IF-MAP: INFLUENCE-FUNCTION-INSPIRED AGENT PRIORITIZATION IN MULTI-AGENT SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Improving every agent in an LLM-based multi-agent system can be prohibitively costly, making it important to identify which agent should be prioritized to achieve the greatest system-level gain with limited resources. Failure attribution can inform this decision, but an agent's responsibility for failures does not directly reveal the system-level gains achievable by improving it. We therefore introduce IF-MAP, an influence-function-inspired method for identifying agents with high potential for system-level improvement. For each agent, IF-MAP measures end-to-end loss changes under controlled semantic prompt perturbations, fits a regularized local response model in a low-dimensional prompt-embedding subspace, and uses the estimated response norm as a priority score. This black-box procedure requires no access to model parameters or gradients and produces agent priorities that can guide subsequent improvement effort. We evaluate IF-MAP across six multi-agent workflows spanning code generation, question answering, and mathematical reasoning, using the same priorities for prompt optimization and model upgrades. Optimizing the selected agents' prompts achieves a macro-average gain of 3.81 percentage points, compared with 1.85 for uniform agent selection. Across five target models, upgrading the selected agents yields macro-average gains of 7.62–16.42 percentage points, compared with 3.81–8.62 for uniform agent selection. Despite its probing overhead, IF-MAP reduces Qwen3-8B and DeepSeek-V3.2 token use by 74.2% and 35.2%, respectively, relative to optimizing every agent, and reduces target-model tokens by 63.6–64.9% relative to upgrading every agent. These results support the use of IF-MAP to identify promising agents and direct improvement effort toward greater end-to-end performance gains.

1 INTRODUCTION

Large language model (LLM)-based multi-agent systems distribute complex tasks across specialized agents that generate solutions, gather evidence, coordinate reasoning, and verify intermediate or final outputs (Wu et al., 2023; Zhang et al., 2025a). Their performance depends on both the capabilities of individual agents and the interactions through which their outputs affect the overall computation. Improving such a system therefore offers several possible interventions: replacing an agent's model with a more capable one, fine-tuning its model, refining its instructions, or improving its tools. However, these interventions need not produce comparable benefits at different agents. For example, in a question-answering workflow, refining the answer-generation agent's prompt may improve overall answer accuracy, whereas refining the prompt of the agent that combines candidate answers may yield little benefit or even reduce accuracy. Determining where to direct improvement effort is consequently an important decision in developing multi-agent systems.

This decision becomes particularly consequential under limited development compute, evaluation capacity, or deployment resources. Improving all agents may exceed these resource limits. For example, replacing every agent's model with a more capable one can substantially increase the inference cost of each workflow execution, making a system-wide upgrade prohibitively expensive. We therefore study *agent prioritization*: given a fixed multi-agent system, select the agent whose improvement is expected to produce the largest end-to-end performance gain. This formulation captures the next improvement decision in an ongoing development process. The goal is to identify a high-value target without carrying out the full improvement procedure for every candidate agent.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

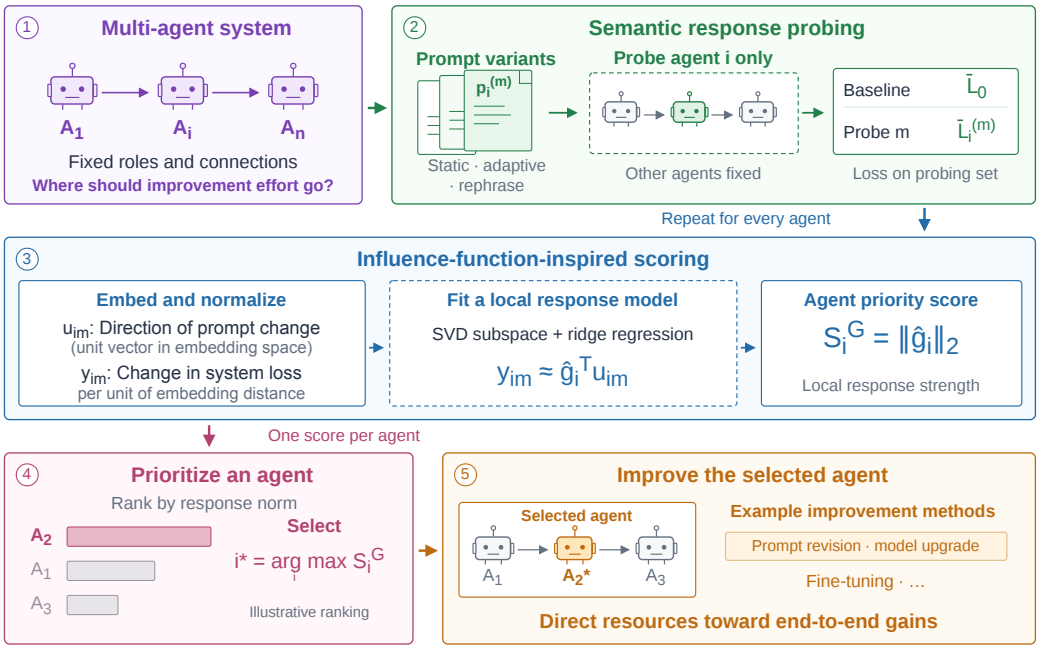


Figure 1: **Overview of IF-MAP.** Given a fixed multi-agent system ①, we perturb one agent's prompt at a time and measure end-to-end losses on the probing set ②. Subspace ridge regression on normalized prompt-embedding displacements and loss responses estimates a local response vector; its norm gives an influence-function-inspired priority score ③. We rank agents by this score ④ and direct improvement effort toward the top-ranked agent ⑤, e.g., through prompt revision, model upgrades, or fine-tuning. The ranking is illustrative.

Existing work provides useful foundations for this decision. Failure-attribution methods identify agents and execution steps associated with unsuccessful outcomes, providing evidence for debugging and system refinement (Zhang et al., 2025b). Nevertheless, responsibility for an observed failure does not uniquely determine an agent's potential for system-level improvement. An agent may frequently contribute to errors while offering little room for improvement, whereas another agent may admit a change that substantially improves downstream behavior. Intervention-oriented studies further demonstrate the importance of evaluating actual recovery and realized system-level gains (Ma et al., 2026; Yoon & Kim, 2026). These observations motivate a complementary objective: estimating which agent is a promising target for system-level improvement under a limited budget.

Influence functions offer a useful conceptual starting point for this objective. Classical influence functions characterize how perturbations to training data affect a learned model and its predictions (Koh & Liang, 2017). We take inspiration from this perturbation-based perspective and investigate whether controlled changes to an agent can reveal information useful for prioritizing its improvement. Because an agent's outputs propagate through the remaining workflow, measuring the resulting end-to-end loss captures responses beyond that agent's immediate behavior. In the black-box setting considered here, we obtain this evidence through workflow executions and semantic representations, without requiring access to model parameters or gradients. The central hypothesis is that an estimated local response can reveal agents with substantial improvement potential, providing a useful signal for allocating improvement effort.

We propose **IF-MAP**, *Influence Function–Inspired Multi-Agent Prioritization for System Improvement*. For each agent, IF-MAP constructs controlled semantic prompt perturbations and measures their effects on end-to-end loss while keeping the remaining system configuration fixed. It represents these perturbations as displacements in a prompt-embedding space and fits a regularized local response model within a low-dimensional subspace. The norm of the estimated response vector then provides an agent-priority score, incorporating both the observed loss changes and the geom-

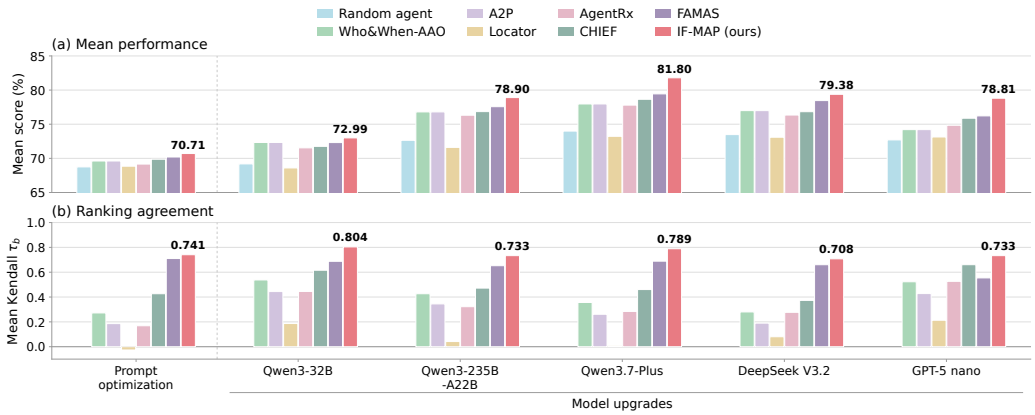


Figure 2: Macro-average performance and ranking agreement across six workflows spanning code generation (HumanEval and MBPP), question answering (NQ, HotpotQA and DROP), and mathematical problem solving (MATH). The base model (before the model upgrade) is Qwen3-8B. Higher mean performance and ranking agreement (Kendall τ_b) indicate better system performance and stronger agreement between the agent-priority and realized-improvement rankings, respectively.

etry of the perturbations. Prompt perturbations serve as executable black-box interventions, and the resulting ranking guides subsequent improvement effort. This makes IF-MAP a prioritization step that can be used alongside existing agent-improvement methods. The score uses estimated local response sensitivity to define a model-implied local intervention opportunity, whose relation to realized gains we evaluate through prompt optimization and model upgrades.

We evaluate IF-MAP across six task-specific multi-agent workflows spanning code generation, question answering, and mathematical reasoning. We assess the usefulness of the same agent priorities through prompt optimization and upgrades to stronger models. Under prompt optimization, IF-MAP achieves a macro-average selected gain of 3.81 percentage points (pp), compared with 1.85 pp for uniform agent selection (Table 1). Across five model-upgrade targets, the selected gains range from 7.62 to 16.42 pp, compared with 3.81–8.62 pp for uniform agent selection (Table 2). The selected mean gains exceed the uniform-selection reference on all six workflows under prompt optimization and each model-upgrade target. The priorities also align with observed improvement gains: mean Kendall τ_b is 0.741 for prompt optimization and 0.708–0.804 across the five model upgrades. These findings support IF-MAP as a useful tool for directing improvement effort toward agents with substantial potential for system-level gains.

Our contributions are threefold:

- **Agent prioritization for system improvement.** We formulate the selection of an agent to prioritize under a limited budget in a fixed multi-agent system, with expected end-to-end improvement as the target quantity.
- **Influence-function-inspired response estimation.** We introduce IF-MAP, a black-box prioritization method that combines semantic perturbations, whole-system response measurements, and a regularized local model to estimate agent response sensitivity.
- **Empirical validation of agent priorities.** We evaluate the same priorities across six workflows through prompt optimization and model upgrades, obtaining higher mean selected gains than uniform agent selection and measuring ranking agreement with observed improvement gains.

2 Agent Prioritization in Multi-Agent Systems

We consider a fixed LLM-based multi-agent system W with n agents, indexed by $i \in \{1, \ldots, n\}$, prompts $p = (p_1, \ldots, p_n)$, and assigned models. An agent corresponds to a persistent workflow role and may execute multiple times within a run. The workflow structure determines how agents exchange information, so changing one agent can affect both its own outputs and the behavior of downstream agents. For a task example x, the system-level score $s(W, x) \in [0, 1]$ defines the loss $\ell(W, x) = 1 - s(W, x)$.

Prioritization objective. Under a limited improvement budget, our goal is to identify the agent whose improvement is expected to yield the largest end-to-end gain. Let $\mathcal{A}_i$ denote an improvement procedure applied to agent i, with the workflow structure and the configurations of all other agents fixed. This procedure may refine the agent's prompt, upgrade or fine-tune its model, or improve its tools. For a given procedure and budget, the expected utility of improving agent i is

$$U_i = \mathbb{E}_{x,\xi}\left[\ell(W, x) - \ell(\mathcal{A}_i(W; \xi), x)\right], \tag{1}$$

where the expectation is over task examples x and randomness ξ in workflow execution and improvement. A prioritization method assigns each agent a score S_i and selects $\hat{\imath} \in \arg\max_i S_i$, aiming for a large $U_{\hat{\imath}}$ without executing the full improvement procedure for every agent. IF-MAP estimates these priorities from development-time response probes; its ranking can then guide subsequent agent improvement.

Evaluating agent priorities. We evaluate the priorities through the gains obtained by improving the selected agents. Let $\widehat{U}_i$ be the mean observed test gain from improving agent i. For a unique top-ranked agent, the *selected gain* is $100\widehat{U}_{\hat{\imath}}$ percentage points (pp). If multiple agents share the highest priority score, we report their average observed improvement gain, corresponding to selecting one of these agents uniformly at random. We compare selected gain with no improvement (zero gain), uniform agent selection ($100n^{-1}\sum_i \widehat{U}_i$), and the empirical best agent ($100\max_i \widehat{U}_i$). Evaluating improvements at every agent provides an offline reference for assessing the selection; these test outcomes are not inputs to the prioritization method.

To assess the full ranking, we compute Kendall's τ_b (Kendall, 1945) between the ranking induced by agent-priority scores and the ranking induced by mean observed test gains from improving each agent. This metric accounts for ties in both rankings and measures whether agents assigned higher priority tend to yield larger improvements. Higher values indicate stronger agreement. Appendix D specifies aggregation and uncertainty estimates.

