LOGIC SIGNATURES FOR TEST-TIME LLM REASONING ROUTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reasoning in Large Language Models (LLMs) is commonly summarized by benchmark scores and improved with a uniform inference recipe. However, this scalar view neglects the interaction between a query's logical demands and a particular solver's failure tendencies. We hypothesize a *failure-to-control principle*: failures indexed by reasoning operations can guide query–solver-specific assistance, before the similar vulnerabilities recur. To test this hypothesis, we propose LOGICROUTE, a training-free framework for pre-response intervention routing. At its core is LOGICSIG, a sparse two-sided representation that separates *what the query requires* from *where the solver tends to fail*. Using offline zero-shot traces, LOGICROUTE constructs solver-specific logic profiles, organizes candidate interventions into general reasoning scaffolds and targeted repairs, and routes each query before the target model responds. Across five open-sourced LLM models spanning 1.7B–27B parameters on MATH500, AIME, GPQA, and MMLU-Pro, LOGICROUTE improves macro accuracy from 51.00% to 62.51%, outperforming semantic demonstration retrieval by 8.19 percentage points. Across the evaluated Qwen, Llama, and Mistral models, macro accuracy increases from 39.55% to 51.79%. Profile substitutions show that useful routes are solver-specific; route perturbations support the utility of logic-feature ordering; and demonstration-bank ablations show that assistance is compositional but not monotonic in context volume. Together, these findings suggest that failures in the past provide a reusable pre-response control signal for targeted assistance without parameter updates.

1 INTRODUCTION

An aggregate benchmark score tells us how often a language model succeeds, but not which assistance would help on the next query (Mousavi et al., 2026). Reasoning problems require different operations, and models with similar overall accuracy can have different weaknesses. Moreover, an in-context demonstration can redirect a model's reasoning behavior, helping one query–solver pair while distracting another. A useful demonstration must therefore match both the query's reasoning requirements and the target solver's vulnerabilities. This motivates our central question: *can a frozen solver's past failures guide the selection of assistance before it answers a new query?*

Several lines of work address parts of this issue. Test-time reasoning methods elicit, sample, or search intermediate reasoning trajectories, in the forms such as Chain of Thought (CoT) and Chain of Draft (CoD) (Wei et al., 2022; Xu et al., 2025). In-context learning adapts the evidence through semantic retrieval, automatically constructed demonstrations, and reasoning-pattern diversity (Luo et al., 2024; Zhang et al., 2022; 2024). Logical reasoning benchmarks and neuro-symbolic methods expose finer-grained reasoning operations (Farjami et al., 2026; Parmar et al., 2024; Luo et al., 2023), while model orchestration fuses generated candidate answers (Jiang et al., 2023) or routes queries among models before answer generation (Wang et al., 2026). Recent error-aware retrieval selects examples previously misclassified by the deployed model (Omidvartehrani et al., 2026), and failed-trace methods use observed failures to choose recovery procedures (Islah et al., 2026). These methods fall short in explicitly connecting a query's logical demands with a solver's recurring failure patterns, leading to gap in addressing the true reasoning weaknesses or distracted guidance.

The key to overcome this challenge is to select an intervention that addresses a likely failure, beyond predicting failure risk alone (Yang et al., 2026). This requires relating the query's reasoning

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.

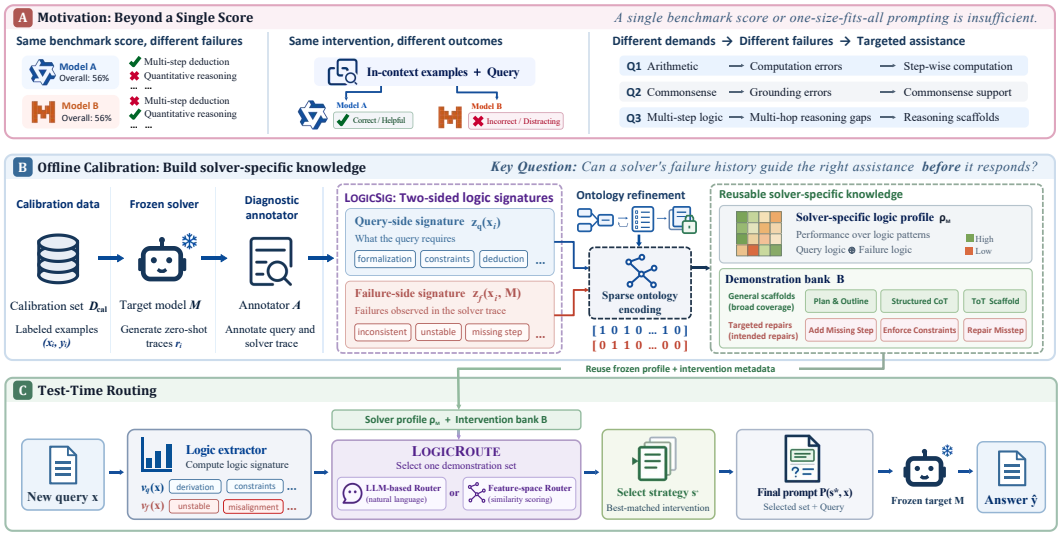


Figure 1: **Approach overview.** Offline calibration produces signatures and a solver-specific profile used to construct the demonstration bank. At test time, the current query and frozen profile guide vulnerability estimation and demonstration selection before the target model responds, without access to its test-time response or correctness.

demands to the solver's vulnerabilities and the repair offered by each candidate intervention. More broadly, computational interfaces for LLMs, ranging from token representations to symbolic reasoning protocols, can be adapted rather than treated as fixed (Pei et al., 2024; 2026). This perspective motivates us to represent the failure history into structured patterns to guide pre-response assistance.

Our working hypothesis, the *failure-to-control principle*, is that failures indexed by reasoning operations provide reusable information for selecting assistance. We operationalize it with LogicSig, a two-sided behavioral representation, and LogicRoute, a framework for pre-response demonstration routing. During offline calibration, a diagnostic annotator examines a query and a frozen solver's zero-shot trace. It records the operations required by the query in $z_q(x)$ and the observed failure patterns in $z_f(x, M)$. These annotations inform a solver-specific profile ρ_M and a bank containing general reasoning scaffolds and targeted repair demonstrations. At test time, LogicRoute combines the new query with the frozen calibration information to select one demonstration set before the target solver answers (Figure 1). The framework requires no parameter updates.

We evaluate on held-out subsets of MATH500, AIME, GPQA-Main, and MMLU-Pro. Across five Qwen-family models spanning 1.7B–27B parameters, LogicRoute-LLM improves macro accuracy from 51.00% for Zero-shot to 62.51%, compared with 48.08% for CoT and 54.32% for semantic demonstration retrieval. Across Qwen, Llama, and Mistral architectures, macro accuracy increases from 39.55% to 51.79% and exceeds semantic retrieval by 9.89%.

Our experiments provide three observations about selective assistance. First, *scalar diagnostics are intervention-incomplete*. Aggregate accuracy, and even a scalar failure probability, can hardly identify the best repair because solvers with identical failure risk may respond differently to the same intervention. Second, *solver conditioning can have strict value*. Whenever two solvers rank candidate interventions differently under the same logical demand, no query-only routing policy can be optimal for both. Third, intervention-bank design exhibits a *coverage–routing trade-off*. Enlarging the candidate bank weakly expands the oracle choice set, but can reduce the real performance when the resulting increase in routing regret exceeds the additional oracle gain. These results explain why effective pre-response routing requires a solver-specific profile, joint query–solver matching, and a compact, structured demonstration bank rather than an indiscriminately large demonstration pool.

Our contributions are:

1. **A failure-to-control formulation of test-time assistance.** We formulate reasoning assistance as pre-response intervention routing over the query–solver pair, rather than as a universal inference recipe or a query-only retrieval problem.

2. **A two-sided representation and routing framework.** We introduce LOGICSIG to connect query-side logical demands with solver-side vulnerabilities, and LOGICROUTE to transform offline behavioral evidence into targeted intervention routing.
3. **An empirical analysis of selective assistance.** We examine how observed performance varies with solver-specific descriptions and demonstrations, logic-space selection, and the composition and size of the candidate bank.
4. **Broad empirical and mechanism-level evidence.** We report aggregate gains across four reasoning benchmarks, five Qwen model scales, and three model families, and examine the roles of profiles, route selection, bank composition, and bank size through ablations.

2 PROBLEM SETUP

This work considers a question: *given a fixed language model and a reasoning query, what form of test-time assistance should be provided to this particular model for this particular query?* Unlike a fixed prompting strategy, this formulation makes assistance specific to both the model and the query.

2.1 SETTING AND INFORMATION BOUNDARY

Let M be the frozen target model, $x \in \mathcal{X}$ a query, and $\mathcal{S}$ a finite set of candidate demonstration strategies. Each strategy contains in-context examples and associated reasoning instructions. Given a prompt constructor P, the target model produces

$$\hat{y} \sim M(\cdot \mid P(s, x)) . \tag{1}$$

A routing policy $\pi(x, M)$ selects one strategy $s \in \mathcal{S}$. It does not observe the target model's response, reasoning trace, gold answer, or correctness on the test query.

The usefulness of a demonstration depends on more than the query alone. Models with similar overall benchmark accuracy may fail on different reasoning operations, and the same model may need different assistance for different queries. An effective routing rule must therefore characterize both (i) the reasoning structure required by the current query and (ii) the reasoning patterns on which the target model is likely to be vulnerable. They must be estimated before the target model responds to the test query, using only the routing inputs specified above.

For each solver, the policy is configured using a frozen calibration-derived profile and demonstration bank; only the new input at test time serves as the query.

