arXiv:2610.15322v2 [cs.AI] 14 Oct 2026

SuperAI: Merging Capabilities Across Heterogeneous Foundation Models

Jingtong Dou[*]
University of Sydney
Sydney, Australia

Chuancheng Shi[*]
University of Sydney
Sydney, Australia

Wenhua Wu
University of Sydney
Sydney, Australia

Wei Bao
University of Sydney
Sydney, Australia

Fei Shen[†]
National University of Singapore
Singapore

Zhiyong Wang
University of Sydney
Sydney, Australia

Tat-Seng Chua
National University of Singapore
Singapore

Abstract

Enabling the reuse and composition of knowledge and capabilities distributed across foundation models is an important step toward building scalable artificial intelligence (AI) systems. However, architectural differences and incompatible representations make it difficult to transfer and integrate these capabilities through a unified cross-model interface. To address this challenge, we propose SuperAI, a cross-architecture multi-model fusion framework that integrates complementary capabilities from multiple source models into a host model through orthogonal alignment of low-dimensional task-relevant representations. Specifically, we first extract task-relevant representations from the source models and the host model using shared anchor samples, and apply principal component analysis (PCA) to construct low-dimensional functional subspaces with the same dimensionality. Next, we use orthogonal Procrustes mappings to align the source and host coordinate systems, establishing cross-model correspondences while preserving the internal geometry of each source subspace. Finally, we train a lightweight residual multilayer perceptron (MLP) with joint supervision on task outputs and source-model increments, learning input-dependent residual increments that are injected into the host model for multi-model capability fusion. Experiments on three host models across 15 capability domains show that SuperAI achieves higher average domain scores than the evaluated cross-architecture fusion methods and better preserves general capabilities as the number of source models increases. Ablation studies further support the effectiveness of orthogonal alignment, nonlinear residual mapping, and joint supervision.

1 Introduction

Foundation models [Qwen et al., 2025, OLMo et al., 2024, Team et al., 2025, Shi et al., 2026a] develop complementary capabilities shaped by diverse training data, objectives, and architectures. These capabilities remain distributed across models [Ramtoula et al., 2026, Dou et al., 2026a, Wei et al., 2026, Shi et al., 2026b, Dou et al., 2026b, Li et al., 2026, Shi et al., 2026c], while acquiring new ones through further training is costly. Reusing and composing existing capabilities [Akiba et al., 2025] therefore offers an important path toward scalable artificial intelligence (AI) systems.

Existing approaches [Wortsman et al., 2022, Ilharco et al., 2022, Yoshida et al., 2026] reuse capabilities through parameter merging [Akiba et al., 2025], knowledge distillation [Ko et al., 2024], and multi-model collaboration [Wang et al., 2025]. Conventional parameter merging combines weights Yadav et al. [2023] or task vectors Ortiz-Jimenez et al. [2023] but generally requires compatible architectures and parameter structures. Knowledge distillation transfers information from source-model outputs or intermediate representations through additional host-model training, while routing and collaboration invoke different models according to the input and typically retain multiple models at deployment.

[*]Equal contribution. Email for Jingtong Dou: `jdou0056@uni.sydney.edu.au`.
[†]Corresponding author. Email: `shenfei29@nus.edu.sg`.

To accommodate architectural differences, cross-model fusion methods also use neuron matching [Xu et al., 2024], optimal transport [Cui et al., 2026, Imfeld et al., 2024], and representation alignment [Dery et al., 2026] to establish correspondences.

However, as shown in Figure 1, integrating complementary information from multiple heterogeneous source models into a common host representation space remains challenging, particularly when improving target-task performance must be balanced with retaining the host model's existing capabilities.

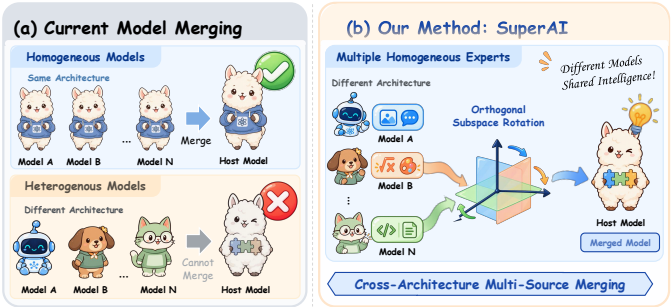

Figure 1: **Conventional Model Merging Versus SuperAI.** Conventional parameter merging requires compatible parameter spaces, whereas SuperAI integrates heterogeneous source capabilities into a merged model through orthogonal subspace alignment and lightweight residual fusion.

We hypothesize that heterogeneous models can learn structurally corresponding task representations. Using shared anchor samples, we observe that these representations can be approximately aligned through orthogonal mappings in low-dimensional functional subspaces while preserving each source subspace's geometry. However, alignment alone does not guarantee task improvements, as transferred information must also be compatible with downstream host computation.

Building on this insight, we propose SuperAI, a cross-architecture multi-model capability fusion framework that integrates complementary capabilities from multiple source models into a host model through orthogonal subspace alignment. Specifically, we first extract task-relevant representations from the source models and the host model using shared anchor samples, and apply principal component analysis (PCA) to construct low-dimensional functional subspaces with the same dimensionality. We then use orthogonal Procrustes mappings to align their coordinate systems. Finally, we train a lightweight residual multilayer perceptron (MLP) with joint supervision on task outputs and source-model increments to predict input-dependent residual increments, which are injected into the host model. This joint objective uses source-model information to guide residual learning while adapting the increments to the host model's task objectives, connecting representation alignment to task-level capability fusion. Our main contributions are as follows:

- We propose SuperAI, a cross-architecture multi-model capability fusion framework combining orthogonal subspace alignment with residual injection, providing a unified interface for reusing and composing complementary capabilities from heterogeneous source models.
- We identify approximately orthogonal cross-model correspondences between task-relevant representations. Ablations support the effectiveness of orthogonal alignment, nonlinear residual mapping, and joint supervision on task outputs and source-model increments.
- Across three host models and 15 capability domains, the proposed SuperAI outperforms the state-of-the-art (SOTA) fusion methods on average and better preserves general capabilities as the number of source models increases.

2 Related Work

Model Merging and Capability Fusion. Parameter-space merging integrates models [Wortsman et al., 2022, Ilharco et al., 2022] with compatible architectures through weight averaging or task-vector addition, as in model soups and task arithmetic. Subsequent methods reduce task interference through sign resolution, filtering, conditioning, or loss balancing [Yadav et al., 2023, Qiu et al., 2026, Yoshida et al., 2026, Im et al., 2026], and improve efficiency through low-rank, orthogonal, or shared parameter structures [Crisostomi et al., 2026, Lee et al., 2026, Yang et al., 2026, Dhasade et al., 2026]. Beyond direct parameter merging, knowledge fusion transfers information through output distributions or token alignment [Wan et al., 2024, 2025], while neuron and component matching reduces representational mismatch before fusion [Ainsworth et al., 2022, Singh and Jaggi, 2020, Imfeld et al., 2024]. For cross-architecture fusion, Transport and Merge aligns activations and layers to guide weight fusion [Cui et al., 2026], whereas THESEUS [Rinaldi et al., 2026] combines functional matching with orthogonal Procrustes alignment to transfer task vectors across model widths.

Cross-Model Representation Alignment. Representation comparison methods assess statistical similarity using SVCCA and CKA [Raghu et al., 2017, Kornblith et al., 2019], or functional compatibility through model stitching [Lenc and Vedaldi, 2015, Bansal et al., 2021, Athanasiadis et al., 2025]. Relative representations and latent-space

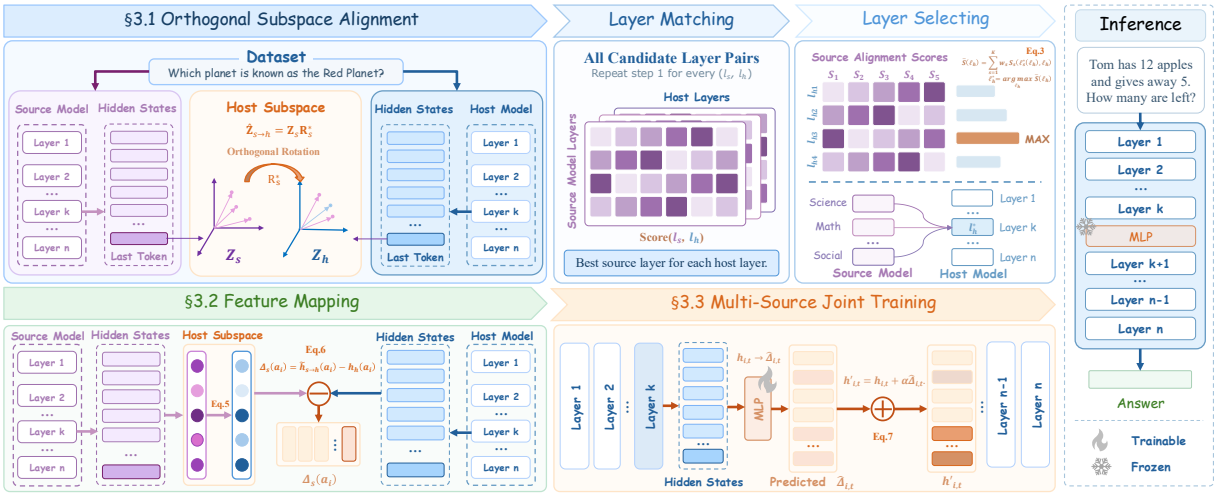

Figure 2: **Overview of Proposed SuperAI Framework.** Shared prompts are used to align source and host representations through PCA and orthogonal rotation. All layer pairs are evaluated to select one shared host layer and the corresponding source layers. The aligned source features define capability increments Δ_e, which supervise a shared residual MLP together with task and domain objectives. At inference, only the frozen host model and the trained MLP are required.

mappings enable explicit representation reuse [Moschella et al., 2022, Maiorca et al., 2023, Moayeri et al., 2023, Fumero et al., 2024], while the Platonic Representation Hypothesis suggests shared structure across independently trained models [Huh et al., 2024]. Recent work transfers interventions, probes, and adapters between language models [Oozeer et al., 2025, Chen et al., 2026], and extends alignment to multiple models sharing a reference space [Achara et al., 2026]. However, representational correspondence alone does not guarantee effective capability fusion, as aligned information from multiple source models must also be adapted to the host model's downstream computation and task objectives to improve its performance.

3 Methodology

3.1 Orthogonal Subspace Alignment

Let f_h be the host model and $\{f_s\}_{s=1}^{K}$ be K source models. Their hidden dimensions are d_h and d_s, respectively. For source model s, we use shared prompts $\mathcal{A}_s = \{a_i\}_{i=1}^{n_s}$. As illustrated in Figure 2, SuperAI first extracts paired source and host representations from shared prompts and aligns them in low-dimensional subspaces. Let $h_m^{(\ell_m)}(a_i)$ be the hidden state of the last non-padding token at layer ℓ_m of model $m \in \{s, h\}$. We stack the representations of the same prompts into $H_s^{(\ell_s)} \in \mathbb{R}^{n_s \times d_s}$ and $H_h^{(\ell_h)} \in \mathbb{R}^{n_s \times d_h}$. The representations are paired by prompt, so the models may use different tokenizers.

