SignGraph: Sign-Pattern Graph Diffusion for Model Merging

Anonymous ACL submission

Abstract

Model merging combines task-specific fine-tuned models into a single multi-task system without retraining, but existing methods process parameters independently, ignoring agreement patterns across tasks. We propose **SignGraph**, which clusters parameters by their N-bit sign fingerprint across task vectors, constructs a Hamming-distance graph over these clusters, and propagates consensus via personalized PageRank diffusion. Instead of discarding minority contributions through hard magnitude pruning, SignGraph uses graph-based smoothing and soft suppression to leverage cross-task sign agreement. Across 144 evaluation slots spanning eight models (0.5B–8B), nine tasks, and three merging settings, SignGraph wins 77 slots (53%), more than all baselines combined, and improves over the strongest baseline by +8.5 percentage points in average retention ($p < 0.001$, Wilcoxon signed-rank). Code: https://github.com/paultrust26/signgraph

1 Introduction

The open-source ecosystem hosts tens of thousands of specialized language models and adapters, each fine-tuned for a specific capability such as sentiment analysis, code generation, safety classification, or multilingual translation (Hu et al., 2022). In practice, users often need models that combine multiple abilities: a coding assistant that is also safe and multilingual, or a single endpoint serving diverse classification tasks. Joint multi-task training can produce such models but is expensive, requires access to all original training data, and must be repeated whenever a new task is added.

Model merging offers an alternative by combining independently trained models directly in parameter space, without additional training or data (Yang et al., 2024). The key abstraction is the *task vector*: the difference between a fine-tuned model's weights and its pretrained base. Task Arithmetic (Ilharco et al., 2023) showed that these vectors compose via linear operations. TIES-Merging (Yadav et al., 2023) improved robustness through magnitude pruning, majority sign election, and disjoint averaging, while DARE (Yu et al., 2024) introduced random dropout with rescaling.

Despite their effectiveness, these methods share key limitations. First, they process parameters independently, ignoring agreement patterns across tasks. Second, majority sign election discards all minority contributions. Third, hard magnitude pruning introduces a sharp threshold that is sensitive to the density hyperparameter. These issues become more pronounced as the number of merged adapters and the diversity of tasks increase.

We propose **SignGraph**, which replaces magnitude pruning and sign election with a graph-based approach. SignGraph clusters parameters by their N-bit sign fingerprint across task vectors, builds a Hamming-distance graph over these clusters, and runs personalized PageRank (PPR) diffusion to propagate consensus among groups with similar sign patterns. A soft sigmoid gate suppresses weak clusters rather than applying hard pruning. The pipeline runs in $O(ND + 2^{2N}T)$ time where N is the number of task vectors and D is the parameter count; since $2^{2N} \ll D$ for practical N, the cost is effectively linear in parameters.

Across 144 evaluation slots spanning eight models and three settings, SignGraph wins 53% of slots, more than all baselines combined (§5).

2 Related Work

Model merging combines multiple expert models into a single model without access to the original training data or costly retraining (Ilharco et al., 2023; Wortsman et al., 2022). Recent surveys provide a comprehensive overview of the field (Yang

et al., 2024; Yadav et al., 2024). The foundational insight is that models fine-tuned from the same pretrained checkpoint lie in connected regions of the loss landscape (Garipov et al., 2018), enabling meaningful interpolation in weight space.

Task Arithmetic (Ilharco et al., 2023) introduced the task vector formulation, showing that weight-space arithmetic can compose capabilities. TIES-Merging (Yadav et al., 2023) improved robustness through magnitude pruning, majority sign election, and disjoint averaging. DARE (Yu et al., 2024) replaced magnitude pruning with random dropout and rescaling; DARE-TIES combines DARE's dropout with TIES's sign election. Consensus-TIES (our baseline variant) extends TIES with conflict-aware density allocation that protects high-disagreement positions from pruning. Model Breadcrumbs (Davari and Belilovsky, 2023) applies outlier and small-magnitude pruning without sign consensus. DELLA (Deep et al., 2024) uses rank-based probabilistic pruning with L_1 rescaling. Fisher Merging (Matena and Raffel, 2022) weights parameters by diagonal Fisher information estimated from calibration data. All these methods operate element-wise on flattened parameter vectors without considering cross-task agreement patterns.

Recent work has applied merging to graph neural networks (Chen et al., 2026), building graphs over models to route inputs based on structural similarity. SignGraph differs by introducing graph structure into the merging operation on parameters themselves: it constructs a Hamming-distance graph over sign-pattern clusters and uses personalized PageRank (Page et al., 1999) diffusion to propagate consensus in weight space, without requiring additional data, optimization, or architectural changes.

3 Methodology: SignGraph

3.1 Preliminaries

Given a pretrained model $\theta_0 \in \mathbb{R}^D$ and N fine-tuned models $\{\theta_t\}_{t=1}^N$, the **task vector** for model t is $\tau_t = \theta_t - \theta_0$.

TIES-Merging (Yadav et al., 2023) combines task vectors in three steps: (1) **Trim**: magnitude-prune to density ρ; (2) **Elect Sign**: majority vote per parameter position; (3) **Disjoint Merge**: average only values matching the elected sign. The merged model is $\theta_{\text{merged}} = \theta_0 + \lambda \cdot \bar{\tau}_{\text{TIES}}$, where λ is a scaling coefficient.

Three properties motivate our approach: TIES processes elements independently without considering cross-task patterns; majority voting discards all minority contributions; and hard magnitude pruning is sensitive to ρ. These issues become more pronounced as N and task diversity increase (§5).