2.2 SOLVER-CONDITIONED DEMONSTRATION SELECTION

Semantic retrieval selects demonstrations resembling the query (Liu et al., 2022), while Pattern-Aware CoT considers the diversity of reasoning patterns in demonstrations (Zhang et al., 2024). Model-routing methods instead select a solver from a candidate model pool (Wang et al., 2026). Our setting fixes the target solver and changes only its demonstration-based assistance. Selection therefore considers both resemblance to the query and guidance for reasoning operations associated with the solver's observed weaknesses. A broader discussion appears in Appendix A.

3 LOGIC-AWARE TEST-TIME REASONING ROUTING

We introduce LOGICSIG, a structured representation of reasoning patterns, and LOGICROUTE, a training-free framework that uses these representations to select query-adaptive demonstrations for a target language model.

3.1 OVERALL FRAMEWORK

LOGICROUTE consists of three stages: *logic characterization*, *demonstration construction*, and *dynamic routing*.

Logic characterization. During offline calibration, the frozen target model M produces a zero-shot reasoning trace for each calibration example. A strong diagnostic model analyzes the query and trace

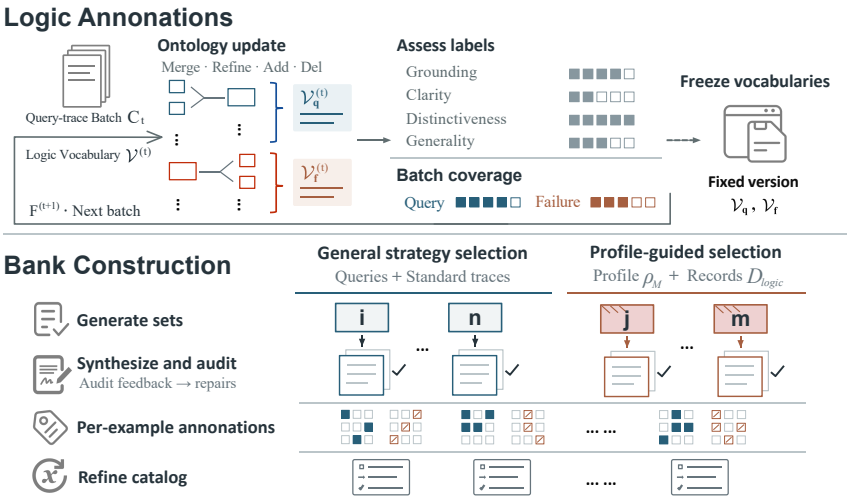


Figure 2: **Construction of LOGICSIG and the demonstration bank.** *Top:* Iterative ontology refinement, quality scoring, and coverage assessment produce fixed vocabularies for multi-hot signatures; feedback is reused in the next batch. *Bottom:* General and profile-guided repair strategies yield audited demonstration sets. Each set includes a complete prompt, per-example operation and intended-repair annotations, and a routing description.

to identify two complementary types of reasoning patterns: the logical operations required by the problem and the failure-side patterns exhibited by the model. The resulting signatures are aggregated to characterize the model's empirical performance under different logic-pattern configurations.

Demonstration construction. Using the target model's logic profile ρ_M, we identify recurring weaknesses and construct targeted demonstration sets to address them. These complement general sets that cover reusable reasoning structures. Each set contains audited demonstrations, logic annotations, and a routing description summarizing its applicability and intended guidance.

Dynamic routing. For each test query, the frozen target model M selects one strategy set from the frozen bank, either by assessing its routing description or by extracting logic features and applying a fixed distance rule. The same model then answers the query using the selected demonstrations.

Figure 2 summarizes the offline construction of logic signatures and the demonstration bank.

3.2 LOGIC SIGNATURES

A *logic signature* describes reasoning from two complementary perspectives: the logical operations involved in a problem and the failure-side patterns associated with the target model.

3.2.1 TRACE-CONDITIONED LOGIC ANNOTATION

Let $\mathcal{D}_{\text{cal}} = \{(x_i, y_i)\}_{i=1}^{N}$ denote the calibration set. For each example x_i, the frozen target model produces a zero-shot reasoning trace $r_i \sim M(\cdot \mid x_i)$. A strong diagnostic model A jointly analyzes the query and the target-model trace:

$$A(x_i, r_i) \to (z_q(x_i), z_f(x_i, M), \epsilon_i)\,, \tag{2}$$

where ϵ_i is a short natural-language audit note.

The query-side logic signature is defined as $z_q(x_i) \in \{0,1\}^{K_q}$, where each active dimension corresponds to a logical operation involved in solving the problem, such as problem formalization, constraint modeling, or multi-step deduction. This signature characterizes the problem's logical structure; the target-model trace provides auxiliary evidence about the operations involved in the attempted solution.

The failure-side logic signature is $z_f(x_i, M) \in \{0,1\}^{K_f}$, which records reasoning patterns observed to be incorrect, unstable, or omitted in the target model's trace. This signature is trace-conditioned and model-specific: it captures failure-side patterns exhibited by model M on query x_i.

Semantically redundant labels are merged and overly broad labels are refined to maintain consistent granularity. After each ontology update, a separate call to the diagnostic model scores each new or revised label for grounding, clarity, distinctiveness, and generality. Query-side and failure-side coverage of the current batch are also rated. Appendix D.1 details the construction procedure.

3.2.2 Ontology Encoding

Raw logic labels produced by the diagnostic model are normalized into two global vocabularies, $\mathcal{V}_q = \{\ell_1^q, \ldots, \ell_{K_q}^q\}, \mathcal{V}_f = \{\ell_1^f, \ldots, \ell_{K_f}^f\}$. The normalized labels are then converted into fixed-dimensional multi-hot vectors:

$$[z_q(x_i)]_j = \mathbb{I}[\ell_j^q \in \mathcal{L}_q(x_i)], \quad [z_f(x_i, M)]_k = \mathbb{I}[\ell_k^f \in \mathcal{L}_f(x_i)]. \tag{3}$$

The complete logic signature is therefore $z(x_i, M) = z_q(x_i) \oplus z_f(x_i, M)$, where $\oplus$ denotes concatenation. This sparse representation provides a common interface for model profiling, demonstration construction, and logic-aware routing.

3.2.3 Model-Specific Logic Profile

We aggregate instance-level signatures to characterize the target model's reliability under different logic-pattern configurations.

Let ℓ denote either an individual logic label or a combination of labels from the predefined taxonomy. We define the empirical accuracy of model M under ℓ as

$$\epsilon_M(\ell) = \mathbb{E}_{x_i \in \mathcal{X}_\ell} \left[\phi(x_i)\right], \tag{4}$$

where $\mathcal{X}_\ell$ is the subset of calibration examples associated with logic pattern ℓ, and $\phi(x_i)$ equals 1 if model M answers x_i correctly and 0 otherwise.

We denote the collection of these calibration accuracies and their associated logic annotations by ρ_M. The profile describes the solver's observed reliability under different reasoning requirements, beyond a single benchmark score. It identifies relevant model vulnerabilities to inform demonstration construction and routing.

3.3 Demonstration Bank

The demonstration bank $\mathcal{B} = \{b_1, \ldots, b_K\}$ contains candidate strategy sets. Each item b_i corresponds to one strategy S_i with n_i problem–solution demonstrations $e_{i,1}, \ldots, e_{i,n_i}$:

$$b_i = (P_{b_i}, \mathcal{F}_i, m_i), \quad \mathcal{F}_i = \left\{\left(\bar{z}_q(e_{i,j}), \bar{z}_r(e_{i,j})\right)\right\}_{j=1}^{n_i}. \tag{5}$$

Here, P_{b_i} is the complete few-shot prompt containing the demonstrations and strategy guidance. The per-example annotations $\bar{z}_q$ encode represented reasoning operations in $\mathcal{V}_q$, while $\bar{z}_r$ encodes intended repair patterns in $\mathcal{V}_f$. The routing description m_i contains three fields: applicable scenarios, observable query features, and reasoning directives.

The default bank contains three general sets and three targeted repair sets, with three to five examples per set. General sets cover reusable reasoning structures; targeted sets address recurring weaknesses identified from ρ_M and the target model's calibration records. The bank is frozen before evaluation, and each query is routed to exactly one complete set. Appendix D.2 details construction.

3.4 Logic-Aware Routing

Given a test query q, the frozen target model M selects one strategy from $\mathcal{S} = \{S_1, \ldots, S_K\}$ before generating its task-solving response. We consider two complementary routing mechanisms: the LLM-based router uses M to assess strategy descriptions, while the feature-space router uses M to extract logic features and selects a strategy by a fixed distance rule. Routing uses the query and frozen calibration-derived information, without access to the test query's solution trace, final answer, reference answer, or correctness.

3.4.1 LLM-BASED ROUTER

The LLM-based router uses the frozen target model M itself to assess context-dependent relationships between a query and the candidate strategies. For a query q_{text}, the routing prompt is

$$P_{\text{route}} = [I_{\text{route}}, q_{\text{text}}, \mathcal{C}_{\text{route}}], \tag{6}$$

where I_{route} specifies the selection objective and $\mathcal{C}_{\text{route}} = \{c_1, \ldots, c_K\}$ is the routing catalog. Each c_i is the routing description of the bank item b_i for strategy S_i in Section 3.3, with three fields:

- **Scenarios:** the problem types and reasoning situations covered by the set.
- **Features:** clues that indicate the strategy's applicability.
- **Directives:** concise reasoning guidance or corrective procedures.

Conditioned on P_{route}, M returns a structured decision $\{\texttt{route_index} : i^\star, \texttt{reasoning} : R_{\text{explain}}\}$, where $i^\star$ identifies the selected strategy $S_{i^\star}$ and R_{explain} briefly explains its relevance. The corresponding prompt $P_{b_{i^\star}}$ is then concatenated with the query for a subsequent answer-generation call to the same model M.