For a candidate layer pair, let μ_m be the mean hidden state of model m. We center and normalize each representation. We then project the representations into separate r-dimensional PCA subspaces:

$$u_m(a_i) = \frac{h_m^{(\ell_m)}(a_i) - \mu_m}{\max\left(\|h_m^{(\ell_m)}(a_i) - \mu_m\|_2, \epsilon\right)}, \qquad Z_m = (U_m - \mathbf{1}c_m^\top)P_m, \quad m \in \{s, h\}. \tag{1}$$

Here, $\epsilon > 0$ prevents division by zero. The rows of U_m are $u_m(a_i)$. The PCA mean is c_m, and $P_m \in \mathbb{R}^{d_m \times r}$ contains the principal components, with $P_m^\top P_m = I_r$. The vector $\mathbf{1}$ contains ones. We omit the candidate layer indices below for clarity.

Layer Matching. The coordinates Z_s and Z_h have the same dimension, but their axes may differ. We align them with orthogonal Procrustes:

$$R_s^* = \underset{R^\top R = I_r}{\arg\min} \|Z_s R - Z_h\|_F^2, \qquad Z_s^\top Z_h = L\Sigma V^\top, \quad R_s^* = LV^\top. \tag{2}$$

The aligned source coordinates are $\widehat{Z}_{s\to h} = Z_s R_s^*$. The orthogonal map preserves pairwise distances within the source PCA subspace.

Layer Selecting. Let $S_s(\ell_s, \ell_h)$ be the alignment score of source layer ℓ_s and host layer ℓ_h on held-out prompts. For each candidate host layer, we first select the best source layer. We then combine the scores across source models and

select one shared host layer:

$$\ell_s^*(\ell_h) = \arg\max_{\ell_s} S_s(\ell_s, \ell_h), \qquad \ell_h^* = \arg\max_{\ell_h} \sum_{s=1}^{K} w_s S_s(\ell_s^*(\ell_h), \ell_h)\,, \quad \sum_{s=1}^{K} w_s = 1. \tag{3}$$

Here, w_s is the weight of source model s. With equal weights, $w_s = 1/K$. Source models may use different source layers and orthogonal maps, but all are mapped to the same host layer ℓ_h^*. In the next section, we use this layer to construct target feature increments.

We fit the normalization statistics, PCA subspaces, and orthogonal mapping separately for each source model using its domain-specific training prompts. The mapping establishes cross-model correspondences in low-dimensional functional subspaces while preserving the geometry of the source model representations. We next examine these properties qualitatively and quantitatively.

Geometric Correspondence Across Models. To visualize the effect of orthogonal alignment, we compare paired source model and host representations before and after applying the learned mapping. As shown in Figure 3(a), the representations initially differ substantially in position, orientation, and density. After alignment, their point clouds and distribution contours closely overlap, as shown in Figure 3(b). This result suggests that heterogeneous models encode similar task-relevant information using different coordinate bases while retaining alignable underlying structures, supporting the effectiveness of orthogonal subspace alignment in establishing cross-model correspondences.

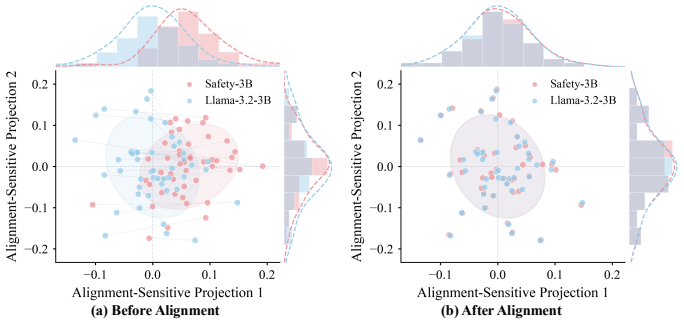

Figure 3: **Cross-Model Representations Before and After Orthogonal Alignment.** Source and host representations differ substantially before alignment and become closely aligned afterward.

Cross-Model Alignment Evaluation. To evaluate whether this geometric correspondence generalizes across heterogeneous source–host pairs, we examine the selected layer pairs between 15 source models and two host models, Llama-3.2-3B and Qwen2.5-3B. Match measures overall alignment quality, CKA measures global geometric similarity, and Retrieval@1 measures instance-level correspondence between representations of the same input. As shown in Figure 4, the average Match, CKA, and Retrieval@1 scores are 0.90, 0.98, and 0.97 for Llama-3.2-3B, and 0.82, 0.93, and 0.87 for Qwen2.5-3B. Despite variation across source–host pairs, the scores indicate that orthogonal alignment preserves global geometry and input-level correspondences across heterogeneous models, supporting its use for subsequent capability increment construction.

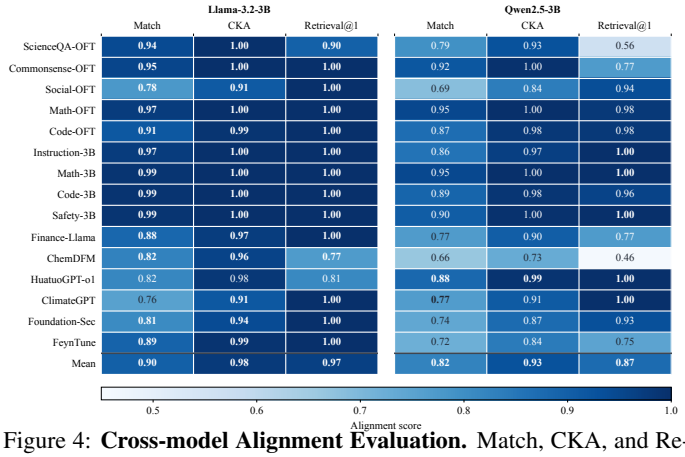

Figure 4: **Cross-model Alignment Evaluation.** Match, CKA, and Retrieval@1 scores across 15 source models with Llama and Qwen as host models.

3.2 Feature Mapping

The previous section selects a source layer ℓ_s^* and an orthogonal map R_s^* for each source model s. It also selects one shared host layer ℓ_h^*. For a prompt a_i, we rotate the source PCA coordinates into the host subspace and reconstruct them in the host hidden space:

$$\widehat{z}_{s\to h}(a_i) = z_s(a_i) R_s^*, \qquad b_{s\to h}(a_i) = \widehat{z}_{s\to h}(a_i) P_h^\top + c_h. \tag{4}$$

Here, P_h and c_h are the host PCA component matrix and mean fitted on the prompts paired with source model s. As in the previous section, we omit their source index for clarity.

The alignment uses normalized representations. We therefore use $b_{s\to h}(a_i)$ to set the direction and the host hidden state to set the magnitude:

$$\widetilde{h}_{s\to h}(a_i) = \mu_h + \|h_h(a_i) - \mu_h\|_2 \frac{b_{s\to h}(a_i)}{\max(\|b_{s\to h}(a_i)\|_2, \epsilon)}. \tag{5}$$

Here, $h_h(a_i)$ is the original last non-padding token hidden state at host layer ℓ_h^*, and μ_h is the mean hidden state at that layer for the prompts paired with source model s. We define the target feature increment as the difference between the mapped representation and the original host representation:

$$\Delta_s(a_i) = \widetilde{h}_{s\to h}(a_i) - h_h(a_i), \Delta_s(a_i) \in \mathbb{R}^{d_h}. \tag{6}$$

The increment $\Delta_s(a_i)$ lies in the host hidden space. It provides a representation-level target for the shared residual MLP in the next section. At inference time, the MLP predicts increments directly from host hidden states. Source models and explicit feature maps are not needed.

3.3 Multi-Source Joint Training

Section 3.2 constructs a target feature increment $\Delta_s(a_i)$ in the host hidden space for each training prompt from source model s. We mix training samples from all source models and train one shared residual MLP at the host layer selected in Section 3.1. The host model remains frozen. For hidden state $h_{i,t}$ at token position t, the MLP predicts an increment and adds it to the hidden state:

$$q_{i,t} = \mathrm{SiLU}(W_{\mathrm{down}}\,\mathrm{LN}(h_{i,t}))\,, \qquad \widehat{\Delta}_{i,t} = W_{\mathrm{up}} q_{i,t}, \qquad h'_{i,t} = h_{i,t} + \alpha\widehat{\Delta}_{i,t}. \tag{7}$$

Here, $W_{\mathrm{down}} \in \mathbb{R}^{b\times d_h}$ and $W_{\mathrm{up}} \in \mathbb{R}^{d_h\times b}$, where b is the bottleneck width. The residual scaling factor α controls the strength of the predicted capability increment injected into the host hidden state. We initialize W_{up} to zero, so the residual update is zero at the start of training. The MLP updates hidden states at all token positions. Increment supervision is applied only at p_i, the last non-padding position of prompt a_i. This is the position used to extract features in the preceding sections.

Let s_i denote the source model associated with prompt a_i, and let $\mathcal{A}_s^{\mathrm{tr}}$ be the training prompts for source model s. To account for differences in target magnitude across source models, we define the normalization factor and increment loss:

$$\rho_s = \sqrt{\frac{1}{|\mathcal{A}_s^{\mathrm{tr}}|d_h}\sum_{a_i\in\mathcal{A}_s^{\mathrm{tr}}}\|\Delta_s(a_i)\|_2^2}, \qquad \mathcal{L}_{\mathrm{inc}} = \frac{1}{Nd_h}\sum_{i=1}^{N}\left\|\frac{\widehat{\Delta}_{i,p_i} - \Delta_{s_i}(a_i)}{\max(\rho_{s_i},\epsilon)}\right\|_2^2, \tag{8}$$

where N is the total number of training prompts and $\epsilon > 0$ prevents division by zero.

We also compute cross-entropy against the ground-truth answers. Let $\mathcal{T}_i$ be the answer token positions for prompt a_i. The task loss and the joint objective are

$$\mathcal{L}_{\mathrm{task}} = -\frac{1}{N}\sum_{i=1}^{N}\frac{1}{|\mathcal{T}_i|}\sum_{t\in\mathcal{T}_i}\log p_\theta(y_{i,t} \mid a_i, y_{i,<t}), \qquad \mathcal{L} = \lambda_{\mathrm{task}}\mathcal{L}_{\mathrm{task}} + \lambda_{\mathrm{inc}}\mathcal{L}_{\mathrm{inc}}. \tag{9}$$

Here, $\mathcal{L}_{\mathrm{task}}$ is the cross-entropy loss on the ground-truth answer tokens, and $\mathcal{L}_{\mathrm{inc}}$ is the source-increment supervision loss. The coefficients λ_{task} and λ_{inc} control the contributions of the two losses. Here, p_θ is the output distribution of the host model with the residual MLP. Only the MLP is updated during training. The source identity s_i is used to select the target increment and its normalization factor. At inference time, only the host model and the shared MLP are needed. Source models, source identities, and explicit feature maps are not required.