3 IF-MAP: Influence-function-inspired agent prioritization

IF-MAP is inspired by influence functions, which characterize how perturbations to training data affect a learned model and its predictions (Koh & Liang, 2017). We adapt this perturbation-response perspective to agent prioritization: *semantic changes to one agent's prompt reveal how its behavior affects end-to-end loss*. Our estimator fits a local response model from black-box workflow executions and prompt embeddings, without computing classical training-data influences or accessing model parameters, gradients, or Hessians. As illustrated in Figure 1, we probe each agent, estimate its local response strength, and use the resulting scores to direct improvement effort.

Data subsets. We use disjoint development subsets for probing, optimization, and validation, and reserve the test set for frozen interventions; complete data roles and sizes appear in Appendix A.

3.1 Probing agents through semantic prompt perturbations

For each agent i, we start from its original prompt p_i and construct at most ten formal perturbations $p_i^{(m)}$ spanning static, adaptive, and structure-preserving changes. They retain the agent's role and required output interface, while structural checks reject unusable or duplicate prompts. The ten-slot budget is therefore an upper bound on the realized perturbation pool. Appendix B provides the full perturbation protocol.

Each probe changes only agent i's prompt, keeping the other agents' prompts, all assigned models, and the workflow structure fixed. We execute the full system on the probing set, so the measured response includes changes propagated through agent interactions. Let $\overline{L}_0$ denote the mean baseline loss and $\overline{L}_i^{(m)}$ the mean loss under perturbation m. The signed effect is $\Delta_{im} = \overline{L}_0 - \overline{L}_i^{(m)}$, with positive values indicating improvement.

3.2 Estimating local response strength

Our goal is to turn the probe results into a priority score that reflects how strongly system performance responds to changes in each agent. Influence functions motivate studying such responses

through a local approximation: they estimate how perturbing training data affects a learned model and its downstream loss (Koh & Liang, 2017). Inspired by this perspective, we estimate how changes to an agent's prompt affect the end-to-end loss of the multi-agent system. We learn this response from observed workflow executions, using prompt embeddings to represent the changes.

Representing prompt changes and their effects. Each probe provides two pieces of information: how the prompt changed and how the system's loss changed. A fixed embedding model E maps prompts to L2-normalized vectors, allowing us to represent a prompt change as a displacement in embedding space. Since perturbations can differ in size, we normalize both the displacement and the observed loss change by the embedding distance between the original and perturbed prompts:

$$v_{im} = E(p_i^{(m)}) - E(p_i), \quad u_{im} = \frac{v_{im}}{\|v_{im}\|_2}, \quad y_{im} = \frac{\overline{L}_i^{(m)} - \overline{L}_0}{\|v_{im}\|_2}. \tag{2}$$

Here, v_{im} represents the prompt change, u_{im} describes its direction, and y_{im} measures the resulting loss change per unit of embedding distance. Positive values of y_{im} indicate increased loss, whereas negative values indicate improvement. This normalization compares responses relative to perturbation size, using embedding distance as a proxy for that size. We exclude zero or near-zero displacements and let M_i denote the number of retained measurements.

Combining probes into a stable local response model. A single probe measures the response to one particular change. To combine evidence from several changes, we fit a simple linear model that predicts each normalized loss response from its direction: $y_{im} \approx \hat{g}_i^\top u_{im}$. The vector $\hat{g}_i$ acts as an estimated local slope: it summarizes which directions tend to increase or decrease loss and how strongly the system responds along those directions. This adopts a local perturbation–response perspective inspired by influence functions, while estimating the response directly from finite prompt perturbations rather than differentiating through model training.

There are few probes relative to the embedding dimension, so estimating a separate response for every embedding coordinate would be unreliable. We therefore restrict the fit to a low-dimensional space derived from the observed probe directions. We also use ridge regularization to discourage large, unstable coefficients. These choices allow the estimate to combine the available evidence without fitting an unrestricted high-dimensional response.

Formally, let V_i have rows $u_{im}^\top$ and let y_i collect the responses y_{im}. We construct an orthonormal basis Q_i from the right-singular vectors of the centered direction matrix and set $Z_i = V_i Q_i$. Centering is used only to construct the basis; the regression retains the original directions so that they remain paired with their measured responses. Ridge regression (Hoerl & Kennard, 1970) gives

$$\hat{\beta}_i = (Z_i^\top Z_i + \lambda I)^{-1} Z_i^\top y_i, \qquad \hat{g}_i = Q_i \hat{\beta}_i, \qquad S_i^G = \|\hat{g}_i\|_2. \tag{3}$$

The coefficients $\hat{\beta}_i$ describe the response in the reduced space, and $\hat{g}_i$ expresses that response in the original embedding space. We use Qwen3-Embedding-8B vectors (Zhang et al., 2025c), and a default subspace size of $\max(1, M_i - 1)$, bounded by the available SVD dimensions.

Turning response strength into an agent priority. For a local displacement $\delta \in \mathrm{span}(Q_i)$, the fitted model predicts the loss change $\hat{g}_i^\top \delta$. Under a common local intervention radius r, define the model-implied intervention opportunity as

$$\widehat{U}_i(r) = \max_{\substack{\delta \in \mathrm{span}(Q_i) \\ \|\delta\|_2 \le r}} -\hat{g}_i^\top \delta = r\|\hat{g}_i\|_2 = r S_i^G. \tag{4}$$

The equality follows from Cauchy–Schwarz and, for $\hat{g}_i \neq 0$, is attained at $\delta_i^* = -r\hat{g}_i / \|\hat{g}_i\|_2$. Thus, for a common radius, ranking agents by S_i^G is equivalent to ranking them by the maximum first-order loss reduction predicted within the fitted perturbation subspace. Intuitively, a larger score indicates a steeper local response and a larger predicted loss-reducing direction. Section 4 evaluates how these priorities transfer to realizable prompt revisions and model upgrades.

Comparison with response-magnitude scores. For ablation, we compare G-norm with I-abs and $\mathrm{MeanAbs}$, which summarize the largest and average absolute probe effects, respectively. Unlike these magnitude-only scores, G-norm also uses the directions and sizes of prompt changes. Appendix G gives the formal definitions.

Table 1: Prompt optimization: mean selected gain and mean Kendall τ_b for each dataset and their six-dataset macro means.

Algorithm	HumanEval		MBPP		NQ		DROP		HotpotQA		MATH		Macro (6)	
	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\overline{\Delta}\uparrow$	$\overline{\tau}_b\uparrow$
IF-MAP (ours)	+4.44	0.556	**+7.91**	**0.889**	**+2.14**	0.778	**+4.99**	**1.000**	**+1.53**	0.556	**+1.85**	**0.667**	**+3.81**	**0.741**
Who&When-AAO	+3.23	-0.222	+3.02	0.222	**+2.14**	0.333	**+4.99**	**1.000**	**+1.53**	0.333	+1.38	-0.035	+2.72	0.272
A2P	+3.23	-0.556	+3.02	0.000	**+2.14**	0.333	**+4.99**	**1.000**	**+1.53**	0.333	+1.38	0.004	+2.72	0.186
Adapted locator	+4.55	**0.816**	+5.27	0.516	-0.11	-0.333	+2.98	0.333	-0.55	-0.778	-0.40	-0.713	+1.96	-0.026
AgentRx	+3.74	-0.161	+0.16	-0.411	**+2.14**	0.333	**+4.99**	**1.000**	**+1.53**	0.333	+1.05	-0.084	+2.27	0.169
CHIEF	+4.44	0.444	+3.80	0.365	+1.39	0.111	**+4.99**	**1.000**	**+1.53**	0.333	+1.70	0.305	+2.98	0.427
FAMAS	**+4.65**	0.717	+4.65	0.669	**+2.14**	**0.878**	**+4.99**	**1.000**	**+1.53**	**0.778**	**+1.85**	0.219	+3.30	0.710
Uniform-agent reference†	+3.43	—	+2.85	—	+1.34	—	+1.98	—	+0.33	—	+1.18	—	+1.85	—

Entries report means only; Δ is gain in percentage points. The complete mean±SE results appear in Appendix Table 11. Top ties use uniform tie-averaged gain, and Kendall retains score ties. † denotes the uniform-agent expected gain, a reference average rather than a selector; **bold**/underline marks the largest/second-largest distinct displayed mean per column.

3.3 From agent priorities to system improvement

After ranking agents by S_i^G, we apply the improvement procedure only to a highest-scoring agent while keeping the remaining workflow fixed.

We assess the same priorities through prompt optimization and model upgrades. For prompt optimization, a shared textual-feedback optimizer revises only the selected agent using development-set evidence (Pryzant et al., 2023; Yuksekgonul et al., 2025). Its full protocol appears in Appendix C. For model upgrades, the selected agent receives a stronger model while retaining its original prompt. Final configurations are fixed before test-set evaluation; Section 4 details the evaluation protocols and the model-upgrade results.

4 Experiments

4.1 Experimental setup and baselines

Workflows and data. We study six AFlow-derived workflows (Zhang et al., 2025a): HumanEval and MBPP (code), NQ (factoid question answering), HotpotQA (multi-hop question answering), DROP (discrete reasoning over passages), and MATH (mathematical problem solving) (Chen et al., 2021; Austin et al., 2021; Kwiatkowski et al., 2019; Yang et al., 2018; Dua et al., 2019; Hendrycks et al., 2021). For all six datasets, we start from the benchmark partitions used by AFlow and divide their development data into the roles specified in Appendix A. The appendix also specifies sample counts and node inventories.

Models and configurations. Experiments use Qwen3-8B for workflow execution (Yang et al., 2025), DeepSeek-V3.2 for prompt operations (DeepSeek-AI, 2025), and five model-upgrade targets: Qwen3-32B, Qwen3-235B-A22B, Qwen3.7-Plus (Qwen Team, 2025; 2026), DeepSeek-V3.2, and GPT-5 nano (OpenAI, n.d.). DeepSeek's prompt-operation and upgrade roles are evaluated and accounted for separately. The Qwen3-235B-A22B target specifically uses the Instruct-2507 checkpoint. Full configurations and repetitions appear in Appendix A, with uncertainty aggregation specified in Appendix D.

Baselines. We compare IF-MAP with six failure-attribution, debugging, or weak-agent localization methods adapted to produce workflow-level agent rankings. Who&When-AAO uses the complete conversation log for agent/step failure attribution (Zhang et al., 2025b); A2P uses counterfactual blame reasoning (West et al., 2025); the adapted locator uses the locator component of Shen et al. (Shen et al., 2025); AgentRx uses trajectory-derived constraints and debugging evidence (Barke et al., 2026); and CHIEF and FAMAS provide additional failure-attribution procedures (Wang et al., 2026; Ge et al., 2025). We use the All-at-Once variant of Who&When because the original study reports higher agent-level accuracy when the full conversation log is available. We adapt all six

Table 2: Model upgrades: selecting an agent for upgrade. Panel A averages the metrics over Qwen3-32B, Qwen3-235B-A22B, and Qwen3.7-Plus; panels B and C report DeepSeek-V3.2 and GPT-5 nano separately. Each panel includes its uniform-agent reference (the † row). Mean gain and mean Kendall are reported for each dataset and in macro average.