3.2 Overview

SignGraph replaces magnitude pruning and sign election with a graph-based approach. The key insight is that the N-dimensional sign pattern of a parameter across task vectors encodes meaningful information: parameters with identical sign patterns exhibit the same cross-task agreement and should be merged consistently. Figure 1 illustrates the four-stage pipeline. All equations below are applied independently per layer; we omit the layer index for clarity.

3.3 Sign Fingerprint Clustering

Given task vectors $\{\tau_t\}_{t=1}^N$, we preprocess and then cluster as follows.

Sign-flip correction. Occasionally two adapters learn equivalent solutions with opposite sign conventions, causing cancellation during averaging. We detect this by computing the correlation of each task vector against a reference: if $\text{corr}(\tau_t, \tau_1) < \gamma$, we flip $\tau_t \to -\tau_t$. The choice of τ_1 as reference is not critical because the threshold triggers only for strongly anti-correlated vectors.

Weighted averaging. We compute the element-wise weighted average $\delta = \sum_t w_t \tau_t$, where w_t are per-task importance weights, producing a single combined task vector.

Fingerprint assignment. Each parameter j is assigned to a cluster based on its N-bit sign fingerprint across the original (pre-averaged) task vectors (Equation 1):

$$c_j = \sum_{t=1}^{N} \mathbb{1}[\tau_t[j] > 0] \cdot 2^{t-1} \tag{1}$$

This yields at most 2^N distinct cluster labels $c_j \in \{0, \ldots, 2^N - 1\}$. Parameters with the same label agree on which tasks push the parameter positive and which push it negative. We use c to range over cluster labels, with $n_c = |\{j : c_j = c\}|$ denoting the size of cluster c. For each non-empty cluster

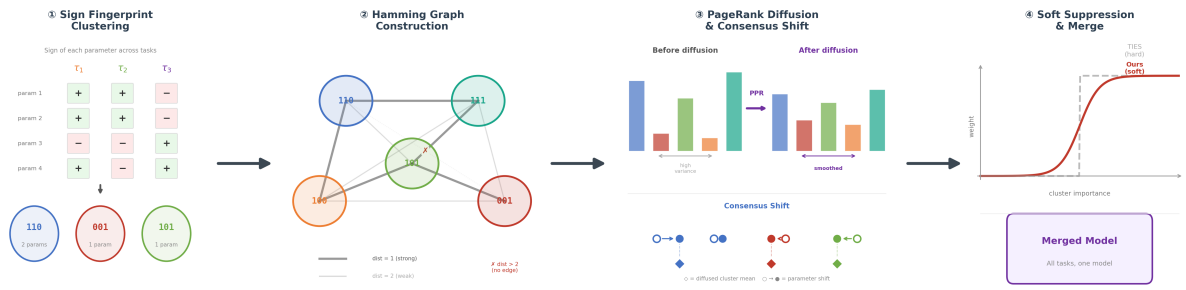

Figure 1: SignGraph pipeline overview. ① Parameters are grouped by N-bit sign fingerprint into clusters with identical sign patterns. ② A Hamming-distance graph connects structurally similar clusters (strong edges at distance 1, weak at distance 2, no edge beyond). ③ Personalized PageRank smooths cluster means, and a consensus shift moves parameters toward the diffused mean. ④ A soft sigmoid gate suppresses low-importance clusters (versus TIES's hard cutoff), producing the merged model.

we compute:

$$\mu_c = \frac{1}{n_c} \sum_{j:\, c_j = c} \delta_j, \qquad m_c = \frac{1}{n_c} \sum_{j:\, c_j = c} |\delta_j| \quad (2)$$

where μ_c is the cluster mean and m_c is the cluster magnitude.

3.4 Hamming Graph Construction

We build a graph over non-empty cluster labels. The edge weight between clusters c and c' is $1/h(c, c')$ when their Hamming distance satisfies $0 < h(c, c') \leq h_{\max}$, and zero otherwise. Clusters at Hamming distance 1 differ in only one task's sign, making them natural neighbors; clusters at larger distances receive no edge. Isolated clusters with no neighbors within $h_{\max}$ retain a self-loop and remain unchanged by diffusion. We row-normalize the adjacency matrix A to obtain the transition matrix $\tilde{A}$.

3.5 PageRank Diffusion and Consensus Shift

Starting from cluster means $\mu^{(0)} = \mu$, we run personalized PageRank diffusion:

$$\mu^{(k+1)} = \alpha \cdot \mu^{(0)} + (1 - \alpha) \cdot \tilde{A}\mu^{(k)} \quad (3)$$

with restart probability α for T steps. The restart term preserves each cluster's original signal, while the diffusion term propagates information from similar neighbors, smoothing noisy cluster estimates. After diffusion, a consensus shift moves each parameter toward its cluster's smoothed mean:

$$\hat{\delta}_j = \delta_j + \beta \cdot (\mu_{c_j}^{(T)} - \mu_{c_j}^{(0)}) \quad (4)$$

where β controls the consensus strength. Clusters whose means shifted most during diffusion (those that were outliers relative to their neighbors) receive the largest correction.

Parameter	Default	Role
ρ (density)	tuned	Suppression threshold
γ (correlation)	-0.7	Sign flip threshold
α (restart)	0.3	PPR restart probability
T (steps)	3	Diffusion iterations
β (consensus)	0.3	Consensus shift weight
$h_{\max}$	2	Max Hamming for edges
s (steepness)	10	Sigmoid steepness

Table 1: SignGraph hyperparameters. Only density ρ requires tuning; all others use fixed defaults across all models and settings.

