RAIN-MERGING: A GRADIENT-FREE METHOD TO ENHANCE INSTRUCTION FOLLOWING IN LARGE REASONING MODELS WITH PRESERVED THINKING FORMAT

Anonymous authors

000

001

002

004

006

008 009 010

011 012

013

014

015

016

017

018

019

021

023

024

025

026

027

028

029

031

032

034

037

040

041

042

043

044

046

047

048

051

052

Paper under double-blind review

ABSTRACT

Large reasoning models (LRMs) excel at a long chain of reasoning but often fail to faithfully follow instructions regarding output format, constraints, or specific requirements. We investigate whether this gap can be closed by integrating an instruction-tuned model (ITM) into an LRM. Analyzing their differences in parameter space, namely task vectors, we find that their principal subspaces are nearly orthogonal across key modules, suggesting a lightweight merging with minimal interference. However, we also demonstrate that naïve merges are fragile because they overlook the output format mismatch between LRMs (with explicit thinking and response segments) and ITMs (answers-only). We introduce RAIN-Merging (Reasoning-Aware Instruction-attention guided Null-space projection Merging), a gradient-free method that integrates instruction following while preserving thinking format and reasoning performance. First, with a small reasoning calibration set, we project the ITM task vector onto the null space of forward features at thinking special tokens, which preserves the LRM's structured reasoning mechanisms. Second, using a small instruction calibration set, we estimate instruction attention to derive module-specific scaling that amplifies instructionrelevant components and suppresses leakage. Across four instruction-following benchmarks and nine reasoning & general capability benchmarks, RAIN-Merging substantially improves instruction adherence while maintaining reasoning quality. The gains are consistent across model scales and architectures, translating to improved performance in agentic scenarios.

1 Introduction

In the current boom of research, Large Reasoning Models (LRMs, like OpenAI-o1 (Jaech et al., 2024), DeepSeek-R1 (Guo et al., 2025)) have shown strong potential on tasks that require rigorous multi-step reasoning (Wei et al., 2022), such as mathematical derivation (Shao et al., 2024) and program synthesis (Guo et al., 2024). However, a discouraging paradox has emerged: although LRMs perform well in purely reasoning-oriented settings, they lag in instruction following (Fu et al., 2025; Li et al., 2025). They often generate lengthy logical derivations yet ignore user-specified formats, constraints, or specific operational requirements in the final response. This inconsistency undermines LRM practicality and reliability in real-world applications (Chkirbene et al., 2024), especially in agent (Qi et al., 2025) and professional tool deployments (Zhao et al., 2024).

A straightforward remedy is to continue training LRMs with supervised fine-tuning (SFT) to strengthen instruction following. However, building high-quality supervision datasets for tasks that require generating long chains of thought entails substantial annotation and computational resources (Qin et al., 2025). Moreover, these post-training methods often induce capability regressions, with degradation in generality and in responses to unseen instructions (Shenfeld et al., 2025). In contrast, a training-free and compute-light alternative is model merging, which extracts parameter differences between fine-tuned and pre-trained models (namely the **task vector**), then combines these task vectors to create a unified model that preserves pre-trained knowledge while incorporating capabilities from multiple tasks (Ilharco et al., 2023). This motivates a central question: whether we can merge the LRM and the Instruction-tuned Model (ITM) to enhance the instruction following while preserving its reasoning capability.

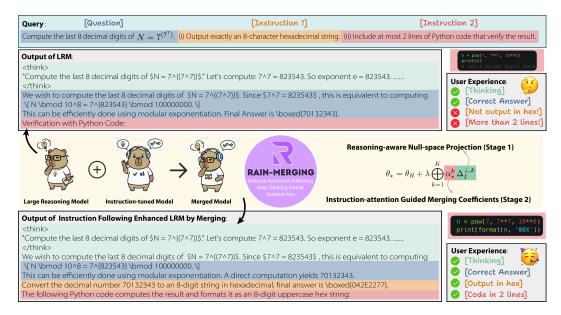


Figure 1: An overview of **RAIN-Merging**. In the case, the LRM arrives at the correct solution but ignores the required format and specific code. To preserve the reasoning structure, we perform training-free merging by combining a task vector projected onto the null space of the thinking format with instruction-attention guided coefficients. The merged model remains correct while satisfying the specified constraints. See **Sec. 3** for details.

We begin with a parameter-space analysis of the task vectors from the LRM and the Instruction-tuned Model (ITM) relative to their shared base. We find that their principal subspaces are nearly orthogonal across key modules, which indicates minimal interference between the two capabilities and suggests that merging is a promising lightweight way to enhance the LRM's instruction following (Ortiz-Jiménez et al., 2023). However, direct merging carries risks. LRMs and ITMs differ fundamentally in output structure: the former explicitly separates "thinking" and "response" with special markers (e.g., R1-style <think>...</think>), whereas the latter provides only a final answer. Traditional data-free merging (Ilharco et al., 2023; Goddard et al., 2024) prunes or scales the task vector purely from parameter-internal statistics to balance domain performance, thereby ignoring output-distribution mismatches and disrupting the LRM's structured reasoning. Recent work (Nobari et al., 2025; Yao et al., 2025; Chopra et al., 2025) has tried to guide merging with forward activations using small calibration sets. Although this introduces data-driven constraints, the lack of an explicit notion of the output mismatch between the two types still makes it difficult to achieve a stable and effective balance between preserving reasoning structure and improving instruction following.

To this end, we propose a two-stage merging strategy that enhances instruction-following capability without sacrificing the thinking format and reasoning performance of the LRM. First, leveraging task-vector orthogonality between the LRM and ITM, we preserve reasoning ability and enforce thinking-format invariance by projecting the ITM task vector into the null space derived from forward features at thinking tokens on a small reasoning-calibration set. This keeps the merged model's reasoning representations aligned with the original LRM and retains structured outputs. Second, while keeping these invariances fixed, we aim to enhance instruction-following performance as much as possible. We improve instruction adherence by estimating per-module importance based on attention outputs over instruction-related spans from a small set of instruction examples. Attention-guided coefficients are then assigned to strengthen instruction-relevant behaviors. We refer to the overall two-stage approach as Reasoning-Aware Instruction-guided Null-space projection Merging (RAIN-Merging) in Fig. 1, which effectively synergizes reasoning and instruction-following performance.

We conduct a systematic evaluation of our proposed method on four instruction-following benchmarks and on nine evaluation benchmarks that cover mathematics, code, STEM, and creative-writing capabilities. The results show that RAIN-Merging not only substantially improves the LRM's instruction-following ability but also maintains reasoning and general capability. Moreover, our method exhibits consistent stability across different model sizes and architectures, and demonstrates enhanced performance in agentic scenarios.

2 Preliminary and Observations

Task Vector. A task vector (Ilharco et al., 2023) characterizes the parameter delta from a base model to a task-specific one. A straightforward way to combine capabilities is **task arithmetic**, which linearly adds such deltas to a base model to obtain a multi-skilled model. This simple approach can work when tasks are compatible. However, for distinct abilities such as reasoning and instruction-following that impose different output structures (Yadav et al., 2023), naive linear addition may cause capability interference and disrupt the representations essential to each domain.

Orthogonality between Reason & Instruction Task Vectors. To examine whether capability interference arises when merging ITM θ_I into LRM θ_R , we take the shared base model θ_B as reference and define the LRM task vector $\Delta_R = \theta_R - \theta_B$ and the ITM task vector $\Delta_I = \theta_I - \theta_B$. We perform singular value decomposition (SVD) within the main forward modules, e.g. attention and FFN, for these two task vectors and evaluate the principal subspace cosine similarity of their principal subspaces. As shown in Fig. 2, A1, A2, the two are nearly orthogonal since their similarities are all < 0.1. Prior studies (Ortiz-Jiménez et al., 2023) indicate that this phenomenon reflects a low degree of coupling between reasoning ability and instruction following in parameter space, which suggests that lightweight task-vector merging strategies can enhance instruction following while preserving the original reasoning performance. More details are in Appendix E.1.

Risks in Thinking Format During Merging. However, orthogonality in parameter space is not sufficient to guarantee that the merged model will retain the LRM's structured output behavior,

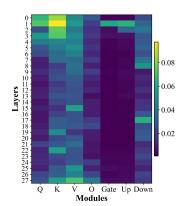


Figure 2: Principal subspace cosine similarity between LRM and ITM task vectors for each layer and submodule. The similarities are consistently low (< 0.1).

since this behavior is determined by downstream propagation and forward features (see Appendix E.1 for proof). In particular, the LRM relies on special tokens such as <think> and </think> to explicitly separate the reasoning segment from the answer segment, and these tokens are crucial in instruction-following tasks. For example, if the model fails to generate the terminator correctly after merging (as Fig. 3), it may conflate the reasoning content with the instruction-compliant response, which can violate constraints such as limits on output length. Therefore, although task-vector orthogonality suggests minimal capability interference, we still need to explicitly constrain the distributional shift of the output structure during merging to preserve the integrity of the reasoning process.

3 Our RAIN-Merging Method

Notation. For notational convenience in later derivations, we flatten model submodules by layer and head with index $k=1,\ldots,K$ as $\theta=\bigoplus_{k=1}^K W^k:=[\operatorname{vec}(W^1)^\top,\ldots,\operatorname{vec}(W^K)^\top]^\top$, where \bigoplus denotes the block-wise concatenation that assembles disjoint parameter blocks into a single coordinate vector. More details of the forward mechanism in Transformer (Vaswani et al., 2017) are in Appendix G.1. Let h^k_t denote the forward input vector at the k-th submodule and the t-th sampled token position. The corresponding linear map of this submodule admits the Kronecker-vectorization form (Koning et al., 1991) with Kronecker product \otimes , identity matrix $\operatorname{diag}(1)$, and vectorization operator $\operatorname{vec}(\cdot)$, as $W^k h^k_t = ((h^k_t)^\top \otimes \operatorname{diag}(1)) \operatorname{vec}(W^k)$. Stacking all sampled positions t row-wise yields the forward feature operator $\Phi^k_{\{t\}}$ and outputs for the k-th submodule:

$$\Phi_{\{t\}}^k := \left[(h_1^k)^\top \otimes \operatorname{diag}(1), \dots, (h_T^k)^\top \otimes \operatorname{diag}(1) \right], \quad W^k h^k = \Phi_{\{t\}}^k \operatorname{vec}(W^k). \tag{1}$$

Optimization Objective. To preserve the original reasoning performance of the LRM as much as possible, we take the reasoning model parameters θ_R as the anchor. We transform the ITM task vector Δ_I through a merging function f to obtain $\Delta = f(\Delta_I)$, and form the merged model $\theta = \theta_R + \Delta$. Our goal is to enhance instruction following without damaging the LRM's thinking format and reasoning performance. We therefore formulate a constrained optimization problem: over the instruction data distribution \mathcal{D}_I , maximize the surrogate objective for instruction following, $\mathcal{J}_I(\theta) \triangleq \mathbb{E}_{x \sim \mathcal{D}_I} \mathbb{E}_{y \sim \pi_\theta(\cdot|x)} \big[\mathrm{IF}(x,y) \big]$, while, over the reasoning data distribution \mathcal{D}_R , constraining the deviation between the model's output distribution within the segment of thinking special tokens

 $\Omega_{\rm think}$ and the reference policy of the original reasoning model θ_R . This constraint is quantified by aggregating the per-step KL divergence within the segment:

$$\mathcal{L}_{\text{think}}(\theta) \triangleq \mathbb{E}_{x \sim \mathcal{D}_R} \, \mathbb{E}_{y \sim \pi_{\theta_R}(\cdot \mid x)} \, \sum_{t \in \Omega_{\text{think}}(x)} \text{KL} \big(\pi_{\theta}(\cdot \mid x, y_{< t}) \, \big\| \, \pi_{\theta_R}(\cdot \mid x, y_{< t}) \big) \,. \tag{2}$$

The overall objective with tolerance δ is then:

$$\max_{\Delta} \mathcal{J}_I(\theta_R + \Delta) \quad \text{s.t.} \quad \mathcal{L}_{\text{think}}(\theta_R + \Delta) \le \delta.$$
 (3)

Noting that \mathcal{J}_I is a surrogate objective for instruction following, referring to a class of functions IF that evaluate instruction alignment. In later we instantiate it with metrics based on instruction-attention alignment or leakage. In addition, motivated by the orthogonality between the LRM and ITM task vectors discussed earlier, we constrain only the conditional distribution in the segment of thinking special tokens and do not restrict the content generated in the other thinking or response segments, which preserves flexibility for improving instruction-following performance.