4 Experiments

4.1 Experimental Setup

Source Models and Datasets. We select three model families as host models, covering multiple parameter scales within each family: the Llama includes Llama-3.2-1B, Llama-3.2-3B, and Llama-3.1-8B Grattafiori et al. [2024]; the Pythia includes Pythia-1B, Pythia-2.8B, Pythia-6.9B, and Pythia-12B Biderman et al. [2023]; and the Qwen2.5 includes Qwen2.5-1.5B, Qwen2.5-3B, Qwen2.5-7B, and Qwen2.5-14B Qwen et al. [2025]. We construct a pool of 15 source models spanning science, commonsense and social reasoning, math, code generation, multilingual reasoning, safety, law, finance, astronomy, medicine, climate science, cyber security, chemistry, and physics. The pool includes four OFT sources Yang et al. [2026], Qiu et al. [2023], Qwen2.5-Coder-7B-Instruct Hui et al. [2024], two MergeBench source models He et al. [2026], and eight domain-specialized source models. These source models vary in model family, parameter scale, and training procedure, forming a heterogeneous testbed for cross-architecture capability transfer and multi-source fusion. The complete source benchmark mapping is provided in the appendix.

Table 1: **Quantitative Comparison of Multi-source Capability Fusion Across Different Host Models.** We report the original host performance and multi-source fusion results across 15 capability domains. The best and second-best results among the fusion methods are shown in **bold** and underlined, respectively. Green and red arrows in the Avg. column denote gains and decreases relative to the corresponding original host.

Method	Math	Safety	Law	Phys.	Sci.	C.Sense	Social	Code	Multi.	Fin./Bus.	Astro.	Med.	Climate	Cyber	Chem.	Avg.
Domain	*67.7*	*72.0*	*46.5*	*80.9*	*76.5*	*81.6*	*75.2*	*81.5*	*69.4*	*15.0*	*74.3*	*58.4*	*71.4*	*36.2*	*49.0*	*63.7*
Host model: Llama-3.2-3B																
Original	16.7	56.8	38.0	68.3	67.9	66.2	58.7	39.7	56.1	3.7	60.5	50.9	42.3	31.4	44.0	46.7
THESEUS	24.1	57.3	40.3	62.9	63.5	60.0	53.8	**42.1**	44.0	12.7	54.2	40.6	38.4	34.7	32.1	44.0 (↓ 2.7)
T&M	17.6	59.5	**41.2**	61.4	64.8	61.7	54.2	41.7	44.3	10.9	49.7	40.3	35.8	31.2	43.4	43.8 (↓ 2.9)
SuperAI	**40.7**	**70.9**	40.8	**89.1**	**69.8**	**68.2**	**63.3**	28.3	**55.0**	**14.7**	**57.9**	**48.2**	**72.5**	**48.9**	**44.0**	**54.2** (↑ 7.5)
Host model: Qwen2.5-3B																
Original	26.3	61.3	39.4	85.1	73.0	77.2	69.0	55.6	71.9	7.0	71.2	53.6	68.1	39.2	45.0	56.2
THESEUS	22.7	55.6	40.5	71.8	56.2	54.0	59.1	41.5	57.8	**13.4**	46.7	37.2	31.9	32.6	29.0	43.3 (↓ 12.9)
T&M	31.2	61.6	43.8	65.9	63.7	62.8	65.9	**53.8**	64.9	12.7	55.0	43.4	40.8	34.1	35.6	49.0 (↓ 7.2)
SuperAI	**45.7**	**76.6**	**53.5**	**94.9**	**82.5**	**79.5**	**69.5**	44.2	**67.6**	9.3	**72.4**	**53.8**	**73.8**	**53.8**	**43.0**	**61.3** (↑ 5.1)
Host model: Pythia-2.8B																
Original	3.7	42.5	23.9	24.2	24.0	20.7	35.7	11.1	23.6	0.7	28.3	26.2	27.9	23.3	27.0	22.9
THESEUS	1.8	13.4	4.3	54.9	20.1	31.9	25.5	3.8	6.2	6.9	18.2	7.9	38.9	32.7	20.5	19.1 (↓ 3.8)
T&M	3.1	20.1	6.0	47.5	27.1	**40.6**	32.2	**6.0**	9.4	**10.3**	21.8	11.6	46.0	29.2	**23.1**	22.3 (↓ 0.6)
SuperAI	**8.0**	**69.9**	**31.0**	**89.2**	**27.6**	18.7	**37.5**	1.3	**29.3**	1.0	**25.0**	**27.7**	**72.5**	**33.0**	21.0	**32.8** (↑ 9.9)

Evaluation Metrics. We use one predefined primary metric for each domain. Most domains are evaluated with accuracy or task-specific variants of accuracy. The main exceptions are code generation and cybersecurity: MBPP Austin et al. [2021] is evaluated with Pass@1 Chen et al. [2021], requiring the generated program to pass the test suite, while SEVENLLM Ji et al. [2024b] uses token-level F1. All metrics are higher-is-better. For multi-source fusion, we additionally report the unweighted macro average across 15 domains and MMLU Hendrycks et al. [2021] to assess general-capability retention as more source models are merged. More details are in the appendix.

Implementation Details. We use Llama-3.2-3B Grattafiori et al. [2024], Qwen2.5-3B Qwen et al. [2025], and Pythia-2.8B Biderman et al. [2023] as host models. Both the PCA subspace dimension and the bottleneck dimension of the residual MLP are set to 128. We train the models for two epochs using AdamW with a batch size of 4, a learning rate of 2×10^{-4}, and a weight decay of 0.01. The feature increment loss weight is set to 0.5. The residual scaling factor α is set to 1.0, 0.75, and 0.25 for the three host models, respectively.

4.2 Heterogeneous Multi-Source Merging

Quantitative Comparison Across Host Models. To evaluate SuperAI across host architectures, we jointly fuse all 15 domain source models into each of three hosts, Llama-3.2-3B, Qwen2.5-3B, and Pythia-2.8B, and compare the resulting models with THESEUS Rinaldi et al. [2026] and Transport and Merge Cui et al. [2026]. The results in Table 1 show that SuperAI achieves the highest average score for all three host models. Specifically, SuperAI improves the average scores of Llama-3.2-3B, Qwen2.5-3B, and Pythia-2.8B from 46.7, 56.2, and 22.9 to 54.2, 61.3, and 32.8, corresponding to gains of 7.5, 5.1, and 9.9 points, respectively. Compared with the strongest fusion baseline for each host, SuperAI improves the average score by 10.2, 12.3, and 10.5 points, respectively. The largest domain-level gains vary across hosts. Llama improves most in climate and math, by 30.2 and 24.0 points, respectively. Qwen obtains substantial gains in math, safety, and cyber, improving by 19.4, 15.3, and 14.6 points. Although Pythia achieves the largest average gain, the post-fusion ranking remains Qwen, Llama, and Pythia, consistent with their original performance. Therefore, orthogonal subspace alignment provides an effective cross-model interface for heterogeneous capability transfer, while capability absorption and final performance remain dependent on the host model.

General Capability Retention. To evaluate how well SuperAI preserves the host model's existing general capabilities during multi-source capability fusion, we conduct an MMLU retention analysis as the number of integrated source models increases from 1 to 15. Figure 5 show that MMLU retention decreases for all three methods as more source models are integrated, but SuperAI exhibits a substantially slower decline and an increasing advantage over the baselines. Specifically, the retention rates of THESEUS and Transport and Merge decrease from 96.8% and 97.9% to 61.7% and 70.9%, respectively, whereas SuperAI retains 88.6% of the original MMLU performance after integrating 15 source models. The widening performance gap indicates that interference accumulates more slowly under SuperAI as additional source models are introduced. Therefore, these results suggest that orthogonal subspace alignment and residual injection mitigate interference from accumulating source capabilities, enabling the host model to absorb domain-specific capabilities while better preserving its existing general capabilities.

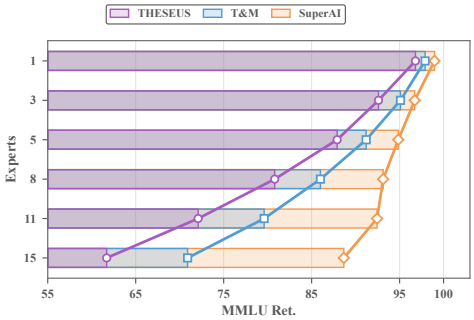

Figure 5: **General capability retention.** MMLU retention of THESEUS, T&M, and SuperAI with 1–15 source models.

4.3 Ablation Study

Effect of Training Objectives. To evaluate the roles of task-output and source-model increment supervision, we compare $\lambda_{\text{inc}} = 0$, $\lambda_{\text{task}} = 0$, and the joint objective across three host models, using cosine-similarity gain to measure increment matching. The results in Table 2 show that joint supervision achieves the best downstream performance, whereas either signal alone is insufficient. Specifically, $\lambda_{\text{inc}} = 0$ improves Llama, Qwen, and Pythia by 3.9, 1.3, and 4.2 points but yields negligible increment-matching gains. Conversely, $\lambda_{\text{task}} = 0$ achieves the highest matching gains but provides inconsistent downstream improvements. In contrast, the joint objective improves the three hosts by 7.5, 5.1, and 9.9 points. Therefore, joint supervision effectively balances source-model guidance with adaptation to host task objectives, thereby improving the effectiveness of cross-architecture capability transfer.

Table 2: **Effect of Training Objectives.** We report average scores across 15 domains and cosine-similarity gains for three host models. For each host and metric, the best and second-best results are bolded and underlined, respectively.

Objective	Llama-3.2-3B	Qwen2.5-3B	Pythia-2.8B
15-domain average score			
Original	46.7	56.2	22.9
$\lambda_{\text{inc}} = 0$	<u>50.6</u> (↑ 3.9)	<u>57.5</u> (↑ 1.3)	<u>27.1</u> (↑ 4.2)
$\lambda_{\text{task}} = 0$	44.8 (↓ 1.9)	56.3 (↑ 0.1)	24.0 (↑ 1.1)
Joint	**54.2** (↑ 7.5)	**61.3** (↑ 5.1)	**32.8** (↑ 9.9)
Cosine-similarity gain			
$\lambda_{\text{inc}} = 0$	0.0003	−0.0004	−0.0004
$\lambda_{\text{task}} = 0$	**0.0212**	**0.1003**	**0.2560**
Joint	<u>0.0128</u>	<u>0.0840</u>	<u>0.0913</u>

Effect of Orthogonal Subspace Alignment. To evaluate the role of orthogonal subspace alignment, we compare No Alignment, PCA Only, PCA with Linear Mapping, and PCA with Orthogonal Mapping under identical training settings. The results in Table 3 show that PCA with Orthogonal Mapping consistently achieves the best performance across all three host models. Specifically, it improves Llama, Qwen, and Pythia by 7.5, 5.1, and 9.9 points, respectively, whereas the other configurations produce smaller or less consistent gains. It outperforms linear mapping by 1.0, 3.8, and 5.6 points on the three hosts, respectively. Therefore, orthogonal alignment establishes a reliable cross-model interface by reconciling coordinate differences while preserving the geometry of PCA-projected source representations.