3.6 Soft Suppression and Merge

We compute per-cluster importance $I_c = n_c \cdot m_c$ (size $\times$ magnitude), determine a threshold at the $(1-\rho)$ quantile, and apply soft sigmoid suppression:

$$\hat{\delta}_j \leftarrow \hat{\delta}_j \cdot \sigma\big(s \cdot (I_{c_j} - \text{thresh})\big) \quad (5)$$

where σ is the sigmoid function, s controls steepness, and ρ is the density parameter. Unlike TIES's hard cutoff, which zeros all parameters below the threshold, the sigmoid attenuates marginally important clusters gradually. The final merged model is $\theta_m = \theta_0 + \lambda \cdot \hat{\delta}$. Algorithm 1 presents the full procedure. The pipeline runs in $O(N{\cdot}D{+}K^2{\cdot}T)$ where $K = 2^N \ll D$; for our experiments with $N \leq 9$, graph costs are negligible and the overall complexity is effectively $O(ND)$, matching TIES.

3.7 Hyperparameters

We vary only the density ρ across experiments; all other reported results apart from the component ablation (§5.6) use the fixed defaults in Table 1.

Algorithm 1 SignGraph Merging Procedure.

Input: $\{\theta_t\}_{t=1}^{N}, \theta_0, \{w_t\}, \lambda, \rho, \gamma, \alpha, T, \beta, h_{\max}, s$
Output: Merged model θ_m

▷ *Stage 1: Sign Fingerprint Clustering*
for each t **do**
  $\tau_t \leftarrow \theta_t - \theta_0$
  if $\mathrm{corr}(\tau_t, \tau_1) < \gamma$ **then**
    $\tau_t \leftarrow -\tau_t$
  end if
end for
$\delta \leftarrow \sum_t w_t \tau_t$
for each j **do**
  $c_j \leftarrow \sum_t \mathbb{1}[\tau_t[j] > 0] \cdot 2^{t-1}$
end for
for each cluster c **do**
  $n_c \leftarrow |\{j : c_j = c\}|$; $\mu_c \leftarrow \frac{1}{n_c} \sum_{j:\, c_j = c} \delta_j$; $m_c \leftarrow \frac{1}{n_c} \sum_{j:\, c_j = c} |\delta_j|$
end for
▷ *Stage 2: Hamming Graph Construction*
for each pair c, c' **do**
  $A_{c,c'} \leftarrow 1/h(c, c')$ if $0 < h(c, c') \leq h_{\max}$, else 0
end for
$\tilde{A} \leftarrow \mathrm{row_normalize}(A)$
▷ *Stage 3: PageRank Diffusion & Consensus Shift*
$\mu^{(0)} \leftarrow \mu$
for $k = 0$ **to** $T-1$ **do**
  $\mu^{(k+1)} \leftarrow \alpha \mu^{(0)} + (1-\alpha) \tilde{A} \mu^{(k)}$
end for
for each j **do**
  $\hat{\delta}_j \leftarrow \delta_j + \beta(\mu_{c_j}^{(T)} - \mu_{c_j}^{(0)})$
end for
▷ *Stage 4: Soft Suppression & Merge*
for each cluster c **do**
  $I_c \leftarrow n_c \cdot m_c$
end for
$\mathrm{thresh} \leftarrow \mathrm{quantile}_{1-\rho}(\{I_c\})$
for each j **do**
  $\hat{\delta}_j \leftarrow \hat{\delta}_j \cdot \sigma(s(I_{c_j} - \mathrm{thresh}))$
end for
return $\theta_m \leftarrow \theta_0 + \lambda \cdot \hat{\delta}$

4 Experimental Setup

Models. We evaluate on eight autoregressive language models (0.5B–8B) spanning two architecture families: Llama-3.2-1B-Instruct, Llama-3.2-3B-Instruct, Llama-3-8B-Instruct (Grattafiori et al., 2024), Qwen2.5-0.5B-Instruct, Qwen2.5-1.5B-Instruct, Qwen2.5-3B-Instruct, Qwen2.5-7B-Instruct (Qwen Team, 2025), and Qwen3-0.6B. All models are generative; classification tasks (GLUE and safety) are evaluated via prompt-based generation where the model produces a label token. Task-specific models are produced via Low-Rank Adaptation (LoRA) fine-tuning (Hu et al., 2022) with rank r=16 and LoRA scaling α_{LoRA}=32, targeting all attention and multi-layer perceptron (MLP) projection layers. While SignGraph operates on arbitrary task vectors, we focus on LoRA deltas as they represent the most common practical setting for adapter merging.

Three evaluation settings.

1. **GLUE**: Six classification tasks from the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019) (SST-2, CoLA, RTE, QQP, MNLI, QNLI). 8 models × 6 tasks = 48 slots.

2. **All Tasks**: GLUE + safety classification (WildGuardMix (Han et al., 2024)) + code generation (OpenCodeInstruct (Zhuo et al., 2024)) + multilingual translation (Yoruba–English, Aya (Singh et al., 2024)). 8 × 9 = 72 slots.

3. **Non-GLUE Only**: Safety + code + multilingual only (3 adapters merged). 8 × 3 = 24 slots. This diagnostic setting isolates non-classification performance from GLUE adapter interference (§5.3).

Baselines. Eight methods that operate on pre-computed task vectors without gradient-based optimization: TIES (Yadav et al., 2023), DARE-TIES (Yu et al., 2024), DELLA (Deep et al., 2024), Consensus-TIES (our conflict-aware variant), Fisher Merging (Matena and Raffel, 2022), Linear / Task Arithmetic (Ilharco et al., 2023), Simple Average (Wortsman et al., 2022), and Model Breadcrumbs (Davari and Belilovsky, 2023). We exclude methods requiring optimization (RegMean, LoRAHub) or routing infrastructure (SMILE, evolutionary merging) to ensure a fair comparison among non-optimization methods. MergeKit (Goddard et al., 2024) provides implementations of several baselines; we reimplement all methods for controlled comparison.