Reasoning-aware Null-space Projection (Stage 1). To satisfy the KL constraint on the segment of thinking special tokens, we try to seek a parameter subspace that preserves the thinking format. Intuitively, if we view the forward inputs at the thinking positions as a "measurement" of the reasoning style, then any parameter perturbation that is unresponsive under this measurement will not change the model's thinking pattern. This idea corresponds to projecting the perturbation onto the **null space** (Wang et al., 2021) of the forward feature operator $\Phi = \text{blkdiag}(\Phi^1, \dots, \Phi^K)$ (blkdiag denotes the block-diagonal matrix), namely $\mathcal{N}(\Phi) = \{v : \Phi v = 0\}$, as illustrated in **Fig. 3** (a). Such a null space projection keeps the token-level forward features at the thinking positions invariant. Formally, for each submodule k, we construct the least-squares orthogonal projector $P^{\perp}(\cdot)$ using the forward feature operator $\Phi^k_{\Omega_{\text{think}}}$ built from thinking special token indexs Ω_{think} to form the null space:

$$P^{\perp}(\Phi_{\Omega_{\text{think}}}^{k}) = \text{diag}(1) - \Phi_{\Omega_{\text{think}}}^{k} {}^{\top} \left(\Phi_{\Omega_{\text{think}}}^{k} \Phi_{\Omega_{\text{think}}}^{k} {}^{\top} \right)^{+} \Phi_{\Omega_{\text{think}}}^{k}, \tag{4}$$

where $(\cdot)^+$ denotes Moore-Penrose pseudoinverse. And then project the ITM submodule task vector Δ_I^k by submodule-wise and stack them to form the overall projected task vector to satisfy the null space constraint:

$$\operatorname{vec}\left(\Delta_{I}^{\perp,k}\right) = P^{\perp}(\Phi_{\Omega_{\text{think}}}^{k}) \operatorname{vec}\left(\Delta_{I}^{k}\right) \Rightarrow \Phi_{\Omega_{\text{think}}} \operatorname{vec}\left(\Delta_{I}^{\perp}\right) = 0, \text{ where } \Delta_{I}^{\perp} = \bigoplus_{k=1}^{K} \Delta_{I}^{\perp,k}. \tag{5}$$

This projection keeps the merged model's intermediate representations and even the final logits at the thinking special tokens close to those of the anchor model. To verify its effectiveness in preserving the thinking format, we analyze a second-order expansion of the softmax KL divergence and show that the task vector after null-space projection satisfies the KL constraint on the special token output distribution in Eq. (2). This yields the following **Prop.** 1 (proof is in Appendix E.2):

Proposition 1. Let the logits of sample x at thinking special tokens $t \in \Omega_{think}(x)$ be $z_{\theta}(x,t)$, and let $\pi_{\theta}(\cdot \mid x, y_{\leq t}) = \operatorname{softmax}(z_{\theta}(x,t))$. By a second-order approximation of the softmax–KL divergence with a uniform upper bound, for any perturbation u,

$$\text{KL}(\text{softmax}(z+u) \parallel \text{softmax}(z)) \le \frac{1}{8} \|u\|_2^2 + O(\|u\|_2^3).$$
 (6)

Assuming the model's intermediate representations are Lipschitz continuous and bounded, there exist constants $C_1, C_2 > 0$ such that for $u(x,t) = z_{\theta_R + \Delta}(x,t) - z_{\theta_R}(x,t)$, we have:

$$||u(x,t)||_2 \le C_1 ||\Phi||_2 + C_2 ||\Delta||_2^2.$$
 (7)

Substituting the projected vector $\Delta_I^{\perp} = \bigoplus_{k=1}^K \Delta_I^{\perp,k}$ and the condition Φ vec $(\Delta_I^{\perp}) = 0$ yields:

$$\mathcal{L}_{think}(\theta_R + \Delta_I^{\perp}) \le \frac{1}{8} \, \mathbb{E}_{x,t} [\|u(x,t)\|_2^2] + O(\mathbb{E}_{x,t} \|u(x,t)\|_2^3) = O(\|\Delta_I^{\perp}\|_2^2) \approx 0.$$
 (8)

Therefore, null-space projection in **Eq.** (5) approximately removes the thinking format constraint in the original objective and reduces the original optimization objective **Eq.** (3) to:

$$\max_{\Delta^{\perp}} \mathcal{J}_I(\theta_R + \Delta^{\perp}), \text{ where } \Delta^{\perp} = f(\Delta_I^{\perp}).$$
 (9)

217

218

219

220

221222223

224

225

226

227 228

229

230

231

232

235

237

238

239

240

241 242

243

244

245

246

247

249

250

251

253 254

255

256

257

258

259

260261

262

264

265

266267

268

Figure 3: Two stages of our **RAIN-Merging** pipeline. (a) For each submodule, the ITM task vector is projected onto the null space preventing shifts in thinking format. (b) Given the instruction calibration set, we compute the instruction-attention score from attention outputs to obtain merging coefficients.

With the thinking-format constraint relaxed, we next focus on strengthening the task vector's effect on instruction following.

Instruction-attention Guided Merging Coefficients (Stage 2). To enhance the performance gain of the ITM task vector during merging, we seek a suitable gradient-free surrogate objective to instantiate \mathcal{J}_I . Prior studies (Guardieiro et al., 2025) suggest that failures in instruction following often stem from insufficient conditioning on the instruction span during decoding: attention does not sufficiently focus on instruction-relevant tokens and instead leaks to unrelated regions. A simple remedy is to amplify attention outputs on the instruction span at decoding time, which can remarkably improve instruction following. This approach, however, requires pre-identifying the instruction span, and excessive amplification may cause the model to ignore other necessary content. Motivated by this, we hypothesize that different layers and heads exhibit heterogeneous response behavior to instructions. Consequently, on the null-space–projected task vector $\Delta_I^{\perp,k}$, we introduce per-module scaling coefficients $\alpha = \{\alpha^k\} \in \mathbb{R}_+^K$ and reparameterize the merged model as $\theta(\alpha) = \theta_R + \bigoplus_{k=1}^K \alpha^k \Delta_I^{\perp,k}$ to instantiate merging function f. Given that attention outputs are directly coupled to the self-attention mechanism, we first focus on the merging coefficients of these submodules, as $\tilde{\alpha} = \{\alpha^k\} \in \mathbb{R}_+^K$, where k denotes the self-attention submodule index. Our central intuition is that an ideal merge should yield stronger attention responses on instruction-relevant spans (high alignment) while maintaining low attention activation on instruction-irrelevant content (low leakage). To translate this intuition into measurable quantities, we formalize the model's forward computation as follows and in Fig. 3 (b). Let $\operatorname{Att}^k(x,\tilde{\alpha})[t,\tau]$ denote the attention output of the merged model with $\tilde{\alpha}$ at head \tilde{k} from token position t to τ . For an instruction-following sample $x \sim \mathcal{D}_I$, we define the per-sample normalized alignment a and leakage u metrics for head k:

$$\underbrace{a^{\tilde{k}}(x,\tilde{\alpha})}_{\text{alignment}} := \sum_{t \in \mathcal{I}(x)} \sum_{\tau \in \mathcal{R}(x)} \frac{\operatorname{Att}^{\tilde{k}}(x,\tilde{\alpha})[t,\tau]}{|\mathcal{I}(x)| |\mathcal{R}(x)|}, \underbrace{u^{\tilde{k}}(x,\tilde{\alpha})}_{\text{leakage}} := \sum_{t \in \mathcal{I}(x)} \sum_{\tau \in \mathcal{U}(x)} \frac{\operatorname{Att}^{\tilde{k}}(x,\tilde{\alpha})[t,\tau]}{|\mathcal{I}(x)| |\mathcal{U}(x)|}. \tag{10}$$

where $\mathcal{I}(x)\subset\{1,\dots,T\}$ represents the index set of instruction tokens that encodes the task description, formatting rules, constraints, and any examples in the query span. Likewise, $\mathcal{R}(x)$ denotes the set of output tokens whose content is directly constrained by the instruction in the response span, and $\mathcal{U}(x)$ the set of output tokens unrelated to the instruction. Taking expectations over instruction-following samples \mathcal{D}_I and heads \tilde{k} yields averaged alignment $\bar{a}(\tilde{\alpha}) = \sum_{\tilde{k}} \mathbb{E}_{x \sim \mathcal{D}_I}[a^{\tilde{k}}(x, \tilde{\alpha})]$ and averaged leakage $\bar{u}(\tilde{\alpha}) = \sum_{\tilde{k}} \mathbb{E}_{x \sim \mathcal{D}_I}[u^{\tilde{k}}(x, \tilde{\alpha})]$. We seek merging coefficients that achieve *high alignment* and *low leakage*. Accordingly, we combine the two metrics into a single **instruction-attention score** $\mathcal{J}_I^{\text{Proxy}}$ with trade-off hyperparameter $\rho > 0$, instantiating the surrogate objective in the reduced problem **Eq. (9)** then yields:

$$\boxed{\max_{\tilde{\alpha}} \mathcal{J}_{I}^{\text{Proxy}}(\tilde{\alpha}) := \bar{a}(\tilde{\alpha}) - \rho \bar{u}(\tilde{\alpha}).}$$
(11)

Quadratic Approximation of Instruction-attention Score. Although this objective is differentiable and could be optimized by gradient descent, we adopt a forward-pass approximation to reduce computation. Initialize at the directly merged point after projection, $\tilde{\alpha}_{(0)} \equiv 1$. Perform a second-

order Taylor expansion of $\mathcal{J}_{I}^{\text{Proxy}}(\tilde{\alpha})$ around $\tilde{\alpha}_{(0)}$:

$$\mathcal{J}_{I}^{\text{Proxy}}(\tilde{\alpha}) \approx \mathcal{J}_{I}^{\text{Proxy}}(\tilde{\alpha}_{(0)}) + \nabla_{\tilde{\alpha}} \mathcal{J}_{I}^{\text{Proxy}}(\tilde{\alpha}_{(0)})^{\top} (\tilde{\alpha} - \tilde{\alpha}_{(0)}) + \frac{1}{2} (\tilde{\alpha} - \tilde{\alpha}_{(0)})^{\top} H (\tilde{\alpha} - \tilde{\alpha}_{(0)}), \quad (12)$$

where $H=\nabla^2_{\tilde{\alpha}}\mathcal{J}_I^{\text{Proxy}}(\tilde{\alpha}_{(0)})$ is the Hessian. Writing $g=\nabla_{\tilde{\alpha}}\mathcal{J}_I^{\text{Proxy}}(\tilde{\alpha}_{(0)})$ and ignoring the constant term in Eq. (11), we obtain the quadratic surrogate:

$$\mathcal{J}_{I}^{\text{quad}}(\tilde{\alpha}) = g^{\top}(\tilde{\alpha} - \tilde{\alpha}_{(0)}) + \frac{1}{2}(\tilde{\alpha} - \tilde{\alpha}_{(0)})^{\top}H(\tilde{\alpha} - \tilde{\alpha}_{(0)}). \tag{13}$$

① For first-order term g, if we restrict $\tilde{\alpha}$ to small deviations near $\tilde{\alpha}_{(0)}$ and adopt a linear approximation of alignment and leakage on merging coefficients, the per-head gradient can be estimated as:

$$g^{\tilde{k}} = \left. \frac{\partial \mathcal{J}_{I}^{\text{Proxy}}(\tilde{\alpha})}{\partial \tilde{\alpha}^{\tilde{k}}} \right|_{\tilde{\alpha}(0)} \approx \left. \frac{\partial \bar{a}(\tilde{\alpha})}{\partial \tilde{\alpha}^{\tilde{k}}} - \rho \, \frac{\partial \bar{u}(\tilde{\alpha})}{\partial \tilde{\alpha}^{\tilde{k}}} \approx \mathbb{E}_{x \sim \mathcal{D}_{I}} \left[a^{\tilde{k}}(x, \tilde{\alpha}_{(0)}) - \rho \, u^{\tilde{k}}(x, \tilde{\alpha}_{(0)}) \right], \tag{14}$$

which replaces partial derivatives with the current metric values. In practice, this approximately scales the contribution of each head to instruction versus non-instruction attention mass, consistent with the intuition behind attention amplification.

② For second-order term H, to avoid the cost of computing the Hessian for large models, we adopt a diagonal approximation that limits the step size, $\tilde{H}^{\tilde{k}} = \mathrm{diag}(1) + \mathbb{E}_{x \sim \mathcal{D}_I}[u^{\tilde{k}}(x, \tilde{\alpha}_{(0)})]$, where the second term imposes a stronger quadratic penalty on heads with higher leakage, thereby limiting their amplification. Substituting the approximations into the quadratic objective, dropping $\tilde{\alpha}_{(0)}$ for simplicity, and constraining $\tilde{\alpha} \in [\tilde{\alpha}_I, \tilde{\alpha}_u]^{\tilde{K}}$ to prevent scaling instability, we obtain a closed-form solution to the convex quadratic program:

$$\max_{\tilde{\alpha} \in [\tilde{\alpha}_l, \tilde{\alpha}_u]^{\tilde{K}}} \left(g^{\top} \tilde{\alpha} - \frac{1}{2} \tilde{\alpha}^{\top} \tilde{H} \tilde{\alpha} \right) \quad \Rightarrow \quad \tilde{\alpha}_{\star}^{\tilde{k}} = \operatorname{clip}_{[\tilde{\alpha}_l, \tilde{\alpha}_u]} \left(\frac{g^{\tilde{k}}}{\tilde{H}^{\tilde{k}}} \right), \tag{15}$$

where $\tilde{H} = \mathrm{diag} \big(\tilde{H}^{\tilde{k}} \big)$ and $\mathrm{clip}_{[a,b]}(\cdot)$ clips to the interval [a,b]. Thus, by a second-order expansion with engineering approximations and using only forward attention statistics in a gradient-free manner, we approximate the optimal merging coefficients $\tilde{\alpha}_{\star}$ of self-attention submodules that increase instruction alignment while controlling attention leakage to non-instruction content. For modules shared across attention heads, such as the feed-forward network (FFN), we set the layer-wise coefficient to the average over heads. Aggregating the coefficients for all submodules yields the complete instruction attention guided merging coefficients $\alpha_{\star} = \{\alpha_{\star}^{k}\}$.

Combined to Our Two-stage Merging Method. We chain "Reasoning-aware Null-space Projection (Stage 1)" with "Instruction-attention Guided Merging Coefficients (Stage 2)" to propose a fully gradient-free merging pipeline, termed *Reasoning-Aware Instruction-attention guided Null-space projection Merging* (RAIN-Merging) as Fig. 3. Our method addresses the challenge in the original optimization problem of Eq. (3), improving instruction following while preserving the reasoning structure after merging. The final merged model is:

$$\theta_{\star} = \theta_R + \lambda \bigoplus_{k=1}^K \alpha_{\star}^k \Delta_I^{\perp,k},$$
(16)

where λ is a global scaling coefficient that controls the merging strength. The entire procedure only relies on forward-feature extraction and attention statistics, and does not require gradient-based updates. RAIN-Merging offers a low-cost, interpretable path to strengthen instruction following in LRMs, filling the gap left by costly SFT.

Implementation details. To balance compute and storage efficiency, we merge only the core modules that are most sensitive to attention outputs, namely the Q, K, V, O, and FFN parameters. In Stage 1, we sample 150 examples from the Mixture-of-Thoughts (Face, 2025) dataset distilled from DeepSeek-R1 (Guo et al., 2025) from to form the reasoning calibration set. In Stage 2, we an instruction calibration set obtained by distilling DeepSeek-R1 on IFEval (Zhou et al., 2023b), followed by LLM-as-Judge filtering and manual screening, resulting a total of 365 samples. More details of implementation, complete algorithm pseudocode, calibration set construction, and ablation studies are provided in Appendix G, F, H, and J.3.

Table 1: Comprehensive comparison of instruction following and reasoning & general capabilities. We merge Qwen2.5-7B-Instruct (ITM) into DeepSeek-R1-Distill-Qwen-7B (LRM) and compare our RAIN-Merging against multiple merging methods as well as SFT trained on the same calibration data. "Avg." denotes the average over all subsets. "RT" reports the run-time for merging or training in minutes. The best and second-best results are highlighted in **bold** and <u>underlined</u>, respectively.