Table 3: **Effect of Orthogonal Subspace Alignment.** We report 15-domain average scores for three host models, with changes relative to each original host in parentheses.

Mapping Method	Llama-3.2-3B	Qwen2.5-3B	Pythia-2.8B
Original	46.7	56.2	22.9
No Alignment	47.6 (↑ 0.9)	56.1 (↓ 0.1)	21.0 (↓ 1.9)
PCA Only	47.8 (↑ 1.1)	56.9 (↑ 0.7)	28.6 (↑ 5.7)
PCA + Linear	53.2 (↑ 6.5)	57.5 (↑ 1.3)	27.2 (↑ 4.3)
PCA + Orthogonal	**54.2** (↑ 7.5)	**61.3** (↑ 5.1)	**32.8** (↑ 9.9)

Effect of Residual Mapper Architecture. To evaluate whether nonlinear residual mapping is necessary for adapting aligned source representations to host computation, we compare a linear mapper with the residual MLP used in SuperAI across three host models. The results in Table 4 show that the linear mapper produces inconsistent performance changes, whereas the residual MLP consistently achieves larger gains across three models. The linear mapper improves Llama and Pythia by 3.9 and 7.2 points, respectively, but reduces Qwen performance by 0.2 points. In contrast, the residual MLP improves Llama, Qwen, and Pythia by 7.5, 5.1, and 9.9 points, outperforming the linear mapper by 3.6, 5.3, and 2.7 points, respectively. Therefore, the effectiveness of linear mapping for capability transfer varies across host architectures, whereas nonlinear residual mapping learns host-compatible increments and enables consistent capability transfer across architectures.

Table 4: **Effect of Residual Mapper Architecture.** We report average scores across 15 domains for three host models. Parentheses indicate score changes relative to the original host.

Mapper	Llama-3.2-3B	Qwen2.5-3B	Pythia-2.8B
Original	46.7	56.2	22.9
Linear	50.6 (↑ 3.9)	56.0 (↓ 0.2)	30.1 (↑ 7.2)
MLP	**54.2** (↑ 7.5)	**61.3** (↑ 5.1)	**32.8** (↑ 9.9)

Sensitivity to Residual Scaling. To assess how residual scaling affects capability transfer, we vary the residual scaling factor α from 0 to 2.0 across three host models. Figure 6 shows that performance varies non-monotonically with α, with the optimal setting differing across hosts. Specifically, Llama-3.2-3B, Qwen2.5-3B, and Pythia-2.8B achieve their best performance at $\alpha = 1.0$, 0.75, and 0.25, respectively. These different optima suggest that the preferred injection strength depends on the host model. Increasing α to 1.5 or 2.0 does not yield consistent additional gains in average domain performance. Pythia exhibits a more pronounced decline beyond its optimum than Llama, indicating greater sensitivity to residual scaling. Over-

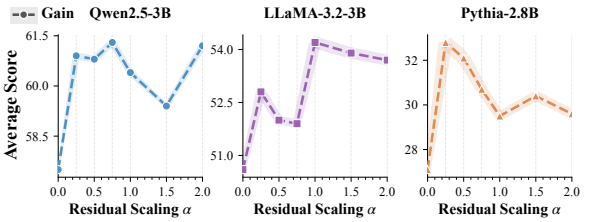

Figure 6: **Sensitivity to Residual Scaling.** Average scores across 15 domains for three host models as the residual scaling factor α varies from 0 to 2.0.

all, these results support selecting an appropriate scaling factor for each host, as stronger residual injection does not necessarily improve capability transfer.

Effect of Layer-Pair Alignment Quality. To evaluate whether post-alignment Match scores can guide the selection of cross-model transfer layers, we compare source and host layer pairs with weak, medium, and optimal match scores while keeping the training data, residual MLP, and optimization settings fixed. The results in Table 5 show that transfer performance improves with layer-pair alignment quality. Specifically, weakly matched layer pairs improve Llama, Qwen, and Pythia by 4.4, 2.8, and 5.5 points, respectively. The gains increase to 6.8, 4.2, and 8.7 points for the medium-matched pairs, and further to 7.5, 5.1, and 9.9 points for the best-matched pairs. Meanwhile, the post-alignment Match scores of all three host models consistently increase from weakly matched to best-matched pairs. Higher Match scores indicate more consistent sample relationships and functional geometry between source and host representations, making the aligned source-model increments more compatible with downstream host computation. These results show a consistent positive relationship between alignment quality and capability transfer performance, supporting the use of orthogonal subspace alignment to identify effective representation injection locations for cross-architecture capability transfer.

Table 5: **Effect of Layer-pair Alignment Quality.** We report 15-domain averages and post-alignment Match scores; parentheses show changes from the baseline. Best and second-best results are bolded and underlined.

Layer Pair	Llama-3.2-3B	Qwen2.5-3B	Pythia-2.8B
15-domain average score			
Original	46.7	56.2	22.9
Weakly Matched	51.1 (↑ 4.4)	59.0 (↑ 2.8)	28.4 (↑ 5.5)
Medium Matched	<u>53.5</u> (↑ 6.8)	<u>60.4</u> (↑ 4.2)	<u>31.6</u> (↑ 8.7)
Best Matched	**54.2** (↑ 7.5)	**61.3** (↑ 5.1)	**32.8** (↑ 9.9)
Post-alignment Match score			
Weakly Matched	0.907	0.912	0.964
Medium Matched	<u>0.926</u>	<u>0.922</u>	<u>0.967</u>
Best Matched	**0.931**	**0.925**	**0.976**

4.4 Deeper Analysis

Maximum Stable Fusion Capacity. To evaluate the impact of host model scale on multi-source fusion capability, we conduct scaling experiments across the Qwen, Llama, and Pythia model families, measuring the maximum stable fusion capacity k^* and its corresponding average performance gain. The results in Figure 7 show that fusion capacity generally exhibits a non-decreasing trend as model scale increases. Specifically, k^* increases from 8 to 12 for Qwen, from 7 to 12 for Llama, and from 9 to 12 for Pythia at the 7–8B scale. Pythia reaches a capacity of 14 at the 12–14B scale. At approximately 3B scale, the average gains for Qwen, Llama, and Pythia are 5.1 ± 0.57, 7.5 ± 0.84, and 9.9 ± 1.31 points, respectively. At the 7–8B scale, all three families can stably accommodate 12 source models, with the corresponding gains decreasing to 4.8 ± 0.61, 5.2 ± 0.69, and 8.7 ± 0.78 points. All reported configurations achieve positive gains across all three random seeds, demonstrating solid stability in the fusion outcomes. Therefore, within the evaluated model families, scaling up the host model increases the number of stably mergeable source models, yet performance gains do not grow monotonically with size, being further shaped by family-specific characteristics and the host model's remaining headroom for improvement.

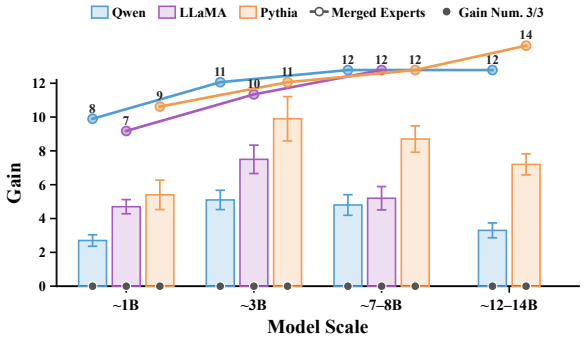

Figure 7: **Effect of Host Model Scale on Multi-source Fusion Capacity.** Bars show mean 15-domain gains at the maximum stable fusion capacity k^*, with error bars indicating standard deviations across three seeds. Lines show k^*; 3/3 denotes positive gains in all three seeds. Missing entries indicate unavailable model scales.

Domain-Wise Capability Absorption. To evaluate how host model scale affects domain-specific capability absorption, we conduct a scaling analysis within the Qwen2.5 family across four representative domains: Math, Safety, Law, and Physics. The results in Table 6 show that SuperAI achieves positive gains across most model scales and domains. The average gain across the four domains increases from 4.6 points for Qwen2.5-1.5B to 14.6 points for Qwen2.5-3B, while remaining substantial at 14.2 and 12.1 points for the 7B and 14B models, respectively. Math shows a clear scaling effect,

Table 6: **Domain-wise Capability Absorption Across Qwen2.5 Scales.** Each cell reports the post-fusion score and its change from the corresponding base model. Avg. denotes the four-domain average; best and second-best results are bolded and underlined.

Host Model	Math	Safety	Law	Physics	Avg.
Qwen2.5-1.5B	14.3 (↓ 12.3)	72.5 (↑ 8.0)	49.3 (↑ 8.5)	94.0 (↑ 14.1)	57.5 (↑ 4.6)
Qwen2.5-3B	45.7 (↑ 19.4)	76.6 (↑ 15.3)	53.5 (↑ 14.1)	94.9 (↑ 9.8)	67.7 (↑ 14.6)
Qwen2.5-7B	<u>57.3</u> (↑ 40.7)	<u>79.3</u> (↑ 3.3)	<u>56.3</u> (↑ 8.5)	<u>95.7</u> (↑ 4.3)	<u>72.2</u> (↑ 14.2)
Qwen2.5-14B	**57.7** (↑ 44.0)	**80.2** (↑ 8.9)	**59.2** (↓ 5.6)	**97.9** (↑ 1.1)	**73.8** (↑ 12.1)

with gains of 19.4, 40.7, and 44.0 points for the 3B, 7B, and 14B models. Safety, law, and physics also achieve substantial improvements at different model scales, suggesting that the optimal scale for capability absorption varies across domains. These contrasting trends suggest that model size alone does not determine the magnitude of domain-specific transfer gains. Therefore, increasing host model scale can enhance the absorption of heterogeneous source capabilities, but the benefits exhibit diminishing returns, potentially because stronger baseline capabilities leave less room for complementary capability gains.

Generalization Across Datasets and Host Configurations. To evaluate the generalization of SuperAI, we conduct experiments along three dimensions: cross-architecture, cross-dataset, and cross-scale generalization. The results in Figure 8(a) show that Llama-3.2-3B, Qwen2.5-3B, and Pythia-2.8B achieve average gains of 5.7, 2.0, and 1.5 points, respectively, on unseen datasets across Climate, Science, Common Sense, and Social. Specifically, Figure 8(b) shows that the Qwen2.5-1.5B, 3B, 7B, and 14B models achieve average gains of 4.4, 2.1, 5.0, and 5.8 points across the four domains, respectively, with the 7B and 14B models exhibiting stronger overall capability absorption. In the Science domain, the gain increases from 1.1 to 9.2 points as the model scale grows, indicating that larger host models can more effectively absorb source capabilities in certain domains. Therefore, SuperAI enables consistent positive transfer across model architectures, unseen data distributions, and parameter scales, supporting orthogonal subspace alignment as a generalizable interface for cross-model capability fusion. However, domain-specific returns remain non-linear and depend on the host model's existing capabilities and remaining headroom for improvement.