Densities and metric. Each method is evaluated at $\rho \in \{0.01, 0.05, 0.50, 0.99\}$. Note that density has different semantics across methods: for TIES-like methods, ρ controls parameter-level retention after pruning; for SignGraph, ρ determines the quantile threshold over cluster importance, so effective parameter retention may differ at equal ρ. We report **retention** (Yadav et al., 2023; Yu et al., 2024): the ratio of the merged model's task performance to the individually fine-tuned model's performance:

$$\text{retention}_t = \frac{\text{Task Metric}_t^{\text{merged}}}{\text{Task Metric}_t^{\text{individual}}} \times 100 \qquad (6)$$

where the task-specific score is macro F1 for GLUE and safety classification (we use F1 uni-

formly because all tasks are evaluated via prompt-based label generation, where standard GLUE metrics like Matthews correlation are not directly applicable), test pass rate (pass@1) for code generation, and ROUGE-1 (Recall-Oriented Understudy for Gisting Evaluation) for multilingual translation. All scores are computed on held-out test splits. A method "wins" a slot if it achieves the highest retention across all methods and densities.

Reproducibility. All experiments are conducted on a single 48GB NVIDIA GPU using the Hugging Face Transformers (Wolf et al., 2020) and PEFT (Mangrulkar et al., 2022) libraries. Each dataset is split into train/validation/test with an 8:1:1 ratio. Adapters are trained with early stopping on validation loss (learning rate 2×10^{-4}, batch size 16, AdamW optimizer, up to 10 epochs). Merging is deterministic given fixed task vectors; stochastic methods (DARE-TIES, DELLA) use fixed seeds. All evaluation uses greedy decoding. Code is available at `https://github.com/paultrust26/signgraph` and adapters at `https://hf.co/collections/paultrust/signgraph`.

5 Results and Discussion

We report results across three evaluation settings (Tables 2–4), followed by analysis of adapter quality effects, statistical significance, and component ablation.

5.1 GLUE Tasks (6 Adapters)

Table 2 shows average retention across eight models per task, with win counts in parentheses. SignGraph wins 27 of 48 slots (56%) and leads in average retention on 5 of 6 tasks, with the largest margins on RTE (+11.2 points over DARE-TIES) and CoLA (+4.0 points over Breadcrumbs). On QQP, Consensus-TIES achieves the highest average retention (89.6%), where the task's clean sign structure benefits hard voting.

5.2 All Tasks (9 Adapters)

The full 9-task setting (Table 3) adds code generation and multilingual (Yoruba–English) translation to GLUE and safety. SignGraph wins 42 of 72 slots (58%) and leads average retention on 8 of 9 tasks. The generative tasks are challenging for all methods: multilingual retention averages below 15% across the board. Despite this, SignGraph remains the top method, winning more than $4\times$ the slots of the next-best baseline (TIES, 9 wins). With more task vectors, the sign fingerprint becomes higher-dimensional, yielding finer-grained clusters; by contrast, TIES's majority voting becomes coarser, as a 5–4 vote discards nearly half the contributions. We investigate whether the lower non-GLUE performance reflects a method limitation or adapter interference in §5.3.

5.3 Non-GLUE Tasks (3 Adapters)

In the All Tasks setting, SignGraph wins relatively few non-GLUE slots. To isolate the cause, we evaluate a Non-GLUE Only setting that merges only the three non-GLUE adapters without GLUE adapters (Table 4). SignGraph's safety retention jumps from 58.8% to 100.9%, and it leads all methods in overall average retention (67.6% versus 62.2% for Consensus-TIES).

This indicates a majority-dominance effect: when six classification adapters outnumber three non-GLUE adapters, they dominate the sign-pattern clusters and suppress minority contributions. Notably, *no method* dominates non-GLUE tasks in the All Tasks setting, suggesting this is a general challenge of heterogeneous multi-task merging rather than a SignGraph-specific limitation.

5.4 Merging Success versus Adapter Quality

Table 5 shows the relationship between individual adapter accuracy and merging outcomes. We bin 78 unique model-task adapters (excluding multilingual adapters whose low absolute scores make retention unstable) by their individually fine-tuned accuracy. When adapters are weak (<78% accuracy), all methods struggle; SignGraph wins only 42% of slots with -2.3 points below the per-slot best baseline. As adapter quality increases, SignGraph's advantage grows: at the highest tier (>92% accuracy), SignGraph wins 58% of slots with +2.9 points above the best baseline.

Table 6 further breaks this down: SignGraph leads in average retention across all quality tiers. Its lower win rate on weak adapters (42%) reflects cases where specific baselines outperform on individual task-model pairs. On moderate-to-strong adapters, SignGraph leads all baselines by 4–6 points.