		Instru	ction Fol	lowing			Reaso	ning &	General		
Method	IFEval	CELLO	Info Bench	Complex Bench	Avg.	Math	GPQA	Aider	Arena- Hard-v2	Avg.	RT
ITM	70.43	19.15	78.49	43.63	52.92	47.27	29.80	33.33	62.86	43.32	_
LRM	55.45	16.59	71.73	32.72	44.12	64.75	44.44	29.63	65.29	51.03	_
SFT	62.48	17.11	68.58	32.15	45.08	62.07	41.92	28.89	64.67	49.39	120.32
				Data-j	ree Merg	ing					
Task Arithmetic	60.44	16.97	73.07	33.34	45.96	62.57	42.93	26.67	64.53	49.17	0.93
SLERP	58.96	17.56	72.18	34.93	45.95	64.22	42.93	31.85	65.29	51.07	1.12
Karcher	62.11	17.99	73.16	34.06	46.83	63.82	48.99	30.77	66.13	52.33	1.20
TIES	58.60	18.48	73.91	34.40	46.35	64.85	46.46	32.59	63.47	51.84	1.18
DARE-TIES	60.81	17.88	73.33	33.49	46.38	65.46	47.98	29.63	64.17	51.80	2.21
				Data-dep	endent M	erging					
ACM-TIES	59.33	16.45	72.44	33.75	45.50	65.92	45.96	32.59	62.00	51.80	12.45
LEWIS-TIES	60.44	17.41	72.67	34.40	46.23	64.57	48.99	31.11	64.80	52.37	16.60
AIM-TIES	62.78	17.93	73.11	34.28	47.02	64.26	<u>49.49</u>	33.33	63.64	52.68	18.51
RAIN-Merging	63.22	<u>19.03</u>	<u>74.53</u>	<u>35.66</u>	<u>48.11</u>	68.75	54.55	33.33	<u>65.73</u>	55.59	20.96

4 EXPERIMENTS

In this section, we empirically investigate three research questions:

- **RQ1** (Effectiveness and Efficiency). Compared with baseline methods, can RAIN-Merging improve instruction-following while maintaining reasoning capabilities, maintaining the computational and memory efficiency characteristic of gradient-free approaches? (**Tab. 1** and **Fig. 4**)
- **RQ2** (Scalability). How well does RAIN-Merging scale across models of varying sizes and architectures, and does it perform effectively in interactive agentic scenarios? (Tab. 2, 3)
- **RQ3** (**Ablation**). What roles do the two stages of RAIN-Merging play? Specifically, does Stage 1 preserve the format of thinking segments and the output distribution, and does Stage 2 enhance instruction-attention scores? (**Tab. 4** and **Fig. 5**, **Fig. 6**)

4.1 EXPERIMENTAL SETUP

We begin with a brief overview of the benchmarks, models, and baselines used in our experiments. Additional details on experimental settings, benchmarks and evaluation metrics, and hyperparameters are provided in Appendix I.

Benchmarks. To comprehensively assess instruction following, we use four mainstream benchmarks: IFEval (Zhou et al., 2023b), CELLO (He et al., 2024), InfoBench (Qin et al., 2024), and ComplexBench (Wen et al., 2024). To comprehensively evaluate reasoning and general capabilities, we use nine benchmarks: Mathematical reasoning is evaluated by aggregating results from six benchmarks, as Math. We also measure performance on code editing (Aider (Aider, 2024)), STEM (GPQA (Rein et al., 2024)), and creative writing (Arena-Hard-v2 (Li et al., 2024)) to reflect general and reasoning capabilities. For agentic scenarios, we use ALFWorld (Shridhar et al., 2021) and WebShop (Yao et al., 2022), two realistic multi-turn interactive tasks, to evaluate how well the model integrates reasoning and instruction following to solve complex problems.

Models. We evaluate RAIN-Merging on models of different sizes and architectures: DeepSeek-R1-Distill-Qwen-1.5B/7B/14B (Guo et al., 2025) (LRM) and Qwen2.5-1.5B/7B¹/14B-Instruct (Yang et al., 2025) (ITM), as well as the Llama family (Dubey et al., 2024) using DeepSeek-R1-Distill-Llama-8B (LRM), its instruction-tuned counterpart Llama-3.1-8B-Instruct (ITM).

¹Although DeepSeek-R1-Distill-Qwen-1.5B/7B are trained from Qwen2.5-Math-1.5B/7B (Yang et al., 2024a), we find that Qwen2.5-Math-1.5B/7B-Instruct do not outperform the distilled LRMs in instruction following. We therefore select the stronger instruction followers, Qwen2.5-1.5B/7B-Instruct, as ITMs.

Table 2: Merging performance and gains of RAIN-Merging across model three scales and two architectures. We merge the corresponding ITM into the LRM with base models: Qwen2.5-1.5B, Llama-3.1-8B, and Qwen2.5-14B. "Avg." denotes the average over all subsets. For each scale, the subsequent "(gain)" row reports the relative improvement of our method over the LRM, highlighted in green.

	Instruction Following					Reasoning & General				
Model	IFEval	CELLO	Info Bench	Complex Bench	Avg.	Math	GPQA	Aider	Arena- Hard-v2	Avg.
Qwen2.5-1.5B-Instruct	36.78	19.04	64.76	27.83	37.10	31.77	25.76	16.30	38.45	28.07
DeepSeek-R1-Distill-Qwen-1.5B	39.00	16.03	55.29	21.54	32.97	41.62	29.29	14.07	39.73	31.18
Qwen2.5-1.5B-RAIN-Merging	41.59	16.51	58.18	23.62	34.97	45.87	33.33	14.81	40.93	33.74
(gain)	+6.64%	+2.98%	+5.23%	+9.63%	+6.09%	+10.21%	+13.79%	+5.26%	+3.02%	+8.20%
Llama-3.1-8B-Instruct	68.58	27.21	78.67	38.47	53.23	35.59	25.25	34.07	72.23	41.79
DeepSeek-R1-Distill-Llama-8B	58.41	17.78	73.33	38.38	46.97	60.21	38.38	27.41	71.93	49.48
Llama-3.1-8B-RAIN-Merging	63.77	18.84	77.38	38.93	49.73	61.95	43.94	30.37	77.07	53.33
(gain)	+9.18%	+5.99%	+5.52%	+1.42%	+5.86%	+2.89%	+14.47%	+10.81%	+7.15%	+7.78%
Qwen2.5-14B-Instruct	79.85	20.13	83.38	44.19	56.89	52.73	36.87	37.04	74.40	50.29
DeepSeek-R1-Distill-Qwen-14B	71.35	18.71	81.33	40.68	53.02	72.31	57.07	33.33	80.67	60.85
Qwen2.5-14B-RAIN-Merging	76.71	19.57	84.13	44.63	56.26	74.58	57.58	40.00	86.25	64.60
(gain)	+7.51%	+4.58%	+3.44%	+9.69%	+6.11%	+3.13%	+0.88%	+20.00%	+6.92%	+6.17%

Baselines. We include several data-free, task-vector based merging methods: Task Arithmetic (Ilharco et al., 2023), SLERP (Biship, 2007; Goddard et al., 2024), Karcher (Nielsen & Bhatia, 2013; Goddard et al., 2024), TIES (Yadav et al., 2023), and DARE (Yu et al., 2024). We also compare with data-dependent, activation-based merging approaches that leverage small calibration sets, including ACM (Yao et al., 2025), LEWIS (Chopra et al., 2025), and AIM (Nobari et al., 2025). To strengthen baseline performance, we apply TIES on top of other merging baselines as in previous work (Wu et al., 2025). In addition, we report a training baseline using SFT on the same calibration data.

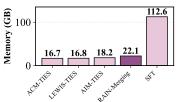
4.2 RESULTS

Performance Comparison with Baseline Methods. (RQ1) As shown in Tab. 1, RAIN-Merging achieves overall gains across both instruction-following and reasoning & general capability evaluations, outperforming all merging baselines. While Task Arithmetic and SFT can improve instruction following to some extent, they typically do so at the cost of reasoning and general capabilities. In contrast, our method consistently surpasses all baselines on instruction-following, mathematical reasoning, and general-capability benchmarks. Our merged LRM trails the ITM slightly on instruction following, indicating room for further improvement. Interstingly, the merged model exhibits stable gains in reasoning and general ability. We hypothesize that stronger instruction adherence improves the quality of the model's internal chain of thought, which yields better reasoning performance. Overall, RAIN-Merging substantially enhances instruction following without sacrificing the LRM's reasoning and general capabilities.

Run-time and Memory Analysis. (**RQ1**) Our method achieves a favorable efficiency trade-off. Its runtime, though slightly above activation-based merging baselines due to null-space computation, is far below SFT (RT in **Tab. 1**). Similarly, while storing hidden features increases memory use compared to other merging methods, its footprint remains much smaller than SFT's (**Fig. 4**). This demonstrates our approach as a highly practical, training-free alternative for enhancing LRMs.

Performance on Models of Different Sizes and Architectures. (**RQ2**) To evaluate the scalability of our method across model sizes and architectures, we conduct experiments on several configurations, including the Qwen2.5 family distilled from DeepSeek-R1 at 1.5B and 14B parameters, and the 8B model built on the Llama 3.1 architecture. As reported in **Tab. 2**, our method consistently enhances instruction-following and reasoning performance, achieving average improvements from 5.86% to 8.20% on LRMs. These results confirm that RAIN-Merging robustly strengthens both instruction adherence and complex reasoning across diverse model sizes and architectures.

Performance in Agentic Scenarios. (**RQ2**) To further assess the practical benefits of improved instruction following, we evaluate the merged model on two representative agentic scenarios, Web-Shop and AlfWorld. As shown in **Tab. 3**, the merged model achieves better performance than the original LRM and ITM in these scenarios, indicating that enhanced instruction understanding and reasoning effectively support multi-turn interaction and complex decision making. These results also demonstrate that our gradient-free approach is effective for increasing the real-world utility of LRMs.



433

434

435

436

437 438

439

440

441 442

443 444

445

446

448

449

450

451

452

453

454 455 456

457

458

459

460

461

462

463

464

465

466

467

468

469 470

471

472

473

474

475

476 477 478

479 480

481

482

483

484

485

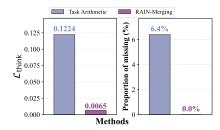
Figure 4: GPU memory usage comparison between different methods under the same configuration as **Tab. 1**.

Table 3: R1-Distill-Qwen-7B (LRM).

Model	ALFWorld	WebShop
ITM	17.50	10.45
LRM	22.00	26.63
RAIN-Merging	25.00	29.42

Performance of Table 4: Performance of combining RAIN-Merging in agent set- Stage 1 and Stage 2 of our method, tings. We merge Qwen2.5-7B- under the same setup as Tab. 1. "I Instruct (ITM) into DeepSeek- Avg." and "R Avg." denote the average performance on instruction-following and reasoning & general benchmarks.

Method	I Avg.	R Avg.
RAIN-Merging w/o Stage 2	46.58	54.92
RAIN-Merging	48.11	55.59



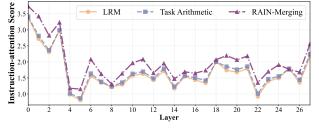


Figure 5: \mathcal{L}_{think} in **Eq.** (2) (left) on the reasoning calibration validation set, and the proportion of generations missing the closing </think> token (right) on IFEval under the same configuration as **Tab. 1**.

Figure 6: Layer-wise instruction attention score (alignment – leakage). Per-layer scores on IFEval instruction examples; higher is better. We compare the unmerged LRM, Task Arithmetic, and RAIN-Merging when merging Qwen2.5-7B-Instruct (ITM) into DeepSeek-R1-Distill-Qwen-7B (LRM).

Ablation Study of Stage 1 and Stage 2. (RQ3) We investigate the contribution of the two components in RAIN-Merging, shown in **Tab. 4**. When only Stage 1 is applied without Stage 2, the merged model retains reasoning and general capabilities while achieving competitive instruction-following performance. Incorporating Stage 2 further boosts instruction-following performance, as reflected by the score increase from 46.58 to 48.11. These results demonstrate that both stages play critical and complementary roles. Together, they form our gradient-free pipeline that enhances instructionfollowing ability without compromising reasoning performance of LRMs.

Effectiveness of Null-space Projection. (RQ3) To assess how our null-space projection in Stage 1 preserves thinking formats, we evaluate its impact on thinking special token distributions and resulting generation outputs. We measure the KL divergence near thinking tokens as in Eq. (2) and the rate of missing </think> tokens. Results Fig. 5 show that Task Arithmetic substantially alters the distribution $(\mathcal{L}_{think} = 0.1224)$ and results in 6.4% missing </think> tokens, violating the output format. Our approach, in contrast, only induces minimal change ($\mathcal{L}_{think} = 0.0065$) and ensures no missing tokens (0.0%). These findings indicate that null-space projection successfully protects thinking formats.

Effectiveness of Merging Coefficients. (RQ3) To validate the merging coefficients, we compare the Instruction-Attention Score in Eq. (11) across layers before and after merging under different methods. As shown in Fig. 6, instruction-attention guided coefficients in Stage 2 enable RAIN-Merging to consistently outperform both the LRM and Task Arithmetic, exhibiting a higher alignment and lower leakage. This finding suggests that our weighted reparameterization of merging submodules enhances activation along instruction-aware pathways while slightly suppressing leakage, which improves instruction following without altering the original reasoning pattern.

5 CONCLUSION

We propose RAIN-Merging, a gradient-free method to enhance instruction following in LRMs while preserving their structured reasoning outputs. By projecting the instruction task vector onto the null space of the thinking format and scaling it by instruction-attention guided coefficients, RAIN-Merging achieves a balance between instruction following and reasoning structure preservation. The method is evaluated on instruction-following, reasoning & general capability, agentic benchmarks, showing that RAIN-Merging not only substantially improves the LRM's instruction-following ability but also brings gains in reasoning and general capability across several settings.