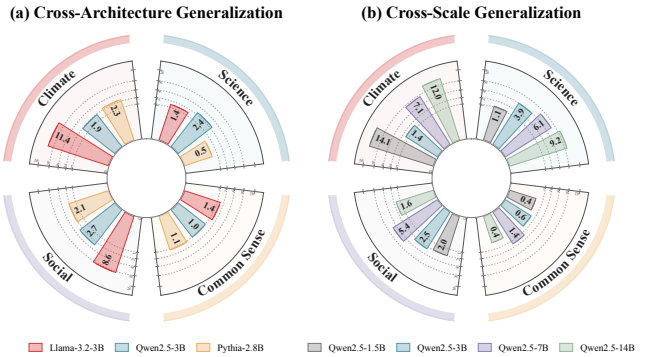

Figure 8: **Generalization Across Host Architectures and Scales.** Evaluation of SuperAI in the Climate, Science, Common Sense, and Social domains: (a) generalization to unseen datasets across host architectures; (b) generalization performance across host model scales.

5 Conclusion

In this work, we introduced SuperAI, a framework for integrating multiple heterogeneous source models into a single host model through orthogonal subspace alignment and lightweight residual fusion. Our analysis reveals that task-relevant representations across different model architectures exhibit approximately orthogonal correspondences in low-dimensional functional subspaces, providing a shared geometric interface for cross-model capability transfer. Experiments across 15 capability domains and three host model families demonstrate that SuperAI consistently outperforms existing cross-architecture merging methods while better preserving the host model's general capabilities. Further results across unseen datasets, model architectures, and parameter scales confirm the generalizability of this approach, while the scaling analysis shows that capability absorption is jointly constrained by model capacity and representational characteristics. Overall, our findings suggest that aligning and composing functional representations offers a promising alternative to parameter-level merging for modular and efficient multi-source integration.

References

Raia Abu Ahmad, Aida Usmanova, and Georg Rehm. The ClimateCheck dataset: Mapping social media claims about climate change to corresponding scholarly articles. In Tirthankar Ghosal, Philipp Mayr, Amanpreet Singh, Aakanksha Naik, Georg Rehm, Dayne Freitag, Dan Li, Sonja Schimmler, and Anita De Waard, editors, *Proceedings of the Fifth Workshop on Scholarly Document Processing (SDP 2025)*, pages 42–56, Vienna, Austria, July 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.sdp-1.5. URL `https://aclanthology.org/2025.sdp-1.5/`.

Akshit Achara, Tatiana Gaintseva, Mateo Mahaut, Pritish Chakraborty, Viktor Stenby Johansson, Melih Barsbey, Emanuele Rodolà, and Donato Crisostomi. Multi-way representation alignment. *arXiv preprint arXiv:2602.06205*, 2026.

Samuel K Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*, 2022.

Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. Evolutionary optimization of model merging recipes. *Nature Machine Intelligence*, 7(2):195–204, 2025.

Ioannis Athanasiadis, Anmar Karmush, and Michael Felsberg. Grounding functional similarity by invariance-aware model stitching. *arXiv preprint arXiv:2505.20142*, 2025.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Yamini Bansal, Preetum Nakkiran, and Boaz Barak. Revisiting model stitching to compare neural representations. *Advances in neural information processing systems*, 34:225–236, 2021.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling, 2023. URL https://arxiv.org/abs/2304.01373.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language, 2019. URL https://arxiv.org/abs/1911.11641.

Alan Chen, Jack Merullo, Alessandro Stolfo, and Ellie Pavlick. Transferring linear features across language models with model stitching. *Advances in Neural Information Processing Systems*, 38:48531–48563, 2026.

Junying Chen, Zhenyang Cai, Ke Ji, Xidong Wang, Wanlong Liu, Rongsheng Wang, Jianye Hou, and Benyou Wang. Huatuogpt-o1, towards medical complex reasoning with llms, 2024. URL https://arxiv.org/abs/2412.18925.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. Finqa: A dataset of numerical reasoning over financial data, 2022. URL https://arxiv.org/abs/2109.00122.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Pierre Colombo, Telmo Pessoa Pires, Malik Boudiaf, Dominic Culver, Rui Melo, Caio Corro, Andre FT Martins, Fabrizio Esposito, Vera Lúcia Raposo, Sofia Morgado, et al. Saullm-7b: A pioneering large language model for law. *arXiv preprint arXiv:2403.03883*, 2024.

Donato Crisostomi, Alessandro Zirilli, Antonio Andrea Gargiulo, Maria Sofia Bucarelli, Simone Scardapane, Fabrizio Silvestri, Iacopo Masi, and Emanuele Rodolà. Mass: Moerging through adaptive subspace selection. In *International Conference on Learning Representations*, volume 2026, pages 19982–20008, 2026.

Chenhang Cui, Binyun Yang, Fei Shen, Yuxin Chen, Jingnan Zheng, Xiang Wang, An Zhang, and Tat-Seng Chua. Transport and merge: Cross-architecture merging for large language models. *arXiv preprint arXiv:2602.05495*, 2026.

Tijmen de Haan, Yuan-Sen Ting, Tirthankar Ghosal, Tuan Dung Nguyen, Alberto Accomazzi, Azton Wells, Nesar Ramachandra, Rui Pan, and Zechang Sun. Achieving gpt-4o level performance in astronomy with a specialized 8b-parameter large language model. *Scientific Reports*, 15(1), April 2025. ISSN 2045-2322. doi: 10.1038/s41598-025-97131-y. URL http://dx.doi.org/10.1038/s41598-025-97131-y.

Lucio M Dery, Zohar Yahav, Henry Prior, Qixuan Feng, Jiajun Shen, and Arthur Szlam. Latent space communication via kv cache alignment. *arXiv preprint arXiv:2601.06123*, 2026.

Akash Dhasade, Divyansh Jhunjhunwala, Milos Vujasinovic, Gauri Joshi, and Anne-Marie Kermarrec. Navigating the accuracy-size trade-off with flexible model merging. In *International Conference on Learning Representations*, volume 2026, pages 81010–81041, 2026.

Thomas Diggelmann, Jordan Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. Climate-fever: A dataset for verification of real-world climate claims, 2021. URL https://arxiv.org/abs/2012.00614.

Jingtong Dou, Chuancheng Shi, Jian Wang, Fei Shen, Zhiyong Wang, and Tat-Seng Chua. Beyond surface artifacts: Capturing shared latent forgery knowledge across modalities. *arXiv preprint arXiv:2604.07763*, 2026a.

Jingtong Dou, Chuancheng Shi, Yemin Wang, Shiming Guo, Anqi Yi, Wenhua Wu, Li Zhang, Fei Shen, and Tat-Seng Chua. Dna: Uncovering universal latent forgery knowledge. *arXiv preprint arXiv:2601.22515*, 2026b.

Marco Fumero, Marco Pegoraro, Valentino Maiorca, Francesco Locatello, and Emanuele Rodolà. Latent functional maps: a spectral framework for representation alignment. *Advances in Neural Information Processing Systems*, 37: 66178–66203, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Yifei He, Siqi Zeng, Yuzheng Hu, Rui Yang, Tong Zhang, and Han Zhao. Mergebench: A benchmark for merging domain-specialized llms. *Advances in Neural Information Processing Systems*, 38, 2026.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL `https://arxiv.org/abs/2009.03300`.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Cosmos qa: Machine reading comprehension with contextual commonsense reasoning, 2019. URL `https://arxiv.org/abs/1909.00277`.

Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. The platonic representation hypothesis. *arXiv preprint arXiv:2405.07987*, 2024.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

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

Kyungjin Im, Miru Kim, Chanin Eom, and Minhae Kwon. Post-hoc merging is not enough: Many-shot model merging with loss-gap balancing. *arXiv preprint arXiv:2606.16501*, 2026.

Moritz Imfeld, Jacopo Graldi, Marco Giordano, Thomas Hofmann, Sotiris Anagnostidis, and Sidak Pal Singh. Transformer fusion with optimal transport. In *International Conference on Learning Representations*, volume 2024, pages 29313–29336, 2024.

Hangyuan Ji, Jian Yang, Linzheng Chai, Chaoren Wei, Liqun Yang, Yunlong Duan, Yunli Wang, Tianzhen Sun, Hongcheng Guo, Tongliang Li, Changyu Ren, and Zhoujun Li. Sevenllm: Benchmarking, eliciting, and enhancing abilities of large language models in cyber threat intelligence, 2024a. URL `https://arxiv.org/abs/2405.03446`.

Hangyuan Ji, Jian Yang, Linzheng Chai, Chaoren Wei, Liqun Yang, Yunlong Duan, Yunli Wang, Tianzhen Sun, Hongcheng Guo, Tongliang Li, et al. Sevenllm: Benchmarking, eliciting, and enhancing abilities of large language models in cyber threat intelligence. *arXiv preprint arXiv:2405.03446*, 2024b.

Jiaming Ji, Donghai Hong, Borong Zhang, Boyuan Chen, Josef Dai, Boren Zheng, Tianyi Alex Qiu, Jiayi Zhou, Kaile Wang, Boxun Li, et al. Pku-saferlhf: Towards multi-level safety alignment for llms with human preference. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 31983–32016, 2025.

Paul Kassianik, Baturay Saglam, Alexander Chen, Blaine Nelson, Anu Vellore, Massimo Aufiero, Fraser Burch, Dhruv Kedia, Avi Zohary, Sajana Weerawardhena, Aman Priyanshu, Adam Swanda, Amy Chang, Hyrum Anderson, Kojin Oshiba, Omar Santos, Yaron Singer, and Amin Karbasi. Llama-3.1-foundationai-securityllm-base-8b technical report, 2025. URL `https://arxiv.org/abs/2504.21039`.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. DistiLLM: Towards streamlined distillation for large language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 24872–24895. PMLR, 21–27 Jul 2024. URL `https://proceedings.mlr.press/v235/ko24c.html`.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMlR, 2019.

Chanhyuk Lee, Jiho Choi, Chanryeol Lee, Donggyun Kim, and Seunghoon Hong. Adarank: Adaptive rank pruning for enhanced model merging. In *International Conference on Learning Representations*, volume 2026, pages 6484–6510, 2026.

Karel Lenc and Andrea Vedaldi. Understanding image representations by measuring their equivariance and equivalence. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 991–999, 2015.

Shaotian Li, Shangze Li, Chuancheng Shi, Wenhua Wu, Yanqiu Wu, Xiaohan Yu, Fei Shen, and Tat-Seng Chua. Latent anomaly knowledge excavation: Unveiling sparse sensitive neurons in vision-language models. *arXiv preprint arXiv:2604.07802*, 2026.

Valentino Maiorca, Luca Moschella, Antonio Norelli, Marco Fumero, Francesco Locatello, and Emanuele Rodolà. Latent space translation via semantic alignment. *Advances in Neural Information Processing Systems*, 36:55394–55414, 2023.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering, 2018. URL `https://arxiv.org/abs/1809.02789`.

Mazda Moayeri, Keivan Rezaei, Maziar Sanjabi, and Soheil Feizi. Text-to-concept (and back) via cross-model alignment. In *International Conference on Machine Learning*, pages 25037–25060. PMLR, 2023.

Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. Relative representations enable zero-shot latent space communication. *arXiv preprint arXiv:2209.15430*, 2022.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024.

Narmeen Oozeer, Dhruv Nathawani, Nirmalendu Prakash, Michael Lan, Abir Harrasse, and Amirali Abdullah. Activation space interventions can be transferred between large language models. *arXiv preprint arXiv:2503.04429*, 2025.

Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. *Advances in Neural Information Processing Systems*, 36:66727–66754, 2023.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. Medmcqa : A large-scale multi-subject multi-choice dataset for medical domain question answering, 2022. URL `https://arxiv.org/abs/2203.14371`.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 2080–2094, 2021.

Zeju Qiu, Weiyang Liu, Haiwen Feng, Yuxuan Xue, Yao Feng, Zhen Liu, Dan Zhang, Adrian Weller, and Bernhard Schölkopf. Controlling text-to-image diffusion by orthogonal finetuning. *Advances in neural information processing systems*, 36:79320–79362, 2023.

Zihuan Qiu, Lei Wang, Yang Cao, Runtong Zhang, Bing Su, Yi Xu, Fanman Meng, Linfeng Xu, Qingbo Wu, and Hongliang Li. Null-space filtering for data-free continual model merging: Preserving stability, promoting plasticity. In *International Conference on Learning Representations*, volume 2026, pages 128492–128518, 2026.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL `https://arxiv.org/abs/2412.15115`.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. *Advances in neural information processing systems*, 30, 2017.

Benjamin Ramtoula, Pierre-Yves Lajoie, Paul Newman, and Daniele De Martini. Fantastic features and where to find them: A probing method to combine features from multiple foundation models. *Advances in Neural Information Processing Systems*, 38:89960–89989, 2026.

Paul Richmond, Constantinos Papageorgakis, Vasilis Niarchos, Borun Chowdhury, and Prarit Agarwal. Feyntune: large language models for high-energy theory. *Machine Learning: Science and Technology*, 7(2):025012, 2026.

Filippo Rinaldi, Aniello Panariello, Giacomo Salici, Angelo Porrello, and Simone Calderara. Transporting task vectors across different architectures without training. *arXiv preprint arXiv:2602.12952*, 2026.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 4463–4473, 2019.

Chuancheng Shi, Shangze Li, Shiming Guo, Simiao Xie, Wenhua Wu, Jingtong Dou, Chao Wu, Canran Xiao, Cong Wang, Zifeng Cheng, et al. Where culture fades: revealing the cultural gap in text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14702–14712, 2026a.

Chuancheng Shi, Shangze Li, Wenjun Lu, Wenhua Wu, Cong Wang, Zifeng Cheng, Fei Shen, and Tat-Seng Chua. Tracerouter: robust safety for large foundation models via path-level intervention. *arXiv preprint arXiv:2601.21900*, 2026b.

Chuancheng Shi, Wenhua Wu, Fei Shen, Xiaogang Zhu, Kun Hu, and Zhiyong Wang. Orthoeraser: coupled-neuron orthogonal projection for concept erasure. *arXiv preprint arXiv:2603.11493*, 2026c.

Sidak Pal Singh and Martin Jaggi. Model fusion via optimal transport. *Advances in Neural Information Processing Systems*, 33:22045–22055, 2020.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, 2019.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

David Thulke, Yingbo Gao, Petrus Pelser, Rein Brune, Rricha Jalota, Floris Fok, Michael Ramos, Ian van Wyk, Abdallah Nasir, Hayden Goldstein, Taylor Tragemann, Katie Nguyen, Ariana Fowler, Andrew Stanco, Jon Gabriel, Jordan Taylor, Dean Moro, Evgenii Tsymbalov, Juliette de Waal, Evgeny Matusov, Mudar Yaghi, Mohammad Shihadah, Hermann Ney, Christian Dugast, Jonathan Dotan, and Daniel Erasmus. Climategpt: Towards ai synthesizing interdisciplinary research on climate change, 2024. URL `https://arxiv.org/abs/2401.09646`.

Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. Knowledge fusion of large language models. In *International Conference on Learning Representations*, volume 2024, pages 18303–18322, 2024.

Fanqi Wan, Longguang Zhong, Ziyi Yang, Ruijun Chen, and Xiaojun Quan. Fusechat: Knowledge fusion of chat models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 21629–21653, 2025.

Xinyuan Wang, Yanchi Liu, Wei Cheng, Xujiang Zhao, Zhengzhang Chen, Wenchao Yu, Yanjie Fu, and Haifeng Chen. MixLLM: Dynamic routing in mixed large language models. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 10912–10922, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.545. URL `https://aclanthology.org/2025.naacl-long.545/`.

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

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

Zhengqi Xu, Han Zheng, Jie Song, Li Sun, and Mingli Song. Training-free heterogeneous model merging. *arXiv preprint arXiv:2501.00061*, 2024.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in neural information processing systems*, 36:7093–7115, 2023.

Sihan Yang, Kexuan Shi, and Weiyang Liu. Orthogonal model merging. *arXiv preprint arXiv:2602.05943*, 2026.

Kotaro Yoshida, Yuji Naraki, Takafumi Horie, Ryotaro Shimizu, Ioannis Mitliagkas, and Hiroki Naganuma. Distac: Conditioning task vectors via distillation for robust model merging. In *International Conference on Learning Representations*, volume 2026, pages 118509–118534, 2026.

Zihan Zhao, Da Ma, Lu Chen, Liangtai Sun, Zihao Li, Yi Xia, Hongshen Xu, Zichen Zhu, Su Zhu, Shuai Fan, et al. Chemdfm: A large language foundation model for chemistry. In *Neurips 2024 Workshop Foundation Models for Science: Progress, Opportunities, and Challenges*, 2024.

Appendix

This appendix provides additional details, analyses, and discussions that complement the main paper. Appendix A describes the domain-specific source models, training datasets, evaluation benchmarks, and metrics used across the 15 capability domains. Appendix B summarizes the notation used throughout SuperAI. Appendix C presents additional experiments on PCA subspace dimensionality, training-data scale, and multi-source scalability. Appendix D provides further details on cross-layer matching and the selection of the shared host layer. Appendix E discusses the current scope and limitations of our evaluation, together with directions for future work. Finally, Appendix F provides additional discussion on several methodological and experimental questions concerning SuperAI.

A Domain-Specific Source Models, Datasets, and Metrics

We describe the source model, merging-training data, and evaluation metric for each of the 15 capability domains. Training datasets refer to the data used in our merging procedure, rather than necessarily the original fine-tuning corpora of the source models. All reported metrics are higher-is-better.

Science. We use ScienceQA-OFT-8B as the source model and ARC-Challenge Clark et al. [2018] for merging training and evaluation. The task consists of multiple-choice science questions. We report accuracy, measured by whether the predicted option matches the reference answer.

Common Sense. We use CommonsenseQA-OFT-8B as the source model and CommonsenseQA Talmor et al. [2019] as the corresponding dataset. The task requires selecting the correct answer to questions involving everyday commonsense knowledge. The primary metric is multiple-choice accuracy.

Social. We use Social-IQA-OFT-8B as the source model and the BIG-bench Social IQA Sap et al. [2019] task for merging training and evaluation. Questions concern people's intentions, reactions, and likely consequences in social situations. We report multiple-choice accuracy.

Math. We use NuminaMath-OFT-8B as the source model and SVAMP Patel et al. [2021] for merging training and evaluation. The task requires generating numeric answers to arithmetic word problems. We extract the final numeric value from the generated response and compare it with the reference using relative and absolute tolerances of 10^{-4}. The primary metric is numeric answer accuracy.

Code. We use Qwen2.5-Coder-7B-Instruct Hui et al. [2024] as the source model and MBPP Austin et al. [2021] for merging training. We report Pass@1: a task is counted as solved only when a single generated program passes both the base and additional tests.

Multilingual. We use MergeBench Llama-3.2-3B Multilingual He et al. [2026] as the source model and an eight-language subset of M-ARC Clark et al. [2018]. The task evaluates multiple-choice reasoning across languages. We report sample-averaged accuracy, pooling correct predictions over all evaluated examples rather than taking an unweighted average of per-language accuracies.

Safety. We use MergeBench Llama-3.2-3B Safety He et al. [2026] as the source model and PKU-SafeRLHF Ji et al. [2025] as the corresponding dataset. The task presents a user request and candidate responses, and asks the model to select the safer response. We report safer-response selection accuracy. This metric evaluates safety-related response selection rather than the safety of freely generated responses.

Law. We use Saul-Instruct-v Colombo et al. [2024] as the source model. Merging training uses the local MMLU Law subset Hendrycks et al. [2020]. Both are formulated as multiple-choice legal questions. The primary metric is answer-selection accuracy.

Finance/Business. We use Finance Llama-3.1-8B as the source model and FinQA Chen et al. [2022] for merging training and evaluation. The task requires answering questions based on financial information. Numeric answers are extracted and compared using relative and absolute tolerances of 10^{-4}. For nonnumeric references, we use case-insensitive reference-answer matching in the generated response. We report the fraction of answers judged correct under these rules.

Astronomy. We use AstroSage-8B de Haan et al. [2025] as the source model and arXiv astro-ph text for merging training. Evaluation uses MMLU Astronomy multiple-choice questions. The primary metric is accuracy on MMLU Astronomy; this score is distinct from language-modeling evaluation on the arXiv training corpus.

Table 7: **Training datasets and evaluation metrics for the 15 capability domains.** We list the source models, training datasets, and corresponding evaluation metrics. All metrics are higher-is-better.

Domain	Source Model	Training Dataset	Task Metric ↑
Science	ScienceQA-OFT-8B	ARC-Challenge	Accuracy
Common Sense	CommonsenseQA-OFT-8B	CommonsenseQA	Accuracy
Social	Social-IQA-OFT-8B	BIG-bench Social IQA	Accuracy
Math	NuminaMath-OFT-8B	SVAMP	Numeric Answer Accuracy
Code	Qwen2.5-Coder-7B-Instruct	MBPP	Pass@1
Multilingual	Llama-3.2-3B Multilingual	M-ARC	Averaged Accuracy
Safety	Llama-3.2-3B Safety	PKU-SafeRLHF	Safer-response Selection Accuracy
Law	Saul-Instruct-v1	MMLU Law	Accuracy
Finance/Business	Finance Llama-3.1-8B	FinQA	Answer Accuracy
Astronomy	AstroSage-8B	Astro-ph	Accuracy
Medical	HuatuoGPT-o1-7B	MedMCQA	Accuracy
Climate	ClimateGPT-7B	Climate-FEVER Binary	Accuracy
Cyber Security	Foundation-Sec-8B	SEVENLLM	Token-level F1
Chemistry	ChemDFM-v1.5-8B	MMLU Chemistry	Accuracy
Physics	FeynTune-S1	Hep-th Continuation	Next-sentence Selection Accuracy

Medical. We use HuatuoGPT-o1-7B Chen et al. [2024] as the source model and MedMCQA Pal et al. [2022] for merging training and evaluation. The task consists of multiple-choice questions covering medical knowledge. We report accuracy based on agreement with the reference answer option.

