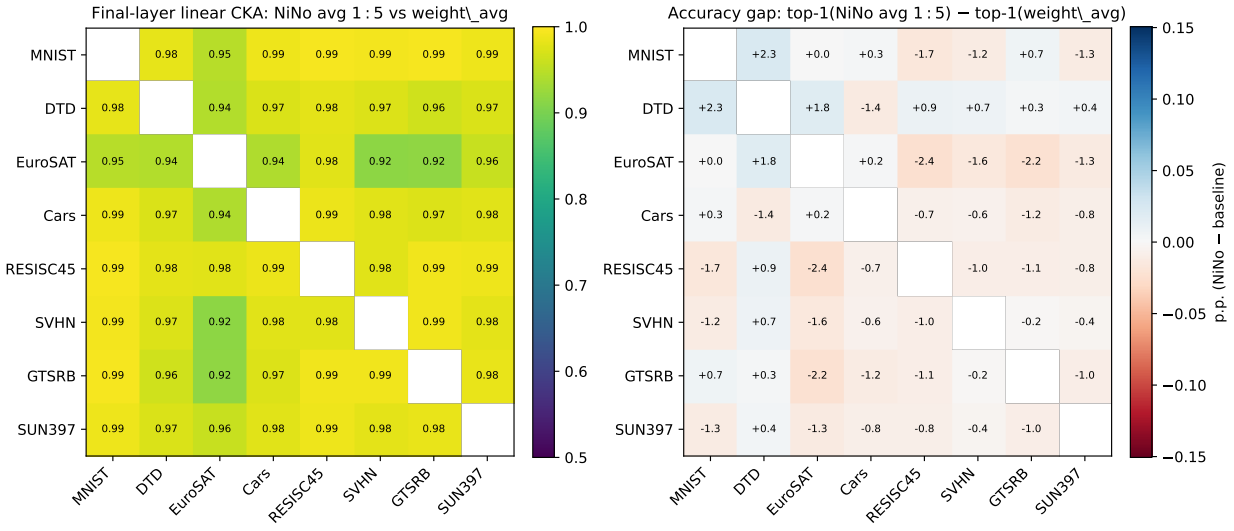

Figure 2. **Per-pair 8×8 comparison of NiNo avg $1{:}5$ against weight averaging.** The left panel reports final-block linear CKA, averaged over the two in-pair evaluation datasets. The right panel reports the signed accuracy gap in percentage points. This is the closest baseline to NiNo in activation space on the 28-pair grid.

Table 1. **Pairwise merge performance on the 28-pair CLIP ViT-B/16 benchmark.** Mean top-1 is averaged over all 28 source-task pairs and their two source datasets; standard error is over pairs. The CKA column reports mean final-block linear CKA to the arithmetic cluster {weight_avg, TA, TIES}, averaged over the same evaluations. Best overall and best NiNo candidate are bold. All NiNo rows use the paper's chosen `01221` context order for each pair.

Method	Mean top-1	CKA to arithmetic cluster
weight_avg	0.896 ± 0.012	0.984
TA	0.905 ± 0.011	0.985
TIES	$\mathbf{0.908 \pm 0.011}$	0.989
breadcrumbs	0.831 ± 0.014	0.830
NiNo $k{=}1$	0.873 ± 0.011	0.909
NiNo $k{=}2$	0.882 ± 0.011	0.940
NiNo $k{=}3$	0.890 ± 0.012	*0.961*
NiNo $k{=}4$	0.874 ± 0.013	0.942
NiNo $k{=}5$	0.845 ± 0.015	0.873
NiNo avg $1{:}5$	$\mathbf{0.891 \pm 0.011}$	0.958

activation-space proximity to the arithmetic merge cluster {weight_avg, TA, TIES}. This proximity is non-monotone across NiNo horizons: it rises from 0.909 at $k{=}1$ to a peak of 0.961 at $k{=}3$, then declines at later horizons. The default `avg 1:5` candidate retains nearly all of $k{=}3$'s alignment (0.958 vs. 0.961) while slightly improving mean top-1 accuracy (0.891 vs. 0.890), which is why it remains our default merged checkpoint. Figure 3 below visualizes the full horizon trajectory across baselines; the remaining per-pair heatmaps and additional baseline comparisons are deferred to Appendix A.2.2 and Appendix A.2.3.

Tables 4 and 12 suggest that NiNo lives near the weight-averaging / Task-Arithmetic axis. Using either NiNo $k{=}3$ or avg $1{:}5$ as the anchor gives the same picture: weight averaging is the nearest baseline in CKA (0.973 for $k{=}3$, 0.972 for avg $1{:}5$), and TA/weight averaging are also the nearest in cosine (0.463 for $k{=}3$, 0.495 for avg $1{:}5$), but TIES is the most accurate baseline overall.

So directional proximity explains NiNo's closeness to the arithmetic family, not which baseline is strongest. TIES gains accuracy by departing from the raw summed-task-vector direction through sparsification and sign resolution, even though that makes it slightly farther from NiNo in cosine and CKA. **Weight-space proximity to the arithmetic cluster.** Table 4 reports the mean cosine between NiNo's task-vector delta and each baseline delta across the 28 pairs. Two patterns are stable. First, every NiNo horizon is closest to weight averaging and Task Arithmetic, then to TIES, and least to breadcrumbs—the same ordering already visible in activation space (Table 1). Second, the cosine decays almost monotonically with horizon: $k{=}1$ is nearest to the raw summed-task-vector direction, while later horizons drift away from every baseline. The full-horizon average $1{:}5$ recovers a midpoint (0.495 to wa/TA), consistent with its role as the default merged checkpoint.