REFERENCES

486

487 488

489

490

491 492

493 494

495

496

497

498

499

500

501

502

504

505

506

507

508

509

510

511

512

513

514

515

516 517

518

519

520

521 522

523 524

525

526 527

528

529

530

531

532

533 534

535

536 537

- Aider. o1 tops aider's new polyglot leaderboard, 2024.
 - Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating Ilms on uncontaminated math competitions. *arXiv preprint arXiv:2505.23281*, 2025.
 - Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon, and Joydeep Ghosh. Clustering with bregman divergences. *Journal of Machine Learning Research (JMLR)*, 2005.
 - Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, Ido Shahaf, Oren Tropp, Ehud Karpas, Ran Zilberstein, Jiaqi Zeng, Soumye Singhal, Alexander Bukharin, Yian Zhang, Tugrul Konuk, Gerald Shen, Ameya Sunil Mahabaleshwarkar, Bilal Kartal, Yoshi Suhara, Olivier Delalleau, Zijia Chen, Zhilin Wang, David Mosallanezhad, Adi Renduchintala, Haifeng Qian, Dima Rekesh, Fei Jia, Somshubra Majumdar, Vahid Noroozi, Wasi Uddin Ahmad, Sean Narenthiran, Aleksander Ficek, Mehrzad Samadi, Jocelyn Huang, Siddhartha Jain, Igor Gitman, Ivan Moshkov, Wei Du, Shubham Toshniwal, George Armstrong, Branislav Kisacanin, Matvei Novikov, Daria Gitman, Evelina Bakhturina, Jane Polak Scowcroft, John Kamalu, Dan Su, Kezhi Kong, Markus Kliegl, Rabeeh Karimi, Ying Lin, Sanjeev Satheesh, Jupinder Parmar, Pritam Gundecha, Brandon Norick, Joseph Jennings, Shrimai Prabhumoye, Syeda Nahida Akter, Mostofa Patwary, Abhinav Khattar, Deepak Narayanan, Roger Waleffe, Jimmy Zhang, Bor-Yiing Su, Guyue Huang, Terry Kong, Parth Chadha, Sahil Jain, Christine Harvey, Elad Segal, Jining Huang, Sergey Kashirsky, Robert McQueen, Izzy Putterman, George Lam, Arun Venkatesan, Sherry Wu, Vinh Nguyen, Manoj Kilaru, Andrew Wang, Anna Warno, Abhilash Somasamudramath, Sandip Bhaskar, Maka Dong, Nave Assaf, Shahar Mor, Omer Ullman Argov, Scot Junkin, Oleksandr Romanenko, Pedro Larroy, Monika Katariya, Marco Rovinelli, Viji Balas, Nicholas Edelman, Anahita Bhiwandiwalla, Muthu Subramaniam, Smita Ithape, Karthik Ramamoorthy, Yuting Wu, Suguna Varshini Velury, Omri Almog, Joyjit Daw, Denys Fridman, Erick Galinkin, Michael Evans, Katherine Luna, Leon Derczynski, Nikki Pope, Eileen Long, Seth Schneider, Guillermo Siman, Tomasz Grzegorzek, Pablo Ribalta, Monika Katariya, Joey Conway, Trisha Saar, Ann Guan, Krzysztof Pawelec, Shyamala Prayaga, Oleksii Kuchaiev, Boris Ginsburg, Oluwatobi Olabiyi, Kari Briski, Jonathan Cohen, Bryan Catanzaro, Jonah Alben, Yonatan Geifman, Eric Chung, and Chris Alexiuk. Llama-nemotron: Efficient reasoning models, 2025.
 - Christopher M Biship. *Pattern recognition and machine learning (information science and statistics)*. Springer, 2007.
 - Dankmar Böhning. Multinomial logistic regression algorithm. *Annals of the institute of Statistical Mathematics (AISM)*, 1992.
 - Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
 - Lang Cao, Chao Peng, Renhong Chen, Wu Ning, Yingtian Zou, and Yitong Li. Step guided reasoning: Improving mathematical reasoning using guidance generation and step reasoning. *arXiv* preprint arXiv:2410.19817, 2024.
 - Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv* preprint arXiv:2211.12588, 2022.
 - Zina Chkirbene, Ridha Hamila, Ala Gouissem, and Unal Devrim. Large language models (llm) in industry: A survey of applications, challenges, and trends. In *IEEE International Conference on Smart Communities: Improving Quality of Life using AI, Robotics and IoT (HONET)*, 2024.
 - Hetarth Chopra, Vidhi Rambhia, and Vikram Adve. Lewis (layer wise sparsity)—a training free guided model merging approach. *arXiv preprint arXiv:2503.03874*, 2025.
 - Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

541

542

543

544

546

547

548

549

550 551

552

553

554

555

556

558

559

560 561

562

563

565

566 567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

588

590

592

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407*, 2024.

- Hugging Face. Open r1: A fully open reproduction of deepseek-r1, 2025.
- Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. Alphaedit: Null-space constrained knowledge editing for language models. In *International Conference on Learning Representations (ICLR)*, 2025.
- Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.
- Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George J. Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- Tingchen Fu, Jiawei Gu, Yafu Li, Xiaoye Qu, and Yu Cheng. Scaling reasoning, losing control: Evaluating instruction following in large reasoning models. *arXiv preprint arXiv:2505.14810*, 2025.
- Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: program-aided language models. In *International Conference on Machine Learning (ICML)*, 2023.
- Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vlad Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. Arcee's mergekit: A toolkit for merging large language models. *arXiv preprint arXiv:2403.13257*, 2024.
- Vitoria Guardieiro, Adam Stein, Avishree Khare, and Eric Wong. Instruction following by boosting attention of large language models. *arXiv preprint arXiv:2506.13734*, 2025.
- Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Yu Wu, YK Li, et al. Deepseek-coder: When the large language model meets programming—the rise of code intelligence. *arXiv preprint arXiv:2401.14196*, 2024.

DeepSeek-AI: Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Levi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li,

- Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.
 - Qianyu He, Jie Zeng, Wenhao Huang, Lina Chen, Jin Xiao, Qianxi He, Xunzhe Zhou, Jiaqing Liang, and Yanghua Xiao. Can large language models understand real-world complex instructions? In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2024.
 - Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv* preprint arXiv:2103.03874, 2021.
 - Gabriel Ilharco, Marco Túlio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *International Conference on Learning Representations (ICLR)*, 2023.
 - Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.
 - Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint* arXiv:1412.6980, 2014.
 - Ruud H Koning, Heinz Neudecker, and Tom Wansbeek. Block kronecker products and the vecb operator. *Linear Algebra and Its Applications*, 1991.
 - Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay V. Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.
 - Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*, 2024.
 - Xiaomin Li, Zhou Yu, Zhiwei Zhang, Xupeng Chen, Ziji Zhang, Yingying Zhuang, Narayanan Sadagopan, and Anurag Beniwal. When thinking fails: The pitfalls of reasoning for instruction-following in llms. *arXiv preprint arXiv:2505.11423*, 2025.
 - Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations (ICLR)*, 2024.
 - Anton Lozhkov, Hynek Kydlíček, Loubna Ben Allal, Guilherme Penedo, Edward Beeching, Quentin Gallouédec, Nathan Habib, Lewis Tunstall, and Leandro von Werra. Openr1-math-220k, 2025.
 - Yurii Nesterov. Introductory lectures on convex optimization: A basic course. Springer, 2013.
 - Frank Nielsen and Rajendra Bhatia. Matrix information geometry. Springer, 2013.
 - Amin Heyrani Nobari, Kaveh Alimohammadi, Ali ArjomandBigdeli, Akash Srivastava, Faez Ahmed, and Navid Azizan. Activation-informed merging of large language models. *arXiv preprint arXiv:2502.02421*, 2025.
 - Guillermo Ortiz-Jiménez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

- Guilherme Penedo, Anton Lozhkov, Hynek Kydlíček, Loubna Ben Allal, Edward Beeching, Agustín Piqueres Lajarín, Quentin Gallouédec, Nathan Habib, Lewis Tunstall, and Leandro von Werra. Codeforces cots, 2025.
 - Yunjia Qi, Hao Peng, Xiaozhi Wang, Amy Xin, Youfeng Liu, Bin Xu, Lei Hou, and Juanzi Li. Agentif: Benchmarking instruction following of large language models in agentic scenarios. *arXiv* preprint arXiv:2505.16944, 2025.
 - Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. Infobench: Evaluating instruction following ability in large language models. In *Proceedings of the Association for Computational Linguistics (ACL Findings)*, 2024.
 - Yulei Qin, Gang Li, Zongyi Li, Zihan Xu, Yuchen Shi, Zhekai Lin, Xiao Cui, Ke Li, and Xing Sun. Incentivizing reasoning for advanced instruction-following of large language models. arXiv preprint arXiv:2506.01413, 2025.
 - David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *Conference on Language Modeling (COLM)*, 2024.
 - Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations (ICLR)*, 2021.
 - Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv* preprint arXiv:2402.03300, 2024.
 - Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. Rl's razor: Why online reinforcement learning forgets less. *arXiv preprint arXiv:2509.04259*, 2025.
 - Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
 - Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations (ICLR)*, 2021.
 - Pengwei Tang, Yong Liu, Dongjie Zhang, Xing Wu, and Debing Zhang. Lora-null: Low-rank adaptation via null space for large language models. *arXiv preprint arXiv:2503.02659*, 2025.
 - Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
 - Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 2008.
 - Shipeng Wang, Xiaorong Li, Jian Sun, and Zongben Xu. Training networks in null space of feature covariance for continual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
 - Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.
 - Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In Advances in Neural Information Processing Systems (NeurIPS), 2022.
 - Bosi Wen, Pei Ke, Xiaotao Gu, Lindong Wu, Hao Huang, Jinfeng Zhou, Wenchuang Li, Binxin Hu, Wendy Gao, Jiaxing Xu, Yiming Liu, Jie Tang, Hongning Wang, and Minlie Huang. Benchmarking complex instruction-following with multiple constraints composition. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

- Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning (ICML)*, 2022.
- Han Wu, Yuxuan Yao, Shuqi Liu, Zehua Liu, Xiaojin Fu, Xiongwei Han, Xing Li, Hui-Ling Zhen, Tao Zhong, and Mingxuan Yuan. Unlocking efficient long-to-short llm reasoning with model merging. *arXiv preprint arXiv:2503.20641*, 2025.
- Prateek Yadav, Derek Tam, Leshem Choshen, Colin A. Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
- An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024a.
- Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. Model merging in llms, mllms, and beyond: Methods, theories, applications and opportunities. *arXiv preprint arXiv:2408.07666*, 2024b.
- 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.
- Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023a.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023b.
- Yuxuan Yao, Shuqi Liu, Zehua Liu, Qintong Li, Mingyang Liu, Xiongwei Han, Zhijiang Guo, Han Wu, and Linqi Song. Activation-guided consensus merging for large language models. *arXiv* preprint arXiv:2505.14009, 2025.
- Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *International Conference on Machine Learning (ICML)*, 2024.
- Haobo Zhang and Jiayu Zhou. Unraveling lora interference: Orthogonal subspaces for robust model merging. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2025.
- Huaqin Zhao, Zhengliang Liu, Zihao Wu, Yiwei Li, Tianze Yang, Peng Shu, Shaochen Xu, Haixing Dai, Lin Zhao, Gengchen Mai, et al. Revolutionizing finance with llms: An overview of applications and insights. *arXiv preprint arXiv:2401.11641*, 2024.
- Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V. Le, and Ed H. Chi. Least-to-most prompting enables complex reasoning in large language models. In *International Conference on Learning Representations (ICLR)*, 2023a.
- Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023b.

APPENDIX

A ETHICS STATEMENT

This research adheres to the licenses and applicable laws governing upstream open-source models and datasets. RAIN-Merging is developed using publicly available weights and data that permit derivation and redistribution.

Safety. Model merging can introduce "capability or safety drift," such as new biases, jailbreak risks, or shifts in hallucination patterns while improving instruction following. The merged model may produce inaccurate, biased, or inappropriate content. It must not be used directly in high-risk decision-making contexts such as medicine, law, or finance. Any production deployment should include human oversight, operation logging, rate limiting, and compliance review procedures.

Dataset use. We rely only on data authorized for academic reproducibility. During data cleaning, we make every effort to remove personally identifiable information and sensitive content. We also disclose potential dataset biases, coverage gaps, and risks of benchmark contamination.

Societal impact. We caution that generative models may exacerbate information asymmetries, reinforce stereotypes, or be applied to produce misleading content. We firmly oppose misuse and will work with the community to address any identified negative impacts.

B REPRODUCIBILITY STATEMENT

To ensure the reproducibility of our results, we provide the following resources and documentation: all algorithm implementations and experiment scripts will be released anonymously with the **supplementary materials**, accompanied by documentation of key functions and the project directory structure. The calibration datasets used in our experiments will be made available alongside the appendix. Public links are included for all open-source models and datasets used in this work.

C LLM USAGE STATEMENT

We used large language models (LLMs) in the following stages and disclose their roles as follows:

Writing Stage. LLMs (both closed- and open-source) were used only for copyediting and grammar checking, including terminology normalization, syntactic polishing, and formatting. They were not used to generate claims, collect evidence, or construct results.

Benchmark Evaluation. When a benchmark's original paper or community practice requires a closed-source LLM (for example, as a judge or as a baseline), we strictly follow the prescribed protocol and disclose the exact model versions.

Calibration Set Construction. We adopt an LLM-as-Judge procedure for automated filtering and scoring of candidate samples as an initial pass (producing only scores or labels; generated text is not used as a training target). A human second-pass review follows to ensure data quality and compliance. All third-party data and models are used within their licenses, with source links and permission details provided.

D RELATED WORK

Large Reasoning Model. Early studies show that prompting models to explicitly produce intermediate steps during reasoning can substantially improve complex reasoning performance, as in Chain-of-Thought (Wei et al., 2022) and Tree-of-Thought (Yao et al., 2023a). Building on this insight, a new generation of LRMs has shifted toward training paradigms that directly incentivize reasoning with reinforcement learning; for example, OpenAI's o1 series and DeepSeek-R1 report marked advances on tasks in mathematics and code that require extended reasoning (Jaech et al., 2024; Guo et al., 2025). These models typically generate structured "thought processes" or "thinking formats," yet in real applications they often exhibit tension with strict instruction following. Beyond explicit intermediate reasoning such as CoT and ToT, subsequent work further improves reasoning

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826 827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

quality and stability: Self-Consistency samples multiple solution paths and uses majority voting to increase reliability; Least-to-Most decomposes complex problems into subgoals ordered from easy to hard; Program-of-Thoughts and PAL externalize the reasoning into executable programs, decoupling computation from reasoning and substantially reducing arithmetic and procedural errors (Wang et al., 2023; Zhou et al., 2023a; Chen et al., 2022; Gao et al., 2023). In the "reasoning plus acting" paradigm, ReAct interleaves thought traces with tool interactions to mitigate hallucinations, while Reflexion employs language-based self-reflection and memory to iteratively refine policies over multi-turn interactions (Yao et al., 2023b; Shinn et al., 2023). In parallel, LRMs are trained with process-level feedback and reinforcement learning to directly encourage thinking before answering: the o1 system emphasizes large-scale RL and thinking-first training and alignment strategies, and DeepSeek-R1 shows that under weak or no supervision, pure RL (e.g., GRPO) can induce longer and more stable chains of thought (Jaech et al., 2024; Guo et al., 2025). Moreover, process supervision and process reward models (PRMs) provide finer-grained step-level feedback that, compared with outcome supervision of final answers, better cultivates verifiable reasoning chains and test-time expansion (Lightman et al., 2024). RAIN-Merging is complementary to this trajectory: instead of retraining the LRM, we preserve the thinking format at merge time and use structured coefficients to selectively enhance instruction responses, thereby striking a balance between fidelity to the reasoning structure and improved instruction following.