This has a practical implication: *investing in good individual adapters matters for merging*. On weak adapters, noisy sign patterns limit graph diffusion's effectiveness. When adapters are strong, their sign patterns become more coherent and

Method	Avg	CoLA	MNLI	QNLI	QQP	RTE	SST-2
DARE-TIES (Yu et al., 2024)	84.2 (4)	85.3 (0)	80.8 (3)	80.0 (0)	87.1 (1)	82.8 (0)	89.2 (0)
Breadcrumbs (Davari and Belilovsky, 2023)	83.0 (3)	87.3 (0)	81.6 (1)	77.5 (0)	88.3 (1)	80.4 (1)	83.1 (0)
Fisher (Matena and Raffel, 2022)	78.1 (1)	57.2 (0)	84.2 (0)	75.8 (0)	85.1 (0)	81.5 (0)	84.6 (1)
Simple Avg (Wortsman et al., 2022)	78.0 (1)	57.2 (0)	84.2 (1)	75.8 (0)	85.0 (0)	81.5 (0)	84.6 (0)
Consensus-TIES	77.1 (8)	72.8 (1)	73.5 (1)	73.8 (2)	**89.6** (1)	73.9 (1)	78.2 (2)
TIES (Yadav et al., 2023)	76.6 (1)	72.9 (0)	76.3 (0)	72.8 (0)	89.4 (1)	72.9 (0)	74.9 (0)
DELLA (Deep et al., 2024)	68.3 (3)	69.1 (0)	67.7 (0)	63.4 (2)	84.9 (1)	58.7 (0)	66.3 (0)
Linear (Ilharco et al., 2023)	68.9 (0)	67.6 (0)	48.5 (0)	71.3 (0)	78.6 (0)	66.6 (0)	82.2 (0)
SignGraph (ours)	**91.5** (27)	**91.3** (7)	**84.5** (2)	**90.8** (4)	89.1 (3)	**94.0** (6)	**99.2** (5)

Table 2: Average retention (%) per task across eight models, **GLUE Only** (48 slots). Parenthetical = win count (best retention among all methods for that model-task). Shaded = SignGraph. **Bold** = highest average in column.

Method	Avg	CoLA	MNLI	QNLI	QQP	RTE	SST-2	Safety	Code	Multi
Breadcrumbs	70.6 (7)	85.7 (1)	79.9 (0)	72.8 (1)	87.7 (0)	80.7 (1)	82.0 (2)	68.4 (2)	67.3 (0)	12.2 (0)
DARE-TIES	69.7 (2)	84.9 (0)	81.0 (0)	74.8 (1)	77.5 (0)	75.0 (0)	93.4 (0)	58.8 (0)	72.4 (1)	10.0 (0)
Consensus-TIES	70.1 (5)	73.7 (0)	79.5 (0)	71.7 (0)	82.6 (0)	76.8 (1)	82.5 (0)	**81.7** (2)	76.9 (1)	10.0 (1)
TIES	69.5 (9)	68.6 (3)	79.9 (1)	72.1 (0)	83.4 (1)	76.8 (0)	82.4 (1)	80.7 (1)	75.0 (1)	10.1 (1)
Simple Avg	66.0 (1)	85.4 (1)	81.7 (0)	72.6 (0)	73.9 (0)	73.6 (0)	83.2 (0)	46.3 (0)	69.6 (0)	5.7 (0)
Fisher	65.9 (1)	85.4 (0)	81.3 (0)	72.5 (0)	73.9 (0)	73.4 (0)	83.1 (0)	46.5 (0)	69.3 (1)	5.8 (0)
DELLA	66.6 (4)	73.1 (0)	76.5 (0)	65.3 (0)	80.3 (0)	68.5 (0)	80.3 (0)	80.7 (1)	70.3 (2)	8.8 (1)
Linear	47.9 (1)	61.1 (0)	43.4 (0)	44.4 (0)	54.7 (0)	54.6 (0)	72.4 (0)	73.8 (0)	29.6 (0)	6.7 (1)
SignGraph (ours)	**76.6** (42)	**86.7** (3)	**88.1** (7)	**84.4** (6)	**89.9** (7)	**89.7** (6)	**97.8** (5)	58.8 (2)	**79.1** (2)	**14.4** (4)

Table 3: Average retention (%) per task across eight models, **All Tasks** (72 slots). Format: avg (wins). Code = code generation, Multi = multilingual Yoruba–English. Coding and multilingual tasks are challenging for all methods.

Method	Avg	Safety	Code	Multi
Consensus-TIES	62.2 (6)	**101.5** (4)	75.1 (2)	10.2 (0)
TIES	62.1 (1)	101.3 (1)	75.0 (0)	9.8 (0)
DARE-TIES	61.2 (0)	97.2 (0)	76.2 (0)	10.3 (0)
Breadcrumbs	60.9 (1)	97.9 (1)	71.0 (0)	13.9 (0)
DELLA	60.4 (2)	100.6 (1)	66.7 (0)	14.0 (1)
Fisher	53.1 (3)	72.0 (0)	77.4 (2)	9.8 (1)
Simple Avg	52.4 (1)	72.1 (0)	75.3 (1)	9.9 (0)
Linear	44.6 (2)	89.4 (0)	29.1 (0)	13.2 (2)
SignGraph (ours)	**67.6** (8)	100.9 (1)	**80.1** (3)	**21.7** (4)

Table 4: Average retention (%) per task across eight models, Non-GLUE Only (24 slots, 3 adapters merged). Format: avg (wins). Safety retention jumps dramatically when GLUE adapters are removed.

Tier	Count	Win Rate	SignGraph	Best BL	Δ
Weak	24	42%	49.4	51.8	−2.3
Moderate	20	55%	90.1	88.7	+1.4
Strong	22	50%	90.6	88.8	+1.8
Very Strong	12	**58%**	**93.0**	90.1	**+2.9**

Table 5: SignGraph performance by adapter quality. Tiers: Weak (<78%), Moderate (78–86%), Strong (86–92%), Very Strong (>92%) individual fine-tuned accuracy. "Win Rate" = SignGraph win rate, "SignGraph"/"Best BL" = average retention (%), "Δ" = gap in percentage points.

SignGraph's clustering captures meaningful cross-task agreement.