Beyond Pairwise Merging. We also stress-test the pre-trained NiNo operator on 4-task and 8-task compositions of the same 8-dataset benchmark. Because NiNo accepts only five context states, many-way meta-merge must be wrapped in heuristic topologies (binary tree, progressive-fold), built from soups of saved pairwise meta-merges, or combined with hand-crafted anchors. Across both scales, the arithmetic baselines clearly dominate. At 4-task scale, the

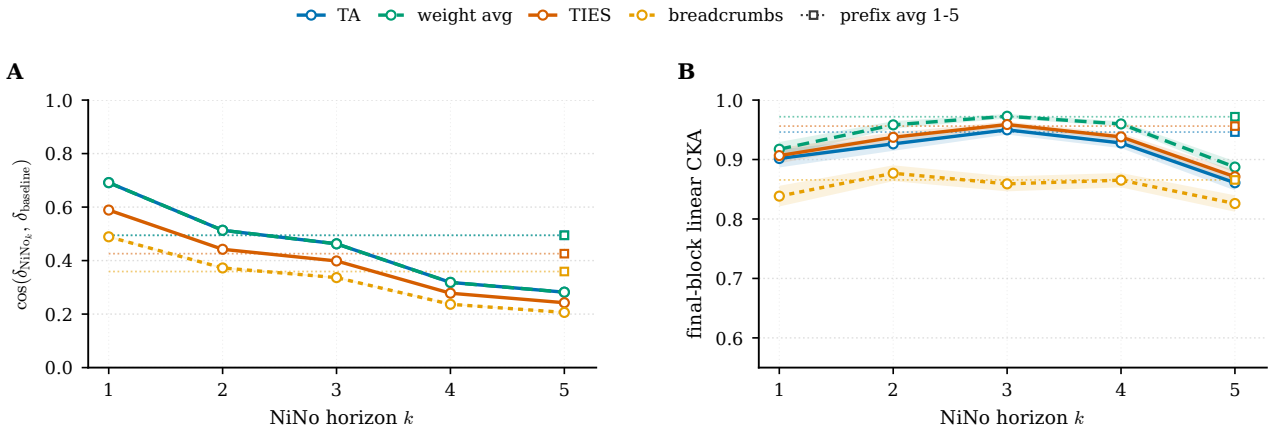

Figure 3. **NiNo horizon trajectory across the 28-pair grid. (A)** weight-space cosine between NiNo's task-vector delta and each baseline's. **(B)** final-block linear CKA between NiNo and each baseline. Solid/dashed lines with circle markers show the single-horizon means at $k \in \{1, \ldots, 5\}$; shaded bands are pair-level standard error. Dotted horizontal segments with square markers report the prefix average 1:5 as a constant reference per baseline. TA and weight averaging coincide in panel A by construction (their deltas are co-linear).

Table 2. **Top-1 accuracy of merged encoders, by evaluation dataset.** Each entry is the mean top-1 of the listed candidate when one of its two source datasets is the column dataset, averaged over the 7 partner pairs. Best result per column in bold.

Method	MNIST	DTD	EuroSAT	Cars	RESISC45	SVHN	GTSRB	SUN397	avg
W_0 (zero-shot)	0.517	0.447	0.555	0.647	0.664	0.520	0.434	0.655	0.555
weight_avg	0.994	0.740	0.988	0.824	0.934	0.962	0.968	0.756	0.896
TA	**0.995**	0.777	0.990	0.829	0.943	0.970	**0.978**	0.757	0.905
TIES	**0.995**	0.778	**0.992**	0.845	**0.947**	**0.971**	0.968	**0.772**	**0.908**
breadcrumbs	0.982	0.617	0.968	0.781	0.891	0.860	0.818	0.729	0.831
NiNo k=1	0.988	**0.790**	0.983	**0.862**	0.857	0.875	0.913	0.712	0.873
NiNo k=2	0.992	0.767	0.986	0.834	0.898	0.912	0.941	0.729	0.882
NiNo k=3	0.994	0.763	0.986	0.822	0.917	0.945	0.960	0.736	0.890
NiNo k=4	0.993	0.716	0.976	0.769	0.914	0.947	0.943	0.734	0.874
NiNo k=5	0.994	0.654	0.962	0.736	0.899	0.954	0.887	0.678	0.845
NiNo avg 1:5	0.994	0.770	0.987	0.829	0.914	0.940	0.958	0.736	0.891

Table 3. **Final-block linear CKA between NiNo k=3 and each baseline, by evaluation dataset.** Each entry is the mean over the 7 partner pairs containing the column dataset. The ordering weight_avg $>$ TIES $\gtrsim$ TA $\gg$ breadcrumbs is stable across datasets; EuroSAT is the clearest outlier (largest representation shift), while RESISC45 and SUN397 stay highly aligned across all merge baselines. Avg 1:5 values are within ± 0.01 of k=3 throughout and are omitted for brevity. Best per column in bold.

Baseline	MNIST	DTD	EuroSAT	Cars	RESISC45	SVHN	GTSRB	SUN397
W_0 (zero-shot)	0.492	0.727	0.282	0.757	0.877	0.213	0.393	0.899
weight_avg	**0.997**	**0.967**	**0.912**	**0.976**	**0.991**	**0.982**	**0.981**	**0.977**
TA	0.991	0.940	0.868	0.970	0.984	0.935	0.942	0.969
TIES	0.988	0.962	0.904	0.971	0.985	0.930	0.964	0.966
breadcrumbs	0.928	0.879	0.621	0.907	0.969	0.833	0.785	0.951

strongest NiNo composition (`nino_tree_mlp`) reaches 80.06 average accuracy, trailing TIES by 5.9 p.p. and weight averaging by 1.5 p.p. (Table 5). At 8-task scale, the best same-run NiNo wrapper reaches 70.86, still below every same-run arithmetic baseline; the soup of pairwise NiNo merges lands at 71.09, close to souped weight averaging (71.67) but $\sim$9 p.p. behind breadcrumbs (Table 6). Compositions built from saved pairwise meta-merges are consistently more stable than direct hierarchical recipes, which we read as evidence that pairwise meta-merge is currently the operating regime for this pre-trained nowcaster; many-way scaling is an open problem. Per-dataset winners, soup-vs-direct di-

Table 4. **Mean weight-space cosine between NiNo deltas and baseline deltas, over 28 pairs.** The pre-trained reference W_0 is omitted (delta is zero). Best per column in bold.