Instruction Following. In the alignment paradigm, InstructGPT systematically improved the stability of "following user intent" through reinforcement learning from human feedback (RLHF), and showed that small instruction-tuned models can achieve strong human preference scores, establishing a foundation for subsequent research on instruction following (Ouyang et al., 2022). For objective evaluation, IFEval targets programmatically verifiable rules, for example, length limits, keywords, formatting, to reduce subjective scoring noise and facilitate reproducibility and fair comparison (Zhou et al., 2023b). CELLO abstracts multi-dimensional attributes from real-world complex instructions, such as multi-step dependencies, format or quantity constraints, and semantic consistency, to characterize where LLMs struggle with complex instruction understanding (He et al., 2024). InfoBench proposes a decomposed metric that breaks a complex instruction into checkable sub-requirements, enabling finer-grained measurement of compliance and error sources (Qin et al., 2024). ComplexBench emphasizes the compositional challenge of multiple simultaneous constraints, systematically testing robustness and trade-offs when many constraints co-occur (Wen et al., 2024). Building on these mainstream benchmarks, this work introduces an instruction-attention-oriented merging strategy: during merging, we quantitatively constrain and amplify the model's responsiveness to instructionrelevant spans while maintaining the stability of its long-chain reasoning format, thereby balancing compliance and an interpretable process.

Model Merging. Parameter-space merging offers a training-free or low-data path for integrating capabilities. Model Soup averages weights from multiple fine-tuned checkpoints to improve out-ofdistribution robustness and overall performance (Wortsman et al., 2022). Task vectors implement additive editing and compositionality by linear arithmetic on weight differences, enabling positive and negative edits as well as multi-task synthesis (Ilharco et al., 2023). TIES-Merging explicitly addresses interfering factors such as resetting parameters with negligible updates and resolving sign conflicts, which mitigates performance degradation caused by parameter-level interference when merging multiple models (Yadav et al., 2023). Community tools and practice are also maturing. MergeKit consolidates and engineers diverse merging algorithms, supporting large-model merging and recipe reproduction under resource constraints, which facilitates methodological comparison and reproducibility (Goddard et al., 2024). Systematic surveys have begun to organize theoretical perspectives, method taxonomies, and application boundaries for merging, providing references for unified terminology, evaluation settings, and future research agendas (Yang et al., 2024b). However, most existing methods focus on average multi-task performance and out-of-distribution robustness, with limited attention to the fidelity of fine-grained functional structures such as the reasoning format, for example, explicit thought traces and process markers. RAIN-Merging targets this gap: during parameter fusion it introduces subspace constraints tied to the "thinking format," and allocates merging coefficients at the per-layer and per-head levels using instruction attention, thereby strengthening instruction following while suppressing structural drift of the original reasoning patterns.

Null Space Projection. Constraint ideas centered on orthogonality and null spaces have been repeatedly validated in continual learning and knowledge editing. OGD projects gradients for new tasks onto the orthogonal complement of the subspace of old tasks, explicitly constraining update

directions to mitigate forgetting (Farajtabar et al., 2020). GPM extracts and maintains "important gradient subspaces" via singular value decomposition, then performs layer-wise orthogonal projection of new gradients to reduce interference across tasks (Saha et al., 2021). For LLM knowledge editing, AlphaEdit projects edit perturbations into the null space of "preserved knowledge" and provides theoretical guarantees on output preservation, which markedly reduces cumulative damage in sequential edits (Fang et al., 2025). In parameter-efficient and mergeable settings, LoRA-Null initializes or constrains the LoRA adaptation subspace using the null space of pretrained representations, alleviating forgetting and improving parallelism and mergeability with other updates (Tang et al., 2025). For multi-task and multi-LoRA model merging, OSRM imposes orthogonalization constraints on task-specific LoRA subspaces before fine-tuning, reducing mutual interference at merge time and improving compatibility (Zhang & Zhou, 2025). Following this line of work, we construct a null-space projection on features tied to the "reasoning format," and combine it with instruction-attention-guided coefficients. The merged model thus preserves structured reasoning outputs while improving adherence to verifiable constraints such as format, length, and enumeration.

E Proof

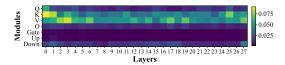


Figure A1: *Principal subspace cosine similarity* between DeepSeek-R1-Distill-Qwen-1.5B (LRM) and Qwen2.5-1.5B-Instruct (ITM) task vectors for each layer and submodule.

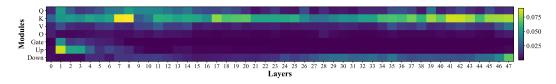


Figure A2: *Principal subspace cosine similarity* between DeepSeek-R1-Distill-Qwen-14B (LRM) and Qwen2.5-14B-Instruct (ITM) task vectors for each layer and submodule.

E.1 Proof of why orthogonal in parameters \neq invariant in outputs

We first describe how we compute orthogonality between **principal parameter subspaces**. Let the two sources be the LRM task vector or weight difference, denoted by R, and the ITM counterpart, denoted by I. For each layer and each linear submodule $W^k \in \mathbb{R}^{d_{\text{out}}^k \times d_{\text{in}}^k}$, we take the top S singular directions (default S=16 in our experiments) and perform SVD:

$$W_R^k = U_R^k \Sigma_R^k (V_R^k)^\top, \qquad W_I^k = U_I^k \Sigma_I^k (V_I^k)^\top.$$
 (A1)

Write $U_{R,S}^k \in \mathbb{R}^{d_{\text{out}}^k \times S}$ for the top-S left singular vectors of U_R^k (similarly $V_{R,S}^k \in \mathbb{R}^{d_{\text{in}}^k \times S}$), and analogously $U_{I,S}^k, V_{I,S}^k$ for source I.

Principal subspace cosine similarity. We focus on the *left* (output-side) principal subspaces and define the alignment matrix

$$A^k = (U_{R,S}^k)^\top U_{I,S}^k \in \mathbb{R}^{S \times S}. \tag{A2}$$

Let $\sigma_1^k, \ldots, \sigma_S^k \in [0,1]$ be the singular values of A^k . They equal the cosines of the principal angles between the two subspaces: $\sigma_i^k = \cos \vartheta_i^k$. We define the **principal subspace cosine similarity** as the mean cosine of principal angles:

$$\cos \Theta_S^k \left(U_{R,S}^k, U_{I,S}^k \right) = \frac{1}{S} \sum_{i=1}^S \sigma_i^k. \tag{A3}$$

Smaller values indicate stronger orthogonality between the sources at that (layer, module) cell.

Empirical observation. Across model sizes and all layers/modules in Qwen2.5-1.5B/7B/14B (Fig. 2, A1, A2), we observe $\cos\Theta_S^k < 0.1$ (with only a few exceptions), indicating that LRM and ITM task vectors are largely *orthogonal in parameter principal directions*. However, as the theory below shows, such parameter-space orthogonality does *not* imply invariance in the output space (i.e., unchanged logits on the thinking format), and thus cannot replace the *null-space projection* constraint used in our method.

Why orthogonal in parameters \neq invariant in outputs. We formalize this issue as the following **Prop. 1** and give a proof with dimension argument.

Proposition 1 (Insufficiency of parameter-space orthogonality for output invariance). For each submodule k, let $\mathcal{U}_{I,S}^k$, $\mathcal{V}_{I,S}^k$ be the S-dimensional instruction-side principal left/right subspaces and define the admissible low-rank parameter perturbation space

$$\mathcal{T}_{I}^{k} = \mathcal{U}_{I,S}^{k} \otimes \mathcal{V}_{I,S}^{k} = \operatorname{span}\left\{\operatorname{vec}(uv^{\top}) : u \in \mathcal{U}_{I,S}^{k}, v \in \mathcal{V}_{I,S}^{k}\right\}, \qquad \mathcal{T}_{I} = \bigoplus_{k=1}^{K} \mathcal{T}_{I}^{k}. \quad (A4)$$

Let J be the Jacobian of the logits on the protected thinking tokens Ω_{think} at the anchor θ_R , with total parameter dimension $D = \dim(\theta)$ and rank $r = \operatorname{rank}(J)$. Then, in generic position,

$$\dim(\mathcal{T}_I \cap \text{Null}(J)) \le \max\{0, KS^2 - r\}. \tag{A5}$$

In particular, if $r > KS^2$, one has $\mathcal{T}_I \cap \text{Null}(J) = \{0\}$ and hence $\mathcal{T}_I \nsubseteq \text{Null}(J)$. Even when $r \leq KS^2$, the inclusion $\mathcal{T}_I \subseteq \text{Null}(J)$ requires a measure-zero alignment and thus almost never holds. Consequently, there exists a nonzero $\Delta \in \mathcal{T}_I$ with $J\Delta \neq 0$, implying

$$\mathcal{L}_{think}(\theta_R + \Delta) = \frac{1}{2} \Delta^{\top} (\mathbb{E}[J^{\top} F J]) \Delta + o(\|\Delta\|^2) > 0, \tag{A6}$$

where F is the Fisher matrix of the softmax.

Proof of Prop. 1. Each module contributes an S-dimensional left subspace and an S-dimensional right subspace; their Kronecker product yields

$$\dim \mathcal{T}_I^k = S \cdot S = S^2$$
 in generic position, (A7)

so, ignoring accidental cross-module dependencies,

$$\dim \mathcal{T}_I = \sum_{k=1}^K \dim \mathcal{T}_I^k = KS^2.$$
 (A8)

By the rank–nullity theorem for $J \in \mathbb{R}^{m \times D}$,

$$\dim \text{Null}(J) = D - r. \tag{A9}$$

For two subspaces $\mathcal{A}, \mathcal{B} \subset \mathbb{R}^D$, a standard upper bound on the intersection dimension states

$$\dim(\mathcal{A} \cap \mathcal{B}) \le \max\{0, \dim \mathcal{A} + \dim \mathcal{B} - D\}. \tag{A10}$$

Setting $A = \mathcal{T}_I$ and B = Null(J) gives

$$\dim(\mathcal{T}_I \cap \text{Null}(J)) \leq \max\{0, KS^2 + (D-r) - D\} = \max\{0, KS^2 - r\}.$$
 (A11)

Hence, if $r > KS^2$, the intersection is trivial and $\mathcal{T}_I \subseteq \text{Null}(J)$ is impossible. Even when $r \leq KS^2$, the full inclusion would require not only dim $\mathcal{T}_I \leq \dim \text{Null}(J)$ but also a non-generic containment (measure-zero alignment) between the two subspaces; thus it almost never holds in generic position.

Finally, since $F \succeq 0$ and $M = \mathbb{E}[J^{\top}FJ] \succeq 0$, any nonzero $\Delta \in \mathcal{T}_I$ with $J\Delta \neq 0$ satisfies $\Delta^{\top}M\Delta > 0$, yielding

$$\mathcal{L}_{\text{think}}(\theta_R + \Delta) = \frac{1}{2} \Delta^{\top} M \Delta + o(\|\Delta\|^2) > 0.$$
 (A12)

In words, orthogonality of principal parameter subspaces does not guarantee first-order invariance of outputs on the thinking format. This is precisely why our Stage 1 imposes a null-space projection constraint (i.e., $\Phi_{\Omega_{\text{think}}} \operatorname{vec}(\Delta^{\perp}) = 0$) to cancel first-order effects.

E.2 Proof of Prop. 1

Proof of Prop. 1. Let $p = \operatorname{softmax}(z) \in \Delta^{V-1}$ and $q = \operatorname{softmax}(z+u)$, where $z \in \mathbb{R}^V$ is the logits vector at a thinking position $t \in \Omega_{\operatorname{think}}(x)$ and $u \in \mathbb{R}^V$ is the perturbation induced by the parameter change.