3.4.2 FEATURE-SPACE ROUTER

The feature-space router is a routing procedure of the same frozen target model M, with strategy selection specified by a fixed logic-distance rule. Let h_M denote logic-pattern extraction performed by M under the ontology. For a test query q, it produces $h_M(q) = \left(v_q^{\text{query}}, v_q^{\text{error}}\right)$. The query-side vector encodes estimated reasoning requirements in $\mathcal{V}_q$. The error-side vector encodes calibration-derived failure patterns in $\mathcal{V}_f$.

For each demonstration $e_{i,j}$ in strategy S_i, the frozen bank provides the corresponding vectors from Section 3.3: $v_{i,j}^{\text{query}} = \bar{z}_q(e_{i,j})$, $v_{i,j}^{\text{error}} = \bar{z}_r(e_{i,j})$. Thus, the error-side comparison matches predicted vulnerabilities to intended repair patterns. Using extracted features, the routing rule computes the average distances

$$d_i^{\text{query}} = \frac{1}{n_i} \sum_{j=1}^{n_i} \left\| v_q^{\text{query}} - v_{i,j}^{\text{query}} \right\|_1, \quad d_i^{\text{error}} = \frac{1}{n_i} \sum_{j=1}^{n_i} \left\| v_q^{\text{error}} - v_{i,j}^{\text{error}} \right\|_1. \tag{7}$$

The combined distance is $D(S_i) = \alpha d_i^{\text{query}} + \beta d_i^{\text{error}}$, where α, β control the contributions of query-side and error-prone logic matching.

Equivalently, M's feature-based routing decision selects the strategy with the smallest weighted average logic distance:

$$S_{\text{best}} = \arg\min_{S_i \in \mathcal{S}} \left(\frac{\alpha}{n_i} \sum_{j=1}^{n_i} \left\| v_q^{\text{query}} - v_{i,j}^{\text{query}} \right\|_1 + \frac{\beta}{n_i} \sum_{j=1}^{n_i} \left\| v_q^{\text{error}} - v_{i,j}^{\text{error}} \right\|_1 \right). \tag{8}$$

This rule favors sets whose examples match both the query's reasoning requirements and the deemed relevant model-specific weaknesses.

Because candidate vectors are encoded offline, test-time routing requires only feature extraction by M and simple distance calculations, with no external stronger model call. This variant tests whether the logic-feature space provides useful information for demonstration selection.

4 EXPERIMENTS

We evaluate the accuracy gains of LOGICROUTE and examine the roles of solver-specific profiles, route selection, and demonstration strategies. This section reports the main comparisons; Appendix C provides the data protocol, per-model results, and detailed ablations.

4.1 EXPERIMENTAL SETUP

Benchmarks and models. We evaluate on four complementary reasoning benchmarks: MATH500, AIME, GPQA-Main, and MMLU-Pro. MATH500 and AIME evaluate mathematical reasoning;

Table 1: Average results across five model scales. The columns report model-averaged accuracy (%); Macro weights all model–benchmark pairs equally. Gen. Tkn equally averages the mean target-answer lengths across model–benchmark pairs, using each solver's tokenizer. Bold/underline mark the highest/second-highest accuracy in each column.

Method	MATH500	AIME	GPQA	MMLU-Pro	Macro	Gen. Tkn
Zero-shot	<u>80.40</u>	19.33	42.05	62.23	51.00	1981.3
Fixed CoT	77.40	17.33	38.18	59.42	48.08	1789.9
Fixed CoD	67.00	13.33	37.95	53.36	42.91	384.8
Random ICL	79.00	22.00	47.95	62.24	52.80	1878.3
DemRAG	78.80	<u>25.33</u>	50.00	63.15	54.32	1843.5
LOGICROUTE-Light	80.20	<u>25.33</u>	<u>52.27</u>	<u>64.90</u>	<u>55.68</u>	1836.5
LOGICROUTE-LLM	**86.80**	**33.33**	**60.23**	**69.67**	**62.51**	1811.7

GPQA-Main and MMLU-Pro cover scientific and multi-domain questions. All evaluations use held-out splits, with split procedures specified in Appendix C.1. The Qwen-scale evaluation covers Qwen3-1.7B, Qwen3-4B, Qwen3-8B, Qwen3-14B, and Qwen3.5-27B. The cross-architecture evaluation uses Llama3.1-8B-Instruct, and Mistral3-3B-Instruct. All target models remain frozen.

Baselines. We compare against the following inference-time reasoning strategies:

- **Zero-shot**: prompts the model to reason step by step without demonstrations;
- **Chain of Thought (CoT)**: uses a fixed set of few-shot demonstrations to elicit explicit, step-by-step reasoning (Wei et al., 2022);
- **Chain of Draft (CoD)**: uses a fixed set of few-shot demonstrations to elicit concise, draft-style reasoning (Xu et al., 2025).
- **Demonstration Retrieval (DemRAG)**: uses Dense Passage Retrieval (DPR) to select the demonstrations with the highest semantic similarity to the query (Mueller et al., 2023);
- **Random ICL**: randomly selects demonstrations from the same demonstration bank, providing a routing-free control;
- **LOGICROUTE-Light**: uses a feature-space router to select the demonstration set with the smallest logic-feature distance to the query;
- **LOGICROUTE-LLM**: uses an LLM-based router to assess the query's reasoning requirements and select the most suitable demonstration set.

Evaluation. We judge each of three independently generated answers against the reference. A question is counted as correct if at least two answers are judged correct. We report accuracy under this correctness-label majority aggregation, with the number of generated tokens (*Gen. Tkn*) as the mean length of the models' answers, averaged over answers. Appendix C.1 specifies correctness-label aggregation and the reported metrics.

4.2 MAIN RESULTS

Table 1 summarizes performance across the five Qwen-family model scales. Across five Qwen models and four benchmarks, LOGICROUTE-LLM reaches 62.51% macro accuracy, compared with 51.00% for Zero-shot and 54.32% for DemRAG.

It also outperforms Fixed CoT and Random ICL and achieves the highest model-averaged accuracy on each benchmark. LOGICROUTE-Light reaches 55.68%, showing gains with a simpler selection rule, although the LLM-based router performs better on average. These comparisons assess the complete routing pipeline; the ablations below examine its components. Appendix C.2 reports the full per-model results.

4.3 CROSS-ARCHITECTURE GENERALIZATION

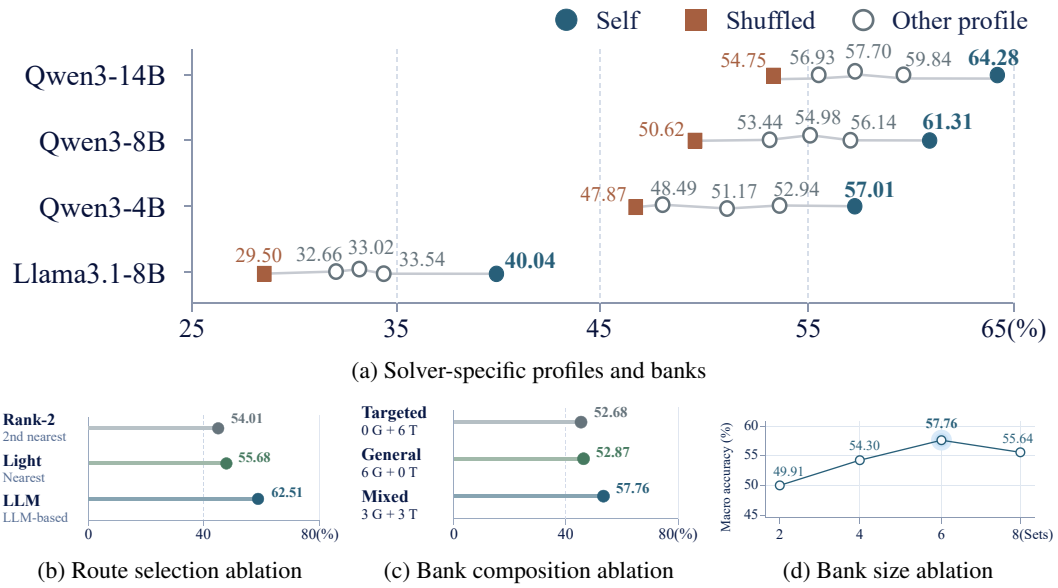
We next evaluate whether the gains extend beyond the Qwen family. Table 2 averages results over Qwen3-4B, Llama3.1-8B-Instruct, and Mistral3-3B-Instruct. Across Qwen3-4B, Llama3.1-

Table 2: Cross-architecture results averaged over Qwen3-4B, Llama3.1-8B-Instruct, and Mistral3-3B-Instruct. Gen. Tkn equally averages the mean target-answer lengths.

Method	MATH500	AIME	GPQA	MMLU-Pro	Macro	Gen. Tkn
Zero-shot	64.67	13.33	30.68	49.51	39.55	2320.0
Fixed CoT	56.67	10.00	27.65	44.67	34.75	2370.0
Fixed CoD	39.33	2.22	31.44	37.76	27.69	1752.6
Random ICL	55.67	13.33	39.01	54.50	40.63	2442.3
DemRAG	54.00	21.11	39.77	52.70	41.90	2487.6
LOGICROUTE-LLM	**67.67**	**24.44**	**53.41**	**61.62**	**51.79**	2324.2

8B-Instruct, and Mistral3-3B-Instruct, LOGICROUTE-LLM reaches 51.79% macro accuracy, improving on Zero-shot by 12.24 percentage points and on DemRAG by 9.89 points. Each model improves on average across the four benchmarks. This supports the view that logic-aware routing captures solver-specific failure structure rather than exploiting a peculiarity of a single model family. Appendix C.2 reports the full per-model results.