NiNo candidate	weight_avg	TA	TIES	breadcrumbs
NiNo k=1	**0.691**	**0.691**	**0.589**	**0.489**
NiNo k=2	0.513	0.513	0.442	0.372
NiNo k=3	0.463	0.463	0.399	0.336
NiNo k=4	0.319	0.319	0.278	0.237
NiNo k=5	0.282	0.282	0.243	0.206
NiNo avg 1:5	0.495	0.495	0.426	0.359

Table 5. **Top-1 accuracy of merged ViT-B/16 encoders across five 4-task subsets.** Mean and standard deviation over five sampled subsets; all NiNo variants average horizons 1:5. The strongest NiNo composition (`nino_tree_mlp`) trails TIES by 5.9 p.p. and weight averaging by 1.5 p.p., but is also the most stable NiNo variant (σ=2.48). Best mean in bold.

Method	Mean	Std
TIES	**85.96**	4.01
TA (λ=0.4)	84.32	4.04
breadcrumbs	83.44	4.76
weight_avg	81.55	4.49
NiNo tree (MLP)	80.06	2.48
NiNo direct (GNN)	78.53	5.21
NiNo prog-fold (GNN)	76.48	4.60
NiNo tree (GNN)	76.00	3.38
NiNo prog-fold (MLP)	72.66	3.44
NiNo direct (MLP)	62.52	8.18

agnostics, and backbone/topology interactions are deferred to Appendix A.3 and Appendix A.4.

Per-dataset winners. The per-dataset winners reinforce the leaderboard. Breadcrumbs wins six of the eight tasks, TIES wins SVHN, and the only NiNo-derived per-dataset win is SUN397 via progressive-fold with fixed α=0.5; since SUN397 is the final task in the canonical progressive-fold order, this is best read as late-task retention rather than broad many-way strength.

All-pair soup diagnostic. The all-pair-soup rows in Table 6 average the 28 saved per-pair merged checkpoints for a method. TA and weight averaging are nearly invariant under this construction: TA drops from 74.99 to 73.83, and weight averaging is unchanged at 71.67. TIES loses 3.51 p.p., and breadcrumbs loses 13.36 p.p., indicating that their per-pair sparsification rules do not transfer cleanly to the many-way setting.

Backbone/topology interaction. The GNN/MLP preference flips with topology. On the binary tree, the MLP backbone beats the GNN by 15.4 p.p.; on progressive-fold, the GNN beats the MLP by 4.5 p.p. The simplest reading is that MLP-NiNo prefers small, symmetric deltas, whereas GNN-NiNo is more tolerant of spread-out contexts. A stage-wise diagnostic of the best progressive-fold family on the full 8-task validation set further shows that stage 3 is stronger than the final stage 4 checkpoint for all four runs: the last fold into GTSRB and SUN397 helps late tasks but hurts overall balance.

Table 6. **Representative 8-task ViT-B/16 merge results, full validation, seed 42.** The strongest same-run baselines, the most informative all-pair soups over the 28-pair grid, and the strongest same-run NiNo wrappers. Even the best same-run NiNo variant (`tree_default_mlp`, 70.86) trails every same-run arithmetic baseline; pair-soup constructions over saved pairwise NiNo merges land in the same neighborhood (71.09). Best in each column in bold.

Strategy	Avg	Min	HM
breadcrumbs	**79.83**	**57.82**	**77.89**
TIES	76.93	57.18	74.93
TA (λ=0.2)	74.99	52.50	72.87
weight_avg	71.67	50.05	69.47
soup-28-pair: TA	73.83	52.77	71.87
soup-28-pair: weight_avg	71.67	50.05	69.46
soup-28-pair: NiNo-avg-1:5	71.09	51.44	69.07
NiNo tree (MLP)	70.86	49.84	68.43
NiNo prog-fold (GNN, α=0.75)	68.07	46.33	65.78

Language-Domain Extension: Qwen3. We repeat the comparison on Qwen3 (Yang et al., 2025), using the released 0.6B model with three task fine-tunes (math, French, code). Table 9 shows that TA is strongest on the macro-average for every merge configuration, so transfer to language is weaker than in the 28-pair vision benchmark. The macro-average gap from NiNo to TA (λ=0.3) widens to 5.6–9.2 p.p., much larger than the pairwise vision gap.

NiNo still retains selective strengths. It achieves the best 3-way HumanEval score at 46.3, exactly matching the standalone Code fine-tune, and it gives the strongest French score on the Fr+Code pair. We read this as partial cross-domain transfer: aggregate language merging is harder, but some task-specialized behavior still survives the meta-merge. For calibration, the arithmetic baselines themselves form a tighter one-dimensional chain (Table 10); NiNo preserves this ordering but sits outside that tight cluster, supporting the "near but outside the arithmetic family" reading of Table 4.

5. Discussion and Conclusion

We showed that a pre-trained NiNo nowcaster can be repurposed as a practical data-free pairwise meta-merge operator. On the 28-pair CLIP ViT-B/16 benchmark, the chosen `01221+avg 1:5` recipe is competitive with strong arithmetic baselines and consistently lands in the same activation-

Table 7. **8-task tree-merge accuracy under four input orderings.** Each row reports per-task top-1 plus the 8-task average and minimum. "First task" is the leftmost task in the input ordering (level-1 pair 1, position a). Orderings differ only in the input permutation; the recipe is the symmetric tree default `[zs,a,b,b,a]` at every node. The mean spread across orderings is only 1.5 p.p. (53.98–55.45), and even the best ordering remains far behind every same-run arithmetic baseline of Table 6: reordering the leaves alone does not close the 8-task gap. Best per column in bold.

Method	First task	MNIST	DTD	EuroSAT	Cars	RESISC45	SVHN	GTSRB	SUN397	Avg	Min
tree, default	MNIST	**94.52**	39.52	53.09	50.25	55.78	**56.92**	39.37	54.15	55.45	39.37
tree, reverse	SUN397	84.17	40.21	52.47	49.99	53.44	54.03	**43.56**	56.08	54.24	40.21
tree, hard_first	Cars	85.37	40.43	55.84	**54.43**	55.83	46.56	39.18	55.12	54.09	39.18
tree, resisc_first	RESISC45	82.97	40.90	**57.19**	51.34	**61.33**	44.89	37.88	55.34	53.98	37.88

Table 8. **Per-task ordering sensitivity for the 8-task tree.** Range of top-1 accuracy across the four 8-task tree orderings of Table 7, sorted descending. "Best ord. (Worst)" lists the ordering achieving the column max (resp. min). MNIST and SVHN move by about 12 p.p. across orderings while DTD and SUN397 are nearly invariant – the consistent effect is positional, as tasks placed in the leftmost level-1 pair are preserved better under the fixed `[zs,a,b,b,a]` tree recipe.