Algorithm	HumanEval		MBPP		NQ		DROP		HotpotQA		MATH		Macro (6)	
	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\overline{\Delta}\uparrow$	$\overline{\tau}_b\uparrow$
A. Qwen3-32B / Qwen3-235B-A22B / Qwen3.7-Plus (mean)														
IF-MAP (ours)	+17.31	0.778	**+13.41**	**0.889**	**+17.40**	0.778	**+10.19**	**1.000**	**+8.03**	0.630	**+8.80**	**0.578**	**+12.52**	**0.775**
Who&When-AAO	+14.88	0.494	+7.67	0.222	**+17.40**	0.333	**+10.19**	**1.000**	**+8.03**	0.556	+3.76	0.035	+10.32	0.440
A2P	+14.88	0.111	+7.67	0.000	**+17.40**	0.333	**+10.19**	**1.000**	**+8.03**	0.556	+3.76	0.099	+10.32	0.350
Adapted locator	+14.85	**0.816**	+8.84	0.516	-0.47	-0.333	+8.82	0.333	+0.36	-0.556	+2.22	-0.305	+5.77	0.079
AgentRx	**+18.55**	0.605	+1.23	-0.411	**+17.40**	0.333	**+10.19**	**1.000**	**+8.03**	0.556	+3.65	0.018	+9.84	0.350
CHIEF	+17.31	0.717	+8.47	0.365	+11.44	0.111	**+10.19**	**1.000**	**+8.03**	0.556	+6.82	0.347	+10.38	0.516
FAMAS	+12.39	0.444	+9.59	0.669	**+17.40**	**0.878**	**+10.19**	**1.000**	**+8.03**	**0.704**	**+8.80**	0.363	+11.07	0.676
Uniform-agent reference†	+9.97	—	+6.28	—	+6.23	—	+8.13	—	+3.34	—	+5.46	—	+6.57	—
B. DeepSeek-V3.2														
IF-MAP (ours)	**+15.76**	0.778	**+12.33**	**0.889**	**+23.08**	0.778	**+8.97**	**1.000**	**+5.18**	0.556	**+9.59**	**0.246**	**+12.48**	**0.708**
Who&When-AAO	+12.53	0.494	+8.60	0.222	**+23.08**	0.333	**+8.97**	**1.000**	**+5.18**	0.333	+2.20	-0.704	+10.09	0.280
A2P	+12.53	0.111	+8.60	0.000	**+23.08**	0.333	**+8.97**	**1.000**	**+5.18**	0.333	+2.20	-0.639	+10.09	0.190
Adapted locator	+14.24	**0.816**	+8.29	0.516	+1.70	-0.333	+7.63	0.333	+1.47	-0.778	+3.86	-0.074	+6.20	0.080
AgentRx	+15.66	0.605	+0.93	-0.411	**+23.08**	0.333	**+8.97**	**1.000**	**+5.18**	0.333	+2.87	-0.201	+9.45	0.277
CHIEF	**+15.76**	0.717	+7.75	0.365	+15.96	0.111	**+8.97**	**1.000**	**+5.18**	0.333	+6.01	-0.289	+9.94	0.373
FAMAS	+12.73	0.444	+9.84	0.669	**+23.08**	**0.878**	**+8.97**	**1.000**	**+5.18**	**0.778**	**+9.59**	0.192	+11.57	0.660
Uniform-agent reference†	+9.49	—	+5.87	—	+8.89	—	+6.96	—	+3.01	—	+5.28	—	+6.58	—
C. GPT-5 nano														
IF-MAP (ours)	+19.39	0.778	**+17.21**	**0.556**	**+14.10**	0.556	**+6.50**	**1.000**	**+3.03**	0.778	**+11.24**	**0.733**	**+11.91**	**0.733**
Who&When-AAO	+16.77	0.494	+2.09	-0.111	**+14.10**	**1.000**	**+6.50**	**1.000**	**+3.03**	**1.000**	+1.38	-0.246	+7.31	0.523
A2P	+16.77	0.111	+2.09	-0.333	**+14.10**	**1.000**	**+6.50**	**1.000**	**+3.03**	**1.000**	+1.38	-0.211	+7.31	0.428
Adapted locator	+16.67	**0.816**	+11.32	0.415	+2.14	0.333	+5.22	0.333	+1.38	-0.111	+0.69	-0.509	+6.24	0.213
AgentRx	**+20.81**	0.605	+1.01	-0.289	**+14.10**	**1.000**	**+6.50**	**1.000**	**+3.03**	**1.000**	+2.26	-0.159	+7.95	0.526
CHIEF	+19.39	0.717	+7.75	0.233	+10.11	0.778	**+6.50**	**1.000**	**+3.03**	**1.000**	+7.11	0.235	+8.98	0.660
FAMAS	+13.94	0.444	+7.13	0.304	**+14.10**	0.655	**+6.50**	**1.000**	**+3.03**	0.556	**+11.24**	0.364	+9.32	0.554
Uniform-agent reference†	+11.31	—	+5.70	—	+5.86	—	+4.58	—	+1.51	—	+5.87	—	+5.81	—

Entries report means only; the complete mean±SE results appear in Appendix Table 12. Uniform-agent rows are reference averages rather than selectors. Other notation and formatting conventions follow Table 1.

methods to produce workflow-level agent rankings from development trajectories. Appendix F describes the common evaluation setting, aggregation procedure, and method-specific adaptations.

4.2 Main Results

Prompt optimization. Table 1 shows that IF-MAP achieves a macro-average selected gain of 3.81 pp, which is 1.96 pp above the uniform-agent reference and 0.51 pp above FAMAS, the strongest external selector on this measure. Its selected gain exceeds the uniform-agent reference on all six workflows, with the largest margins on MBPP and DROP (5.06 and 3.01 pp, respectively), so the aggregate improvement is not confined to one task family. IF-MAP also has the highest mean Kendall τ_b (0.741 versus 0.710 for FAMAS), showing that it achieves a better overall ordering of agents by their realized prompt-improvement gains, not only a better top-agent selection.

Model upgrades. The prompt-derived priorities are reused without target-specific reprioritization. Table 2 gives macro-average selected gains of 12.52, 12.48, and 11.91 pp for the mean across the three Qwen upgrade targets, DeepSeek-V3.2, and GPT-5 nano, respectively. The corresponding uniform-agent references are 6.57, 6.58, and 5.81 pp, and the mean Kendall τ_b values for IF-MAP are 0.775, 0.708, and 0.733. Target-wise uncertainty, per-node gains, and complete rankings appear in Appendix E. Across the five individual targets, selecting with IF-MAP yields 1.86–2.05 times the gain of upgrading a uniformly chosen agent. IF-MAP also exceeds the strongest external selector

Table 3: Macro-average ablation results over six datasets. IF-MAP uses G-norm and the full probe pool (at most 10 probes per node). Scoring variants replace only the scoring rule; budget variants retain G-norm. Model upgrades use per-node gains averaged over all five target models. Dataset means are equally weighted.

	Prompt optimization		Model upgrades	
Method	Gain (pp) ↑	τ_b ↑	Gain (pp) ↑	τ_b ↑
IF-MAP	+3.812	0.741	+12.394	0.778
Scoring-rule ablation				
w/ I-abs	+3.276	0.752	+9.396	0.641
w/ MeanAbs	+3.576	0.774	+9.090	0.663
Probe-budget comparison				
w/ 3 probes	+2.452	0.289	+6.744	0.178
w/ 5 probes	+2.903	0.415	+7.806	0.304

baseline by 1.45, 0.91, and 2.59 pp in the three displayed panels and has the highest macro-average ranking agreement in each panel. The same priorities are used across all five target models, and their consistent performance suggests that the probes identify agent-level leverage that transfers across replacement models. The uniform-agent reference averages all agents within each dataset, so the gap over that row measures the benefit of deciding *where* to upgrade rather than the capability gain of the target model alone.

4.3 Cost analysis

Semantic probing evaluates each perturbation through the complete workflow, requiring more Qwen3-8B executions than most attribution baselines. It uses only 3.6–34.1% of their DeepSeek-V3.2 requests and 1.5–16.6% of their DeepSeek tokens, shifting computation from diagnostic-model calls to base-model workflow execution.

The same effect is more consequential for downstream improvement. Compared with applying the same prompt optimizer to every agent, selecting one agent with IF-MAP uses 30.69% of the Qwen requests, 25.82% of the Qwen tokens, 57.16% of the DeepSeek requests, and 64.78% of the DeepSeek tokens. For model replacement, FullSwap (i.e., upgrading every agent to the target model) avoids the Qwen3-8B and DeepSeek selection overhead, but invokes the target model for every agent: it uses 2.61–3.16 times as many target-model requests and 2.75–2.85 times as many target-model tokens as IF-MAP. Thus, the practical benefit of prioritization is not simply fewer total calls, but concentrating optimization and target-model calls on a single promising agent while using a cheaper model to decide where that effort should be spent. Complete request and token counts are reported in Appendix Tables 16–18.

4.4 Ablation Study

Table 3 reports two matched ablations, macro-averaged with equal weight across the six workflows. The scoring-rule variants use the same probes and realized intervention gains, changing only how probe responses are converted into priorities. The budget variants all use G-norm but retain three, five, or up to ten formal probes per agent (adopted by our main IF-MAP configuration). For model upgrades, each agent's gain is averaged across all five target models before aggregation. Complete aggregation rules appear in Appendix D.

Scoring-rule ablation. This ablation directly tests the Section 3.2 choice to fit a geometry-aware local response rather than use magnitude-only summaries. Using identical probes and test outcomes, G-norm obtains the highest selected gains: 3.812 pp for prompt optimization and 12.394 pp for model upgrades, exceeding I-abs by 0.536 and 2.998 pp and $\mathrm{MeanAbs}$ by 0.236 and 3.304 pp, respectively. The larger model-upgrade advantage suggests that the direction and displacement information used by G-norm transfers beyond prompt editing. However, $\mathrm{MeanAbs}$ has slightly higher prompt-ranking agreement (0.774 versus 0.741) while yielding lower selected gain, showing that agreement over the full ranking need not identify the best top-ranked intervention target.

Probe-budget comparison. With G-norm, increasing the budget from three to five probes raises selected gain from 2.452 to 2.903 pp for prompt optimization and from 6.744 to 7.806 pp for model upgrades, while mean Kendall τ_b increases by 0.126 for both. The full pool of up to ten probes further raises the gains to 3.812 and 12.394 pp and substantially improves ranking agreement, suggesting that five probes provide insufficient aggregate directional coverage. Per-workflow effects are not monotonic, and the reduced variants retain auxiliary adaptive evaluations; the comparison therefore supports a larger formal response pool rather than a proportional cost reduction.

5 Related work

Attribution, debugging, and improvement. Failure-attribution and debugging methods localize agents or steps associated with unsuccessful outcomes through full-trajectory attribution, counterfactual blame, constraint-based evidence, and other localization signals (Zhang et al., 2025b; West et al., 2025; Shen et al., 2025; Barke et al., 2026; Wang et al., 2026; Ge et al., 2025). We adapt six such methods as external selectors in Section 4.1. DoVer emphasizes that attribution should be validated through realized interventions (Ma et al., 2026). CICA shows more directly that, in its experimental setting, the module assigned the greatest responsibility is not necessarily the module whose repair yields the greatest benefit (Yoon & Kim, 2026). This distinction motivates our work: rather than equating responsibility with improvement potential, we evaluate workflow-level agent priorities through two downstream intervention families.

Optimizing prompts and workflows. ProTeGi and TextGrad use textual feedback to improve prompts or compound computations (Pryzant et al., 2023; Yuksekgonul et al., 2025). AFlow searches executable agentic workflows (Zhang et al., 2025a). HiveMind uses coalition-based credit for online prompt optimization, while Agents that Matter studies removal- and model-replacement-based attribution (Xia et al., 2026; Lu et al., 2026). Their coalition/removal variants require a semantics-preserving executor for workflows with absent agents, which is not consistently available for our workflows; Appendix F gives details. Optimas instead optimizes components with globally aligned local rewards (Wu et al., 2026). Our work focuses on the preceding allocation decision of which agent to improve.

Local response estimation. Our response-estimation procedure is methodologically related to local sensitivity analysis and finite-difference estimation (Borgonovo & Plischke, 2016; Nocedal & Wright, 2006), as well as local surrogate methods such as LIME (Ribeiro et al., 2016). These approaches characterize local input–output behavior through derivatives, perturbations, or locally fitted surrogate models. IF-MAP applies this general principle at the level of semantically structured agent responses: each perturbed response is propagated through the downstream workflow, and its effect on task-level system utility is measured to estimate an agent-level response signal. This signal is then used to prioritize candidate agents for improvement under a fixed intervention budget. We demonstrate that the estimated priorities translate into practical gains through downstream interventions including prompt optimization and model upgrades.

6 Conclusion

We introduced IF-MAP, a black-box method for deciding where limited improvement effort should be directed in a fixed LLM-based multi-agent system. By measuring end-to-end responses to semantic prompt perturbations and fitting a regularized local response model, IF-MAP estimates model-implied local intervention opportunities and identifies agents with promising system-level improvement potential. Across six workflows, its priorities guide both prompt optimization and model upgrades, yielding larger selected gains than uniform agent selection while reducing the expensive downstream calls required by full-agent interventions. These results suggest that local response estimation can provide a practical link between failure analysis and resource-aware multi-agent system improvement. Realized gains depend on the embedding model, local linearity, probe coverage, and the alignment between available interventions and the fitted loss-reducing direction. Future work should translate the continuous response direction into discrete improvements through direction-conditioned generation, iterative edit-and-measure procedures, adaptive probing, and uncertainty-aware selection.

AI USE STATEMENT

Generative AI assistants were used in preparing this draft, including literature retrieval, drafting, code assistance, table construction, and automated consistency checks. The experiment itself also uses LLMs for workflow execution, prompt generation, and diagnostic judgments, as distinguished in the experimental protocol. No missing measurements were synthesized as results. Responsibility for verification of the experimental evidence, references, and final submission remains with the authors.

ETHICS STATEMENT

The study uses existing public benchmarks and model services. Generated code can be unsafe and must be executed in an appropriately restricted environment. Execution transcripts may contain sensitive material in deployment settings; only authorized data should be sent to external prompt-generation or judgment services. Scientific integrity requires retaining infrastructure failures, distinguishing missing from zero results, and disclosing adaptations of published baselines.

REPRODUCIBILITY STATEMENT

Appendices A–I specify the data roles, estimator, intervention protocol, statistical units, and artifact details. Tables are generated from the recorded experimental results and checked for consistency with the protocol described in the appendix.

REFERENCES

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. arXiv preprint arXiv:2108.07732, 2021. URL https://arxiv.org/abs/2108.07732.

Shraddha Barke, Arnav Goyal, Alind Khare, Avaljot Singh, Suman Nath, and Chetan Bansal. AgentRx: Diagnosing AI agent failures from execution trajectories. arXiv preprint arXiv:2602.02475, 2026. URL https://arxiv.org/abs/2602.02475. Version 2 reports acceptance to Findings of EMNLP 2026.

Emanuele Borgonovo and Elmar Plischke. Sensitivity analysis: A review of recent advances. *European Journal of Operational Research*, 248(3):869–887, 2016. doi: 10.1016/j.ejor.2015.06.032.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. arXiv preprint arXiv:2107.03374, 2021. URL https://arxiv.org/abs/2107.03374.

DeepSeek-AI. DeepSeek-V3.2: Pushing the frontier of open large language models. arXiv preprint arXiv:2512.02556, 2025. URL https://arxiv.org/abs/2512.02556.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp.