4.4 ABLATION STUDIES



(a) Solver-specific profiles and banks

(b) Route selection ablation

(c) Bank composition ablation

(d) Bank size ablation

Figure 3: **Ablation summaries (macro accuracy, %).** (a) With the answer-generating solver fixed, points show Self (the solver's own descriptions and bank), cross-model substitutions (routing descriptions and targeted sets jointly transferred), and Shuffled (descriptions permuted across version identifiers with the Self bank retained). Gray lines span the minimum and maximum configuration means (Table 7). (b) Rank-2 and LOGICROUTE-Light select the second-nearest and nearest strategies in logic-feature space, respectively; LOGICROUTE-LLM uses LLM-based routing (Table 8). (c) Six-set banks contain six general (G) sets, six targeted (T) sets, or three of each (Table 9). (d) Banks contain 2, 4, 6, or 8 candidate sets, with equal numbers of G and T sets (Table 10).

Solver-specific profiles. We keep the answer-generating solver fixed and jointly vary the source of the routing candidate information and the demonstration bank. The **Self** configuration uses the target solver's own candidate information and demonstration bank. The **Shuffled** control permutes Self's complete description across version identifiers while retaining the Self demonstration bank and its version-to-demonstration mapping. Self achieves the highest accuracy on every benchmark for all four solvers, outperforming the transferred and shuffled configurations (Figure 3(a)). Appendix C.3 details the cross-model substitutions and the profile-alignment control.

Route selection. Selecting the second-nearest rather than nearest logic-feature strategy reduces macro accuracy from 55.68% to 54.01% (Figure 3(b)). This aggregate comparison supports the usefulness of the logic-feature ordering. LOGICROUTE-LLM outperforms both feature-based choices on average, showing an advantage for query-conditioned LLM routing over the evaluated distance rule. Appendix C.4 reports the individual exceptions.

Bank composition and size. Across Qwen3-1.7B–14B, a mixed six-set bank reaches 57.76% macro accuracy, versus 52.87% for six general sets and 52.68% for six targeted sets (Figure 3(c)). The six-set bank has the highest observed mean among the evaluated configurations. Reducing the bank to four sets lowers macro accuracy to 54.30%, while expanding it to eight sets lowers macro accuracy to 55.64% (Figure 3(d)). These results favor offering both categories as alternatives and show that a larger candidate bank does not necessarily improve accuracy. Figure 3 summarizes the comparisons and their respective model groups. Appendices C.5 and C.6 provide the configurations and mechanism-level interpretation.

Scope of the evidence. Together, these ablations support solver-specific alignment of routing information and demonstrations, the utility of LOGICSIG for intervention selection, and the complementary value of general scaffolds and targeted repairs. Macro accuracy varies non-monotonically across the tested bank sizes, indicating that bank expansion alone cannot ensure improvement. These findings motivate joint consideration of candidate coverage and routing reliability in bank design.

5 CONCLUSION AND LIMITATIONS

We introduced LOGICROUTE, a training-free framework that uses LOGICSIG to match a query's logical demands with solver-specific vulnerabilities and select assistance before the solver responds. Across the evaluated models and benchmarks, it improves average reasoning accuracy over zero-shot inference. Selection comparisons, joint description–bank substitutions, and route perturbations support the utility of logic-aware, solver-specific assistance, while composition ablations favor banks containing both general and targeted strategies.

These findings concern behavioral regularities rather than internal reasoning mechanisms, and the logic ontology may inherit biases or omissions from the annotation models. Offline calibration and bank construction incur additional costs, and LLM-based routing adds inference overhead. Future work includes lighter-weight annotation, online profile adaptation, and end-to-end cost modeling.

AI USE STATEMENT

We used generative AI to assist with literature searches, comparisons with related work, and manuscript editing. The authors take responsibility for the final claims, code, and reported evidence.

ETHICS STATEMENT

LOGICROUTE improves auditability via logic signatures, solver profiles, intervention metadata, and routing rationales. It does not certify scientific truth and should not replace empirical validation, domain-expert review, or safety review in high-stakes use.

REPRODUCIBILITY STATEMENT

Section 4 gives benchmarks, models, baselines, metrics, and main results. Appendix B reports logic-signature analyses; Appendix C provides splits, implementation, full results, and ablations; Appendix D details annotation and bank construction. Neither calibration nor routing accesses held-out reference answers, held-out evaluator outputs, or the target model's task-solving traces, answers, or correctness on held-out test queries.

REFERENCES

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Ali Farjami, Luca Redondi, and Marco Valentino. Logic-parametric neuro-symbolic nli: Controlling logical formalisms for verifiable llm reasoning. *arXiv preprint arXiv:2601.05705*, 2026.

Nizar Islah, Istabrak Abbes, Irina Rish, Sarath Chandar, and Eilif B Muller. Failed reasoning traces tell you what is fixable (but not by reading them). *arXiv preprint arXiv:2606.05145*, 2026.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14165–14178, 2023.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, William B Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for gpt-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd workshop on knowledge extraction and integration for deep learning architectures*, pp. 100–114, 2022.

Man Luo, Shrinidhi Kumbhar, Mihir Parmar, Neeraj Varshney, Pratyay Banerjee, Somak Aditya, Chitta Baral, et al. Towards logiglue: A brief survey and a benchmark for analyzing logical reasoning capabilities of language models. *arXiv preprint arXiv:2310.00836*, 2023.

Man Luo, Xin Xu, Yue Liu, Panupong Pasupat, and Mehran Kazemi. In-context learning with retrieved demonstrations for language models: A survey. *arXiv preprint arXiv:2401.11624*, 2024.

Seyed Mahed Mousavi, Edoardo Cecchinato, Lucia Hornikova, and Giuseppe Riccardi. Garbage in, reasoning out? why benchmark scores are unreliable and what to do about it. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 1747–1759, 2026.

Aaron Mueller, Kanika Narang, Lambert Mathias, Qifan Wang, and Hamed Firooz. Meta-training with demonstration retrieval for efficient few-shot learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 6049–6064, 2023.

Xuefei Ning, Zinan Lin, Zixuan Zhou, Zifu Wang, Huazhong Yang, and Yu Wang. Skeleton-of-thought: Prompting llms for efficient parallel generation. In *International Conference on Learning Representations*, volume 2024, pp. 917–967, 2024.

Soroush Omidvartehrani, Mohammadamin Habibollah, Mohammadreza Daviran, and Davood Rafiei. Edgelm: Edge demonstrations for language models' table understanding. *arXiv preprint arXiv:2608.04390*, 2026.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Wang. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 3806–3824, 2023.

Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. Logicbench: Towards systematic evaluation of logical reasoning ability of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13679–13707, 2024.

Zhengqi Pei, Anran Zhang, Shuhui Wang, and Qingming Huang. Modeling language tokens as functionals of semantic fields. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 40114–40128. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/pei24c.html.

Zhengqi Pei, Qingming Huang, and Shuhui Wang. When LLMs develop languages: Symbolic communication for efficient multi-agent reasoning. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=ovpL0ujD6j.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems*, 36:38154–38180, 2023.

KaShun Shum, Shizhe Diao, and Tong Zhang. Automatic prompt augmentation and selection with chain-of-thought from labeled data. In *Findings of the association for computational linguistics: EMNLP 2023*, pp. 12113–12139, 2023.

Chenxu Wang, Hao Li, Yiqun Zhang, Linyao Chen, Jianhao Chen, Ping Jian, Qiaosheng Zhang, and Shuyue Hu. Icl-router: In-context learned model representations for llm routing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 33413–33421, 2026.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*, 2025.

Chih-Hsuan Yang, Jingyan Jiang, Cheng-Hau Yang, Vikram Vasudevan, Huihuo Zheng, Venkatram Vishwanath, and Rajeev Thakur. Llms can predict failure risk, but struggle to predict which collaboration protocol pays off: Cost-aware protocol routing across reasoning tasks. *arXiv preprint arXiv:2608.14927*, 2026.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Yi-Kai Zhang, De-Chuan Zhan, and Han-Jia Ye. Capability instruction tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 25958–25966, 2025.

Yufeng Zhang, Xuepeng Wang, Lingxiang Wu, and Jinqiao Wang. Pattern-aware chain-of-thought prompting in large language models. *arXiv preprint arXiv:2404.14812*, 2024.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*, 2022.

A Related Work

Test-time reasoning. A broad line of work improves language-model reasoning by modifying inference-time computation. Chain-of-Thought (CoT) elicits intermediate reasoning steps (Wei et al., 2022), while self-consistency aggregates multiple reasoning trajectories to improve robustness (Wang et al., 2022). Tree of Thoughts introduces explicit search over intermediate reasoning states (Yao et al., 2023). Efficiency-oriented methods such as Chain of Draft and Skeleton of Thought reduce reasoning verbosity or restructure generation (Xu et al., 2025; Ning et al., 2024). These approaches improve how a model reasons at test time, but typically apply a common reasoning policy without explicitly modeling which logical operations are difficult for a particular model.

In-context demonstration selection. The effectiveness of in-context learning depends strongly on the supplied demonstrations (Brown et al., 2020; Liu et al., 2022). Semantic retrieval methods select examples close to the test query in representation space (Liu et al., 2022; Luo et al., 2024), while Auto-CoT and Automate-CoT automate the construction or selection of reasoning demonstrations (Zhang et al., 2022; Shum et al., 2023). Pattern-Aware CoT further shows that reasoning patterns can provide useful selection signals beyond surface similarity (Zhang et al., 2024). Our setting is related but differs in the routing criterion: LogicRoute selects demonstrations according to both the logical requirements of the current query and the predicted vulnerabilities of the target model, rather than semantic similarity or reasoning-pattern similarity alone.