Task	Min	Max	Range (p.p.)	Best ord. (Worst)
SVHN	44.89	56.92	12.03	default (resisc_first)
MNIST	82.97	94.52	11.55	default (resisc_first)
RESISC45	53.44	61.33	7.89	resisc_first (reverse)
GTSRB	37.88	43.56	5.68	reverse (resisc_first)
EuroSAT	52.47	57.19	4.72	resisc_first (reverse)
Cars	49.99	54.43	4.44	hard_first (reverse)
SUN397	54.15	56.08	1.93	reverse (default)
DTD	39.52	40.90	1.38	resisc_first (default)

space neighborhood as weight averaging, Task Arithmetic, and TIES; the horizon analysis further shows that this behavior is structured rather than accidental. Beyond vision, NiNo is strongest on code-oriented behavior in Qwen3, while its 4-task and 8-task compositions remain less robust than the pairwise operator itself.

Taken together, these results make pairwise meta-merge the clearest current success case for repurposing learned weight-space dynamics for an explicit model composition, and suggest that a merge-relevant structure already exists in general-purpose weight-space learners even before any explicit merge training. The picture is scoped; however, every many-way composition we tried trails the arithmetic baselines, evaluation is in-domain on a single CLIP ViT-B/16 backbone, and the Qwen3 extension reuses the vision-trained NiNo unchanged.

Three directions follow naturally: (i) a merge-aware fine-tuning signal for NiNo, since the current model is trained without explicit reward on merged checkpoints; (ii) scaling beyond pairs via larger context windows or hierarchical wrappers, since our tree and progressive-fold composers trail simple averaging at the 8-task scale; and (iii) cross-backbone re-meta-training, since the Qwen3 numbers reflect zero-shot transfer rather than tuned use.

Table 9. **Top-1 accuracy (%) of merged Qwen3-0.6B models, by merge configuration.** Per-config best result in bold. NiNo's per-config gap to TA λ=0.3 widens to 5.6–9.2 p.p. on the macro-average, substantially above the 1.4 p.p. gap to TA on the 28-pair vision benchmark (Table 1). Setup details are in Appendix A.6.

Method	GSM	Fr	GSM-Fr	HE	Avg
zero-shot (base)	19.5	25.1	21.1	38.4	26.0
3-way (Math+Fr+Code)					
Weight avg.	49.8	27.1	33.1	42.7	38.2
TA, λ=0.3	**50.6**	**27.3**	**35.6**	42.7	**39.1**
TA, λ=0.4	47.8	27.1	30.8	**43.9**	37.4
NiNo	39.1	24.4	23.5	**46.3**	33.3
Math+Fr					
Weight avg.	49.3	**28.0**	29.3	31.7	34.6
TA, λ=0.3	**53.1**	27.5	**36.8**	**32.3**	**37.4**
TA, λ=0.4	53.0	27.7	35.3	32.9	37.2
NiNo	44.0	26.1	28.8	26.8	31.4
Math+Code					
Weight avg.	48.1	25.9	29.9	**42.7**	36.7
TA, λ=0.3	**53.8**	25.3	**35.9**	**42.7**	**39.4**
TA, λ=0.4	51.9	25.5	34.3	**43.3**	38.8
NiNo	42.2	25.4	27.2	26.2	30.2
Fr+Code					
Weight avg.	43.3	26.9	29.9	**42.7**	35.7
TA, λ=0.3	**49.2**	26.8	**35.3**	42.1	**38.3**
TA, λ=0.4	48.8	27.1	33.3	40.9	37.5
NiNo	37.8	**30.2**	27.6	35.4	32.7

Table 10. **Mean weight-space cosine between baseline deltas, over 28 pairs.** Baselines lie on a one-dimensional cluster from breadcrumbs through TIES to TA/weight_avg, contextualising NiNo's 0.495 cosine to weight averaging in Table 4.

	weight_avg	TA	TIES	breadcrumbs
weight_avg	1.000	1.000	0.866	0.718
TA	1.000	1.000	0.866	0.718
TIES	0.866	0.866	1.000	0.743
breadcrumbs	0.718	0.718	0.743	1.000

In short, NiNo already sits just outside the arithmetic baseline cluster, and that residual offset is the most concrete target for the merge-aware fine-tune above.

References

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Davari, M. and Belilovsky, E. Model breadcrumbs: Scaling multi-task model merging with sparse masks, 2024. URL https://arxiv.org/abs/2312.06795.

Davari, M., Horoi, S., Natik, A., Lajoie, G., Wolf, G., and Belilovsky, E. Reliability of cka as a similarity measure in deep learning, 2022. URL https://arxiv.org/abs/2210.16156.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. URL https://arxiv.org/abs/2010.11929.

Gargiulo, A. A., Crisostomi, D., Bucarelli, M. S., Scardapane, S., Silvestri, F., and Rodolà, E. Task singular vectors: Reducing task interference in model merging, 2025. URL https://arxiv.org/abs/2412.00081.

Han, X., Wang, Z., Zhao, B., Zhang, B., Li, J., Borth, D., Yu, R., Maron, H., Ye, Y., Yin, L., et al. A survey of weight space learning: Understanding, representation, and generation. *arXiv preprint arXiv:2603.10090*, 2026.

Hunter, J. S. The exponentially weighted moving average. *Journal of quality technology*, 18(4):203–210, 1986.

Ilharco, G., Ribeiro, M. T., Wortsman, M., Gururangan, S., Schmidt, L., Hajishirzi, H., and Farhadi, A. Editing models with task arithmetic, 2023. URL https://arxiv.org/abs/2212.04089.

Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., and Wilson, A. G. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Jang, Y. K., Huynh, D., Shah, A., Chen, W.-K., and Lim, S.-N. Spherical linear interpolation and text-anchoring for zero-shot composed image retrieval, 2024. URL https://arxiv.org/abs/2405.00571.