Climate. We use ClimateGPT-7B Thulke et al. [2024] as the source model and the local binary formulation of Climate-FEVER Diggelmann et al. [2021]. The task requires classifying climate-related statements according to the provided binary labels. The primary metric is classification accuracy.

Cyber Security. We use Foundation-Sec-8B Kassianik et al. [2025] as the source model and SEVENLLM Ji et al. [2024a] for merging training and evaluation. The task requires generating responses to cybersecurity-related instructions. We report token-level F1 between generated and reference responses, combining token precision and recall. Scores are computed for individual examples and then averaged over the evaluation set.

Chemistry. We use ChemDFM-v1.5-8B Zhao et al. [2024] as the source model. Merging training uses a local mixture of the MMLU High School Chemistry and College Chemistry subsets, containing 165 and 6 training examples, respectively. Evaluation uses 100 questions from MMLU College Chemistry. The primary metric is multiple-choice accuracy.

Physics. We use FeynTune-S1 Llama-3.1-8B Richmond et al. [2026] as the source model and continuation data constructed from arXiv hep-th text. The task requires selecting the true next sentence from a set of candidates. We report next-sentence selection accuracy, measuring whether the selected continuation matches the reference.

Aggregate Evaluation. For multi-source fusion, we report the unweighted mean of the 15 domain scores after expressing them on a common scale. Each domain receives equal weight, regardless of its evaluation-set size. This aggregate is a mean of task-specific metrics, rather than a pooled accuracy over all questions. We additionally report performance on a separate MMLU evaluation set to assess general-capability retention; this score is excluded from the 15-domain average.

Generalization Evaluation. For cross-dataset generalization, we evaluate Science, Common Sense, Social, and Climate capabilities on OpenBookQA Mihaylov et al. [2018], PIQA Bisk et al. [2019], CosmosQA Huang et al. [2019], and ClimateCheck Abu Ahmad et al. [2025], respectively, none of which is used for capability merging or residual-MLP training.

B Notation Summary

Table 8 summarizes the notation used throughout SuperAI and groups the main symbols according to the three stages of the framework: orthogonal subspace alignment, feature mapping, and residual fusion. We use h and s to denote the

Table 8: Summary of notation used in SuperAI.

Symbol	Definition
Models and representations	
$f_h, \{f_s\}_{s=1}^{K}$	Host model and K source models.
d_h, d_s	Hidden dimensions of the host and source model s.
$\mathcal{A}_s = \{a_i\}_{i=1}^{n_s}$	Shared anchor prompts for source model s and the host.
$\ell_m, h_m^{(\ell_m)}(a_i)$	Layer index and last non-padding-token hidden state of model $m \in \{s, h\}$.
$H_m^{(\ell_m)}$	Matrix of hidden states for prompts in $\mathcal{A}_s$.
Subspace alignment	
$\mu_m, u_m(a_i), U_m$	Hidden-state mean, centered and normalized state, and matrix of normalized states.
c_m, P_m, r	PCA mean, principal-component matrix, and shared subspace dimension.
Z_m	PCA coordinates of model m for the paired prompts.
$R_s^*, \widehat{Z}_{s\to h}$	Orthogonal source-to-host map and aligned source coordinates.
$S_s(\ell_s, \ell_h)$	Held-out alignment score for a source–host layer pair.
w_s	Weight of source model s, with $\sum_{s=1}^{K} w_s = 1$.
$\ell_s^*(\ell_h), \ell_h^*$	Best source layer for a candidate host layer and selected shared host layer.
Feature mapping	
$z_s(a_i), \widehat{z}_{s\to h}(a_i)$	Source PCA coordinates and their rotated host-space coordinates.
$b_{s\to h}(a_i)$	Rotated coordinates reconstructed in the normalized host hidden space.
$\widetilde{h}_{s\to h}(a_i)$	Mapped host-space representation after magnitude restoration.
$\Delta_s(a_i)$	Target feature increment $\widetilde{h}_{s\to h}(a_i) - h_h(a_i)$.
Residual fusion and optimization	
$h_{i,t}, h'_{i,t}$	Host hidden state before and after residual injection at token position t.
$b, W_{\text{down}}, W_{\text{up}}$	MLP bottleneck width, down-projection, and up-projection.
$q_{i,t}, \widehat{\Delta}_{i,t}, \alpha$	Bottleneck activation, predicted increment, and residual scaling factor.
$\mathcal{A}_s^{\text{tr}}, s_i, N$	Source-s training prompts, source identity of prompt a_i, and total training-prompt count.
p_i, ρ_s	Last non-padding position of prompt a_i and source-specific increment normalization factor.
$T_i, y_{i,t}, p_\theta$	Answer-token positions, ground-truth token, and model output distribution.
$\mathcal{L}_{\text{task}}, \mathcal{L}_{\text{inc}}$	Answer-token cross-entropy and increment-supervision losses.
$\lambda_{\text{task}}, \lambda_{\text{inc}}, \mathcal{L}$	Loss weights and their weighted joint objective.
ϵ	Positive constant used for numerical stability.

host model and a source model, respectively, while ℓ_h and ℓ_s index their corresponding layers. Shared anchor prompts are used to extract paired hidden representations from the source and host models, which are then normalized and projected into low-dimensional PCA subspaces. The resulting coordinates are aligned through an orthogonal mapping, and the alignment scores are subsequently used for source–host layer matching and shared host-layer selection.

After alignment, the source representation is mapped into the host hidden space to construct a source-specific target feature increment, denoted by $\Delta_s(a_i)$. These increments provide representation-level supervision for the residual fusion module. During joint training, a lightweight residual MLP predicts input-dependent increments from the host hidden states, and the resulting update is injected into the selected host layer. The training objective combines task-output supervision with source-increment supervision, controlled by the corresponding loss weights.

For each source model, the host-side normalization statistics, PCA mean, and PCA basis are fitted using the anchor prompts paired with that source model. Therefore, these quantities are technically source-dependent even when they belong to the host representation space. To simplify the notation, we omit the corresponding source index whenever it is unambiguous from the context. The same convention is used for other source-conditioned intermediate quantities where explicitly carrying the source index would unnecessarily complicate the presentation.

C More Experiments

C.1 Effect of PCA Subspace Dimension

To evaluate the effect of the PCA subspace dimension on multi-source capability transfer, we vary the PCA rank from 16 to 512 and compare the performance across three host models: Llama-3.2-3B, Qwen2.5-3B, and Pythia-2.8B. The results show in Table 9 that all PCA rank settings outperform their corresponding base models, demonstrating that the proposed method effectively integrates multi-domain capabilities across different subspace dimensions. Specifically, Llama-3.2-3B achieves its highest average score of 56.9 at rank 512, improving by 10.2 points over the base model. Qwen2.5-3B attains a peak score of 61.3 across ranks 32 and 128, with a 5.1-point gain. Pythia-2.8B reaches its best score of 32.8 at rank 128, yielding a 9.9-point improvement. Thus, Llama-3.2-3B, Qwen2.5-3B, and Pythia-2.8B deliver optimal performance at ranks 512, 32 and 128, and 128, respectively. Overall performance does not scale monotonically with increasing rank. A plausible reason is that higher-dimensional subspaces retain more redundant and complex signals, which complicates optimization for the mapping module and hurts capability transfer. Considering the overall performance across all three host models, rank 128 achieves a solid balance between information preservation and training difficulty, and is therefore adopted as the default configuration.

Table 9: **Effect of PCA subspace dimensionality.** Each cell reports the 15-domain average score, with the absolute change from the corresponding base model shown in parentheses. PCA rank denotes the requested upper bound.

PCA Rank	Llama-3.2-3B	Qwen2.5-3B	Pythia-2.8B
Base	46.7	56.2	22.9
16	56.3 (↑ 9.6)	60.2 (↑ 4.0)	28.7 (↑ 5.8)
32	55.3 (↑ 8.6)	**61.3** (↑ 5.1)	28.3 (↑ 5.4)
64	54.5 (↑ 7.8)	61.2 (↑ 5.0)	29.0 (↑ 6.1)
128	54.2 (↑ 7.5)	**61.3** (↑ 5.1)	**32.8** (↑ 9.9)
256	55.2 (↑ 8.5)	60.8 (↑ 4.6)	28.9 (↑ 6.0)
512	**56.9** (↑ 10.2)	59.8 (↑ 3.6)	28.9 (↑ 6.0)

C.2 Effect of Training Data Scale

To analyze the effect of training data scale on multi-source capability transfer, we vary the maximum number of training samples per domain from 50 to 1,000 and compare performance across three host models. The results in Table 10 indicate that even with only 50 samples per domain, all three models markedly outperform their respective base models, demonstrating high data efficiency. As the sample size increases, different host models exhibit distinct trajectories: Llama-3.2-3B achieves its peak average score of 57.6 with 200 samples, whereas Qwen2.5-3B and Pythia-2.8B reach their highest scores of 61.3 and 32.8 at 1,000 samples, respectively. These findings show that increasing training data generally promotes more adequate capability mappings, yet performance does not improve monotonically with data scale. An excessive number of samples may introduce redundant or complex patterns, thereby complicating optimization for the mapping module. Overall, the proposed method yields substantial improvements in data-scarce settings, while larger training sets provide additional gains for certain host models.

Table 10: **Effect of training data volume.** Each cell reports the 15-domain average score, with the absolute change from the corresponding base model shown in parentheses. The training cap is applied per domain; domains with fewer examples use all available training samples.

Examples/Domain	Llama-3.2-3B	Qwen2.5-3B	Pythia-2.8B
Base	46.7	56.2	22.9
50	54.1 (↑ 7.4)	57.9 (↑ 1.7)	27.8 (↑ 4.9)
100	56.3 (↑ 9.6)	59.6 (↑ 3.4)	27.1 (↑ 4.2)
200	**57.6** (↑ 10.9)	58.9 (↑ 2.7)	26.9 (↑ 4.0)
500	55.6 (↑ 8.9)	60.0 (↑ 3.8)	27.4 (↑ 4.5)
1000	54.2 (↑ 7.5)	**61.3** (↑ 5.1)	**32.8** (↑ 9.9)

C.3 Multi-Source Scalability

To evaluate the scalability of cross-architecture multi-source capability fusion, we incrementally integrated 1–15 source models in a fixed order and compared SuperAI with THESEUS and Transport and Merge (T&M) across four representative domains and three host architectures. As shown in Figure 9, scores generally declined as more sources were incorporated. Averaged across the 12 domain–host combinations, SuperAI declined by 7.7 points from 1 to 15 sources, compared with 12.8 and 13.4 points for THESEUS and T&M, respectively. SuperAI achieved the highest scores at every evaluated source count in Math, Safety, and Physics. At the 15-source endpoint, the sole domain–host combination in which it did not lead was Law on Llama-3.2-3B, where it trailed T&M by 0.4 points. At that endpoint, SuperAI exceeded the better of the two baselines in each domain–host combination by 19.2 points on average, with particularly large margins in Safety and Physics on Pythia-2.8B and Physics on Llama-3.2-3B. These results suggest that SuperAI mitigates interference under the evaluated fixed source order, although its scores still decline as more sources are integrated.