Reasoning diagnostics and structured representations. Logical reasoning benchmarks provide more fine-grained views of model capability than aggregate task accuracy. LogicBench evaluates models across diverse logical reasoning patterns (Parmar et al., 2024), while LogiGLUE studies logical reasoning capabilities across tasks (Luo et al., 2023). Neuro-symbolic approaches such as Logic-LM improve reasoning by translating problems into formal representations that can be solved by external symbolic systems (Pan et al., 2023). These works demonstrate the value of structured reasoning representations, but their primary objective is evaluation or faithful problem solving. In contrast, LogicSig uses a constrained logic representation as a behavioral interface between offline diagnosis and test-time intervention.

Capability-aware routing and orchestration. Related systems combine multiple models or tools, with different information requirements. LLM-Blender ranks and fuses already generated candidate answers (Jiang et al., 2023), while HuggingGPT orchestrates models for task execution (Shen et al., 2023). Capability Instruction Tuning learns model-capability representations for selecting a solver before candidate-answer generation (Zhang et al., 2025). Our setting instead fixes the target solver and selects its demonstration-based assistance before task-solving answer generation. LogicRoute uses solver-specific calibration information to match query requirements and predicted weaknesses with appropriate demonstrations.

B Logic Signature Results

This section reports separate, post-hoc analyses of logic annotations and observed solver performance, including empirical profiles, correctness classification, and clustering diagnostics. Using zero-shot reasoning traces from MATH500, AIME (2021–2025), GPQA-Main, and MMLU-Pro, we encode the annotated logic patterns as sparse multi-hot representations and analyze five Qwen-family models ranging from 1.7B to 27B, with Mistral3-3B-Instruct included for cross-architecture profiling. We assess the resulting representations through empirical model profiling, trace-conditioned correctness classification, and unsupervised clustering.

B.1 Empirical Profiling

We group calibration examples according to their active logic-pattern configurations and compute the empirical accuracy of each model within each group. MATH500 and AIME are grouped into the *Math* domain, while GPQA-Main and MMLU-Pro form the *General* domain. For each configuration, we report the number of associated examples n and the corresponding model accuracy. The five Qwen-family models are used to examine variation across model scales, while Mistral3-3B-Instruct is additionally included to assess architecture-specific differences.

Table 3 presents solver-specific accuracy profiles across representative logic-pattern configurations. Each configuration is defined by a set of active logic-pattern indices, and the corresponding accuracy measures how reliably each model solves calibration examples exhibiting that configuration.

The results show an overall scale-related improvement in reasoning performance: larger Qwen models generally achieve higher accuracy across logic configurations.

However, this trend is not uniform. Several configurations exhibit substantial non-monotonicity across model scales, indicating that aggregate model capacity does not translate into uniformly stronger reasoning across all logic patterns. The profiles also reveal architecture-specific differences. For example, Mistral3-3B-Instruct exhibits a distinct accuracy distribution from Qwen3-4B, particularly on the *General* configurations. These results motivate solver-specific profiling rather than assuming that reasoning vulnerabilities are shared across models of similar scale.

Table 3: Solver-specific empirical accuracy profiles across representative logic-pattern configurations. **Pattern** denotes the set of active logic-pattern indices, and n is the number of calibration examples associated with each configuration. All remaining entries report inference accuracy (%).

Pattern	n	Qwen3-1.7B	Qwen3-4B	Qwen3-8B	Qwen3-14B	Qwen3.5-27B	Mistral3-3B-Instruct
Math							
$\{6, 7, 8, 11\}$	6	0.00	0.00	0.00	0.00	66.67	33.33
$\{1, 5, 6, 7\}$	6	0.00	0.00	0.00	16.67	33.33	0.00
$\{1, 5, 6\}$	10	40.00	70.00	50.00	60.00	70.00	60.00
$\{0, 1, 6\}$	16	62.50	62.50	75.00	75.00	100.00	56.25
$\{5, 6, 8\}$	34	82.35	85.29	94.12	91.18	100.00	85.29
General							
$\{0, 1, 11\}$	9	11.11	22.22	0.00	44.44	66.67	22.22
$\{0, 2, 10, 13\}$	17	17.65	47.06	64.71	58.82	64.71	47.06
$\{0, 6, 10, 13\}$	31	22.58	51.61	54.84	48.39	61.29	35.48
$\{10, 12, 13\}$	34	38.24	52.94	61.76	58.82	73.53	58.82
$\{0, 6, 10\}$	52	26.92	28.85	46.15	46.15	80.77	36.54

B.2 Predictive Validation

We use the query-side and failure-side signatures of recorded zero-shot query–trace pairs as features for post-hoc correctness classification. This analysis assesses associations between trace-conditioned annotations and response correctness. To determine whether this representation captures predictive structure beyond surface-level textual similarity, we evaluate its ability to predict model behavior on held-out queries.

Specifically, for each model M, we train a binary correctness predictor g_M with

$$\hat{y} = g_M(h(z_q(x), z_f(x, M))) \in \{0, 1\},$$

where $y = 0$ denotes a correct response and $y = 1$ denotes an error.

We compare five feature transformations h: concatenation, elementwise difference, pairwise similarities and distances, summary statistics, and a combined representation. The classifiers are logistic regression, support vector machines with linear and RBF kernels, random forest, and XGBoost.

For this analysis, each model–benchmark annotation dataset is randomly split into training and test partitions using a class-stratified 80/20 ratio and random seed 42. These classification partitions are constructed separately from the intervention-evaluation splits in Appendix C.1. For each model–benchmark pair, we report the highest test-set macro-averaged F1 together with the corresponding accuracy on the same test partition.

Table 4 reports F1 scores and the accuracy. Accuracy ranges from 0.5222 to 0.9600, while macro-F1 ranges from 0.4808 to 0.7396. Performance varies across model–benchmark pairs. These exploratory results describe associations between trace-conditioned annotations and correctness. The logic feature vectors successfully capture reasoning vulnerabilities regardless of the underlying LLM's parameter size, proving effective even for smaller models.

Table 4: Trace-conditioned correctness classification using logic features. Each model–benchmark pair reports the highest test-set macro-F1 across the compared configurations and the accuracy of that same configuration.

	MATH500		AIME		GPQA		MMLU-Pro	
Model	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Qwen3.5-27B	0.9600	0.7396	0.6296	0.6250	0.6889	0.6023	0.8257	0.5141
Qwen3-14B	0.8700	0.6220	0.8148	0.5897	0.5667	0.5543	0.6473	0.5258
Qwen3-8B	0.8400	0.6863	0.8148	0.5897	0.5222	0.5086	0.6183	0.5723
Qwen3-4B	0.8700	0.6542	0.8148	0.5897	0.5667	0.5417	0.6763	0.6510
Qwen3-1.7B	0.7400	0.6212	0.9259	0.4808	0.6889	0.5912	0.6846	0.6752

B.3 Clustering Analysis

To examine whether the proposed logic signatures capture meaningful structural variations among reasoning instances, we perform an unsupervised clustering analysis in the logic feature space.

We perform unsupervised clustering separately on the logic representations extracted from each benchmark. The evaluated algorithms include K-Means, agglomerative clustering, DBSCAN, and spectral clustering, together with alternative feature constructions such as concatenation and difference representations. Clustering quality is evaluated using the Silhouette Coefficient, the Calinski–Harabasz Index, and the Davies–Bouldin Index. Higher Silhouette and Calinski–Harabasz scores, and lower Davies–Bouldin score indicate a better-separated and more compact clustering structure.

Table 5 reports the best-performing clustering configuration for each benchmark. The Silhouette Coefficients range from 0.1582 for AIME to 0.7233 for MATH500, indicating different degrees of geometric separation under the selected representations and algorithms.

Table 5: Clustering Quality Assessment Results

Dataset	Feature	Algorithm	n_{clusters}	Silhouette	Calinski	Davies
MATH500	concatenate	spectral	2	0.7233	23.8531	0.1976
AIME	difference	kmeans	10	0.1582	16.6740	1.3863
GPQA	concatenate	spectral	3	0.5174	10.8582	0.4987
MMLU-Pro	difference	spectral	3	0.3380	49.5441	1.0615

Together, these results demonstrate that logic signatures resolve the benchmark-specific granularity of reasoning structure: they recover compact dominant configurations when clear modes are present, while preserving richer distinctions for highly compositional problems. This structural sensitivity provides a principled basis for logic-aware instance characterization and subsequent adaptive reasoning intervention.

C Experimental Details and Additional Results

C.1 Experiment Setup

Datasets and Splits. We construct the training and held-out test splits as described below. No data augmentation, manual filtering, or label correction is applied to the benchmark questions.

- **MATH500**: We split the 500 problems into training and test subsets at a 4:1 ratio, approximately preserving the difficulty distribution.
- **AIME**: The 103 problems from 2021–2024 form the training pool, and the 30 problems from AIME 2025 form the held-out test set.
- **GPQA-Main**: We partition the 448 questions into 360 calibration questions and 88 test questions, yielding an approximately 4:1 split.

- **MMLU-Pro**: We construct a 1,204-example subset through stratified interval sampling across ten regularly spaced portions of the full benchmark. We then partition this subset into 964 calibration examples and 240 test examples, yielding an approximately 4:1 split.