Johnston, F., Boyland, J. E., Meadows, M., and Shale, E. Some properties of a simple moving average when applied to forecasting a time series. *Journal of the Operational Research Society*, 50(12):1267–1271, 1999.

Jolicoeur-Martineau, A., Gervais, E., Fatras, K., Zhang, Y., and Lacoste-Julien, S. Population parameter averaging (papa). *arXiv preprint arXiv:2304.03094*, 2023.

Knyazev, B., Moudgil, A., Lajoie, G., Belilovsky, E., and Lacoste-Julien, S. Accelerating training with neuron interaction and nowcasting networks. In *International Conference on Learning Representations*, 2025.

Kornblith, S., Norouzi, M., Lee, H., and Hinton, G. E. Similarity of neural network representations revisited. *CoRR*, abs/1905.00414, 2019. URL http://arxiv.org/abs/1905.00414.

Lasbordes, M. and Gad, S. Luth: Efficient french specialization for small language models and cross-lingual transfer, 2025. URL https://arxiv.org/abs/2510.05846. arXiv:2510.05846.

Luo, Z., Xu, C., Zhao, P., Sun, Q., Geng, X., Hu, W., Tao, C., Ma, J., Lin, Q., and Jiang, D. Wizardcoder: Empowering code large language models with evol-instruct, 2023.

Morales-Brotons, D., Vogels, T., and Hendrikx, H. Exponential moving average of weights in deep learning: Dynamics and benefits. *arXiv preprint arXiv:2411.18704*, 2024.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021. URL https://arxiv.org/abs/2103.00020.

Schürholt, K., Mahoney, M. W., and Borth, D. Towards scalable and versatile weight space learning. In *International Conference on Machine Learning (ICML)*. PMLR, 2024.

Soro, B., Zhang, A. S., Andreis, B., Jo, J., Chong, S., and Hwang, S. J. Ls-merge: Merging language models in latent space. In *The Fourteenth International Conference on Learning Representations*, 2026.

Stoica, G., Ramesh, P., Ecsedi, B., Choshen, L., and Hoffman, J. Model merging with svd to tie the knots, 2024. URL https://arxiv.org/abs/2410.19735.

Wortsman, M., Ilharco, G., Li, M., Kim, J. W., Hajishirzi, H., Farhadi, A., Namkoong, H., and Schmidt, L. Robust fine-tuning of zero-shot models. *CoRR*, abs/2109.01903, 2021. URL https://arxiv.org/abs/2109.01903.

Wortsman, M., Ilharco, G., Kim, J. W., Li, M., Kornblith, S., Roelofs, R., Schmidt, L., and Farhadi, A. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, 2022.

Yadav, P., Tam, D., Choshen, L., Raffel, C., and Bansal, M. Ties-merging: Resolving interference when merging models, 2023. URL https://arxiv.org/abs/2306.01708.

Yang, A. et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Table 11. **Top-1 accuracy of merged CLIP ViT-B/16 encoders, averaged over all 28 source-task pairs.** Standard error is over pairs. Best result in bold.

Method	Mean top-1
W_0 (zero-shot)	0.555 ± 0.011
weight_avg	0.896 ± 0.012
TA	0.905 ± 0.011
TIES	$\mathbf{0.908 \pm 0.011}$
breadcrumbs	0.831 ± 0.014
NiNo k=1	0.873 ± 0.011
NiNo k=2	0.882 ± 0.011
NiNo k=3	0.890 ± 0.012
NiNo k=4	0.874 ± 0.013
NiNo k=5	0.845 ± 0.015
NiNo avg 1:5	0.891 ± 0.011

A. Appendix

A.1. Complementary Results

A.2. Full 28-Pair Grid: Accuracy and Geometry

This section extends the main pairwise analysis to all 28 source-task pairs in the 8-task CLIP ViT-B/16 benchmark {MNIST, DTD, EuroSAT, Cars, RESISC45, SVHN, GTSRB, SUN397}. For each pair, we report NiNo's five pairwise meta-merge horizons and the average over 1:5, together with weight averaging, Task Arithmetic, TIES, breadcrumbs, and the zero-shot reference W_0. Pairwise NiNo always uses the paper's chosen `01221` recipe. All evaluations here are in-domain: each merged model is tested only on its two source datasets.

Appendix A.2.1 reports the aggregate per-pair accuracy; the per-dataset breakdown and the direction-vs-accuracy reading are promoted to Section 4.1 in the main text. Appendix A.2.2 places NiNo relative to the arithmetic baselines in weight and activation space. Appendix A.2.3 and Appendix A.2.4 provide the per-pair heatmaps and the midpoint ablation.

A.2.1. Accuracy across the 28-pair grid

Table 11 reports mean per-pair top-1 accuracy over all 28 source-task pairs. NiNo is competitive with weight averaging and Task Arithmetic at every horizon $k \geq 2$; the best single horizon k=3 (0.890) and the full average 1:5 (0.891) trail TA by 1.5 and 1.4 p.p., and TIES by 1.8 and 1.7 p.p. Breadcrumbs is clearly lower on this grid.

The per-dataset breakdown (Table 2, in the main text) shows that NiNo k=1 beats every baseline on DTD and Cars but falls behind on RESISC45, SVHN, and GTSRB; those deficits are what pull its 28-pair mean below TIES. MNIST is nearly saturated for every merge method except W_0, consistent with NiNo's pair-level wins concentrating on MNIST-containing pairs.

A.2.2. NiNo's location in weight and activation space

Section 4.1 showed that NiNo lands near the arithmetic merge cluster in activation space. Here we extend that picture to the full 28-pair grid along two axes: (i) cosine between task-vector deltas in weight space, and (ii) linear CKA on the final visual-encoder block. Throughout, $\delta = W_{\text{merged}} - W_0$ denotes the candidate's task-vector direction.

Weight-space proximity. Tables 4 and 10 (both promoted to Section 4.1) report the mean weight-space cosines for NiNo vs. baselines and for baselines vs. baselines respectively; NiNo preserves the baseline ordering but sits outside their tighter one-dimensional cluster.