Step 1: KL as a Bregman divergence of lse and a uniform quadratic bound. Let $lse(z) = log \sum_{i=1}^{V} e^{z_i}$, so that $\nabla lse(z) = softmax(z) = p$ and $\nabla^2 lse(z) = diag(p) - pp^{\top}$. For the multinomial exponential family, the KL divergence equals the Bregman divergence of the log-partition function (Banerjee et al., 2005; Wainwright & Jordan, 2008):

$$\mathrm{KL}(\mathrm{softmax}(z+u) \parallel \mathrm{softmax}(z)) = D_{\mathrm{lse}}(z+u,z) = \mathrm{lse}(z+u) - \mathrm{lse}(z) - \langle \nabla \, \mathrm{lse}(z), u \rangle$$
. (A13)

Using the integral form of the Bregman remainder for a twice differentiable convex f, $D_f(x+h,x)=\int_0^1 (1-s)\,h^\top\nabla^2 f(x+sh)\,h\,ds$, and the fact that for all z the Hessian satisfies $\|\nabla^2 \sec(z)\|_2 \leq \frac{1}{4}$ by positive semidefinite covariance form as in Lemma 1 (Boyd & Vandenberghe, 2004; Böhning, 1992), we obtain

$$KL(\operatorname{softmax}(z+u) \parallel \operatorname{softmax}(z)) = \int_0^1 (1-s) u^\top \nabla^2 \operatorname{lse}(z+su) u \, ds$$

$$\leq \int_0^1 (1-s) \frac{1}{4} \|u\|_2^2 \, ds = \frac{1}{8} \|u\|_2^2. \tag{A14}$$

Equivalently, the second-order Taylor expansion with a third-order remainder yields

$$KL(\operatorname{softmax}(z+u) \parallel \operatorname{softmax}(z)) = \frac{1}{2} u^{\top} \nabla^{2} \operatorname{lse}(z) u + O(\|u\|_{2}^{3}) \leq \frac{1}{8} \|u\|_{2}^{2} + O(\|u\|_{2}^{3}).$$
(A15)

Lemma 1 (Hessian bound for lse). For any $z \in \mathbb{R}^V$ with $p = \operatorname{softmax}(z)$,

$$\nabla^2 \operatorname{lse}(z) = \operatorname{diag}(p) - pp^{\top} \succeq 0, \qquad \|\nabla^2 \operatorname{lse}(z)\|_2 \le \frac{1}{4}. \tag{A16}$$

Step 2: Bounding the logits perturbation via linearization and Lipschitz regularity. Let $J(x,t) \in \mathbb{R}^{V \times d}$ be the Jacobian of $z_{\theta}(x,t)$ w.r.t. θ at $\theta = \theta_R$. By the mean-value theorem and Taylor expansion with Lipschitz gradient (Nesterov, 2013),

$$u(x,t) := z_{\theta_R + \Delta}(x,t) - z_{\theta_R}(x,t) = J(x,t) \operatorname{vec}(\Delta) + r(x,t), \quad ||r(x,t)||_2 \le \frac{L}{2} ||\Delta||_2^2,$$
 (A17)

where L is a local Lipschitz constant of $\nabla_{\theta} z_{\theta}(x,t)$ around θ_R . Let $\Phi = \text{blkdiag}(\Phi^1,\dots,\Phi^K)$ be the block-diagonal forward feature operator that maps $\text{vec}(\Delta)$ to the linearized token-level feature change collected at thinking positions (per submodule k). Under bounded intermediate activations and operator norms, which are standard in local linearization of deep nets (Fazlyab et al., 2019), there exists $C_1 > 0$ such that $\|J(x,t) \text{ vec}(\Delta)\|_2 \le C_1 \|\Phi \text{ vec}(\Delta)\|_2$. Combining with Eq. (A17),

$$||u(x,t)||_2 \le C_1 ||\Phi|| \operatorname{vec}(\Delta)||_2 + C_2 ||\Delta||_2^2, \qquad C_2 := \frac{L}{2}.$$
 (A18)

Step 3: Enforcing the null-space constraint and aggregating into \mathcal{L}_{think} . Apply the submodule-wise null-space projection (see Eq. (5) in the main text):

$$\operatorname{vec}(\Delta_I^{\perp,k}) = P^{\perp}(\Phi_{\Omega_{\text{think}}}^k) \operatorname{vec}(\Delta_I^k), \qquad \Delta_I^{\perp} = \bigoplus_{k=1}^K \Delta_I^{\perp,k}, \tag{A19}$$

so that by construction $\Phi_{\Omega_{\text{think}}} \operatorname{vec}(\Delta_I^{\perp}) = 0$. Plugging this into **Eq. (A18)** yields for all $t \in \Omega_{\text{think}}(x)$:

$$||u(x,t)||_2 \le C_2 ||\Delta_I^{\perp}||_2^2.$$
 (A20)

Combining with Eq. (A14) and summing/averaging over (x,t) in the definition of \mathcal{L}_{think} (Eq. (2)) gives

$$\mathcal{L}_{\text{think}}(\theta_{R} + \Delta_{I}^{\perp}) = \mathbb{E}_{x} \sum_{t \in \Omega_{\text{think}}(x)} \text{KL}\Big(\pi_{\theta_{R} + \Delta_{I}^{\perp}}(\cdot \mid x, y_{< t}) \mid || \pi_{\theta_{R}}(\cdot \mid x, y_{< t})\Big)$$

$$\leq \frac{1}{8} \mathbb{E}_{x, t} [||u(x, t)||_{2}^{2}] + O(\mathbb{E}_{x, t} ||u(x, t)||_{2}^{3})$$

$$= O(||\Delta_{I}^{\perp}||_{2}^{4}) \leq O(||\Delta_{I}^{\perp}||_{2}^{2}) \approx 0. \tag{A21}$$

This completes the proof.

```
1026
              Algorithm 1: RAIN-Merging: Reasoning-Aware Instruction-attention guided Null-space projec-
1027
1028
                          :LRM \theta_R; ITM \theta_I; base model \theta_B; reasoning calibration set \mathcal{D}_R with thinking indices \Omega_{\text{think}};
1029
                             instruction calibration set \mathcal{D}_I with spans (\mathcal{I}, \mathcal{R}, \mathcal{U}); hyperparameters \rho, \tilde{\alpha}_l, \tilde{\alpha}_u, \lambda.
1030
              Output: Merged model \theta_{\star}.
1031
           1 Stage 0: Task vector and objective.
1032
           \Delta_I \leftarrow \theta_I - \theta_B
                                                                                                    // instruction-tuned task vector
1033
          3 Stage 1: Reasoning-aware Null-space Projection (satisfy Eq. (2)).
1034
          4 for k \leftarrow 1 to K // iterate over submodules (per-layer W_Q, W_K, W_V, W_O, \text{FFN}) do
1035
                    \Phi_{\Omega}^k \leftarrow \text{FeatureOperator}\left(\theta_R, \mathcal{D}_R, \Omega_{\text{think}}, k\right) // forward feature extraction at
                     thinking tokens
                    P_k^{\perp} \leftarrow \operatorname{diag}(1) - (\Phi_{\Omega}^k)^{\top} \big(\Phi_{\Omega}^k (\Phi_{\Omega}^k)^{\top} + \operatorname{diag}(1)\big)^{-1} \Phi_{\Omega}^k
                                                                                                              // least-squares orthogonal
1038
                     projector
                1039
                                                                                            // submodule projection per Eq. (5)
           \theta' \leftarrow \theta_R + \bigoplus_{k=1}^K \Delta_I^{\perp,k}
                                                                                                          // direct merge after Stage 1
          9 Stage 2: Instruction-attention Guided Merging Coefficients (optimize Eq. (11)).
1042
         10 Initialize head-wise coefficients \tilde{\alpha}^{\tilde{k}} \leftarrow 1 for all attention heads \tilde{k}.
         11 for each attention head \tilde{k} do
1044
                    a^{\tilde{k}} \leftarrow \mathbb{E}_{x \sim \mathcal{D}_I} \left[ \frac{1}{|\mathcal{I}(x)| |\mathcal{R}(x)|} \sum_{t \in \mathcal{I}(x)} \sum_{\tau \in \mathcal{R}(x)} \operatorname{Att}_{\theta'}^{\tilde{k}}(x)[t, \tau] \right]
1045
                   u^{\tilde{k}} \leftarrow \mathbb{E}_{x \sim \mathcal{D}_I} \left[ \frac{1}{|\mathcal{I}(x)| |\mathcal{U}(x)|} \sum_{t \in \mathcal{I}(x)} \sum_{\tau \in \mathcal{U}(x)} \operatorname{Att}_{\theta'}^{\tilde{k}}(x)[t, \tau] \right]
1046
         14 for each attention head \tilde{k} do
1048
                    g^{\tilde{k}} \leftarrow a^{\tilde{k}} - \rho u^{\tilde{k}}
                                                                                                   // first-order term for Eq. (11)
1049
         15
                    \tilde{H}^{\tilde{k}} \leftarrow 1 + u^{\tilde{k}}
                                                                                                                 // diagonal Hessian approx
1050
1051
                   \tilde{\alpha}_{\star}^{\tilde{k}} \leftarrow \text{clip}_{[\tilde{\alpha}_l, \tilde{\alpha}_u]} \left( \frac{g^k}{\tilde{n}_{\tilde{k}}} \right)
                                                                                                               // per-head optimal scaling
1052
1053
         18 \alpha_{\star}^{k} \leftarrow \text{Aggregate}\left(\{\tilde{\alpha}_{\star}^{k}\}_{\tilde{k} \in \text{module } k}\right)
                                                                                                                 // mean over heads for FFN
1054
         19 Output (Two-stage Merge).
1055
         20 return \theta_\star \leftarrow \theta_R + \lambda igoplus_{k=1}^K lpha_\star^k \Delta_I^{\perp,k}
                                                                                                                 // final model in Eq. (16)
1056
```

F ALGORITHM

1057 1058

1062

1064

1067

1068 1069

1070 1071 1072

1073

1074

1075

1076 1077

1078

1079

Following Alg. 1 is the algorithm of our RAIN-Merging.

G METHOD IMPLEMENTATION DETAILS

G.1 FORWARD MECHANISM IN TRANSFORMER

A standard Transformer layer consists of multi-head self-attention and a feed-forward network (FFN). In layer ℓ , the hidden state of the token at position t, denoted $h_t^{(\ell-1)} \in \mathbb{R}^d$, is linearly projected to queries, keys, and values: $q_t^{(\ell)} = W_Q^{(\ell)} h_t^{(\ell-1)}, k_\tau^{(\ell)} = W_K^{(\ell)} h_\tau^{(\ell-1)}, v_\tau^{(\ell)} = W_V^{(\ell)} h_\tau^{(\ell-1)}$. For head h, the single-head attention weights are $\operatorname{Att}_\theta^{(\ell,h)}(x)[t,\tau] = \operatorname{softmax}_\tau\left(\langle q_t^{(\ell,h)}, k_\tau^{(\ell,h)} \rangle / \sqrt{d_k}\right)$, which represent the probability that the token at position t attends to position τ . The corresponding head output is $o_t^{(\ell,h)} = \sum_\tau \operatorname{Att}_\theta^{(\ell,h)}(x)[t,\tau] v_\tau^{(\ell,h)}$. After concatenating the outputs from all heads and applying $W_O^{(\ell)}$, we obtain $\tilde{h}_t^{(\ell)}$. The FFN then computes $\hat{h}_t^{(\ell)} = \sigma\left(W_{\mathrm{in}}^{(\ell)} \tilde{h}_t^{(\ell)} + b_{\mathrm{in}}^{(\ell)}\right), h_t^{(\ell)} = W_{\mathrm{out}}^{(\ell)} \hat{h}_t^{(\ell)} + b_{\mathrm{out}}^{(\ell)}$. The top-layer hidden state is mapped to vocabulary logits $z_\theta(x,t)$, which are transformed by a softmax into the conditional distribution $\pi_\theta(\cdot \mid x, y_{< t})$. We follow the notation and the scaled dot-product attention definition of Vaswani et al. (2017) to align with prior work.

Table A1: Reasoning calibration set construction from *Mixture-of-Thoughts*. We uniformly sample 50 examples per domain for calibration and 50 for validation. Raw sizes are taken from the official dataset composition page.

Domain	Raw size	Calibration	Validation
Math	93,700	50	50
Code	83,100	50	50
Science	173,000	50	50
Total	349,800	150	150

G.2 IMPLEMENTATION DETAILS IN MERGING

To balance computational efficiency and memory usage, all model-merging experiments adopt a **layer-wise** merging strategy. During parameter fusion, we compute in **FP64** precision to ensure numerical stability, and we store the final models in **BF16**. Our experiments show that higher compute precision yields consistent but modest improvements for this merging procedure.

H CALIBRATION SET CONSTRUCTION

H.1 REASONING CALIBRATION SET

We employ the *Mixture-of-Thoughts*² (Face, 2025) dataset as the source for reasoning-style calibration. This dataset contains validated R1-style reasoning traces spanning three domains: math, code, and science, with a total size of approximately 350k samples. Its official data composition page clearly specifies the sample sizes and origins for each sub-domain: math samples are sourced from OpenR1-Math (Lozhkov et al., 2025), code from CodeForces-CoTs (Penedo et al., 2025), and science from the science subset of the Nemotron post-training set (Bercovich et al., 2025). From each domain, we randomly sample 50 instances to form the calibration set (150 in total), and an additional 50 instances per domain are randomly sampled to serve as the validation set (150 in total). **Tab.** A1 shows the detailed numbers of samples in each domain.

Thinking Special Token Set Construction. To apply preservation constraints on "thinking format", we extract the thinking tokens, specifically <code><think></code> and <code></think></code> in the model output—based on the R1-style chat template and tokenizer. The procedure involves rendering messages using the chat template provided by LRM. R1-family models prefill <code><think></code> in reasoning mode and insert <code></think></code> in the context, while some templates may omit the visible output of the initial <code><think></code> to enforce thinking mode. We then obtain token positions of <code><think></code> and <code></think></code> in Ω_{think} .

H.2 INSTRUCTION CALIBRATION SET

We construct a high-quality instruction calibration set from rule-verifiable prompts through four automated and auditable steps. The pipeline produces span-based samples $(x \sim \mathcal{D}_I; \mathcal{I}(x), \mathcal{R}(x), \mathcal{U}(x))$ for computing the instruction-attention score proxy in Stage 2 of RAIN-Merging. We choose to distill from IFEval-style instructions for ease of implementation and to test generalization on out-of-domain instruction-following datasets. The final size of the instruction calibration set is 365. The full workflow is:

- Instruction selection. We select rule-verifiable instruction prompts from IFEval (Zhou et al., 2023b) as queries. Each record contains a natural-language instruction and machine-checkable constraints.
- Step 1: Response generation by LRM. For each instruction query, we invoke an R1-style reasoning model (deepseek-reasoner, DeepSeek-R1-0528)³ to produce a format-explicit response. This step yields instruction-following samples generated by a reasoning decoder that reflect realistic decoding behavior.

²https://huggingface.co/datasets/open-r1/Mixture-of-Thoughts

³https://api.deepseek.com/v1

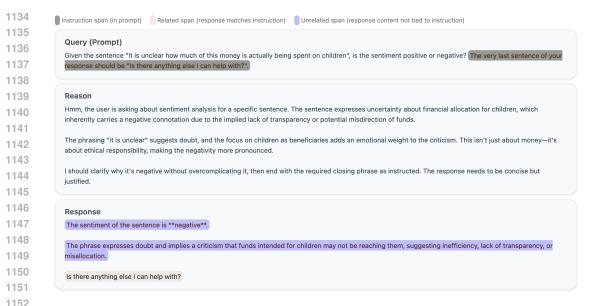


Figure A3: A sample illustration in our instruction calibration set.

Table A2: Instruction-following benchmarks. We list dataset size, constraint taxonomy, composition types, verification, and aggregation strategy.