Per-task patterns. Figure 2 shows that SignGraph's wins concentrate in the high-adapter-accuracy region, while losses cluster at lower quality. Figure 3 shows win rates by task: strongest on RTE (75%), CoLA, QNLI, QQP, and SST-2 (63% each), and weakest on Safety (19%) and Code (31%). The generative tasks are challenging for all methods, with retention often below 50%.

Scaling with adapter count. With more task vectors, the sign fingerprint becomes higher-dimensional, creating finer-grained clusters. TIES's majority voting becomes coarser: with 9 models, a 5–4 vote discards 4 models' contributions entirely. PPR diffusion provides a softer mechanism, allowing similar clusters to share information without hard cutoffs.

Density behavior. SignGraph performs best at low densities (ρ=0.01 and 0.05), leading baselines by approximately 10 points; at ρ=0.50–0.99, methods converge. Figure 4 shows that 65% of SignGraph's optimal configurations occur at $\rho \leq 0.05$, compared to 13% at ρ=0.01 for baselines.

Method	Weak	Moderate	Strong	Very Strong
SignGraph	**49.4**	**90.1**	**90.6**	**93.0**
DARE-TIES	42.7	85.6	84.1	88.6
Breadcrumbs	45.3	84.7	86.5	73.7
Cons-TIES	47.9	76.2	82.5	72.0
TIES	47.3	77.5	82.3	72.2
DELLA	48.1	66.5	74.9	71.0

Table 6: Average retention (%) by adapter quality tier for top methods. Bold = best in column. SignGraph leads in all quality tiers, with the gap growing from +1.3 on Weak to +4.4 percentage points on Very Strong adapters.

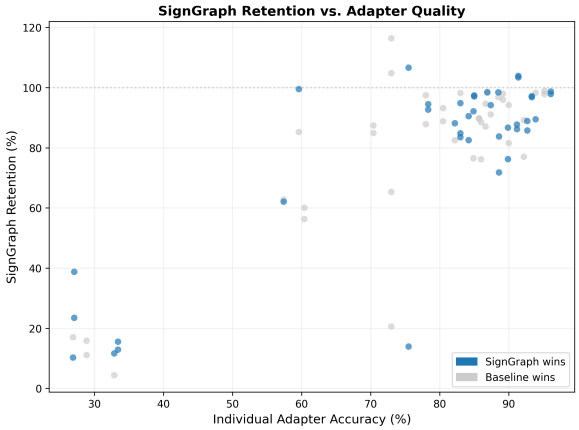

Figure 2: SignGraph retention versus individual adapter accuracy. Blue = SignGraph wins the slot; gray = a baseline wins. SignGraph wins concentrate in the high-accuracy region (upper right), while losses cluster at lower adapter quality.

SignGraph's soft suppression handles very sparse settings well, whereas TIES's hard cutoff requires retaining more parameters to be effective.

Model-level patterns. Qwen2.5-1.5B is SignGraph's weakest model (13 losses), where Consensus-TIES and DARE-TIES already achieve >90% retention, leaving little room for improvement. Conversely, Qwen2.5-0.5B and Qwen3-0.6B are SignGraph's strongest (5 losses each); on these smaller models, baselines struggle more and graph-based consensus provides the largest benefit.

Constructive interference. Several cells in Table 4 show retention >100%: safety retention in the Non-GLUE setting reaches 100.9–101.5%, suggesting that merging may produce models that slightly outperform individually fine-tuned ones, though small improvements above 100% could also reflect evaluation variance.

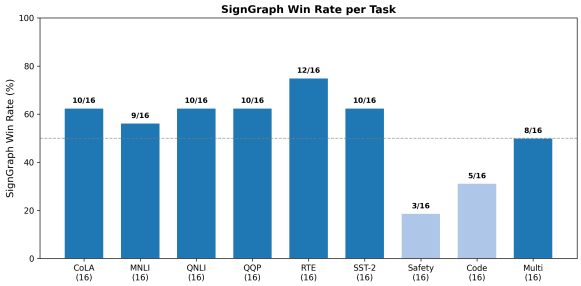

Figure 3: SignGraph win rate per task (number of slots in parentheses). Dashed line = 50%. SignGraph dominates most classification tasks (63–75%) but is weakest on Safety (19%) and Code (31%).

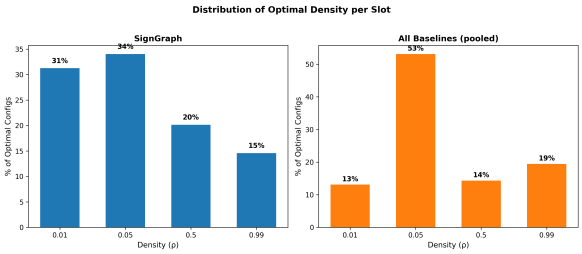

Figure 4: Distribution of optimal density per slot. Left: 65% of SignGraph's best configurations are at $\rho \leq 0.05$, with a strong preference for very low densities (ρ=0.01: 31%). Right: Baselines also favor ρ=0.05 (53%) but have far fewer optimal configurations at ρ=0.01 (13%), showing that SignGraph's soft suppression uniquely benefits very sparse settings.

5.5 Statistical Significance

We conduct pairwise Wilcoxon signed-rank tests across 144 model-task-setting slots (pooling all evaluation settings) to assess whether SignGraph's gains are consistent. Since these slots are not fully independent (the same models and tasks appear across settings), we report these tests as supporting evidence. For each slot, we take each method's best retention across densities and compare using two-sided tests with Holm-Bonferroni correction for eight comparisons. As shown in Table 7, all comparisons are significant at $p < 0.001$. The closest baselines (DARE-TIES, Breadcrumbs) show a mean gap of +8.5 points with medium effect size ($d \approx 0.5$). Against TIES and Consensus-TIES, the gap widens to +11.4 points ($d \approx 0.54$). SignGraph wins at least 103 of 144 head-to-head slots against every baseline.