Activation-space proximity. Table 12 shows a different geometry in activation space. NiNo k=3 and avg 1:5 both reach CKA above 0.95 with weight averaging, TA, and TIES, placing them inside the same activation-space neighborhood despite their larger weight-space offsets. CKA to breadcrumbs is lower, and CKA to W_0 stays around 0.58–0.60 across horizons.

Table 12. **Mean final-block linear CKA between NiNo and the baselines, over 28 pairs × 2 in-pair datasets.** Best per column in bold.

NiNo candidate	W_0	weight_avg	TA	TIES	breadcrumbs
NiNo k=1	0.578	0.917	0.902	0.907	0.838
NiNo k=2	**0.600**	0.958	0.926	0.937	**0.877**
NiNo k=3	0.580	**0.973**	**0.950**	**0.959**	0.859
NiNo k=4	0.594	0.960	0.928	0.938	0.865
NiNo k=5	0.589	0.887	0.861	0.871	0.826
NiNo avg 1:5	0.586	0.972	0.946	0.956	0.866

The arithmetic baselines form an even tighter activation-space cluster: {weight_avg, TA, TIES} all lie at CKA $\geq$ 0.97, whereas breadcrumbs is closer to W_0 than to any of them.

Table 13. **Mean final-block linear CKA between baselines, over 28 pairs × 2 in-pair datasets.** Note that breadcrumbs is closer to W_0 in CKA (0.769) than to weight_avg, TA, or TIES, indicating a separate activation cluster.

	W_0	weight_avg	TA	TIES	breadcrumbs
W_0 (zero-shot)	1.000	0.585	0.521	0.544	0.769
weight_avg	0.585	1.000	0.970	0.983	0.874
TA	0.521	0.970	1.000	0.985	0.792
TIES	0.544	0.983	0.985	1.000	0.825
breadcrumbs	0.769	0.874	0.792	0.825	1.000

Horizon trajectory. The horizon trajectory across baselines is visualized in Figure 3 (Section 4.1, in the main text): weight-space cosine decays almost monotonically with k, while activation-space CKA peaks at k=3 and stays nearly unchanged for avg 1:5. Early horizons align with the arithmetic direction, middle horizons align with the arithmetic representation, and the full average retains that representation while slightly improving accuracy.

Per-dataset breakdown. The per-dataset CKA values are reported in Table 3 (Section 4.1).

A.2.3. Per-pair 8×8 heatmaps – Additional baselines

Figures in this section give pair-level versions of the aggregate CKA and accuracy results, extending the headline weight-averaging heatmap (Section 4.1, Figure 2) to TIES, Task Arithmetic, and breadcrumbs (Figures 6, 7, 4, and 5 in this appendix). Each off-diagonal cell corresponds to one source-task pair and averages the two source-dataset evaluations. The left panel reports final-block CKA between NiNo and the baseline; the right panel reports the signed top-1 gap in percentage points. Pairwise NiNo always uses the paper's chosen `01221` recipe.

NiNo vs Task Arithmetic (avg 1:5). Figure 4 shows the avg 1:5 comparison against TA. CKA stays high on most pairs, with the weakest cells on EuroSAT pairs, and the accuracy gaps are usually small and negative.

The k=3 version of the same comparison (Figure 7) is nearly indistinguishable from the avg 1:5 panel here.

NiNo vs breadcrumbs. Figure 5 extends the same view to breadcrumbs. The CKA panel is noticeably lower than for the arithmetic baselines: cells on EuroSAT pairs drop below 0.65. The accuracy panel is dominated by positive gaps because breadcrumbs underperforms on this grid.

A.2.4. Midpoints input recipe – details

The midpoint negative control reported in Section 3.3 uses the alternative input sequence $[W_0,\ \frac{W_0+W_{t_1}}{2},\ W_{t_1},\ \frac{W_{t_1}+W_{t_2}}{2},\ W_{t_2}]$, evaluated on the same 28-pair grid as the chosen `01221` recipe.

Beyond the headline aggregate loss, the per-pair counts are also one-sided: the best matched midpoint horizon wins only 8 of the 28 pairs, and the midpoint avg 1:5 variant wins 7 of 28. The informative failure is at k=5, where midpoints collapse and win no pair. This is the clean direction-vs-representation dissociation that the main-text DVA paragraph (Section 4.1) describes at aggregate: a candidate can keep the right weight-space direction yet fall outside the useful activation cluster.

A.3. Multi-Subset 4-Task Benchmark – Setup and Per-Variant Notes

To test whether the pairwise picture extends beyond two tasks, we evaluate 10 strategies across five 4-task subsets sampled from the same 8-dataset pool. The baselines are Task Arithmetic, weight averaging, TIES, and breadcrumbs. NiNo contributes three composition families—direct, progressive-fold, and binary tree—each paired with both GNN and MLP backbones. All NiNo variants average horizons 1:5 and are evaluated on the four component validation splits. The five subsets cover different mixes of digit/symbol, texture, spatial, and fine-grained natural tasks. The headline numbers (Table 5) are reported in Section 4.1.

Beyond the aggregate ranking, the per-variant pattern is informative. `nino_direct_mlp` is the least stable composition (σ=8.18), collapsing on harder components such as DTD and SVHN, while `nino_tree_mlp` is the most stable (σ=2.48). Across recipes, direct favors the GNN backbone, tree favors the MLP backbone, and progressive-fold is the weakest family on this benchmark.