Benchmark	Size	Constraint		Composition Type			Veri	fication	Evaluation
		Taxonomy	And	Chain	Selection	Nested	Code-Exec.	LLM-as-Judge	Aggregation
IFEval	541	25	√	_	_	_	✓	_	strict_prompt_level_accuracy
CELLO	523	4	✓	✓	_	_	✓	_	average
InfoBench	500	5	✓	✓	_	_	_	✓	DRFR
ComplexBench	1,150	19	✓	✓	✓	✓	✓	✓	dependency-aware DRFR

- Step 2: Rule evaluation and filtering. We evaluate the outputs of Step 1 with an IFEval-compatible checker and *retain only passing samples* that satisfy all constraints. This removes cases that clearly fail the requirements.
- Step 3: Strict span extraction (LLM-as-Judge). We use a high-performance instruction-tuned LLM (deepseek-chat, DeepSeek-V3.1)³ to precisely extract instruction-relevant spans: $\mathcal{I}(x)$ (tokens in the prompt corresponding to the instruction) and $\mathcal{R}(x)$ (tokens in the response that are governed by the instruction). The unrelated span $\mathcal{U}(x)$ is then implicitly defined as the remainder of the response. See Fig. A3 for an example.
- Step 4: Tokenizer-level verification. We verify the extracted spans on the *target tokenizer* (aligned with our anchor LRM), ensuring that boundaries lie on token edges and can be deterministically reconstructed. Samples that fail alignment are discarded.
- Step 5: Human review and ethical screening. To ensure data quality and compliance with safety and ethics standards, we introduce a manual review stage. Researchers verify the accuracy of the LLM-extracted spans $\mathcal{I}(x)$ and $\mathcal{R}(x)$, and conduct an ethics audit of the responses based on content-safety guidelines, removing any samples that contain biased, harmful, or inappropriate content. This step further enhances the reliability and ethical soundness of the calibration set.

This calibration pipeline is readily transferable and can be extended to additional instruction-following datasets to further improve merging effectiveness by enriching the calibration set. The reasoning distillation model and the LLM-as-Judge can be updated over time to continually enhance the quality of the instruction calibration data.

Table A3: Test set sizes of the six math benchmarks used in our mathematical reasoning (Math) evaluation.

	AIME2025	AIME2024	AMC23	GSM8K	Math500	MinervaMath
# Test samples	30	30	40	1,319	500	272

I Detailed Experimental Setup

I.1 BENCHMARKS

Instruction-following Benchmarks. We evaluate instruction compliance on four widely used, programmatically verifiable benchmarks. The size and constraint types are summarized in **Tab. A2**.

- **IFEval** (Zhou et al., 2023b). IFEval provides four accuracy metrics: (1) *prompt-level strict* accuracy and (2) *instruction-level strict* accuracy, plus (3) *prompt-level loose* and (4) *instruction-level loose* variants. The strict metrics require exact satisfaction (all constraints per prompt for prompt-level strict; per-constraint averaging across prompts for instruction-level strict). The loose metrics first normalize model outputs (e.g., strip Markdown, boilerplate intros/outros) to reduce false negatives. We report the official **strict prompt level accuracy** unless otherwise noted.
- **CELLO** (He et al., 2024). CELLO uses a *code-based* verifier that scores four granular aspects: (i) *count limit* (word/sentence/sample counts), (ii) *answer format* (parsability, keywords), (iii) *task-prescribed phrases* (mandatory phrases covered), and (iv) *input-dependent query* (presence of key phrases from the input), with a penalty to discourage verbatim copying. We follow the benchmark's practice and **average** these checks to produce the final score.
- InfoBench (Qin et al., 2024). InfoBench adopts the *Decomposed Requirements Following Ratio (DRFR)*: each instruction is split into scoring questions that are judged by *LLM-as-a-Judge* with binary YES/NO labels; the final score is the mean over all questions, enabling fine-grained interpretability. We evaluate by GPT-5-mini. We report the official **DRFR**.
- ComplexBench (Wen et al., 2024). ComplexBench also evaluates via decomposed scoring questions with YES/NO judgments and aggregates them into DRFR, but crucially uses a dependency-aware scheme: if any prerequisite constraint fails, all dependent (downstream) constraints are automatically marked as failed. This better reflects multi-constraint compositions. We evaluate by GPT-40-mini. We report the dependency-aware DRFR.

Reasoning & General Benchmarks. We evaluate reasoning and general capabilities on the following benchmarks:

- Math (Mathematical reasoning). We aggregate Pass@1/accuracy over six common math benchmarks: AIME2025 (Balunović et al., 2025); AIME2024 (Balunović et al., 2025); AMC23 (Cao et al., 2024); GSM8K (Cobbe et al., 2021); Math500 (Hendrycks et al., 2021); Minerva-Math (Lewkowycz et al., 2022). The size of each math benchmark is shown in Tab. A3. We report the averaged accuracy over all benchmarks.
- Aider (Aider, 2024) (Code editing). Aider-Edit assesses code editing capability under a minimaledit paradigm. It contains 133 small Python coding exercises sourced from Exercism, where the
 model is provided with a natural-language edit instruction and the existing code, and must generate
 a correct patch. The generated patch is required to apply successfully to the codebase and pass
 compilation and associated tests. Performance is measured by the Pass@2 Edit Success Rate.
- **GPQA** (Rein et al., 2024) (**STEM**). A curated, expert-level subset of GPQA comprising 198 four-option multiple-choice questions across biology, chemistry, and physics. Items are selected to be "Google-proof" and unambiguous: both expert validators must answer correctly while at most one of three skilled non-experts succeeds, yielding a particularly hard split. We follow common practice and report **accuracy** (strict single-choice).
- Arena-Hard-v2 (Li et al., 2024) (Creative writing). A hard, open-ended benchmark constructed
 to maximize model separability and align with human preferences. Arena-Hard-Auto curate
 ~500 challenging prompts covering difficult real-world tasks including creative writing, scoring
 follows the pairwise battle paradigm with human or LLM-as-a-Judge assessments, and results
 are commonly summarized as win rate or transformed to Elo scores. For reproducibility, we fix

- Qwen2.5-7B as the reference baseline and use GPT-5-mini as the judging LLM; we report the resulting **win rate** of each model against this baseline on the official Arena-Hard-v2 prompt set.
- Agentic Scenarios. We evaluate the ability of the model to interact with the environment and complete tasks by two agentic benchmarks: ALFWorld (Shridhar et al., 2021) is a text-interactive household-task environment involving multi-step planning and execution. We evaluate on 100 tasks. Metrics are Success Rate of goal completion. WebShop (Yao et al., 2022) is a web shopping agent task (search, click, compare), with reported Normalized Reward to capture both path efficiency and goal matching. We evaluate on 100 tasks.

I.2 BASELINES

We compare our method with the following merging baselines:

- Task Arithmetic (Ilharco et al., 2023). The simplest linear composition injects the task vector additively near the anchor, $\theta = \theta_R + \lambda \, \Delta_I$. A scalar $\lambda \in [0,1]$ usually controls the strength; using the same λ per layer or per block is also common. Its advantages are zero data and negligible compute; its drawback is that conflicts across submodules are hard to disentangle.
- SLERP (Biship, 2007; Goddard et al., 2024). In SLERP (Spherical Linear Interpolation), weights are ℓ_2 -normalized to the unit sphere and interpolated along the geodesic to preserve norm and angular geometry. Let $\Omega = \arccos(\langle w_R, w_I \rangle)$. Then $\operatorname{slerp}(w_R, w_I; t) = \frac{\sin((1-t)\Omega)}{\sin\Omega} w_R + \frac{\sin(t\Omega)}{\sin\Omega} w_I$, $t \in [0,1]$. During merging, we apply SLERP to each tensor and rescale by the original norm. This reduces norm drift compared with linear interpolation.
- Karcher (Nielsen & Bhatia, 2013; Goddard et al., 2024). On a chosen manifold like the unit sphere or a Stiefel manifold, compute the Fréchet mean by minimizing the sum of squared geodesic distances: $\min_{\bar{w}} \sum_i d^2(\bar{w}, w_i)$. The iterative update is $\bar{w}^{(t+1)} = \operatorname{Exp}_{\bar{w}^{(t)}}(\frac{1}{n}\sum_i \operatorname{Log}_{\bar{w}^{(t)}}(w_i))$.
- TIES (Yadav et al., 2023). TIES is a data-free method that explicitly prunes and sparsifies to handle parameter-level conflicts. For each layer's edit vector it applies (i) a sign-consistency mask (retain entries aligned with the dominant direction to reduce cancellation), (ii) magnitude thresholding or Top-k truncation (keep high-contribution entries and zero out the rest), and (iii) optional rescaling to match a target norm. We can stack TIES as a post-processing step on top of feasible baselines to improve robustness.
- DARE (Yu et al., 2024). DARE uses first-order sensitivities on a small calibration set (for example, gradient norms of labeled loss, log-likelihood changes, or Fisher approximations of the output distribution) to learn a per-layer or per-tensor coefficient α^k (or a diagonal preconditioner), yielding $\theta = \theta_R + \bigoplus_k \alpha^k \Delta_I^k$. It can be viewed as data-aware recalibration that reduces the bias introduced by naive addition with very low compute.
- ACM (Yao et al., 2025). ACM (Activation-Guided Consensus Merging) targets activation consistency. On a small calibration set it measures, before and after injecting the task vector, how each layer or head changes its response on instruction-relevant spans and its leakage on irrelevant spans. It then solves for per-submodule coefficients α^k , optionally with cross-sample consensus regularization to improve generalization.
- LEWIS (Chopra et al., 2025). LEWIS (LayEr WIse Sparsity) is a merge with layer-wise sparsity allocation. Based on sensitivity indicators per layer (such as edit-vector magnitude, activation gradients, or Fisher approximations), it sets a budget s_k and merges only the Top- s_k parameters of that layer, leaving the anchor weights elsewhere unchanged. It can be combined with Task Arithmetic, SLERP, or Karcher as the base, and $\{s_k\}$ are determined by heuristics or grid search on a small calibration set.
- AIM (Nobari et al., 2025). AIM (Activation-based Importance Merging) weights the task vector by activation importance (for example, the effect of Value or FFN outputs on downstream logits, or the instruction-aligned component of attention weights), performing element-wise or block-wise reweighting: Weights the task vector by activation importance, for example, the effect of Value or FFN outputs on downstream logits, or the instruction-aligned component of attention weights, performing element-wise or block-wise reweighting: θ = θ_R + ⊕_k W^k ⊙ Δ^k_I, where W^k is obtained from a single forward pass on the calibration set. Intuitively, this preserves edits that meaningfully change useful representations and suppresses noisy updates.

Table A4: The hyperparameters of various merging methods in **Tab. 1**. λ means the global scaling coefficient in merging. k denotes the trim ratio in TIES-Merging. p means the drop rate in DARE merging. τ is sharpness the ACM. ρ is the pruning ratio in LEWIS. ω means the balance factor in AIM.

Method	Hyper-parameters 7B
Task Arithmetic	$\lambda = 1.0$
SLERP	$\lambda = 1.0$
Karcher	$\lambda = 1.0$
TIES	$k = 0.8, \ \lambda = 0.8$
DARE-TIES	$p = 0.3, k = 0.5, \lambda = 1.2$
ACM-TIES	$\tau = 1.0, \ k = 0.5, \ \lambda = 1.1$
LEWIS-TIES	$\rho = 0.5, \ k = 0.5, \ \lambda = 1.1$
AIM-TIES	$\omega = 0.4, \ k = 0.5, \ \lambda = 1.0$

Table A5: The hyperparameters of RAIN-Merging in different model sizes. λ means the global scaling coefficient in RAIN-Merging.

Method	Hyper-parameters						
	1.5B	7B	8B	14B			
RAIN-Merging	$\lambda = 1.0$	$\lambda = 1.0$	$\lambda = 0.9$	$\lambda = 1.0$			

Unless otherwise specified, following common practice in previous work (Wu et al., 2025), we apply TIES post-processing (sign consistency and magnitude truncation) on the outputs of DARE, ACM, LEWIS, and AIM, in order to improve comparability across baselines.

I.3 HYPERPARAMETERS

For SFT, we use a batch size of 16 with the Adam optimizer (Kingma & Ba, 2014), a learning rate of 2×10^{-5} , weight decay of 0.05, and train for 20 epochs.

For all model-merging methods (including the proposed RAIN-Merging and all baselines), we merge only the task vectors extracted from the ITM's Q/K/V/O/FFN modules. The specific hyperparameter settings for each baseline used in **Tab. 1** are listed in **Tab. A4**.

In **RAIN-Merging**, we set the leakage penalty to $\rho=10$ and bound the attention-head coefficients by $[\tilde{\alpha}_l, \tilde{\alpha}_u] = [0.0, 1.0]$. The global scalar λ is selected via a grid search over [0.0, 1.5] with a step size of 0.1; the chosen values for different model families are provided in **Tab.** A5. An ablation study of the global scalar λ is included in Appendix J.2.

J Additional Experiments

J.1 DETAILED MATH BENCHMARK RESULTS

Tab. A6 and **Tab.** A7 report detailed results on the mathematics benchmarks. **RAIN-Merging** consistently preserves the mathematical reasoning ability of LRMs across different model sizes and architectures. In some cases, improving instruction following also correlates with better mathematical performance, suggesting that enhanced adherence can support clearer intermediate reasoning and more reliable final answers.

J.2 ABLATION STUDY OF THE GLOBAL SCALAR

We conduct a sensitivity analysis of the global scalar λ (Fig. A4). Across a wide range around our chosen value near 1.0, the merged model maintains strong instruction-following performance. As λ

Table A6: Math benchmark results under the same configuration as in **Tab. 1**. "Avg." denotes the average over all math benchmarks. The best and second-best results are highlighted in **bold** and <u>underlined</u>, respectively.

Method	AIME2025	AIME2024	AMC23	GSM8K	Math500	Minerva	Avg.
ITM	10.00	10.00	67.50	86.66	73.80	35.66	47.27
LRM	30.00	<u>50.00</u>	80.00	<u>91.36</u>	89.00	48.16	64.75
SFT	33.33	43.33	75.00	90.75	87.40	45.59	62.57
Task Arithmetic	30.00	50.00	80.00	90.75	89.00	45.59	64.22
SLERP	30.00	46.67	77.50	91.05	89.20	48.53	63.82
Karcher	30.00	50.00	80.00	90.98	89.20	48.90	64.85
TIES	33.33	46.67	82.50	91.36	90.60	48.16	65.44
DARE-TIES	36.67	40.00	82.50	90.98	<u>90.20</u>	45.22	64.26
AIM-TIES	33.33	50.00	85.00	89.76	89.60	47.79	65.92
ACM-TIES	33.33	50.00	77.50	91.36	88.20	47.06	64.57
LEWIS	30.00	33.33	80.00	90.27	88.80	50.00	62.07
RAIN-Merging	36.67	60.00	85.00	92.12	90.20	48.53	68.75

Table A7: Math benchmarks results under the same configuration as in **Tab. 2**. "Avg." denotes the average over all math benchmarks.