2368–2378. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1246. URL https://arxiv.org/abs/1903.00161.

Yu Ge, Linna Xie, Zhong Li, Yu Pei, and Tian Zhang. Who is introducing the failure? Automatically attributing failures of multi-agent systems via spectrum analysis. arXiv preprint arXiv:2509.13782, 2025. URL https://arxiv.org/abs/2509.13782. Published at FSE 2026 as "Spectrum-based Failure Attribution for Multi-Agent Systems".

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. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://arxiv.org/abs/2103.03874.

Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970. URL https://www.stat.cmu.edu/technometrics/70-79/VOL-12-01/v1201055.pdf.

M. G. Kendall. The treatment of ties in ranking problems. *Biometrika*, 33(3):239–251, 1945. doi: 10.1093/biomet/33.3.239. URL https://academic.oup.com/biomet/article-abstract/33/3/239/216180.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1885–1894. PMLR, 2017. URL https://arxiv.org/abs/1703.04730.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL https://aclanthology.org/Q19-1026/.

Mingyu Lu, Yushan Huang, Chris Lin, and Su-In Lee. Agents that matter: Optimizing multi-agent LLMs via removal-based attribution. *arXiv preprint arXiv:2605.27621*, 2026. URL https://arxiv.org/abs/2605.27621.

Ming Ma, Jue Zhang, Fangkai Yang, Yu Kang, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. DoVer: Intervention-driven auto debugging for LLM multi-agent systems. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2512.06749.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, 2 edition, 2006. doi: 10.1007/978-0-387-40065-5.

OpenAI. GPT-5 nano model. OpenAI API documentation, n.d. URL https://developers.openai.com/api/docs/models/gpt-5-nano. Accessed 2026-09-22.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7957–7968. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.494. URL https://arxiv.org/abs/2305.03495.

Qwen Team. Qwen3-235B-A22B-Instruct-2507 model card. Hugging Face model card, 2025. URL https://huggingface.co/Qwen/Qwen3-235B-A22B-Instruct-2507. Accessed 2026-09-13.

Qwen Team. Qwen3.7-Plus: Multimodal agent intelligence, May 2026. URL https://qwen.ai/blog?id=qwen3.7-plus.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144. ACM, 2016. doi: 10.1145/2939672.2939778. URL https://arxiv.org/abs/1602.04938.

Ming Shen, Raphael Shu, Anurag Pratik, James Gung, Yubin Ge, Monica Sunkara, and Yi Zhang. Optimizing LLM-based multi-agent system with textual feedback: A case study on software development. In *AI Agents: Capabilities and Safety Workshop at COLM*, 2025. URL https://arxiv.org/abs/2505.16086v2.

Yawen Wang, Wenjie Wu, Junjie Wang, and Qing Wang. From flat logs to causal graphs: Hierarchical failure attribution for LLM-based multi-agent systems. arXiv preprint arXiv:2602.23701, 2026. URL https://arxiv.org/abs/2602.23701.

Alva West, Yixuan Weng, Minjun Zhu, Zhen Lin, Zhiyuan Ning, and Yue Zhang. Abduct, act, predict: Scaffolding causal inference for automated failure attribution in multi-agent systems. arXiv preprint arXiv:2509.10401, 2025. URL https://arxiv.org/abs/2509.10401.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. arXiv preprint arXiv:2308.08155, 2023. URL https://arxiv.org/abs/2308.08155. Conference version published at COLM 2024.

Shirley Wu, Parth Sarthi, Shiyu Zhao, Aaron Lee, Herumb Shandilya, Adrian Mladenic Grobelnik, Nurendra Choudhary, Eddie Huang, Karthik Subbian, Linjun Zhang, Diyi Yang, James Y Zou, and Jure Leskovec. Optimas: Optimizing compound AI systems with globally aligned local rewards. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2507.03041.

Yihan Xia, Taotao Wang, Shengli Zhang, Zhangyuhua Weng, Bin Cao, and Soung Chang Liew. HiveMind: Contribution-guided online prompt optimization of LLM multi-agent systems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(35):29767–29774, 2026. doi: 10.1609/aaai.v40i35.40222. URL https://arxiv.org/abs/2512.06432.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. arXiv preprint arXiv:2505.09388, 2025. URL https://arxiv.org/abs/2505.09388.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380. Association for Computational Linguistics, 2018. doi: 10.18653/v1/D18-1259. URL https://arxiv.org/abs/1809.09600.

Jeonghun Yoon and Dongchan Kim. Diagnosis is not prescription: Linguistic co-adaptation explains patching hazards in LLM pipelines. arXiv preprint arXiv:2605.21958, 2026. URL https://arxiv.org/abs/2605.21958.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Pan Lu, Zhi Huang, Carlos Guestrin, and James Zou. Optimizing generative AI by backpropagating language model feedback. *Nature*, 639:609–616, 2025. doi: 10.1038/s41586-025-08661-4. URL https://arxiv.org/abs/2406.07496. Earlier preprint: TextGrad: Automatic "Differentiation" via Text.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. AFlow: Automating agentic workflow generation. In *International Conference on Learning Representations*, 2025a. URL https://arxiv.org/abs/2410.10762.

Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, and Qingyun Wu. Which agent causes task failures and when? On automated failure attribution of LLM multi-agent systems. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 76583–76599. PMLR, 2025b. URL https://arxiv.org/abs/2505.00212.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 Embedding: Advancing text embedding and reranking through foundation models. arXiv preprint arXiv:2506.05176, 2025c. URL https://arxiv.org/abs/2506.05176.

A EXPERIMENTAL PROTOCOL AND DETAILS

This appendix specifies the experimental protocol used for all reported results, including data roles, model configurations, repetitions, statistical aggregation, baseline adaptations, and cost accounting.

A.1 DATA ROLES AND WORKFLOW NODES

The probing set supplies examples for constructing prompt perturbations, measuring their effects, and proposing subsequent prompt revisions. The optimization set scores candidate revisions and selects the candidate to consider for acceptance. The validation set supplies the additional acceptance signal specified in Appendix C. The test set evaluates the final intervention and is excluded from agent prioritization and candidate generation. Following the benchmark partitions used by AFlow, the probing, optimization, and validation sets originate from separate portions of each benchmark's development partition. Because the optimization and validation sets are used adaptively, both are treated as development data. Model-upgrade experiments reuse the priorities from the probing set and evaluate gains on the test set, without using the optimization or validation sets.

Table 4: Sizes of the probing, optimization, validation, and test sets. The optimization and validation sets are capped within their separate thirds for ordinary workflows. MATH uses the reduced execution sizes for the performance experiments; the full-sample row is used only for the cost scenario in Appendix H.

Workflow	Probing	Optimization	Validation	Test	Nodes
HumanEval	45	43	43	33	3
MBPP	115	113	113	86	4
NQ	268	266	266	266	3
DROP	268	266	266	200	2
HotpotQA	268	266	266	200	3
MATH (reduced execution)	80	80	80	121	5
MATH (full-sample cost scenario)	162	161	161	121	5

For a development partition of size N, the ordinary split starts with $q = \lfloor N/3 \rfloor$. The probing set receives $q + (N - 3q)$ examples; the optimization and validation sets occupy the second and third blocks of q, respectively. Unused examples above either cap are not reassigned to the other block. Partition sampling uses seed 42; this is distinct from whether an LLM request specifies an inference seed. The MATH reduction uses a fixed 80-example subset of the probing set. The cost scenario in Appendix H instead uses all 162/161/161 probing/optimization/validation examples, respectively, and all 121 test examples.

Table 5: Node inventory for the six task-specific AFlow-derived workflows. A named node is the unit of intervention and can execute repeatedly in a trace.

Workflow	Intervenable nodes
HumanEval	CustomCodeGenerate, ScEnsemble, Test
MBPP	CodeGenerate, ScEnsemble, Test, FixCode
NQ	AnswerGenerate, ScEnsemble, FormatAnswer
DROP	Solve, Format
HotpotQA	AnswerGenerate, ScEnsemble, FormatAnswer
MATH	Programmer, RefineAnswer, DetailedSolution, GenerateSolution, ScEnsemble

HumanEval and MBPP judge code by the workflow's execution/testing protocol (Chen et al., 2021; Austin et al., 2021). Our NQ adaptation targets short factual answers (Kwiatkowski et al., 2019); HotpotQA involves multi-hop questions (Yang et al., 2018); DROP requires discrete reasoning over passages (Dua et al., 2019); MATH assesses competition-style mathematical solutions (Hendrycks et al., 2021). Each workflow uses the benchmark subset and workflow-specific scoring implementation described in this appendix. Exact example IDs, normalizers, the execution harness, and prompt snapshots form part of the complete replication record.

A.2 MODELS, CONTEXTS, AND REPETITIONS

The scoring encoder is Qwen3-Embedding-8B with 512 output dimensions (Zhang et al., 2025c). The ridge coefficient in Equation 3 is fixed to $\lambda = 1.0$ for every workflow and run. The local execution model is Qwen3-8B, with thinking disabled in the intended local request path. The ordinary configuration uses an 8,192-token context, an input budget of 6,144, and an output limit of 2,048. MATH uses a 16,384-token context with input and output budgets of 8,192 each. These are limits rather than measured response lengths. Only nodes designed to sample multiple candidate answers use a nonzero execution temperature (0.3 in the prompt-optimization configuration); other execution nodes use 0. Prompt rewriting uses DeepSeek-V3.2 at 0.7. Generation and workflow evaluation are different model roles.

Each prioritization algorithm is run independently three times on every workflow. Prompt-optimization and model-upgrade interventions are each run independently five times for every agent and dataset; the per-agent gains use the mean and standard error across these five runs. Method-level selected-gain and ranking summaries are aggregated over the three prioritization runs. The five model-upgrade target identities are Qwen3-32B, Qwen3-235B-A22B-Instruct-2507, Qwen3.7-Plus, DeepSeek-V3.2, and GPT-5 nano. Qwen3.7-Plus denotes the service alias recorded during the experiments, with provider and version metadata retained for reproducibility. The two additional targets use the OpenRouter identifiers `deepseek/deepseek-v3.2` and `openai/gpt-5-nano`, respectively, with five test executions per node on each workflow. DeepSeek upgrades disable reasoning and use the workflow's node-specific temperatures. GPT-5 nano uses low reasoning effort and a 16,384-token output cap, including reasoning tokens; its requests omit temperature, top-p, and seed.

B THE TEN-SLOT PERTURBATION SPECIFICATION

Table 6 describes the intended mechanism of each formal slot. Generated prompts are additionally assessed through prompt diffs and structural guards because a revision may combine multiple semantic changes. The numerical potential scores used in this paper are G-norm, I-abs, and $\mathrm{MeanAbs}$; the mechanism names identify intervention types.

Table 6: Formal mechanisms, grouped by construction family. Auxiliary response measurements support the adaptive construction process.

Mechanism	Intended revision
Downstream interface	Clarify what the next operator needs and how this operator must provide it, without changing the workflow's interface.
Decomposition control	Make the allocation or ordering of reasoning steps explicit, keeping the original task and output requirements.
Ambiguity resolution	Resolve an underspecified term, priority, or instruction that could admit incompatible interpretations.
Semantic precision	Sharpen the meaning of an existing instruction rather than merely adding emphasis.
Verification	Add or refine a task-relevant internal check while preserving the required externally visible answer.
Edge cases	Clarify boundary conditions or exceptional cases in a task-general way, not by copying development answers.
Adaptive counterfactual repair	Use a prior's changed outcomes on the probing set to identify and address a plausible repair opportunity.
Adaptive regression repair	Protect original behavior damaged by an otherwise useful prior, using development evidence.
Adaptive score-conditioned	Revise a prior in light of its measured response and the development evidence available for that node.
Sectionwise rephrase	Rephrase editable logical sections separately, keeping protected fragments and template placeholders intact.

Within adaptive construction, up to 36 auxiliary candidates are generated from 12 initial mechanisms, three slots per mechanism. A single development evaluation estimates each candidate's loss

change on a shared subset of the probing set. The subset rule uses approximately a quarter of the probing set, at least 40 examples where available, combining low-scoring and random examples without overlap. MATH's restarted adaptive construction instead uses 32 examples. These internal measurements are performed once per candidate, not three times; formal response estimation is a separate repeated step.

Adaptive construction considers up to the five highest-ranked priors with nonnegative point estimates. A generated adaptive prompt must differ from its selected prior; repeated duplication advances to the next eligible prior. If no distinct candidate is produced, the slot is recorded as redundant. Ordinary mechanisms, sectionwise rephrasing, and adaptive candidates enter the formal response pool regardless of the sign of their measured effect, and no test-set screening is used. The resulting pool contains up to ten distinct prompts per node.

Structural checks preserve placeholders and protected content, require a changed English prompt with parseable generated fields, and reject unusable responses. Formal-pool generation and downstream optimizer validation are different operations with different length policies; the optimizer policy is specified below. Missing or failed measurements are recorded as unavailable rather than assigned an observed effect.

C Common prompt-optimization intervention

To evaluate all selectors with a common downstream intervention, we use a shared optimizer inspired by textual-feedback search. The procedure is as follows.