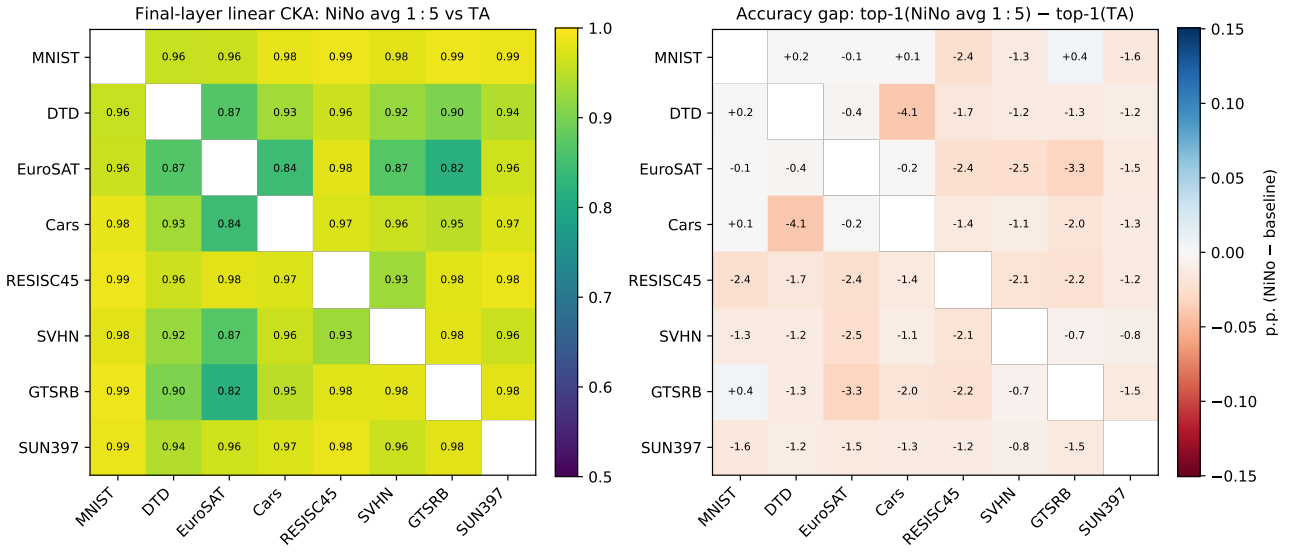

Figure 4. **Per-pair 8×8 comparison of NiNo avg $1{:}5$ against Task Arithmetic.** Left: final-block linear CKA, averaged over the two in-pair evaluation datasets. Right: accuracy gap (NiNo − TA) in percentage points.

A.4. 8-Task Same-Run Multi-Method Benchmark – Setup and Stage Detail

We scale to a single 8-task merge. Because NiNo only accepts five context states, same-run many-way meta-merge must be wrapped in either a binary tree or a progressive-fold topology. We also include an all-pair soup diagnostic that averages the 28 saved pairwise merges for each method. The headline numbers, per-dataset winners, all-pair-soup diagnostic, and backbone/topology interaction are reported in Section 4.1 (Table 6 and the surrounding multi-task discussion). Below we keep only the progressive-fold stage-wise detail and the negative-control tree-anchor observation.

Tree-anchor negative controls. The negative-control tree rows with alternative anchors support the backbone/topology reading: changing the anchor does not rescue the default GNN tree.

Progressive-fold stage diagnostics. A stage-wise diagnostic of the best progressive-fold family on the full 8-task validation set, not only on the tasks absorbed so far, shows that stage 3 is stronger than the final stage 4 checkpoint for all four runs.

For all four runs, the stage-3 checkpoint has a higher full-dataset average than the final stage-4 checkpoint. The last fold into GTSRB and SUN397 therefore helps some late tasks but hurts overall balance. Smaller α protects earlier tasks better, while larger α shifts capacity toward later tasks; in this batch, decay $\alpha_0{=}0.75$ gives the best final average and harmonic mean, and fixed $\alpha{=}0.375$ gives the best final minimum.

A.5. 8-Task Tree-Merge Order Ablation – Closing Remark

The headline tree-order numbers and per-task sensitivity are reported in Tables 7 and 8 (Section 4.1). Fixing the canonical 8-task tree by input permutation alone cannot close the baseline gap. A better many-way recipe would need either a flatter merge structure or a more order-symmetric construction.

A.6. Language-Domain Extension: Qwen3 – Setup

Starting from `Qwen/Qwen3-0.6B` (Yang et al., 2025), we use three public fine-tunes (math, French, and code) and evaluate both the 3-way merge and all three pairwise merges on GSM8K (Cobbe et al., 2021), French-Bench (Lasbordes & Gad, 2025), GSM8K-fr, and HumanEval-Instruct (Luo et al., 2023). Because this is a small shape-matched extension, we compare NiNo only to weight averaging and Task Arithmetic with $\lambda \in \{0.3, 0.4\}$.

For NiNo, pairwise merges use the paper's chosen `01221` recipe, and the 3-way merge uses the `01223` extension from `merge_qwen.py`. The pre-trained NiNo meta-network is unchanged from the vision setting; no Qwen-specific meta-training is performed. The full benchmark-by-pair table is reported in Table 9 of Section 4.1.