Models and Implementation. All experiments run on a server with eight NVIDIA RTX 4090 GPUs, using PyTorch for model execution and vLLM for accelerated inference. The primary evaluation includes Qwen3-1.7B, Qwen3-4B, Qwen3-8B, Qwen3-14B, and Qwen3.5-27B; the cross-architecture evaluation additionally includes Llama3.1-8B-Instruct and Mistral3-3B-Instruct. All target models are evaluated without parameter updates, with their built-in deep-thinking mode disabled. GPT models are used for diagnostic annotation and demonstration generation, and Qwen3-14B for answer evaluation.

Inference Settings. Target-model inference and answer evaluation use a maximum of 8,192 tokens, a temperature of 0.1, top-$p = 0.5$, and top-$k = 20$. Logic-feature encoding uses the same settings.

Generation Seeds. For each local vLLM generation call, the implementation draws a fresh seed. The three candidate-generation calls each obtain a fresh draw under this mechanism. This statement concerns generation randomness, separately from seeds used for data splitting or other analyses.

Correctness-Label Aggregation and Evaluation Metrics. For each test question, we generate three candidate answers in separate runs. The LLM evaluator procedure evaluates each candidate against the reference and records a binary label $c_{i,r} \in \{0, 1\}$, where 1 denotes correctness for question i in run r. The aggregation operates on these recorded labels. For R selected run files, let $\mathcal{I}$ be the intersection of their question indices. We compute

$$\bar{c}_i = \mathbb{I}\left[\sum_{r=1}^{R} c_{i,r} > \frac{R}{2}\right], \quad \text{Accuracy} = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \bar{c}_i.$$

Accuracy is computed from the aggregated correctness labels defined above. *Gen. Tkn* is computed by re-tokenizing the saved answer text with the answer-generating solver's tokenizer.

C.2 Full Results across Models

Table 6 reports accuracy and *Gen. Tkn* for all seven models across the four benchmarks. With equal weighting across those combinations, LogicRoute-LLM achieves 58.70% accuracy, compared with 46.42% for Zero-shot and 50.22% for DemRAG.

Across the five Qwen scales, LogicRoute-LLM outperforms all baselines listed in Table 6 on each benchmark for each model GPQA exhibits the largest mean gain over Zero-shot across the seven models, with accuracy increasing from 37.01% to 58.28%. LogicRoute-LLM achieves the highest mean accuracy among the compared methods for each evaluated model.

Under the same averaging, the mean *Gen. Tkn* is approximately 2,038 for LogicRoute-LLM, compared with 2,139 for Zero-shot and 2,136 for DemRAG. The accuracy gains are therefore not accompanied by longer outputs on average.

C.3 Ablation on Solver-Specific Profiles

We evaluate cross-model transfer with the answer-generating solver fixed, jointly replacing the routing candidate descriptions and the targeted correction demonstrations with those associated with a source model. This design explicitly controls the source of both information components supplied before answer generation. For example, the Qwen3-8B configuration for the Qwen3-4B solver uses Qwen3-8B's candidate descriptions for routing and a bank containing the shared general sets and Qwen3-8B-specific targeted sets; Qwen3-4B still generates the answer under the selected version.

The **Self** configuration uses the target solver's own candidate information and demonstration bank. The separate **Shuffled** control starts from Self and permutes the complete description blocks across version identifiers. In Table 7, **Self** achieves the highest accuracy for every solver across all four benchmarks. Its macro accuracies are 64.28%, 61.31%, 57.01%, and 40.04% for Qwen3-14B, Qwen3-8B, Qwen3-4B, and Llama3.1-8B-Instruct, respectively. Jointly substituting another model's candidate information and demonstration bank lowers accuracy even between Qwen models

Table 6: Evaluation across all seven models. Each entry reports accuracy (%) with average generated tokens (*Gen. Tkn*) in parentheses. LOGICROUTE-LLM denotes the full logic-aware LLM router.

Method	MATH500	AIME	GPQA	MMLU-Pro
Qwen3.5-27B				
Zero-shot	94.00 (1371)	33.33 (5372)	64.77 (2731)	83.33 (1276)
Fixed CoT	95.00 (1428)	36.67 (4407)	64.77 (2610)	85.06 (1091)
Fixed CoD	82.00 (253)	26.67 (1478)	55.68 (185)	73.03 (66)
DemRAG	94.00 (1408)	46.67 (5086)	87.50 (3052)	87.14 (1303)
Random ICL	93.00 (1420)	40.00 (5347)	84.09 (3157)	85.48 (1246)
LOGICROUTE-LLM	**96.00** (1341)	**50.00** (5124)	**88.64** (3154)	**91.29** (1223)
Qwen3-14B				
Zero-shot	85.00 (1058)	23.33 (3297)	42.05 (1921)	65.56 (889)
Fixed CoT	77.00 (1350)	20.00 (2943)	40.91 (1629)	61.00 (755)
Fixed CoD	76.00 (543)	13.33 (1033)	38.64 (110)	55.60 (44)
DemRAG	84.00 (1026)	23.33 (3031)	54.55 (1401)	70.95 (682)
Random ICL	84.00 (1116)	26.67 (3282)	52.27 (1471)	65.98 (661)
LOGICROUTE-LLM	**89.00** (992)	**33.33** (2765)	**61.36** (1418)	**73.44** (533)
Qwen3-8B				
Zero-shot	80.00 (1224)	13.33 (3742)	35.23 (1966)	60.58 (838)
Fixed CoT	77.00 (834)	10.00 (3413)	30.68 (1291)	57.68 (613)
Fixed CoD	71.00 (302)	20.00 (1634)	34.09 (72)	56.02 (77)
DemRAG	80.00 (1287)	20.00 (3799)	43.18 (1277)	62.66 (476)
Random ICL	79.00 (1295)	16.67 (4094)	40.91 (966)	58.92 (725)
LOGICROUTE-LLM	**89.00** (1151)	**33.33** (3909)	**55.68** (1056)	**67.22** (719)
Qwen3-4B				
Zero-shot	77.00 (1164)	16.67 (3651)	43.18 (1884)	58.09 (867)
Fixed CoT	74.00 (1227)	16.67 (3246)	28.41 (1269)	52.70 (592)
Fixed CoD	60.00 (105)	3.33 (740)	34.09 (66)	48.55 (41)
DemRAG	75.00 (1001)	23.33 (3883)	34.09 (1490)	50.62 (537)
Random ICL	74.00 (1268)	16.67 (3471)	36.36 (1577)	55.19 (496)
LOGICROUTE-LLM	**83.00** (1162)	**30.00** (3788)	**53.41** (1608)	**61.64** (510)
Qwen3-1.7B				
Zero-shot	66.00 (1070)	10.00 (2804)	25.00 (1702)	43.57 (799)
Fixed CoT	64.00 (1455)	3.33 (3512)	26.14 (1470)	40.66 (663)
Fixed CoD	46.00 (190)	3.33 (435)	27.27 (239)	33.61 (82)
DemRAG	61.00 (1048)	13.33 (2624)	30.68 (1782)	44.40 (676)
Random ICL	65.00 (997)	10.00 (2787)	26.14 (1607)	45.64 (583)
LOGICROUTE-LLM	**77.00** (1048)	**20.00** (2337)	**42.05** (1696)	**54.77** (700)
Llama3.1-8B-Instruct				
Zero-shot	43.00 (1433)	3.33 (3669)	18.18 (2243)	38.17 (1309)
Fixed CoT	43.00 (1869)	0.00 (4301)	21.59 (1935)	38.17 (1028)
Fixed CoD	32.00 (1692)	0.00 (5215)	31.82 (799)	34.02 (349)
DemRAG	38.00 (1812)	**10.00** (4326)	34.09 (1877)	41.49 (958)
Random ICL	45.00 (1699)	3.33 (3147)	32.95 (2055)	43.57 (1028)
LOGICROUTE-LLM	**55.00** (1488)	**10.00** (4448)	**42.05** (1567)	**53.11** (774)
Mistral3-3B-Instruct				
Zero-shot	**74.00** (2166)	20.00 (5589)	30.68 (2371)	52.28 (1494)
Fixed CoT	53.00 (2907)	13.33 (5889)	32.95 (2298)	43.15 (1879)
Fixed CoD	26.00 (4463)	3.33 (6036)	28.41 (620)	30.71 (905)
DemRAG	49.00 (3316)	30.00 (5779)	51.14 (2727)	65.98 (2145)
Random ICL	48.00 (3352)	20.00 (6229)	47.73 (2877)	64.73 (2108)
LOGICROUTE-LLM	65.00 (2691)	**33.33** (5557)	**64.77** (2424)	**70.12** (1873)

of different scales. For Llama3.1-8B-Instruct, configurations using Qwen-derived candidate information and demonstration banks achieve 46.00–48.00% on MATH500 and 3.33–6.67% on AIME, compared with 55.00% and 10.00% using **Self**, respectively.

The **Shuffled** control also lowers accuracy relative to **Self**. For Qwen3-8B on AIME, accuracy decreases from 33.33% to 13.33%, supporting the utility of preserving profile–scenario alignment in this setting. Together, these results support solver-specific failure profiles rather than assuming that a single profile transfers unchanged across models.

Table 7: Cross-model substitution of routing descriptions and targeted demonstration sets, with the solver fixed. Self uses the solver's own descriptions and bank; Shuffled permutes the complete description blocks across version identifiers while retaining the Self bank. Each entry reports answer accuracy (%) with average generated tokens (*Gen. Tkn*) in parentheses.