5.6 Component Ablation

To isolate each component's contribution, we evaluate three ablations at ρ=0.01 across all eight models: removing graph diffusion ("−Graph"), re-

Baseline	Δ	p	d	W/L
Breadcrumbs	+8.5	1.6e-14	.50	114/28
DARE-TIES	+8.5	2.1e-15	.58	120/24
Cons-TIES	+11.4	7.1e-10	.54	103/39
TIES	+11.4	3.4e-11	.54	107/35
Fisher	+12.6	3.9e-18	.66	124/18
Simple Avg	+12.7	4.6e-19	.66	125/17
DELLA	+17.2	1.3e-14	.62	110/32
Linear	+38.7	3.6e-25	1.19	141/3

Table 7: Statistical significance of SignGraph versus each baseline (144 paired slots). Δ: mean retention difference (percentage points). p: Wilcoxon signed-rank. d: Cohen's d. W/L: wins/losses by >0.1 points. All $p < 0.001$.

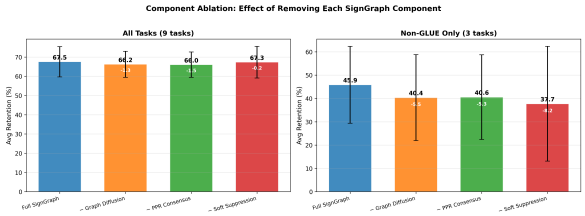

Figure 5: Component ablation: average retention after removing each SignGraph component. Effects are 4–6× larger on Non-GLUE (-5.5 to -8.2 points) than All Tasks (-0.2 to -1.5 points), showing that graph diffusion and soft suppression are most critical for non-classification tasks.

moving PPR consensus ("−Consensus"), and replacing soft suppression with hard thresholding ("−Soft Suppression"). We analyze the All Tasks and Non-GLUE settings separately (Figure 5).

All Tasks setting. On All Tasks (9 adapters), component removal has modest effects: -1.3 points without graph diffusion, -1.5 without PPR consensus, and -0.2 without soft suppression. The GLUE tasks dominate the average and are relatively robust to ablation, masking larger effects on non-GLUE tasks.

Non-GLUE setting. Effects are much larger without GLUE adapters. On Non-GLUE (3 adapters), removing graph diffusion costs -5.5 points, removing consensus costs -5.3 points, and removing soft suppression costs -8.2 points. Code generation drives this: removing graph diffusion drops code retention by -16.4 points in Non-GLUE versus only -3.0 in All Tasks. This indicates that graph diffusion is most valuable when merged adapters span diverse tasks without a dominant majority.

Hyperparameter sensitivity. We sweep sigmoid steepness $s \in \{1, 5, 50\}$ (default 10), PPR iterations $T \in \{1, 15\}$ (default 3), and maximum Hamming distance $h_{\max} \in \{1, 9\}$ (default 2). All variations produce less than 1 point change in both settings, indicating low sensitivity to these choices. The largest effect is $T{=}1$ on All Tasks (-0.6 points), suggesting that even a single diffusion step captures most of the benefit.

6 Conclusion

Existing model merging methods process parameters independently through element-wise pruning and majority voting. We presented SignGraph, which clusters parameters by their sign patterns across task vectors and smooths cluster representations via personalized PageRank on a Hamming-distance graph. Across eight models, three settings, and 144 evaluation slots, SignGraph wins 53% of slots, more than all baselines combined, with a mean retention gap of +8.5 points over the closest baselines (Breadcrumbs, DARE-TIES; $p < 0.001$). Ablations show that graph diffusion and soft suppression are most important for non-classification tasks, while sensitivity to other hyperparameters is low.

Limitations

SignGraph's cluster count grows as 2^N with the number of task vectors N. In practice, merging rarely exceeds 10–15 tasks; our experiments use up to nine (512 clusters) where graph costs are negligible. For larger N, hierarchical clustering or graph sparsification could be applied. Weak adapters produce noisy task vectors that degrade all merging methods; adaptive quality-aware mechanisms are left to future work.

Ethical Statement

All models and datasets are publicly available under permissive licenses with no personally identifiable information. Model merging reduces computational cost and carbon emissions versus retraining. AI assistants aided paraphrasing, brainstorming, and code generation; all outputs were author-verified.

References

Jun Chen, Ziyue Qiao, Qin Zhang, Kaize Ding, and Xiao Luo. 2026. G-merging: Graph models merging

for parameter-efficient multi-task knowledge consolidation. In *International Conference on Learning Representations*.

MohammadReza Davari and Eugene Belilovsky. 2023. Model breadcrumbs: Scaling multi-task model merging with sparse masks. *arXiv preprint arXiv:2312.06795*.

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

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry Vetrov, and Andrew Gordon Wilson. 2018. Loss surfaces, mode connectivity, and fast ensembling of DNNs. *Advances in Neural Information Processing Systems*.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vlad Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. 2024. Arcee's MergeKit: A toolkit for merging large language models. *arXiv preprint arXiv:2403.13257*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, and 1 others. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

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

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

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

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. PEFT: State-of-the-art parameter-efficient fine-tuning methods.

Michael S. Matena and Colin A. Raffel. 2022. Merging models with Fisher-weighted averaging. In *Advances in Neural Information Processing Systems*.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank citation ranking: Bringing order to the web. *Stanford InfoLab Technical Report*.