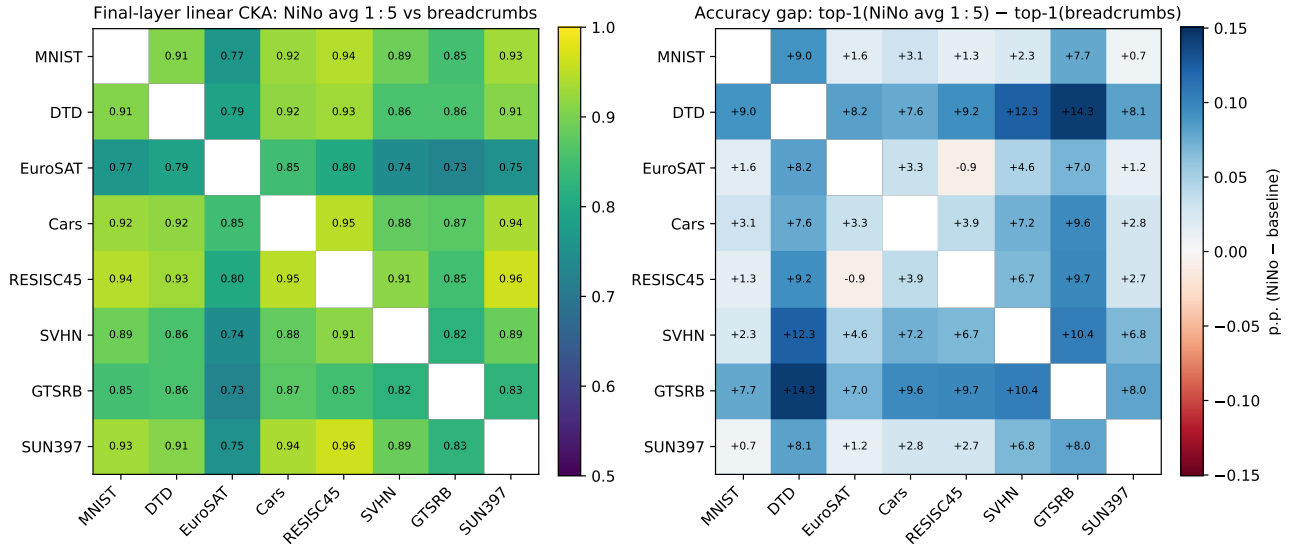

Figure 5. **NiNo avg** 1:5 **against breadcrumbs.** CKA is noticeably lower (cells on EuroSAT pairs drop below 0.65), and the accuracy panel is dominated by positive gaps because breadcrumbs underperforms on this grid.

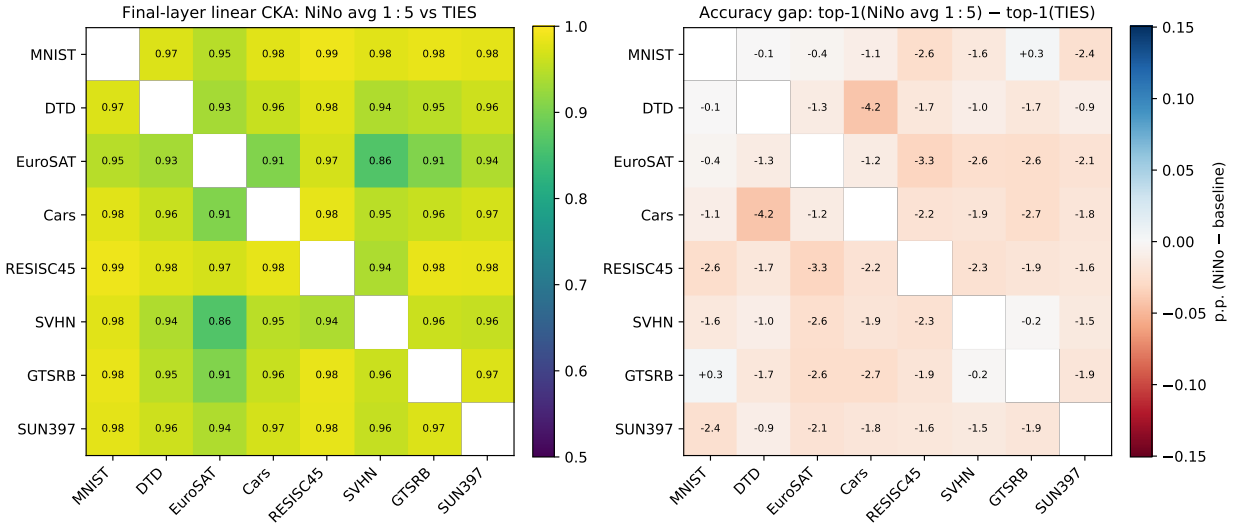

Figure 6. **Per-pair** 8×8 **comparison of NiNo avg** 1:5 **against TIES.** Same construction as Figure 2: CKA on the left, signed accuracy gap (NiNo − TIES) on the right. Compared to the weight-averaging panel, CKA stays high but the accuracy panel shifts more negative on DTD- and Cars-involving pairs, making visible the same direction-vs-accuracy trade-off that Section 4.1's DVA paragraph argues at the aggregate level.

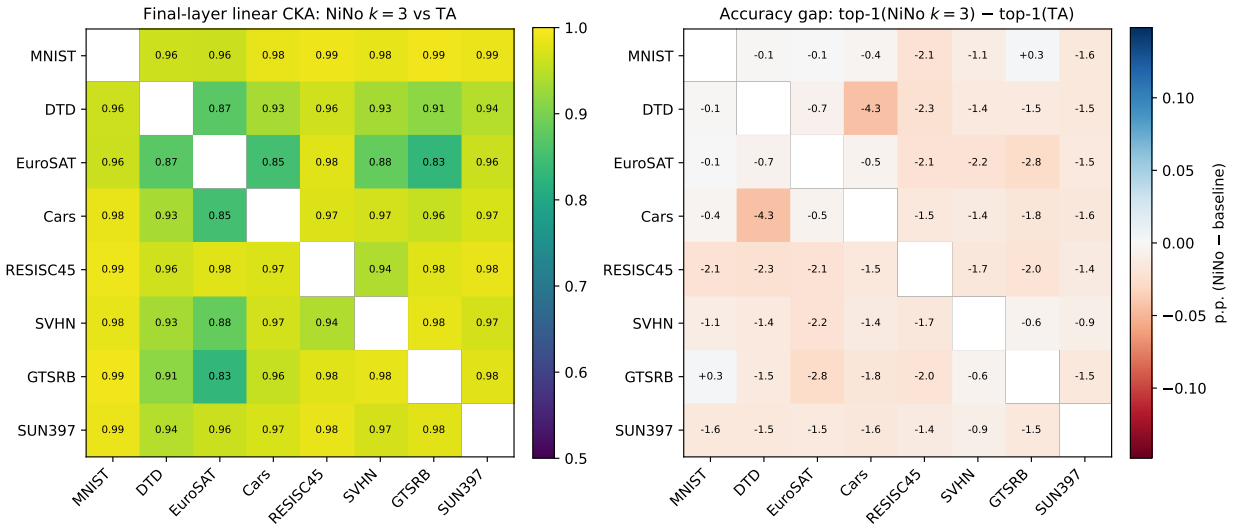

Figure 7. **Per-pair 8×8 comparison of NiNo k=3 against Task Arithmetic.** Same construction as Figure 2: CKA on the left, signed accuracy gap on the right. Using the single best NiNo horizon (k=3) as the anchor, the picture is nearly indistinguishable from the avg 1:5 vs TA appendix view (Figure 4); together with the wa and TIES heatmaps already in this section, this rounds out the pair-level geometry against the three arithmetic baselines.