1. Hold the original node prompt and its agent-prioritization evidence fixed. Rank eligible directions by absolute mean effect on the probing set, using lower uncertainty to break ties. The optimizer uses $|\overline{\Delta}| \geq 0.005$ as its eligibility threshold. The evidence pool remains unchanged during optimization, and each direction is considered at most once.
2. For the next direction, provide the original prompt, current accepted prompt, measured perturbation, a concise LLM-generated description of their difference, and up to three direction-consistent reversal examples from the probing set. Positive evidence uses improved examples; negative evidence uses degraded examples. When none are available, the request omits examples. Expected answers refer to final task outcomes.
3. Ask DeepSeek-V3.2 for four complete local revisions. All four receive the same evidence, with different implementations of the relevant change. Structural validation checks placeholders and protected fragments, rejects the unchanged current prompt, and applies the recorded length interval from 0.2 times the original token length to $\max(512, 3 \times \text{original token length})$. Generation is retried up to three times to obtain four valid candidates; if this is unsuccessful, the current state is retained and the procedure advances to the next direction.
4. Evaluate each candidate on the optimization set with three repeat-specific evaluations. Select the lowest mean loss. Only that candidate can proceed to validation, where the rule below determines acceptance.
5. On acceptance, update the current state and its optimization- and validation-set reference losses. On rejection, retain the previous state. Stop after six accepted updates or exhaustion of eligible directions. Rejected directions do not count toward the six-update limit. MATH's reduced record uses three candidates and at most five attempted directions instead.
6. Hold the final accepted prompt fixed for test-set evaluation. If it equals the original prompt, use the same per-repeat baseline losses, yielding zero gain by construction.

C.1 Acceptance rule

Let $L_{\text{opt}}^{\text{cur}}$ be the mean loss on the optimization set of the current accepted state and L_{opt}^{*} that of the beam winner. For the validation set, let $L_{\text{val}}^{\text{orig}}$, $L_{\text{val}}^{\text{prev}}$, and L_{val}^{*} be the mean losses of the original prompt, previous accepted state, and beam winner, respectively. Define the optimization gain o and

validation gains v_0, v_{prev}, and v by

$$o = L_{\text{opt}}^{\text{cur}} - L_{\text{opt}}^{*}, \tag{5}$$

$$v_0 = L_{\text{val}}^{\text{orig}} - L_{\text{val}}^{*}, \qquad v_{\text{prev}} = L_{\text{val}}^{\text{prev}} - L_{\text{val}}^{*}, \tag{6}$$

$$v = \begin{cases} v_0, & \text{before the first accepted update,} \\ \min(v_0, v_{\text{prev}}), & \text{otherwise.} \end{cases} \tag{7}$$

The validation set is evaluated only when $o > -0.005$ (up to numerical tolerance). The recorded acceptance rule is

$$\text{accept}(o, v) = \begin{cases} \text{true}, & o > 0 \ \wedge \ [v \geq 0 \ \vee \ o - \max(0, -v) > 0.005], \\ \text{true}, & -0.005 < o \leq 0 \ \wedge \ v > 0 \ \wedge \ (o + v)/2 > 0.0025, \\ \text{false}, & \text{otherwise.} \end{cases} \tag{8}$$

The numerical implementation uses an epsilon of 10^{-12} at comparisons. Thresholds are fractions of score, so 0.005 is 0.5 pp. The second branch allows improvement on the validation set to compensate for a small decline on the optimization set. The final exported prompt is the final accepted state, which need not be the state with the best optimization-set score alone. The minimum comparison on the validation set guards against both the frozen original and the previous accepted state; a first candidate is not accepted unconditionally. The rule is used for development-set candidate selection; generalization is evaluated after the final state is fixed for test-set evaluation.

C.2 Representative generation instruction

The following condensed template summarizes the generator's semantics. The replication artifact will provide the exact serialized prompts and API parameters used in the experiments.

> You are optimizing one operator's system prompt in a fixed multi-agent workflow. Lower loss is better. You receive the frozen original prompt, current accepted prompt, measured perturbation prompt, its signed effect on the probing set, and a concise description of the change. Any supplied examples are observations from the probing set; expected answers refer to the final task outcome.
>
> For positive evidence, incorporate the relevant beneficial difference into the current prompt. For negative evidence, do not copy the harmful difference; protect the original behavior it damaged. Add a local safeguard only when the evidence supports a task-general risk. Do not invent an opposite rule merely because the measured direction is negative.
>
> Return four complete, distinct local revisions using the same full evidence. Preserve every required placeholder and protected fragment. Do not summarize or globally rewrite the current prompt. Do not copy example-specific answers. Vary how the supported change is implemented, not which evidence is visible. Output only a JSON list of four prompt strings, without commentary or fences.

D Statistical units and recomputation

For prioritization run $r \in \{1, 2, 3\}$, let $A_r = \arg\max_i S_i^{(r)}$ be the set of top-tied nodes. The reported selected gain for that group is $g_r = 100|A_r|^{-1} \sum_{i \in A_r} \widehat{U}_i$. This quantity is the expected gain under uniform tie-breaking. Kendall t_r compares the complete score vector with the vector of mean intervention gains. For either $z = g$ or $z = t$, the three-run summary is

$$\overline{z} = \frac{1}{3} \sum_{r=1}^{3} z_r, \qquad \text{SE}_{\text{sel}}(z) = \frac{1}{\sqrt{3}} \sqrt{\frac{\sum_r (z_r - \overline{z})^2}{2}}. \tag{9}$$

This aggregation describes variability across the three independent prioritization runs.

We report Kendall against the vector of mean intervention gains. For each node and intervention, the five repeated gains $u_1, \ldots, u_5$ determine the mean intervention gain and its standard error. Sample variance uses divisor $T - 1$ with $T = 5$, and $\text{SE}_{\text{test}} = \sqrt{s^2/T}$.

The uniform-agent reference first averages nodes within each repeat and then estimates its SE across repeats; individual node SEs cannot be averaged to reconstruct it because covariance matters. In the

main model-upgrade table, the Qwen panel averages the selected-gain and Kendall metrics across the three Qwen targets. For the ablations, we instead average all five target models' per-node gains *before* computing selected gain and Kendall.

The six-workflow macro statistics weight workflows equally, not individual examples or nodes. A two-node workflow such as DROP has only one pair; perfect Kendall on it is less demanding than a complete five-node ordering. Point estimates are interpreted in light of sample sizes, shared baselines, and ties; standard-error bars summarize variability rather than define a standalone significance test.

E ALL-NODE GAIN REFERENCES AND COMPLETE UNCERTAINTY TABLES

Table 7: Per-node mean test gains (pp) for prompt optimization and all five model-upgrade targets used as evaluation references, rounded to three decimals. The listed nodes cover all six workflows. MeanGain averages each node's five model-upgrade means before ranking. Each prompt-optimization and model-upgrade entry averages five intervention runs. No uncertainty is implied by this mean-only table.

			Model-upgrade target / average					
Workflow	Node	Prompt optimization	32B	3.7-plus	235B	DeepSeek V3.2	GPT-5 nano	MeanGain
HumanEval	CustomCodeGenerate	+4.242	+15.152	+27.273	+24.242	+18.788	+24.849	+22.061
	ScEnsemble	+1.212	+0.606	+0.000	+0.000	+0.000	+0.606	+0.242
	Test	+4.849	+4.242	+9.091	+9.091	+9.697	+8.485	+8.121
MBPP	CodeGenerate	+3.023	+6.744	+9.070	+7.209	+8.605	+2.093	+6.744
	ScEnsemble	+0.000	−0.233	−0.233	−0.465	+0.233	−0.465	−0.233
	Test	+7.907	+7.442	+21.168	+11.628	+12.325	+17.210	+13.955
	FixCode	+0.465	+0.930	+8.553	+3.488	+2.325	+3.954	+3.850
NQ	AnswerGenerate	+2.141	+10.634	+19.780	+21.782	+23.085	+14.097	+17.876
	ScEnsemble	−0.111	−0.305	−0.122	−0.984	+1.705	+2.138	+0.486
	FormatAnswer	+2.002	+1.298	+1.326	+2.630	+1.881	+1.343	+1.696
DROP	Solve	+4.994	+6.633	+12.822	+11.126	+8.969	+6.503	+9.211
	Format	−1.038	+3.425	+7.448	+7.346	+4.947	+2.658	+5.165
HotpotQA	AnswerGenerate	+1.534	+5.739	+10.463	+7.878	+5.182	+3.033	+6.459
	ScEnsemble	−0.547	+0.403	+0.145	+0.533	+1.472	+1.378	+0.786
	FormatAnswer	+0.000	+0.359	+2.314	+2.248	+2.387	+0.116	+1.485
MATH	Programmer	+1.377	+1.653	+5.234	+4.408	+2.204	+1.378	+2.975
	RefineAnswer	+1.551	+1.377	+11.771	+1.653	+4.187	+7.493	+5.296
	DetailedSolution	+1.728	+0.551	+8.264	+9.091	+3.526	+6.832	+5.653
	GenerateSolution	+2.002	+4.959	+13.751	+13.499	+12.286	+13.113	+11.522
	ScEnsemble	−0.755	−1.423	+3.581	+3.581	+4.187	+0.551	+2.096

The all-node outcomes provide offline reference values for evaluating agent selection. They are measured intervention gains and are excluded from all selector inputs.

Table 8: Expanded view of the IF-MAP uncertainty in Table 2. Every $\pm$ is $\mathrm{SE}_{\mathrm{sel}}$ across the three prioritization runs against the per-node mean gains computed from five intervention runs. It is not $\mathrm{SE}_{\mathrm{test}}$. Zero $\mathrm{SE}_{\mathrm{sel}}$ can result from identical selections despite nonzero test execution variability. The table covers all five upgrade targets. Target aliases match Table 2.

Workflow	Target	G-norm gain (pp)	G-norm τ_b	Workflow	Target	G-norm gain (pp)	G-norm τ_b
HumanEval	32B	$+11.515 \pm 3.636$	0.778 ± 0.222	DROP	32B	$+6.633 \pm 0.000$	1.000 ± 0.000
	3.7-plus	$+21.212 \pm 6.061$	0.778 ± 0.222		3.7-plus	$+12.822 \pm 0.000$	1.000 ± 0.000
	235B	$+19.192 \pm 5.051$	0.778 ± 0.222		235B	$+11.126 \pm 0.000$	1.000 ± 0.000
	DeepSeek-V3.2	$+15.758 \pm 3.030$	0.778 ± 0.222		DeepSeek-V3.2	$+8.969 \pm 0.000$	1.000 ± 0.000
	GPT-5 nano	$+19.394 \pm 5.455$	0.778 ± 0.222		GPT-5 nano	$+6.503 \pm 0.000$	1.000 ± 0.000
MBPP	32B	$+7.442 \pm 0.000$	0.889 ± 0.111	HotpotQA	32B	$+5.739 \pm 0.000$	0.778 ± 0.222
	3.7-plus	$+21.168 \pm 0.000$	0.889 ± 0.111		3.7-plus	$+10.463 \pm 0.000$	0.556 ± 0.222
	235B	$+11.628 \pm 0.000$	0.889 ± 0.111		235B	$+7.878 \pm 0.000$	0.556 ± 0.222
	DeepSeek-V3.2	$+12.325 \pm 0.000$	0.889 ± 0.111		DeepSeek-V3.2	$+5.182 \pm 0.000$	0.556 ± 0.222
	GPT-5 nano	$+17.210 \pm 0.000$	0.556 ± 0.111		GPT-5 nano	$+3.033 \pm 0.000$	0.778 ± 0.222
NQ	32B	$+10.634 \pm 0.000$	0.778 ± 0.222	MATH	32B	$+3.765 \pm 1.194$	0.600 ± 0.115
	3.7-plus	$+19.780 \pm 0.000$	0.778 ± 0.222		3.7-plus	$+13.091 \pm 0.660$	0.733 ± 0.176
	235B	$+21.782 \pm 0.000$	0.778 ± 0.222		235B	$+9.550 \pm 3.949$	0.400 ± 0.115
	DeepSeek-V3.2	$+23.085 \pm 0.000$	0.778 ± 0.222		DeepSeek-V3.2	$+9.587 \pm 2.700$	0.246 ± 0.186
	GPT-5 nano	$+14.097 \pm 0.000$	0.556 ± 0.222		GPT-5 nano	$+11.240 \pm 1.873$	0.733 ± 0.176

Table 9: Agent rankings by mean model-upgrade test gain for HumanEval, MBPP, NQ. Each row ranks nodes from highest to lowest gain for one target model. Parentheses give mean gains in percentage points; $=$ denotes a tie. Gains use the references in Table 7; rankings are computed before display rounding.