Configuration	MATH500	AIME	GPQA	MMLU-Pro
Solver: Qwen3-14B				
Qwen3-8B	86.00 (1021)	23.33 (3504)	53.41 (1800)	68.05 (625)
Qwen3.5-27B	87.00 (1089)	26.67 (3548)	56.82 (1587)	68.88 (582)
Llama3.1-8B	84.00 (1025)	20.00 (3964)	55.68 (1167)	68.05 (652)
Shuffled	82.00 (1249)	20.00 (4078)	52.27 (1462)	64.73 (642)
Self	**89.00** (992)	**33.33** (2765)	**61.36** (1418)	**73.44** (533)
Solver: Qwen3-8B				
Qwen3-4B	84.00 (1330)	23.33 (4071)	46.59 (1095)	65.98 (654)
Qwen3-14B	81.00 (1227)	20.00 (3813)	48.86 (1229)	63.90 (597)
Llama3.1-8B	85.00 (1193)	26.67 (3162)	47.73 (1302)	65.15 (613)
Shuffled	83.00 (1312)	13.33 (4072)	44.32 (756)	61.83 (736)
Self	**89.00** (1151)	**33.33** (3909)	**55.68** (1056)	**67.22** (719)
Solver: Qwen3-4B				
Qwen3-1.7B	74.00 (1305)	23.33 (4078)	39.77 (1636)	56.85 (572)
Qwen3-8B	80.00 (1293)	26.67 (3348)	46.59 (1249)	58.51 (538)
Llama3.1-8B	79.00 (1041)	26.67 (3836)	40.91 (1419)	58.09 (509)
Shuffled	76.00 (1210)	20.00 (3377)	38.64 (1282)	56.85 (529)
Self	**83.00** (1162)	**30.00** (3788)	**53.41** (1608)	**61.64** (510)
Solver: Llama3.1-8B-Instruct				
Qwen3-4B	48.00 (1798)	3.33 (5026)	36.36 (1843)	44.40 (646)
Qwen3-8B	46.00 (1530)	6.67 (4597)	35.23 (1804)	42.74 (751)
Mistral-3B	49.00 (1684)	6.67 (4755)	34.09 (1860)	44.40 (801)
Shuffled	45.00 (1589)	0.00 (4183)	30.68 (1901)	42.32 (961)
Self	**55.00** (1488)	**10.00** (4448)	**42.05** (1567)	**53.11** (774)

C.4 Ablation on Routing Strategies

We compare demonstration-selection strategies to assess the contribution of logic-aware routing.

DemRAG selects demonstrations by semantic similarity. **Rank-2** selects the second-nearest strategy under the proposed logic-space distance. Table 8 shows that logic-aware routing improves aggregate accuracy over Chain of Thought prompting and semantic retrieval.

Averaged equally across the five Qwen scales and four benchmarks, CoT, DemRAG, Rank-2, LogicRoute-Light, and LogicRoute-LLM achieve 48.08%, 54.32%, 54.01%, 55.68%, and 62.51% accuracy, respectively. LogicRoute-Light improves macro accuracy over Fixed CoT and DemRAG in this comparison. Its advantage over Rank-2 indicates that the proposed distance provides useful routing information, although this ordering does not hold for every model–benchmark pair. LogicRoute-LLM achieves the highest mean accuracy, providing additional gains over distance-based selection.

Table 8: Routing mechanism ablation across Qwen-family models. Each entry reports accuracy (%) with average generated tokens (*Gen. Tkn*) in parentheses. Rank-2 and LOGICROUTE-Light select the second-nearest and nearest strategies in logic-feature space. LOGICROUTE-LLM uses the target LLM for query-conditioned routing.

Method	MATH500	AIME	GPQA	MMLU-Pro
Qwen3.5-27B				
Fixed CoT	95.00 (1428)	36.67 (4407)	64.77 (2610)	85.06 (1091)
DemRAG	94.00 (1408)	46.67 (5086)	87.50 (3052)	87.14 (1303)
Rank-2	95.00 (1389)	43.33 (5301)	86.36 (2931)	87.97 (1271)
LOGICROUTE-Light	**96.00** (1305)	46.67 (5337)	**89.77** (2998)	88.80 (1179)
LOGICROUTE-LLM	**96.00** (1341)	**50.00** (5124)	88.64 (3154)	**91.29** (1223)
Qwen3-14B				
Fixed CoT	77.00 (1350)	20.00 (2943)	40.91 (1629)	61.00 (755)
DemRAG	84.00 (1026)	23.33 (3031)	54.55 (1401)	70.95 (682)
Rank-2	85.00 (1294)	16.67 (3351)	52.27 (1436)	67.63 (656)
LOGICROUTE-Light	87.00 (1038)	20.00 (3553)	51.14 (1126)	70.12 (621)
LOGICROUTE-LLM	**89.00** (992)	**33.33** (2765)	**61.36** (1418)	**73.44** (533)
Qwen3-8B				
Fixed CoT	77.00 (834)	10.00 (3413)	30.68 (1291)	57.68 (613)
DemRAG	80.00 (1287)	20.00 (3799)	43.18 (1277)	62.66 (476)
Rank-2	78.00 (1553)	16.67 (4210)	44.32 (1295)	61.00 (715)
LOGICROUTE-Light	81.00 (1168)	23.33 (3566)	47.73 (1380)	62.24 (646)
LOGICROUTE-LLM	**89.00** (1151)	**33.33** (3909)	**55.68** (1056)	**67.22** (719)
Qwen3-4B				
Fixed CoT	74.00 (1227)	16.67 (3246)	28.41 (1269)	52.70 (592)
DemRAG	75.00 (1001)	23.33 (3883)	34.09 (1490)	50.62 (537)
Rank-2	75.00 (1156)	26.67 (4006)	38.64 (1336)	58.51 (621)
LOGICROUTE-Light	75.00 (1171)	23.33 (4305)	40.91 (1548)	61.00 (495)
LOGICROUTE-LLM	**83.00** (1162)	**30.00** (3788)	**53.41** (1608)	**61.64** (510)
Qwen3-1.7B				
Fixed CoT	64.00 (1455)	3.33 (3512)	26.14 (1470)	40.66 (663)
DemRAG	61.00 (1048)	13.33 (2624)	30.68 (1782)	44.40 (676)
Rank-2	62.00 (941)	13.33 (2401)	29.55 (1365)	42.32 (683)
LOGICROUTE-Light	62.00 (941)	13.33 (2401)	31.82 (1247)	42.32 (705)
LOGICROUTE-LLM	**77.00** (1048)	**20.00** (2337)	**42.05** (1696)	**54.77** (700)

C.5 Ablation on Demonstration Bank Composition

We compare mixed and single-category demonstration banks while holding the number of candidate sets fixed at six.

The full bank contains three general enhancement sets and three targeted repair sets. The *w/o Target* variant uses six general enhancement sets, whereas *w/o General* uses six targeted repair sets. Table 9 reports the results.

Across four Qwen models from 1.7B to 14B and four benchmarks, the full configuration achieves 57.76% macro accuracy, compared with 52.87% for *w/o Target* and 52.68% for *w/o General*. Both single-category banks improve over CoT, but the mixed bank yields the highest aggregate accuracy. This pattern supports the complementary value of general reasoning guidance and solver-specific correction within the candidate bank, rather than attributing the gains to either category alone.

Table 9: Demonstration-bank composition ablation. All banks contain six sets: general only (*w/o Target*), targeted only (*w/o General*), or three of each (LogicRoute-LLM). Each entry reports accuracy (%) with average generated tokens (*Gen. Tkn*) in parentheses.

Method	MATH500	AIME	GPQA	MMLU-Pro
Qwen3-14B				
Fixed CoT	77.00 (1350)	20.00 (2943)	40.91 (1629)	61.00 (755)
w/o Target	**89.00** (996)	26.67 (3207)	59.09 (1404)	71.78 (632)
w/o General	85.00 (943)	**33.33** (3613)	57.95 (1430)	69.29 (507)
LogicRoute-LLM	**89.00** (992)	**33.33** (2765)	**61.36** (1418)	**73.44** (533)
Qwen3-8B				
Fixed CoT	77.00 (834)	10.00 (3413)	30.68 (1291)	57.68 (613)
w/o Target	86.00 (1101)	23.33 (3165)	50.00 (1458)	64.73 (786)
w/o General	84.00 (1084)	20.00 (4522)	46.59 (1075)	59.75 (631)
LogicRoute-LLM	**89.00** (1151)	**33.33** (3909)	**55.68** (1056)	**67.22** (719)
Qwen3-4B				
Fixed CoT	74.00 (1227)	16.67 (3246)	28.41 (1269)	52.70 (592)
w/o Target	80.00 (1005)	20.00 (3899)	45.45 (1489)	59.75 (486)
w/o General	78.00 (1259)	26.67 (3991)	43.18 (1223)	**62.24** (516)
LogicRoute-LLM	**83.00** (1162)	**30.00** (3788)	**53.41** (1608)	61.64 (510)
Qwen3-1.7B				
Fixed CoT	64.00 (1455)	3.33 (3512)	26.14 (1470)	40.66 (663)
w/o Target	71.00 (937)	13.33 (2394)	35.23 (1341)	50.62 (483)
w/o General	74.00 (1285)	16.67 (2045)	36.36 (1550)	49.79 (669)
LogicRoute-LLM	**77.00** (1048)	**20.00** (2337)	**42.05** (1696)	**54.77** (700)

C.6 Ablation on Demonstration Bank Size

We vary the number of candidate demonstration sets while maintaining equal numbers of general enhancement and targeted repair sets. The 2-, 4-, 6-, and 8-set banks contain one, two, three, and four sets of each category, respectively. LogicRoute uses the six-set bank by default.

Across the four Qwen models and four benchmarks in Table 10, the 2-, 4-, 6-, and 8-set banks achieve mean accuracies of 49.91%, 54.30%, 57.76%, and 55.64% respectively. Mean accuracy increases as the bank grows from two to six sets, then declines at eight. The six-set bank therefore performs best among the evaluated configurations, indicating that expanding the candidate bank does not necessarily improve accuracy.