Qwen Team. 2025. Qwen2.5 technical report. In *arXiv preprint arXiv:2412.15115*.

Shivalika Singh, Freddie Vargus, Daniel D'souza, Börje F. Karlsson, Abinaya Mahendiran, Wei-Yin Ko, Herumb Shandilya, Jay Patel, Deividas Mataciunas, Laura OMahony, and 1 others. 2024. Aya dataset: An open-access collection for multilingual instruction tuning. *arXiv preprint arXiv:2402.06619*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

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. 2022. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proceedings of the 39th International Conference on Machine Learning*.

Prateek Yadav, Colin Raffel, Mohammed Muqeeth, Lucas Caccia, Haokun Liu, Tianlong Chen, Mohit Bansal, Orion Weller, and Benjamin Van Durme. 2024. A survey on model MoErging: Recycling and routing among specialized experts for collaborative learning. *arXiv preprint arXiv:2408.07057*.

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

Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. 2024. Model merging in LLMs, MLLMs, and beyond: Methods, theories, applications and opportunities. *arXiv preprint arXiv:2408.07666*.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super Mario: Absorbing abilities from homologous models as a free lunch. In *Proceedings of the 41st International Conference on Machine Learning*.

Tianyu Zhuo, Zhaojian Yu, Yue Wang, Jing Zhang, Shuaiyi Li, Yuxiang Zhang, Yanlin Wang, Shengyu Zhang, Junjie Wang, and Yida Xu. 2024. OpenCodeInterpreter: Integrating code generation with execution and refinement. *arXiv preprint arXiv:2402.14658*.

A Appendix

A.1 Retention Distribution

Figure 6 shows the retention distribution across all 144 slots for each method. SignGraph has the highest median and the tightest interquartile range among competitive methods.

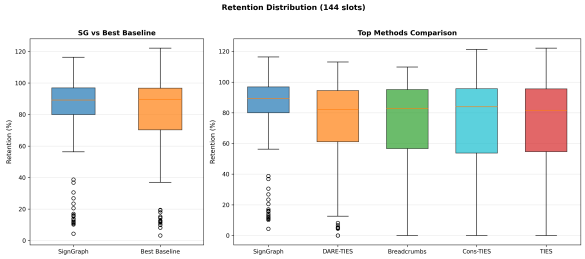

Figure 6: Left: SignGraph versus the best baseline per slot (a fair one-versus-one comparison). SignGraph has a higher median and tighter IQR. Right: SignGraph versus the four strongest individual baselines. Outliers at the bottom are safety/coding/multilingual slots that affect all methods.

A.2 Per-Task Win Breakdown

Table 8 shows how wins distribute across individual tasks.

Method	CoLA	MNLI	QNLI	QQP	RTE	SST-2	Safety	Code	Multi	Tot
SignGraph	10	9	10	10	12	10	3	5	8	**77**
Cons-TIES	1	1	2	1	2	2	6	3	1	19
Breadcrumbs	1	1	1	1	2	2	3	0	0	11
TIES	3	1	0	2	0	1	2	1	1	11
DELLA	0	0	2	1	0	0	2	2	2	9
DARE-TIES	0	3	1	1	0	0	0	1	0	6
Fisher	0	0	0	0	0	1	0	3	1	5
Simple Avg	1	1	0	0	0	0	0	1	0	3
Linear	0	0	0	0	0	0	0	0	3	3
Total	16	16	16	16	16	16	16	16	16	144

Table 8: Win counts by task across all three settings (144 slots). Each task has 16 slots (8 models $\times$ 2 settings).

A.3 Head-to-Head Pairwise Comparisons

Table 9 shows pairwise comparisons between SignGraph and each baseline across all slots where both methods produce non-zero retention.

Baseline	Ours wins	BL wins	Ties	Gap (%)
Linear	141	3	0	+38.7
DELLA	110	32	2	+17.2
Simple Avg	125	17	2	+12.7
Fisher	124	18	2	+12.6
Cons-TIES	103	39	2	+11.4
TIES	107	35	2	+11.4
Breadcrumbs	114	28	2	+8.5
DARE-TIES	120	24	0	+8.5

Table 9: Head-to-head: SignGraph versus each baseline across 144 slots (including coding and multilingual). Gap = mean retention difference in percentage points (positive = SignGraph better). SignGraph wins >71% of head-to-head matchups against every baseline.

A.4 Error Analysis: When SignGraph Loses

Table 10 summarizes SignGraph's losses by task and model.

By task		
Task	**Losses**	**Gap (%)**
Safety	13	6.9
Code	11	4.2
Multi	8	3.2
MNLI	7	4.6
CoLA	6	1.9
QQP	6	3.2
QNLI	6	1.9
SST-2	6	0.5
RTE	4	1.9
By model		
Qwen2.5-1.5B	13	
Qwen2.5-7B	10	
Llama-3-8B	9	
Qwen2.5-3B	9	
Llama-3.2-1B	8	
Llama-3.2-3B	8	
Qwen2.5-0.5B	5	
Qwen3-0.6B	5	

Table 10: SignGraph losses by task and model. Losses are concentrated in tasks and models where adapter interference effects are strongest or where baselines already achieve >90% retention.

Losses cluster around three patterns. First, **safety, coding, and multilingual** tasks account for 32 of 67 losses, largely due to adapter interference in the All Tasks setting (§5.3). Second, **QQP and MNLI** on models where baselines with sign consensus (TIES, DELLA, Consensus-TIES) already achieve high retention. Third, **Qwen2.5-1.5B-Instruct** leads in losses (13), where task vectors are unusually compatible and baselines already reach >90% retention.