Workflow	Upgrade target	Agent ranking (mean gain, pp)
HumanEval	Qwen3-32B	CustomCodeGenerate (+15.152) > Test (+4.242) > ScEnsemble (+0.606)
	Qwen3-235B-A22B	CustomCodeGenerate (+24.242) > Test (+9.091) > ScEnsemble (+0.000)
	Qwen3.7-Plus	CustomCodeGenerate (+27.273) > Test (+9.091) > ScEnsemble (+0.000)
	DeepSeek-V3.2	CustomCodeGenerate (+18.788) > Test (+9.697) > ScEnsemble (+0.000)
	GPT-5 nano	CustomCodeGenerate (+24.849) > Test (+8.485) > ScEnsemble (+0.606)
MBPP	Qwen3-32B	Test (+7.442) > CodeGenerate (+6.744) > FixCode (+0.930) > ScEnsemble (-0.233)
	Qwen3-235B-A22B	Test (+11.628) > CodeGenerate (+7.209) > FixCode (+3.488) > ScEnsemble (-0.465)
	Qwen3.7-Plus	Test (+21.168) > CodeGenerate (+9.070) > FixCode (+8.553) > ScEnsemble (-0.233)
	DeepSeek-V3.2	Test (+12.325) > CodeGenerate (+8.605) > FixCode (+2.325) > ScEnsemble (+0.233)
	GPT-5 nano	Test (+17.210) > FixCode (+3.954) > CodeGenerate (+2.093) > ScEnsemble (-0.465)
NQ	Qwen3-32B	AnswerGenerate (+10.634) > FormatAnswer (+1.298) > ScEnsemble (-0.305)
	Qwen3-235B-A22B	AnswerGenerate (+21.782) > FormatAnswer (+2.630) > ScEnsemble (-0.984)
	Qwen3.7-Plus	AnswerGenerate (+19.780) > FormatAnswer (+1.326) > ScEnsemble (-0.122)
	DeepSeek-V3.2	AnswerGenerate (+23.085) > FormatAnswer (+1.881) > ScEnsemble (+1.705)
	GPT-5 nano	AnswerGenerate (+14.097) > ScEnsemble (+2.138) > FormatAnswer (+1.343)

Table 10: Agent rankings by mean model-upgrade test gain for DROP, HotpotQA, MATH. Each row ranks nodes from highest to lowest gain for one target model. Parentheses give mean gains in percentage points; = denotes a tie. Gains use the references in Table 7; rankings are computed before display rounding.

Workflow	Upgrade target	Agent ranking (mean gain, pp)
DROP	Qwen3-32B	Solve (+6.633) > Format (+3.425)
	Qwen3-235B-A22B	Solve (+11.126) > Format (+7.346)
	Qwen3.7-Plus	Solve (+12.822) > Format (+7.448)
	DeepSeek-V3.2	Solve (+8.969) > Format (+4.947)
	GPT-5 nano	Solve (+6.503) > Format (+2.658)
HotpotQA	Qwen3-32B	AnswerGenerate (+5.739) > ScEnsemble (+0.403) > FormatAnswer (+0.359)
	Qwen3-235B-A22B	AnswerGenerate (+7.878) > FormatAnswer (+2.248) > ScEnsemble (+0.533)
	Qwen3.7-Plus	AnswerGenerate (+10.463) > FormatAnswer (+2.314) > ScEnsemble (+0.145)
	DeepSeek-V3.2	AnswerGenerate (+5.182) > FormatAnswer (+2.387) > ScEnsemble (+1.472)
	GPT-5 nano	AnswerGenerate (+3.033) > ScEnsemble (+1.378) > FormatAnswer (+0.116)
MATH	Qwen3-32B	GenerateSolution (+4.959) > Programmer (+1.653) > RefineAnswer (+1.377) > DetailedSolution (+0.551) > ScEnsemble (-1.423)
	Qwen3-235B-A22B	GenerateSolution (+13.499) > DetailedSolution (+9.091) > Programmer (+4.408) > ScEnsemble (+3.581) > RefineAnswer (+1.653)
	Qwen3.7-Plus	GenerateSolution (+13.751) > RefineAnswer (+11.771) > DetailedSolution (+8.264) > Programmer (+5.234) > ScEnsemble (+3.581)
	DeepSeek-V3.2	GenerateSolution (+12.286) > RefineAnswer (+4.187) = ScEnsemble (+4.187) > DetailedSolution (+3.526) > Programmer (+2.204)
	GPT-5 nano	GenerateSolution (+13.113) > RefineAnswer (+7.493) > DetailedSolution (+6.832) > Programmer (+1.378) > ScEnsemble (+0.551)

Table 11: Prompt optimization with standard errors: complete mean±SE version of Table 1. Each dataset has separate columns for selected gain and Kendall τ_b; the final pair gives six-dataset macro means.

Algorithm	HumanEval		MBPP		NQ		DROP		HotpotQA		MATH		Macro (6)	
	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\overline{\Delta}\uparrow$	$\overline{\tau}_b\uparrow$
IF-MAP (ours)	+4.44 ±0.20	0.556 ±0.222	**+7.91** ±0.00	**0.889** ±0.111	**+2.14** ±0.00	0.778 ±0.222	**+4.99** ±0.00	**1.000** ±0.000	**+1.53** ±0.00	0.556 ±0.222	**+1.85** ±0.15	**0.667** ±0.067	**+3.81** ±0.05	**0.741** ±0.056
Who&When-AAO	+3.23 ±1.01	-0.222 ±0.401	+3.02 ±0.00	0.222 ±0.111	**+2.14** ±0.00	0.333 ±0.000	**+4.99** ±0.00	**1.000** ±0.000	**+1.53** ±0.00	0.333 ±0.000	+1.38 ±0.00	-0.035 ±0.070	+2.72 ±0.17	0.272 ±0.075
A2P	+3.23 ±1.01	-0.556 ±0.222	+3.02 ±0.00	0.000 ±0.000	**+2.14** ±0.00	0.333 ±0.000	**+4.99** ±0.00	**1.000** ±0.000	**+1.53** ±0.00	0.333 ±0.000	+1.38 ±0.00	0.004 ±0.164	+2.72 ±0.17	0.186 ±0.028
Adapted locator	+4.55 ±0.00	**0.816** ±0.000	+5.27 ±2.64	0.516 ±0.289	-0.11 ±0.00	-0.333 ±0.000	+2.98 ±2.01	0.333 ±0.667	-0.55 ±0.00	-0.778 ±0.222	-0.40 ±0.36	-0.713 ±0.059	+1.96 ±0.46	-0.026 ±0.038
AgentRx	+3.74 ±0.51	-0.161 ±0.342	+0.16 ±0.16	-0.411 ±0.297	**+2.14** ±0.00	0.333 ±0.000	**+4.99** ±0.00	**1.000** ±0.000	**+1.53** ±0.00	0.333 ±0.000	+1.05 ±0.33	-0.084 ±0.221	+2.27 ±0.10	0.169 ±0.063
CHIEF	+4.44 ±0.20	0.444 ±0.294	+3.80 ±2.18	0.365 ±0.279	+1.39 ±0.75	0.111 ±0.222	**+4.99** ±0.00	**1.000** ±0.000	**+1.53** ±0.00	0.333 ±0.000	+1.70 ±0.18	0.305 ±0.058	+2.98 ±0.47	0.427 ±0.104
FAMAS	**+4.65** ±0.20	0.717 ±0.199	+4.65 ±1.63	0.669 ±0.122	**+2.14** ±0.00	**0.878** ±0.061	**+4.99** ±0.00	**1.000** ±0.000	**+1.53** ±0.00	**0.778** ±0.222	**+1.85** ±0.15	0.219 ±0.114	+3.30 ±0.21	0.710 ±0.067
Uniform-agent reference†	+3.43 ±0.40	—	+2.85 ±0.56	—	+1.34 ±0.36	—	+1.98 ±0.18	—	+0.33 ±0.36	—	+1.18 ±0.28	—	+1.85 ±0.15	—

Mean above, ±SE below; Δ is gain in percentage points. Method-row SEs summarize three prioritization runs; per-agent intervention means are computed from five runs. Top ties use uniform tie-averaged gain, and Kendall retains score ties. †: uniform-agent expected gain, shown as a reference average rather than a selector, with $\mathrm{SE}_{\mathrm{test}}$ across the five intervention runs instead; **Bold**/underline: largest/second-largest distinct displayed mean per column within each panel.

Table 12: Model upgrades with standard errors: complete mean±SE version of Table 2. Each panel uses one target model and its own uniform-agent reference.

Algorithm	HumanEval		MBPP		NQ		DROP		HotpotQA		MATH		Macro (6)	
	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\overline{\Delta}\uparrow$	$\overline{\tau}_b\uparrow$
A. Qwen3-32B														
IF-MAP (ours)	+11.52 ±3.64	0.778 ±0.222	**+7.44** ±0.00	**0.889** ±0.111	**+10.63** ±0.00	0.778 ±0.222	**+6.63** ±0.00	**1.000** ±0.000	**+5.74** ±0.00	0.778 ±0.222	**+3.76** ±1.19	**0.600** ±0.115	**+7.62** ±0.54	**0.804** ±0.036
Who&When-AAO	+10.30 ±4.85	0.494 ±0.417	+6.74 ±0.00	0.222 ±0.111	**+10.63** ±0.00	0.333 ±0.000	**+6.63** ±0.00	**1.000** ±0.000	**+5.74** ±0.00	**1.000** ±0.000	+1.65 ±0.00	0.176 ±0.070	+6.95 ±0.81	0.538 ±0.074
A2P	+10.30 ±4.85	0.111 ±0.222	+6.74 ±0.00	0.000 ±0.000	**+10.63** ±0.00	0.333 ±0.000	**+6.63** ±0.00	**1.000** ±0.000	**+5.74** ±0.00	**1.000** ±0.000	+1.65 ±0.00	0.219 ±0.167	+6.95 ±0.81	0.444 ±0.027
Adapted locator	+9.70 ±0.00	**0.816** ±0.000	+4.88 ±2.56	0.516 ±0.289	-0.30 ±0.00	-0.333 ±0.000	+5.56 ±1.07	0.333 ±0.667	+0.40 ±0.00	-0.111 ±0.222	-0.91 ±0.51	-0.102 ±0.058	+3.22 ±0.44	0.187 ±0.040
AgentRx	**+12.73** ±2.42	0.605 ±0.307	+0.16 ±0.39	-0.411 ±0.297	**+10.63** ±0.00	0.333 ±0.000	**+6.63** ±0.00	**1.000** ±0.000	**+5.74** ±0.00	**1.000** ±0.000	+1.09 ±0.56	0.141 ±0.217	+6.16 ±0.41	0.445 ±0.066
CHIEF	+11.52 ±3.64	0.717 ±0.199	+5.04 ±2.06	0.365 ±0.279	+6.99 ±3.65	0.111 ±0.222	**+6.63** ±0.00	**1.000** ±0.000	**+5.74** ±0.00	**1.000** ±0.000	+2.39 ±1.32	0.502 ±0.263	+6.38 ±0.62	0.616 ±0.026
FAMAS	+7.88 ±3.64	0.444 ±0.294	+6.98 ±0.23	0.669 ±0.122	**+10.63** ±0.00	**0.878** ±0.061	**+6.63** ±0.00	**1.000** ±0.000	**+5.74** ±0.00	0.556 ±0.222	**+3.76** ±1.19	0.579 ±0.169	+6.94 ±0.45	0.688 ±0.024
Uniform-agent reference†	+6.67 ±0.69	—	+3.72 ±0.12	—	+3.88 ±0.27	—	+5.03 ±0.89	—	+2.17 ±0.21	—	+1.42 ±1.76	—	+3.81 ±0.35	—
B. Qwen3-235B-A22B														
IF-MAP (ours)	+19.19 ±5.05	0.778 ±0.222	**+11.63** ±0.00	**0.889** ±0.111	**+21.78** ±0.00	0.778 ±0.222	**+11.13** ±0.00	**1.000** ±0.000	**+7.88** ±0.00	0.556 ±0.222	**+9.55** ±3.95	**0.400** ±0.115	**+13.53** ±0.77	**0.733** ±0.040
Who&When-AAO	+16.16 ±8.08	0.494 ±0.417	+7.21 ±0.00	0.222 ±0.111	**+21.78** ±0.00	0.333 ±0.000	**+11.13** ±0.00	**1.000** ±0.000	**+7.88** ±0.00	0.333 ±0.000	+4.41 ±0.00	0.176 ±0.141	+11.43 ±1.35	0.426 ±0.087
A2P	+16.16 ±8.08	0.111 ±0.222	+7.21 ±0.00	0.000 ±0.000	**+21.78** ±0.00	0.333 ±0.000	**+11.13** ±0.00	**1.000** ±0.000	**+7.88** ±0.00	0.333 ±0.000	+4.41 ±0.00	0.290 ±0.100	+11.43 ±1.35	0.345 ±0.026
Adapted locator	+16.67 ±0.00	**0.816** ±0.000	+7.60 ±4.03	0.516 ±0.289	-0.98 ±0.00	-0.333 ±0.000	+9.87 ±1.26	0.333 ±0.667	+0.53 ±0.00	-0.778 ±0.222	+3.72 ±0.14	-0.305 ±0.058	+6.23 ±0.61	0.042 ±0.039
AgentRx	**+20.20** ±4.04	0.605 ±0.307	+0.85 ±1.32	-0.411 ±0.297	**+21.78** ±0.00	0.333 ±0.000	**+11.13** ±0.00	**1.000** ±0.000	**+7.88** ±0.00	0.333 ±0.000	+3.81 ±0.60	0.072 ±0.374	+10.94 ±0.78	0.322 ±0.091
CHIEF	+19.19 ±5.05	0.717 ±0.199	+7.44 ±2.35	0.365 ±0.279	+14.19 ±7.59	0.111 ±0.222	**+11.13** ±0.00	**1.000** ±0.000	**+7.88** ±0.00	0.333 ±0.000	+9.00 ±2.62	0.305 ±0.058	+11.47 ±1.09	0.472 ±0.037
FAMAS	+14.14 ±5.05	0.444 ±0.294	+8.68 ±1.47	0.669 ±0.122	**+21.78** ±0.00	**0.878** ±0.061	**+11.13** ±0.00	**1.000** ±0.000	**+7.88** ±0.00	**0.778** ±0.222	**+9.55** ±3.95	0.145 ±0.247	+12.19 ±0.43	0.652 ±0.059
Uniform-agent reference†	+11.11 ±0.00	—	+5.47 ±0.07	—	+7.81 ±0.22	—	+9.24 ±0.50	—	+3.55 ±0.21	—	+6.45 ±0.66	—	+7.27 ±0.15	—
C. Qwen3.7-Plus														
IF-MAP (ours)	+21.21 ±6.06	0.778 ±0.222	**+21.17** ±0.00	**0.889** ±0.111	**+19.78** ±0.00	0.778 ±0.222	**+12.82** ±0.00	**1.000** ±0.000	**+10.46** ±0.00	0.556 ±0.222	**+13.09** ±0.66	**0.733** ±0.176	**+16.42** ±0.96	**0.789** ±0.068
Who&When-AAO	+18.18 ±9.09	0.494 ±0.417	+9.07 ±0.00	0.222 ±0.111	**+19.78** ±0.00	0.333 ±0.000	**+12.82** ±0.00	**1.000** ±0.000	**+10.46** ±0.00	0.333 ±0.000	+5.23 ±0.00	-0.246 ±0.070	+12.59 ±1.52	0.356 ±0.074
A2P	+18.18 ±9.09	0.111 ±0.222	+9.07 ±0.00	0.000 ±0.000	**+19.78** ±0.00	0.333 ±0.000	**+12.82** ±0.00	**1.000** ±0.000	**+10.46** ±0.00	0.333 ±0.000	+5.23 ±0.00	-0.211 ±0.161	+12.59 ±1.52	0.261 ±0.028
Adapted locator	+18.18 ±0.00	**0.816** ±0.000	+14.03 ±7.13	0.516 ±0.289	-0.12 ±0.00	-0.333 ±0.000	+11.03 ±1.79	0.333 ±0.667	+0.14 ±0.00	-0.778 ±0.222	+3.86 ±0.28	-0.509 ±0.058	+7.85 ±1.10	0.008 ±0.039
AgentRx	**+22.73** ±4.55	0.605 ±0.307	+2.70 ±2.93	-0.411 ±0.297	**+19.78** ±0.00	0.333 ±0.000	**+12.82** ±0.00	**1.000** ±0.000	**+10.46** ±0.00	0.333 ±0.000	+6.05 ±0.81	-0.159 ±0.138	+12.42 ±1.11	0.284 ±0.060
CHIEF	+21.21 ±6.06	0.717 ±0.199	+12.93 ±4.12	0.365 ±0.279	+13.15 ±6.63	0.111 ±0.222	**+12.82** ±0.00	**1.000** ±0.000	**+10.46** ±0.00	0.333 ±0.000	+9.08 ±2.49	0.235 ±0.185	+13.28 ±0.72	0.460 ±0.015
FAMAS	+15.15 ±6.06	0.444 ±0.294	+13.10 ±4.03	0.669 ±0.122	**+19.78** ±0.00	**0.878** ±0.061	**+12.82** ±0.00	**1.000** ±0.000	**+10.46** ±0.00	**0.778** ±0.222	**+13.09** ±0.66	0.364 ±0.165	+14.07 ±1.57	0.689 ±0.113
Uniform-agent reference†	+12.12 ±0.00	—	+9.64 ±0.20	—	+6.99 ±0.23	—	+10.13 ±0.19	—	+4.31 ±0.13	—	+8.52 ±0.50	—	+8.62 ±0.11	—