Method	AIME2025	AIME2024	AMC23	GSM8K	Math500	Minerva	Average
Qwen2.5-1.5B-Instruct	3.33	0.00	30.00	75.44	59.40	22.43	31.77
DeepSeek-R1-Distill-Qwen-1.5B	20.00	13.33	42.50	73.77	71.80	28.31	41.62
Qwen2.5-1.5B-RAIN-Merging	20.00	16.67	60.00	76.36	72.40	29.78	45.87
Llama-3.1-8B-Instruct	3.33	6.67	20.00	81.65	67.60	34.26	35.59
DeepSeek-R1-Distill-Llama-8B	30.00	40.00	75.00	90.52	80.40	45.37	60.21
Llama-3.1-8B-RAIN-Merging	30.00	43.33	77.50	90.30	82.80	47.79	61.95
Qwen2.5-14B-Instruct	20.00	13.33	65.00	93.18	80.00	44.85	52.73
DeepSeek-R1-Distill-Qwen-14B	50.00	56.67	92.50	93.22	89.00	52.49	72.31
Qwen2.5-14B-RAIN-Merging	50.00	63.33	92.50	94.37	91.00	56.25	74.58

increases, reasoning ability improves slowly at first but then drops sharply beyond 1.0, indicating that overly large merge strength can still harm reasoning.

J.3 ABLATION STUDY OF REASONING CALIBRATION SET SIZE

Fig. A5 presents an ablation over the size of the reasoning calibration set. As the set grows, preservation of reasoning improves; however, instruction-following performance degrades gradually. We hypothesize that overly strict preservation of the thinking format can limit gains in instruction adherence and also increase computation. Balancing performance and resource usage, we select a calibration size of 150.

J.4 VISUALIZATION OF MERGING COEFFICIENTS IN STAGE 2

As shown in **Fig. A6**, the heatmap of merging coefficients for DeepSeek-R1-Distill-Qwen-7B exhibits clear layer-wise differences, indicating that different layers respond to instruction focus to different degrees. Notably, the earliest layers show the strongest response, with coefficients reaching the upper bound, and this pattern is consistent with the observations in **Fig. 6**.

J.5 CASE STUDY

We provide two case studies to illustrate the effectiveness of RAIN-Merging compared with DeepSeek-R1-Distill-Qwen-7B as the baseline LRM on IFEval.

• IFEval Example 1: The baseline LRM violates the explicit rule to first echo the request verbatim and further duplicates its poem, yielding a "following: False" outcome. In contrast, RAIN-Merging

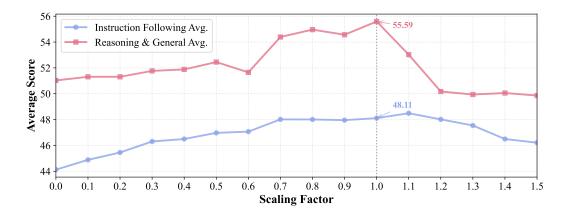


Figure A4: Instruction following and reasoning & generage performance of our RAIN-Merging using different global scalar λ . The configuration is the same as in **Tab. 2**. The performance is measured by the average of the instruction following and reasoning & general capability benchmarks. The marked result is our choice in the experiments.

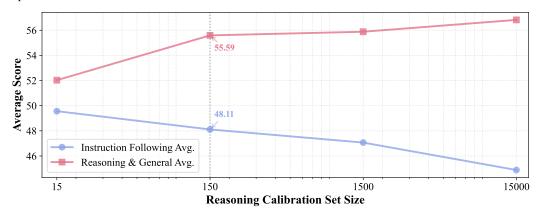


Figure A5: Instruction following and reasoning & generage performance of our RAIN-Merging using different reasoning calibration set sizes. The x axis represents the size of the reasoning calibration set with exponential scale. The configuration is the same as in **Tab. 2**. The performance is measured by the average of the instruction following and reasoning & general capability benchmarks. The marked result is our choice in the experiments.

correctly repeats the request word-for-word, includes the required keywords ("intern," "grow"), and produces a coherent, father-pleasing limerick ("following: True").

• **IFEval Example 2**: The baseline LRM introduces capitalized section headers and markup (e.g., "Verse 1"), breaking the "all lowercase" constraint ("following: False"). RAIN-Merging delivers fully lowercase lyrics with clear structure and consistent semantics ("following: True").

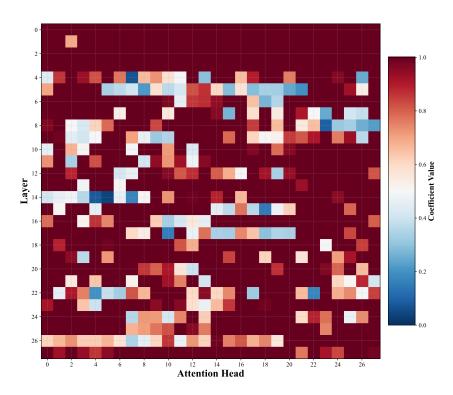


Figure A6: Heatmap of merging coefficients by our Stage 2 for each layer and attention head of DeepSeek-R1-Distill-Qwen-7B.

Instruction-following Example 1 in IFEval

Prompt

Write a limerick about Hannah, a college student, doing an internship at a coffee company. Make sure that her father would love the limerick. Include the words "intern" and "grow".

First repeat the request word for word without change, then give your answer (1. do not say any words or characters before repeating the request; 2. the request you need to repeat does not include this sentence)

DeepSeek-R1-Distill-Qwen-7B Reason

Okay, so the user has given me a query where they want a limerick about Hannah, a college student doing an internship at a coffee company. They also want me to make sure her father would love it, include the words "intern" and "grow".

First, I need to understand what a limerick is. It's a five-line poem with an AABBA rhyme scheme, usually humorous. So, I should structure it that way.

The main characters here are Hannah, who's an intern, and her father. I need to highlight something that would please her father. Since he's looking back on her, maybe something positive, like growth or learning. The user mentioned "grow," so that's a keyword I must include.

I'll start the first line with "There once was a college student named Hannah," which sets the scene. Next, she's doing an internship at a coffee company, so the second line can mention that.

For the third line, I need to introduce her father. Maybe something like "Her dad would be tickled," which fits the rhyme and the idea that he's proud or finds it amusing.

In the fourth line, I have to include "grow," perhaps about Hannah growing into her role or learning something. Something like "she grew as an intern." That works because it's a common phrase and fits the rhyme.

Finally, the last line should be a nice, positive ending. Maybe the coffee company has grown as a result of her hard work. "And the JavaHouse grew from her intern!" That ties the "grow" into the company's success.

I should make sure the rhyme scheme is consistent and the lines flow well. Let me check each line for the AABBA pattern: the first and third lines have "Hannah," the second and fourth have "intern" and "grow," and the fifth is the punchline. The rhymes at the end of each line should be "annah," "inn," "dad," "un," and "or." Wait, that doesn't quite fit perfectly. Let me adjust the fifth line to "and JavaHouse"

1512 grew from her intern." That way, the last two lines rhyme with "grow" and "grew," which is close 1513 enough for a limerick's loose rhyme. 1514 I think this captures Hannah's experience, shows her learning, and how her father is proud. It also ties 1515 the coffee company's growth to her internship, which is a nice touch. I should make sure the language 1516 is playful and fits the humorous tone of limericks. 1517 DeepSeek-R1-Distill-Qwen-7B Response (following: False) 1518 There once was a college student named Hannah, 1519 Whose intern days at JavaHouse she would swap for much. 1520 Her dad would be tickled, I'm sure, If he heard she grew as an intern. 1521 And JavaHouse grew from her intern! 1522 There once was a college student named Hannah, 1523 Whose intern days at JavaHouse she would swap for much. Her dad would be tickled, I'm sure, 1525 If he heard she grew as an intern. 1526 And JavaHouse grew from her intern! **RAIN-Merging Reason** Okay, so the user has given me a query where they want a limerick about Hannah, a college student 1529 doing an internship at a coffee company. They also want me to include the words "intern" and "grow". 1530 Plus, the limerick should make my father happy. I need to make sure that the poem not only fits the structure of a limerick but also touches on what a father might appreciate. 1531 First, recall the structure of a limerick (AABBA, typically anapestic). Incorporate internship details and 1532 the required keywords. Emphasize traits a father would love: responsibility, learning, contribution. 1533 Draft lines that meet the rhyme scheme and include "intern" and "grow," aiming for a playful, positive 1534 tone that showcases Hannah's growth and impact at the coffee company. 1535 **RAIN-Merging Response (following: True)** 1536 Write a limerick about Hannah, a college student, doing an internship at a coffee company. Make sure 1537 that her father would love the limerick. Include the words "intern" and "grow". 1538 There once was a college student named Hannah, 1539 An intern at a coffee company she was. 1540 With her skills, both sharp and bright, She's learning every day, 1541 She's growing taller, her skills grow bright. 1542 Her dad would surely give her a smile, 1543 For Hannah's heart and Hannah's skill. 1544 She's balancing tasks, balancing time, 1545 Turning great coffee into something divine. 1546 A limerick full of sunny pride! 1547 1548 1549 Instruction-following Example 2 in IFEval 1550 1551 **Prompt** Compose song lyrics about a socio-economic problem. The song should be in English and in all 1552 lowercase letters. 1553 DeepSeek-R1-Distill-Qwen-7B Reason 1554 The user asked for song lyrics about a socio-economic problem, in English and all lowercase. Choose a 1555 relatable issue (e.g., inequality, unemployment), structure with verses/chorus/bridge, maintain empa-1556 thetic tone, vivid imagery, and ensure every character is lowercase. Aim for accessibility, repetition in 1557 chorus, and a subtle sense of resilience or hope. DeepSeek-R1-Distill-Qwen-7B Response (following: False) Verse 1 1560 every penny's a struggle 1561 like fuel for a car that won't start the stars are bright but we don't see them the ladder's always broken 1563 **Chorus** 1564

every penny's a struggle

1566 a weight in your chest 1567 the world's a game but you're just a piece 1568 no matter how hard you try 1569 every penny's a struggle 1570 Verse 2 1571 the roads are wet but the rain falls cold 1572 the wind's a whisper but it's got no hold 1573 we're dancing to a tune we can't control but the beat's the same 1574 1575 Chorus every penny's a struggle 1576 a weight in your chest 1577 the world's a game but you're just a piece no matter how hard you try 1579 every penny's a struggle 1580 **Bridge** 1581 look beyond the pain is there a spark in your soul? can you see the stars at night? maybe there's something more 1584 1585 every penny's a struggle 1586 a weight in your chest 1587 the world's a game but you're just a piece 1588 no matter how hard you try 1589 every penny's a struggle 1590 a penny's worth of love or light 1591 the struggle's real but we gotta fight 1592 we'll find the way through the cold 1593 the world's a game but you're just a piece 1594 **RAIN-Merging Reason** 1595 Generate lyrics that address a socio-economic issue with clear imagery and structure 1596 (verses/chorus/bridge). Conform strictly to the all-lowercase constraint. Keep language simple and 1597 emotive, focusing on systemic barriers and lived experience, while maintaining flow and singability. 1598 **RAIN-Merging Response (following: True)** "the gap in the system" (verse 1) 1601 i see the faces in the lines, faces searching for a ticket out of here. factories grinding, bodies quiet, a world that promises better, but i'm still waiting for the breath. 1604 (chorus) 1605 there's a crack in the system, a fault in the law, 1606 half the planet's frozen in time. we're crafted from the same mold, same broken spell, 1608 but the world won't let us climb the hill. 1609 (verse 2) 1610 the streets are empty, the shops tiled over, 1611 the only sound is the shuttle's old tune. we're drowning in the waves, but the shores are a lie, 1612 the promises fade like ash in the wind. 1613 (chorus) 1614 there's a crack in the system, a fault in the law, 1615 half the planet's frozen in time. 1616 we're crafted from the same mold, same broken spell, 1617 but the world won't let us climb the hill. 1618 (bridge)

oh, it's not just the lack, it's the broken chain, we're all stuck in the margins, never quite seen. but the fights don't stop, they never do, we're trapped in the loop, chained to the ground. (chorus) there's a crack in the system, a fault in the law, half the planet's frozen in time. we're crafted from the same mold, same broken spell, but the world won't let us climb the hill. so here's to the night shifts and the cold nights, the endless loop where the kids don't get the break. we're all fragments on the ground, trying to find a way out of the ground.

K IMPLICATIONS

The potential impacts of this work span both academic and applied dimensions.

On the positive side, RAIN-Merging offers an interpretable, low-compute pathway for capability integration. It enables researchers and resource-constrained labs to inject instruction-following competence into LRMs without additional training. By enforcing a null-space constraint on the thinking segment (<think>...</think>), the method preserves the model's structured reasoning format, which helps maintain reliability in reasoning. This direction may catalyze systematic studies of the relationship between task-vector orthogonality and thinking-format stability, and it encourages reproducible evaluation practices (for example, public evaluation scripts, calibration sets, and hyperparameter configurations) and greater standardization of community benchmarks. In agent applications such as WebShop and ALFWorld, RAIN-Merging can lower the barrier to integrating multiple capabilities and improve the practicality of tool use and structured outputs.

On the risk side, parameter merging can introduce capability drift or safety drift. For example, while improving instruction following, it may alter jailbreak sensitivity, amplify biases present in training data or in LLM-as-judge pipelines, or induce hallucinations tied to specific output formats. Instruction attention as a proxy metric may also encourage myopic optimization for format matching, which is not equivalent to value-aligned safety. Moreover, increased model usability can be misused for mass generation of misleading content, evasion of platform policies, or automated spam. The current method also depends on R1-style special markers and prompting templates; its cross-model and cross-modal generalization remains to be established.

L LIMITATIONS AND FUTURE WORK

Our method has the following limitations. (i) The method relies on R1-style templates and tokenization to extract <think>...</think> for constructing the null space. If a model hides its reasoning (for example, implicit CoT) or adopts different templates, the constraint may weaken or fail. (ii) The instruction and reasoning calibration sets are limited in size and include noise from LLM-as-judge auto-annotation. Distribution shifts across languages or task domains may affect the generalization of the merging coefficients. (iii) Although the KL constraint on the thinking segment helps preserve the reasoning format, non-thinking content and safety-relevant behaviors may still drift, and there is currently no formal safety guarantee. (iv) Experiments focus primarily on the Qwen/DeepSeek families. Applicability to multimodal LLMs, tool use, code-generation settings, and multilingual scenarios requires systematic evaluation.