D Construction of Logic Signatures and Demonstrations

This appendix describes the prompts, data requirements, and validation procedures used by the construction utilities. The workflow comprises logic-ontology induction from target-model re-

Table 10: Demonstration-bank size ablation. Each entry reports accuracy (%) with average generated tokens (*Gen. Tkn*) in parentheses. LOGICROUTE-LLM denotes the default six-set configuration, containing three general enhancement sets and three targeted repair sets.

Method	MATH500	AIME	GPQA	MMLU-Pro
Qwen3-14B				
Fixed CoT	77.00 (1350)	20.00 (2943)	40.91 (1629)	61.00 (755)
2 sets	86.00 (1009)	23.33 (3485)	51.14 (1414)	69.71 (585)
4 sets	88.00 (1103)	26.67 (3235)	57.95 (1429)	71.37 (629)
8 sets	**89.00** (963)	30.00 (3033)	**61.36** (1399)	70.95 (570)
LOGICROUTE-LLM	**89.00** (992)	**33.33** (2765)	**61.36** (1418)	**73.44** (533)
Qwen3-8B				
Fixed CoT	77.00 (834)	10.00 (3413)	30.68 (1291)	57.68 (613)
2 sets	80.00 (1203)	23.33 (3692)	47.73 (1078)	63.49 (611)
4 sets	86.00 (1165)	23.33 (3719)	51.14 (1038)	64.73 (681)
8 sets	86.00 (1283)	26.67 (3938)	54.55 (1203)	65.98 (688)
LOGICROUTE-LLM	**89.00** (1151)	**33.33** (3909)	**55.68** (1056)	**67.22** (719)
Qwen3-4B				
Fixed CoT	74.00 (1227)	16.67 (3246)	28.41 (1269)	52.70 (592)
2 sets	77.00 (1061)	23.33 (4001)	37.50 (1701)	58.92 (542)
4 sets	81.00 (1142)	30.00 (3905)	46.59 (1558)	60.58 (506)
8 sets	81.00 (1226)	**33.33** (4156)	46.59 (1419)	61.00 (484)
LOGICROUTE-LLM	**83.00** (1162)	30.00 (3788)	**53.41** (1608)	**61.64** (510)
Qwen3-1.7B				
Fixed CoT	64.00 (1455)	3.33 (3512)	26.14 (1470)	40.66 (663)
2 sets	69.00 (1106)	13.33 (2308)	29.55 (1379)	45.23 (701)
4 sets	72.00 (1094)	**20.00** (2460)	37.50 (1632)	51.87 (640)
8 sets	**78.00** (995)	16.67 (2061)	39.77 (1449)	49.38 (799)
LOGICROUTE-LLM	77.00 (1048)	**20.00** (2337)	**42.05** (1696)	**54.77** (700)

sponses, model profiling from precomputed sample-level signatures, and the construction of candidate demonstration sets with routing metadata.

D.1 Logic Ontology Construction and Signature Representation

1. Calibration inputs. Let $\mathcal{D}_{\text{cal}} = \{(x_i, y_i)\}_{i=1}^{N}$ denote the calibration pool for a target model M, where x_i is a query and y_i its reference answer. The construction stage consumes previously collected zero-shot responses r_i^M, containing the model's reasoning and final prediction. In the question–answer pairs supplied to the diagnostic model, the answer field contains r_i^M. The reference answer is used subsequently for demonstration synthesis.

2. Joint ontology induction and refinement. Starting from an existing ontology, the implementation constructs batches by drawing question–response pairs in round-robin order from the declared sources, with 50 pairs per batch by default. For batch $\mathcal{C}_t$, a diagnostic LLM returns the complete updated query-side and failure-side vocabularies:

$$\mathcal{V}^{(t)} = G\Big(\mathcal{V}^{(t-1)}, \mathcal{C}_t, F^{(t-1)}\Big), \tag{9}$$

where $\mathcal{V}^{(t)} = (\mathcal{V}_q^{(t)}, \mathcal{V}_f^{(t)})$ and $F^{(t-1)}$ is the preceding round's scoring feedback. Query-side labels describe required reasoning operations, whereas failure-side labels describe incorrect, unstable, or omitted reasoning evidenced by the supplied responses. The prompt explicitly distinguishes these two roles and instructs the model not to invent failures in correct reasoning.

Each label contains a name, category, definition, representative scenarios, and a difficulty level. The update prompt preserves useful existing labels and their order where possible, while allowing additions and definition revisions supported by the current batch. For MATH500 and AIME, the prompt emphasizes formalization, constraints, and derivation; for GPQA and MMLU-Pro, it additionally considers knowledge integration, contextual interpretation, and distractors.

3. Label-quality assessment and versioning. After each update, a separate call to the same diagnostic LLM scores every newly added or revised label on four criteria: *grounding*, *clarity*, *distinctiveness*, and *generality*. Each criterion receives an integer score from 1 to 5, and the reported mean is their arithmetic average. The model also rates query-side and failure-side coverage of that batch on the same scale, even when no labels have changed.

The complete vocabularies, scores, source information, and ontology identifiers are saved after every batch. Iteration ends after all supplied records have been consumed. Downstream processing uses a fixed ontology version with a fixed coordinate order.

4. Sample-level signature interface. Let $\mathcal{V}_q = \{\ell_1^q, \ldots, \ell_{K_q}^q\}$ and $\mathcal{V}_f = \{\ell_1^f, \ldots, \ell_{K_f}^f\}$ denote the frozen vocabularies, indexed in a fixed order. For a separately annotated calibration example, let $\mathcal{L}_q(x_i) \subseteq \mathcal{V}_q$ contain the required query-side operations and $\mathcal{L}_f(x_i, r_i^M) \subseteq \mathcal{V}_f$ contain the failures identified in the recorded response of M. The corresponding multi-hot representation is defined by

$$[z_q(x_i)]_j = \mathbb{I}\big[\ell_j^q \in \mathcal{L}_q(x_i)\big], [z_f(x_i, M)]_k = \mathbb{I}\Big[\ell_k^f \in \mathcal{L}_f(x_i, r_i^M)\Big]. \tag{10}$$

The concatenated signature is

$$z(x_i, M) = z_q(x_i) \oplus z_f(x_i, M) \in \{0,1\}^{K_q + K_f}, \tag{11}$$

where $\oplus$ denotes concatenation and the dependence on the recorded response is implicit in $z_f(x_i, M)$. Given the label assignments, conversion to a multi-hot vector is deterministic.

The resulting sparse representation provides a unified interface for model profiling, correctness prediction, targeted demonstration construction, and logic-aware routing. At test time, the router reuses the frozen ontology to assess the relevance of calibration-derived failure patterns to each query.

D.2 Demonstration Bank Construction

The bank $\mathcal{B}$ contains six candidate strategy sets: three general sets (A–C) and three target-model-specific repair sets (D–F). Each bank item corresponds to a complete few-shot prompt P_b containing

several problem–solution pairs, together with selection and routing metadata. The number of examples per set is configurable from three to five. The router selects one set for each query.

1. General strategy selection. A strong LLM receives the candidate query texts and the standard reasoning traces. It is prompted to identify reusable reasoning structures and propose three complementary strategies with representative source problems. Each strategy specifies a concise reasoning rule, ontology-linked pattern labels, and a rationale for each selected index. Selection aims to cover varied logical structures.

2. Profile-guided repair selection. For targeted selection, let $c_i \in \{0, 1\}$ denote the evaluated correctness of the target model's calibration answer. The implementation groups records separately on each side of the signature, both by individual active labels and by identical activation configurations. For a group p with index set $\mathcal{I}_p$ and support $n_p = |\mathcal{I}_p|$, it records empirical conditional accuracy and error count:

$$\widehat{a}_M(p) = \frac{1}{n_p} \sum_{i \in \mathcal{I}_p} c_i, \qquad e_p = \sum_{i \in \mathcal{I}_p} (1 - c_i). \tag{12}$$

Groups below a configurable minimum support are omitted from the profile.

The targeted-selection prompt receives this profile, and D_{logic}, which contains each candidate's query, index, correctness, and query/failure vectors. It proposes three corrective meta-rules with supporting examples. These source failures are subsequently rewritten as correct demonstrations.

3. Demonstration synthesis and auditing. For each selected problem, the generation prompt receives the original query, reference answer, and strategy definition. For targeted strategies, it additionally receives the model profile, frozen ontology definitions, and the selected examples' diagnostic records. The LLM is instructed to produce a concise, correct English solution that makes the strategy's reasoning procedure concrete, retaining necessary intermediate derivations and checks.

A separate call to the same LLM audits each generated solution for completeness, reasoning correctness, grounded premises, and consistency of the derivation with the reference answer. Solutions failing any check are regenerated using the audit feedback; the default permits up to two repair rounds after the initial generation. The implementation saves the generated candidates and audit outcomes for each attempt.

4. Routing-catalog refinement. After each family of strategy sets has been generated and audited, an LLM produces a preliminary routing description grounded in the strategy definitions and completed demonstrations. A subsequent joint pass refines all six descriptions to clarify their applicability and reduce redundant wording. The resulting catalog contains four fields:

- **Version:** the identifier of the corresponding strategy set.
- **Scenarios:** a short description of the problems to which the strategy applies.
- **Features:** concrete cues observable in a new query before solving it.
- **Directives:** concise, action-oriented guidance summarizing the intended reasoning or corrective procedure.

This pass revises routing descriptions without rewriting or merging the demonstration sets.

Through this decoupled architecture, the router do not need to parse complex underlying defects. Instead, it efficiently matches explicit trigger features with corresponding logic and remediation signatures, ensuring precise scheduling of a potentially useful set $b \in \mathcal{B}$.