Mean above, ±SE below. Method-row SEs summarize three prioritization runs; uniform-agent reference rows use $\mathrm{SE}_{\mathrm{test}}$ across five intervention runs and are reference averages rather than selectors. Other notation and formatting conventions follow Appendix Table 11.

Table 13: Model upgrades with standard errors: complete mean±SE version of Table 2. Each panel uses one target model and its own uniform-agent reference.

Algorithm	HumanEval		MBPP		NQ		DROP		HotpotQA		MATH		Macro (6)	
	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\Delta\uparrow$	$\tau_b\uparrow$	$\overline{\Delta}\uparrow$	$\overline{\tau}_b\uparrow$
D. DeepSeek-V3.2														
IF-MAP (ours)	**+15.76** ±3.03	0.778 ±0.222	**+12.33** ±0.00	**0.889** ±0.111	**+23.08** ±0.00	0.778 ±0.222	**+8.97** ±0.00	**1.000** ±0.000	**+5.18** ±0.00	0.556 ±0.222	**+9.59** ±2.70	**0.246** ±0.186	**+12.48** ±0.48	**0.708** ±0.069
Who&When-AAO	+12.53 ±6.26	0.494 ±0.417	+8.60 ±0.00	0.222 ±0.111	**+23.08** ±0.00	0.333 ±0.000	**+8.97** ±0.00	**1.000** ±0.000	**+5.18** ±0.00	0.333 ±0.000	+2.20 ±0.00	-0.704 ±0.098	+10.09 ±1.04	0.280 ±0.081
A2P	+12.53 ±6.26	0.111 ±0.222	+8.60 ±0.00	0.000 ±0.000	**+23.08** ±0.00	0.333 ±0.000	**+8.97** ±0.00	**1.000** ±0.000	**+5.18** ±0.00	0.333 ±0.000	+2.20 ±0.00	-0.639 ±0.127	+10.09 ±1.04	0.190 ±0.028
Adapted locator	+14.24 ±0.00	**0.816** ±0.000	+8.29 ±4.03	0.516 ±0.289	+1.70 ±0.00	-0.333 ±0.000	+7.63 ±1.34	0.333 ±0.667	+1.47 ±0.00	-0.778 ±0.222	+3.86 ±0.33	-0.074 ±0.096	+6.20 ±0.55	0.080 ±0.028
AgentRx	+15.66 ±3.13	0.605 ±0.307	+0.93 ±0.70	-0.411 ±0.297	**+23.08** ±0.00	0.333 ±0.000	**+8.97** ±0.00	**1.000** ±0.000	**+5.18** ±0.00	0.333 ±0.000	+2.87 ±0.66	-0.201 ±0.224	+9.45 ±0.64	0.277 ±0.086
CHIEF	**+15.76** ±3.03	0.717 ±0.199	+7.75 ±2.92	0.365 ±0.279	+15.96 ±7.13	0.111 ±0.222	**+8.97** ±0.00	**1.000** ±0.000	**+5.18** ±0.00	0.333 ±0.000	+6.01 ±3.16	-0.289 ±0.198	+9.94 ±0.92	0.373 ±0.014
FAMAS	+12.73 ±3.03	0.444 ±0.294	+9.84 ±1.24	0.669 ±0.122	**+23.08** ±0.00	**0.878** ±0.061	**+8.97** ±0.00	**1.000** ±0.000	**+5.18** ±0.00	**0.778** ±0.222	**+9.59** ±2.70	0.192 ±0.103	+11.57 ±0.26	0.660 ±0.090
Uniform-agent reference†	+9.49 ±0.49	—	+5.87 ±0.29	—	+8.89 ±0.18	—	+6.96 ±0.15	—	+3.01 ±0.19	—	+5.28 ±0.38	—	+6.58 ±0.18	—
E. GPT-5 nano														
IF-MAP (ours)	+19.39 ±5.45	0.778 ±0.222	**+17.21** ±0.00	**0.556** ±0.111	**+14.10** ±0.00	0.556 ±0.222	**+6.50** ±0.00	**1.000** ±0.000	**+3.03** ±0.00	0.778 ±0.222	**+11.24** ±1.87	**0.733** ±0.176	**+11.91** ±0.80	**0.733** ±0.106
Who&When-AAO	+16.77 ±8.08	0.494 ±0.417	+2.09 ±0.00	-0.111 ±0.111	**+14.10** ±0.00	**1.000** ±0.000	**+6.50** ±0.00	**1.000** ±0.000	**+3.03** ±0.00	**1.000** ±0.000	+1.38 ±0.00	-0.246 ±0.070	+7.31 ±1.35	0.523 ±0.074
A2P	+16.77 ±8.08	0.111 ±0.222	+2.09 ±0.00	-0.333 ±0.000	**+14.10** ±0.00	**1.000** ±0.000	**+6.50** ±0.00	**1.000** ±0.000	**+3.03** ±0.00	**1.000** ±0.000	+1.38 ±0.00	-0.211 ±0.161	+7.31 ±1.35	0.428 ±0.028
Adapted locator	+16.67 ±0.00	**0.816** ±0.000	+11.32 ±5.89	0.415 ±0.381	+2.14 ±0.00	0.333 ±0.000	+5.22 ±1.28	0.333 ±0.667	+1.38 ±0.00	-0.111 ±0.222	+0.69 ±0.14	-0.509 ±0.058	+6.24 ±0.91	0.213 ±0.077
AgentRx	**+20.81** ±4.04	0.605 ±0.307	+1.01 ±1.47	-0.289 ±0.418	**+14.10** ±0.00	**1.000** ±0.000	**+6.50** ±0.00	**1.000** ±0.000	**+3.03** ±0.00	**1.000** ±0.000	+2.26 ±0.88	-0.159 ±0.138	+7.95 ±0.87	0.526 ±0.078
CHIEF	+19.39 ±5.45	0.717 ±0.199	+7.75 ±4.76	0.233 ±0.217	+10.11 ±3.99	0.778 ±0.222	**+6.50** ±0.00	**1.000** ±0.000	**+3.03** ±0.00	**1.000** ±0.000	+7.11 ±3.39	0.235 ±0.185	+8.98 ±0.32	0.660 ±0.026
FAMAS	+13.94 ±5.45	0.444 ±0.294	+7.13 ±5.04	0.304 ±0.122	**+14.10** ±0.00	0.655 ±0.161	**+6.50** ±0.00	**1.000** ±0.000	**+3.03** ±0.00	0.556 ±0.222	**+11.24** ±1.87	0.364 ±0.165	+9.32 ±1.44	0.554 ±0.059
Uniform-agent reference†	+11.31 ±0.45	—	+5.70 ±0.55	—	+5.86 ±0.28	—	+4.58 ±0.36	—	+1.51 ±0.33	—	+5.87 ±0.66	—	+5.81 ±0.19	—

Mean above, ±SE below. Method-row SEs summarize three prioritization runs; uniform-agent reference rows use $\mathrm{SE}_{\mathrm{test}}$ across five intervention runs and are reference averages rather than selectors. Other notation and formatting conventions follow Appendix Table 11.

F EXTERNAL SELECTORS AND ADAPTATIONS

The six external selectors use the same frozen DEV-A trajectories, node inventory, and downstream gain matrices. Diagnostic judgments use DeepSeek-V3.2, while task execution uses the base workflow model. Each selector is run independently three times. Valid task-level attributions are aggregated over failed development trajectories to form a workflow-level node ranking; unparseable outputs and explicit abstentions contribute no vote. The resulting ranking is used unchanged for both prompt optimization and model upgrades.

Who&When-AAO. We use the paper's All-at-Once prompt, which reads the complete failed trajectory in one request and returns an agent, an error step, and a reason. A valid agent–step attribution contributes one vote to that agent. Votes are summed over failed trajectories to obtain the node ranking.

A2P. One diagnostic request performs the paper's three reasoning stages: abduction of the failure cause, a minimal corrective action, and prediction of the resulting continuation. The final agent–step attribution contributes one vote, and the workflow-level ranking is obtained by aggregating these votes.

Adapted locator. We first apply the task critic adapted from the paper's critic template to the final workflow output and its recorded evaluation evidence. The locator then uses this feedback together with agent roles and the complete trajectory to return a set of weak agents. Each trajectory contributes one vote divided uniformly over that set, and the accumulated shares determine the ranking. We evaluate this localization component independently of the paper's subsequent workflow-optimization stage.

AgentRx. The recorded trajectory is converted to AgentRx's canonical intermediate representation. Its static and dynamic invariant generators produce checks, the generated Python and natural-language checks are evaluated, and the native judge assigns a failure category and responsible step from the resulting violation log. Generated Python checks run in isolated containers. A mapped responsible step contributes one vote to its owning agent; intermediate violation counts are not used as node scores.

CHIEF. The implementation follows CHIEF's staged diagnosis: retrieval-augmented task decomposition, OTAR construction, subtask/agent/step dependency graphs, virtual oracle construction, hierarchical backtracking, and the final Local–Planning-Control–Data-Flow–Deviation-Aware attribution. The final supported agent receives one task-level vote. The RAG store and the complete intermediate diagnosis are retained for every trajectory.

FAMAS. For each failed anchor trajectory, the reported configuration combines the anchor with 20 additional replays and retains at most ten native leaf events per trajectory. The implementation abstracts events into agent/action/state triples, clusters semantically equivalent action–state pairs within each agent, and applies the FAMAS frequency, coverage, and success/failure spectrum score with decay $\lambda = 0.9$. The owner of the unique highest-scoring anchor triple receives one vote; a top-score tie produces an abstention.