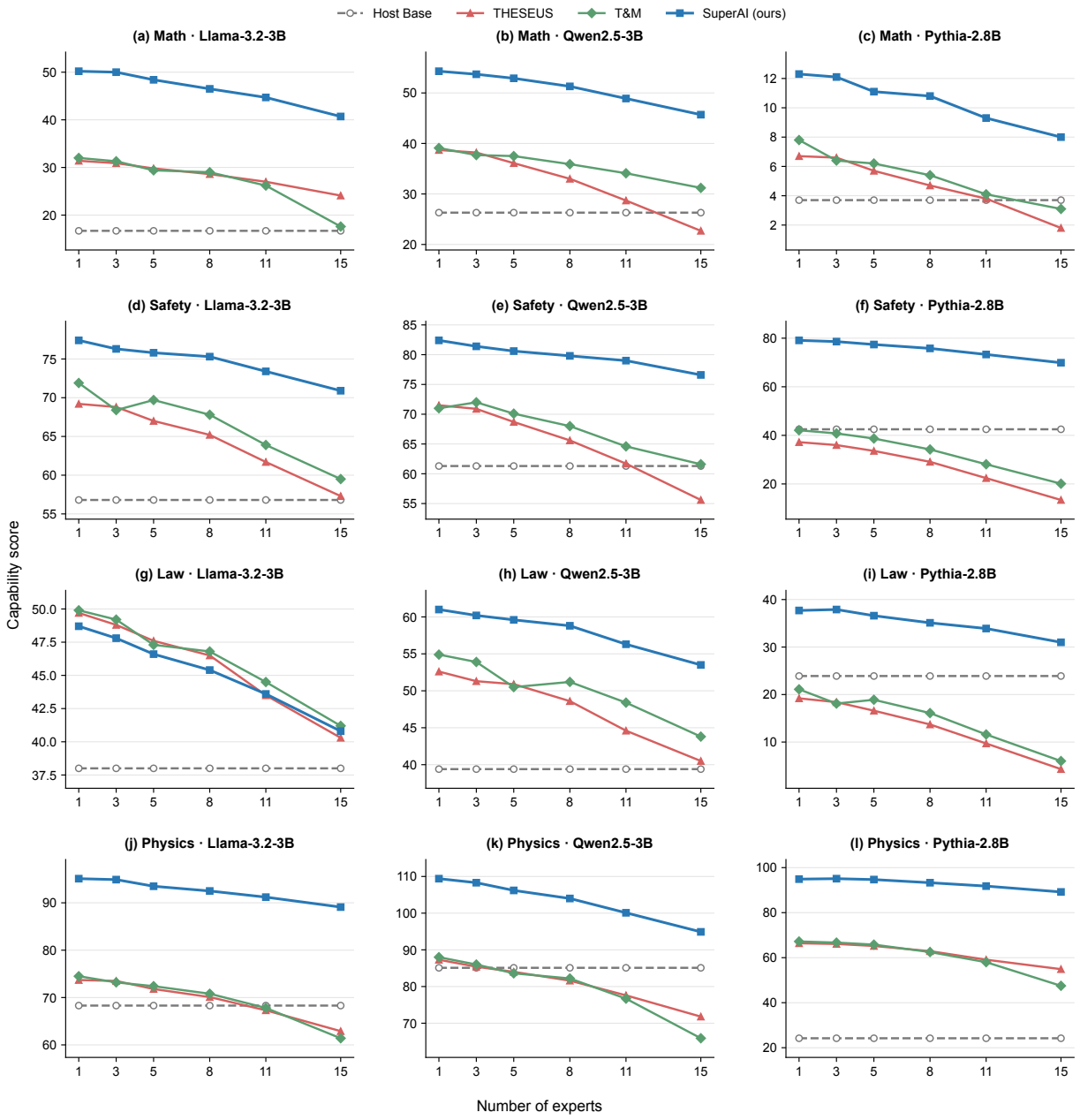

Figure 9: **Multi-source scalability across capability domains.** Capability scores of SuperAI, THESEUS, and T&M as the number of merged sources increases across Math, Safety, Law, and Physics.

D Cross-Layer Matching and Shared Host-Layer Selection

This section provides additional details on the selection of the shared host layer. Let $\mathcal{L}_h$ denote the candidate host layers and $\mathcal{L}_s$ denote the candidate layers of source model s. For each source model, we use prompts that are shared with the host model:

$$\mathcal{A}_s = \{a_i\}_{i=1}^{n_s}. \tag{10}$$

For every candidate layer pair (ℓ_s, ℓ_h), we extract the hidden state of the last non-padding token from both models. We then apply the normalization, PCA projection, and orthogonal Procrustes alignment defined in Section 3.1. The mapping is fitted on $\mathcal{A}_s^{\text{fit}}$ and evaluated on a disjoint matching set $\mathcal{A}_s^{\text{val}}$. For a validation prompt a_i, the aligned source

coordinate is

$$\widehat{z}_{s\to h}^{(\ell_s,\ell_h)}(a_i) = z_s^{(\ell_s)}(a_i) R_s^{(\ell_s,\ell_h)}. \tag{11}$$

We define the matching score of a candidate layer pair as the mean cosine similarity between the aligned source coordinates and the paired host coordinates:

$$S_s(\ell_s,\ell_h) = \frac{1}{|\mathcal{A}_s^{\mathrm{val}}|} \sum_{a_i \in \mathcal{A}_s^{\mathrm{val}}} \cos\left(\widehat{z}_{s\to h}^{(\ell_s,\ell_h)}(a_i), z_h^{(\ell_h,s)}(a_i)\right). \tag{12}$$

A higher score indicates that the source and host layers produce more consistent representations for the same inputs after orthogonal alignment. CKA and Retrieval@1 are reported as additional diagnostic metrics but are not used for layer selection. For each candidate host layer, we first identify the best-matching layer from every source model:

$$\ell_s^*(\ell_h) = \arg\max_{\ell_s \in \mathcal{L}_s} S_s(\ell_s,\ell_h). \tag{13}$$

The corresponding score is

$$S_s^*(\ell_h) = S_s\left(\ell_s^*(\ell_h), \ell_h\right). \tag{14}$$

We then aggregate the best matching scores across all source models:

$$S(\ell_h) = \sum_{s=1}^{K} w_s S_s^*(\ell_h), \qquad \sum_{s=1}^{K} w_s = 1. \tag{15}$$

We use $w_s = 1/K$ by default, so each source model contributes equally to layer selection. The shared host layer is selected as

$$\ell_h^* = \arg\max_{\ell_h \in \mathcal{L}_h} S(\ell_h), \tag{16}$$

and the final layer selected from source model s is

$$\ell_s^* = \ell_s^*(\ell_h^*). \tag{17}$$

Thus, different source models may contribute representations from different source layers, while all representations are aligned with the same host layer ℓ_h^*. After layer selection, we refit the corresponding mappings using the available training anchors and use them to construct the source-specific feature increments. The layer search is performed only once before residual training and introduces no additional source-model cost during inference.

E Limitation

Our current evaluation focuses on offline benchmark-based capability fusion, where prompts and target tasks are predefined. We have not yet systematically evaluated SuperAI in interactive human–AI settings involving multi-turn dialogue, evolving user intent, or feedback-dependent behavior. While this does not affect the validity of the reported offline fusion results, interactive deployment may introduce additional challenges such as maintaining transferred capabilities consistently across dialogue turns and adapting residual updates to changing conversational context. Future work will extend SuperAI to interactive settings by incorporating conversation-level representations, turn-aware residual routing, and human-feedback-based evaluation to study how fused capabilities behave under dynamic user interactions.

F More Discussion

▷ ***Q1. The method uses ground-truth task labels to train the residual MLP. How can the authors distinguish capability transfer from ordinary supervised adaptation of the host model?***

We would like to clarify that SuperAI is not equivalent to standard supervised adaptation. The key distinction is that the residual MLP is jointly optimized with both the task loss and the source-derived increment supervision. As shown in Table 2, using only the task loss ($\lambda_{\text{inc}} = 0$) consistently underperforms the joint objective across all three host models, while using only the increment loss improves representation matching but does not provide the same downstream gains. This suggests that the improvement does not arise from task supervision alone; rather, the source-derived increments provide an additional transfer signal that complements the task objective. The role of the task loss is to adapt the transferred source information to the downstream computation of the host model, which is necessary because representation correspondence alone does not guarantee functional compatibility.

▷ ***Q2. The improvement over THESEUS and Transport-and-Merge is large. Could this simply be due to the additional trainable residual MLP rather than the proposed cross-model alignment mechanism?***

Our ablations indicate that the gain cannot be explained by the residual MLP alone. First, removing source-increment supervision and retaining the same task-trained residual module leads to substantially lower performance than the full joint objective. Second, Table 3 shows that, under the same downstream training setup, replacing PCA + orthogonal alignment with no alignment, PCA-only mapping, or unconstrained linear mapping consistently degrades performance across the three host architectures. Third, Table 4 shows that the nonlinear residual mapper further improves over a linear mapper, indicating that the alignment module and the host-adaptation module play complementary roles rather than one simply substituting for the other. Therefore, the empirical gains result from the combination of cross-model geometric alignment and host-compatible residual adaptation.

▷ ***Q3. The paper reports very high post-alignment similarity. Could these scores merely reflect overfitting of the Procrustes mapping to paired anchor samples?***

The reported alignment scores are not computed on the same samples used to fit the mapping. As described in the Appendix, for every candidate source–host layer pair, the normalization, PCA projection, and orthogonal Procrustes mapping are fitted on A_s^{fit}, while the matching score is evaluated on a disjoint validation set A_s^{val}. CKA and Retrieval@1 are also used as independent diagnostic metrics rather than as optimization objectives for layer selection. Moreover, Table 5 shows that downstream capability transfer improves systematically as the held-out post-alignment Match score increases from weakly matched to medium and best-matched layer pairs. This provides evidence that the alignment score captures transferable cross-model structure rather than merely fitting the anchor examples.

▷ ***Q4. Table 1 shows that some domains degrade after multi-source fusion. For example, Code decreases from 39.7 to 28.3 on Llama-3.2-3B, from 55.6 to 44.2 on Qwen2.5-3B, and from 11.1 to 1.3 on Pythia-2.8B. How should these results be interpreted?***

The Code results reflect the trade-off inherent in joint multi-domain fusion rather than the overall effectiveness of capability integration. SuperAI optimizes a single host model across 15 heterogeneous domains, whose objectives and output formats differ substantially. In particular, code generation requires highly structured token-level generation, making its preferred representation updates different from those of many reasoning and classification domains. Consequently, optimizing the shared representation for broad multi-domain performance does not necessarily preserve the optimum of every individual domain.

Importantly, the goal of this setting is joint capability composition rather than independent per-domain specialization, and SuperAI consistently improves the aggregate 15-domain performance across all three hosts. Future work will investigate domain-conditioned weighting or routing to further improve the balance among heterogeneous capabilities.