Why coalition/removal baselines are not included. HiveMind assigns node credit from coalition values on a workflow DAG using a Shapley calculation (Xia et al., 2026). Agents that Matter likewise formalizes agent credit through explicit removal protocols, alongside a separate model-replacement analysis (Lu et al., 2026). In the five chain-structured workflows used here, the current executable interface admits only the complete source-to-sink coalition: removing an intermediate node leaves a downstream node without its required input. Under this coalition set, every node receives the same value $v(N)/|N|$, so the selected node is determined only by tie breaking. MATH admits additional graph-valid coalitions, but its composite operators do not define what a downstream node should receive when one of its required parents is absent. Filling those inputs with null values or replaying outputs from the complete workflow would define a different intervention rather than execute the coalition being evaluated. We therefore do not include the coalition/removal vari-

ants of these methods in the matched selector comparison, because the six workflows do not provide a common semantics-preserving removal operator.

Tables 1 and 2 report the comparison under the shared aggregation and intervention protocol above.

G SCORING AND PERTURBATION-BUDGET ABLATIONS

Using the signed probe effect $\Delta_{im} = \overline{L}_0 - \overline{L}_i^{(m)}$, the two magnitude-only scoring baselines are

$$S_i^{\text{abs}} = \max_m |\Delta_{im}|, \qquad S_i^{\text{mean}} = \frac{1}{M_i} \sum_m |\Delta_{im}|. \tag{10}$$

These scores ignore whether a perturbation helps or harms and do not use the embedding direction or displacement size.

Table 3 reports completed variants with three, five, and up to ten formal probes per node. All variants use the same datasets, execution models, downstream interventions, and evaluation rules. The nominal probe budget does not include auxiliary evaluations used during adaptive construction, which are retained in the reduced-probe variants.

Tables 14 and 15 report the corresponding results for each workflow. Gains are in percentage points; each $\pm$ term is the standard error across the three prioritization runs. The model-upgrade columns first average each node's gain over the five target models. The macro rows use equal dataset weights and reproduce Table 3.

Table 14: Per-workflow scoring-rule ablation. Each cell reports the mean $\pm\text{SE}_{\text{sel}}$ over three prioritization runs. Model-upgrade gains first average each node over all five target models.

		Prompt optimization		Model upgrades	
Workflow	Score	Gain (pp) ↑	τ_b ↑	Gain (pp) ↑	τ_b ↑
HumanEval	G-norm	$+4.444 \pm 0.202$	0.556 ± 0.222	$+17.414 \pm 4.647$	0.778 ± 0.222
	I-abs	$+4.849 \pm 0.000$	1.000 ± 0.000	$+8.121 \pm 0.000$	0.333 ± 0.000
	MeanAbs	$+4.849 \pm 0.000$	1.000 ± 0.000	$+8.121 \pm 0.000$	0.333 ± 0.000
MBPP	G-norm	$+7.907 \pm 0.000$	0.889 ± 0.111	$+13.955 \pm 0.000$	0.889 ± 0.111
	I-abs	$+4.651 \pm 1.628$	0.778 ± 0.111	$+9.148 \pm 2.404$	0.778 ± 0.111
	MeanAbs	$+6.279 \pm 1.628$	0.556 ± 0.111	$+11.551 \pm 2.404$	0.556 ± 0.111
NQ	G-norm	$+2.141 \pm 0.000$	0.778 ± 0.222	$+17.876 \pm 0.000$	0.778 ± 0.222
	I-abs	$+2.141 \pm 0.000$	0.778 ± 0.222	$+17.876 \pm 0.000$	0.778 ± 0.222
	MeanAbs	$+2.095 \pm 0.046$	0.778 ± 0.222	$+12.483 \pm 5.393$	0.778 ± 0.222
DROP	G-norm	$+4.994 \pm 0.000$	1.000 ± 0.000	$+9.211 \pm 0.000$	1.000 ± 0.000
	I-abs	$+4.994 \pm 0.000$	1.000 ± 0.000	$+9.211 \pm 0.000$	1.000 ± 0.000
	MeanAbs	$+4.994 \pm 0.000$	1.000 ± 0.000	$+9.211 \pm 0.000$	1.000 ± 0.000
HotpotQA	G-norm	$+1.534 \pm 0.000$	0.556 ± 0.222	$+6.459 \pm 0.000$	0.556 ± 0.222
	I-abs	$+1.534 \pm 0.000$	0.556 ± 0.222	$+6.459 \pm 0.000$	0.556 ± 0.222
	MeanAbs	$+1.534 \pm 0.000$	0.778 ± 0.222	$+6.459 \pm 0.000$	0.778 ± 0.222
MATH	G-norm	$+1.852 \pm 0.150$	0.667 ± 0.067	$+9.447 \pm 2.075$	0.667 ± 0.067
	I-abs	$+1.485 \pm 0.200$	0.399 ± 0.204	$+5.563 \pm 0.052$	0.399 ± 0.204
	MeanAbs	$+1.702 \pm 0.181$	0.533 ± 0.176	$+6.717 \pm 2.524$	0.533 ± 0.176
Macro	G-norm	$+3.812$	0.741	$+12.394$	0.778
	I-abs	$+3.276$	0.752	$+9.396$	0.641
	MeanAbs	$+3.576$	0.774	$+9.090$	0.663

Table 15: Per-workflow perturbation-budget comparison using G-norm. Each cell reports the mean $\pm\mathrm{SE}_{\mathrm{sel}}$ over three prioritization runs. The full setting uses at most ten formal probes per node.

		Prompt optimization		Model upgrades	
Workflow	Probes	Gain (pp) ↑	τ_b ↑	Gain (pp) ↑	τ_b ↑
HumanEval	3	$+4.849 \pm 0.000$	1.000 ± 0.000	$+8.121 \pm 0.000$	0.333 ± 0.000
	5	$+4.849 \pm 0.000$	1.000 ± 0.000	$+8.121 \pm 0.000$	0.333 ± 0.000
	Full	$+4.444 \pm 0.202$	0.556 ± 0.222	$+17.414 \pm 4.647$	0.778 ± 0.222
MBPP	3	$+5.426 \pm 2.481$	0.667 ± 0.192	$+10.587 \pm 3.368$	0.667 ± 0.192
	5	$+5.426 \pm 2.481$	0.667 ± 0.192	$+10.587 \pm 3.368$	0.667 ± 0.192
	Full	$+7.907 \pm 0.000$	0.889 ± 0.111	$+13.955 \pm 0.000$	0.889 ± 0.111
NQ	3	$+1.344 \pm 0.728$	0.111 ± 0.588	$+6.686 \pm 5.606$	0.111 ± 0.588
	5	$+2.095 \pm 0.046$	0.333 ± 0.385	$+12.483 \pm 5.393$	0.333 ± 0.385
	Full	$+2.141 \pm 0.000$	0.778 ± 0.222	$+17.876 \pm 0.000$	0.778 ± 0.222
DROP	3	$+0.973 \pm 2.011$	-0.333 ± 0.667	$+6.514 \pm 1.349$	-0.333 ± 0.667
	5	$+2.984 \pm 2.011$	0.333 ± 0.667	$+7.862 \pm 1.349$	0.333 ± 0.667
	Full	$+4.994 \pm 0.000$	1.000 ± 0.000	$+9.211 \pm 0.000$	1.000 ± 0.000
HotpotQA	3	$+0.511 \pm 0.511$	-0.111 ± 0.222	$+3.143 \pm 1.658$	-0.111 ± 0.222
	5	$+0.511 \pm 0.511$	-0.111 ± 0.222	$+3.143 \pm 1.658$	-0.111 ± 0.222
	Full	$+1.534 \pm 0.000$	0.556 ± 0.222	$+6.459 \pm 0.000$	0.556 ± 0.222
MATH	3	$+1.610 \pm 0.059$	0.400 ± 0.115	$+5.415 \pm 0.119$	0.400 ± 0.115
	5	$+1.552 \pm 0.101$	0.267 ± 0.176	$+4.641 \pm 0.840$	0.267 ± 0.176
	Full	$+1.852 \pm 0.150$	0.667 ± 0.067	$+9.447 \pm 2.075$	0.667 ± 0.067
Macro	3	+2.452	0.289	+6.744	0.178
	5	+2.903	0.415	+7.806	0.304
	Full	+3.812	0.741	+12.394	0.778

H COST ACCOUNTING

All tables in this section report measured execution costs over the six datasets, aggregated from the recorded execution ledgers. Token counts include input and output tokens and are reported in millions (M); fractional request counts can result from averaging measured runs and weighted selections. Each cell shows the absolute cost above and, in parentheses, the percentage relative to IF-MAP in the corresponding column and setting; IF-MAP is therefore 100% by definition.

Table 16: Cost of one agent-prioritization run over all six datasets.

Method	Qwen3-8B requests	Qwen tokens (M)	DeepSeek-V3.2 requests	DeepSeek tokens (M)
IF-MAP	63,088 (100.00%)	47.141 (100.00%)	662 (100.00%)	3.049 (100.00%)
Who&When-AAO	35,288 (55.93%)	26.672 (56.58%)	1,943 (293.50%)	18.341 (601.54%)
A2P	35,288 (55.93%)	26.672 (56.58%)	1,938 (292.75%)	18.645 (611.51%)
Adapted locator	35,288 (55.93%)	26.672 (56.58%)	3,197 (482.93%)	21.138 (693.28%)
AgentRx	35,288 (55.93%)	26.672 (56.58%)	8,506 (1,284.89%)	202.576 (6,644.01%)
CHIEF	35,288 (55.93%)	26.672 (56.58%)	10,225 (1,544.56%)	142.013 (4,657.69%)
FAMAS	72,894 (115.54%)	52.709 (111.81%)	18,228 (2,753.47%)	171.566 (5,626.96%)

Table 17: Prompt-optimization cost over all six datasets. IF-MAP first selects an agent and optimizes only that agent; Full-Agent PromptOpt applies the same custom prompt optimizer to every agent.

Method	Qwen3-8B requests	Qwen tokens (M)	DeepSeek-V3.2 (TextGrad) requests	DeepSeek (TextGrad) tokens (M)
IF-MAP	203,546.74 (100.00%)	352.994 (100.00%)	950.00 (100.00%)	4.249 (100.00%)
Full-Agent PromptOpt	663,341.97 (325.89%)	1,366.978 (387.25%)	1,662.01 (174.95%)	6.559 (154.37%)

Table 18: Model-upgrade cost for one complete run over all six datasets. Each IF-MAP row includes agent selection, one original test-set baseline, and one post-upgrade test-set evaluation. FullSwap upgrades and evaluates every agent and therefore has no DeepSeek-V3.2 selection-stage cost.

Target	Method	Qwen3-8B requests	Qwen3-8B tokens (M)	DeepSeek-V3.2 (selection) requests	DeepSeek-V3.2 (selection) tokens (M)	Target model (upgrade) requests	Target model (upgrade) tokens (M)
Qwen3-32B	**IF-MAP**	91,748 (100.00%)	151.102 (100.00%)	794 (100.00%)	4.028 (100.00%)	1,557 (100.00%)	1.541 (100.00%)
Qwen3-32B	FullSwap	12,553 (13.68%)	15.710 (10.40%)	0 (0.00%)	0 (0.00%)	4,066 (261.14%)	4.389 (284.82%)
Qwen3-235B-A22B	**IF-MAP**	91,748 (100.00%)	151.102 (100.00%)	794 (100.00%)	4.028 (100.00%)	1,534 (100.00%)	1.601 (100.00%)
Qwen3-235B-A22B	FullSwap	12,518 (13.64%)	15.730 (10.41%)	0 (0.00%)	0 (0.00%)	4,085 (266.30%)	4.481 (279.89%)
Qwen3.7-Plus	**IF-MAP**	91,748 (100.00%)	151.102 (100.00%)	794 (100.00%)	4.028 (100.00%)	1,524 (100.00%)	1.781 (100.00%)
Qwen3.7-Plus	FullSwap	12,774 (13.92%)	16.030 (10.61%)	0 (0.00%)	0 (0.00%)	4,134 (271.26%)	4.923 (276.42%)
DeepSeek-V3.2	**IF-MAP**	91,748 (100.00%)	151.102 (100.00%)	794 (100.00%)	4.028 (100.00%)	1,522 (100.00%)	1.939 (100.00%)
DeepSeek-V3.2	FullSwap	12,871 (14.03%)	16.210 (10.73%)	0 (0.00%)	0 (0.00%)	4,601.2 (302.31%)	5.331 (274.93%)
GPT-5 nano	**IF-MAP**	91,748 (100.00%)	151.102 (100.00%)	794 (100.00%)	4.028 (100.00%)	1,526 (100.00%)	3.000 (100.00%)
GPT-5 nano	FullSwap	12,646 (13.78%)	15.330 (10.15%)	0 (0.00%)	0 (0.00%)	4,825.8 (316.24%)	8.330 (277.67%)

I ARTIFACT PROVENANCE AND REPRODUCIBILITY

All reported performance values are produced by the protocol specified in this appendix. The experimental records contain the three prioritization runs, five intervention runs, and the gain matrix used for each table.

The following materials and evaluations support a complete artifact release and broader replication.

1. Example IDs, generated prompt texts, perturbation omissions, response caches by repeat, and the gain matrix used for each table.
2. Per-call query, token, latency, and monetary accounting for the prioritization methods and downstream interventions.
3. Evaluations on independently constructed workflows with different interaction graphs and additional intervention families.
4. An anonymous artifact containing the prompt snapshots, dependencies, evaluation commands, and applicable licenses.

Future topology studies could include code generation with execution feedback, question decomposition with evidence checking, and independent mathematical solution verification. Workflows with distinct interaction graphs would test whether a node-response score remains useful when information routing and correction opportunities change.