# RGDT-BENCH: BENCHMARKING LLM REASONING FOR RULE-GOVERNED DECISIONS AND THEIR JUSTIFICATIONS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

We study reasoning in **Rule-Governed Decision Tasks (RGDTs)**, where models apply external rules to case facts and justify decisions, as required in policy, contract, and compliance settings. Beyond the deductive capability emphasized by standard mathematical and logical reasoning tasks, RGDTs require interpreting rules and their applicability, assessing conditions from evidence, combining judgments under rules and exceptions, and providing checkable justifications. These demands motivate a new benchmark that assesses existing LLMs' reasoning abilities by evaluating both their decisions and their stated grounds. We introduce ***RGDT-Bench***, providing 202.1K condition-level supervision slots across four task tracks and eight supported task–probe combinations that vary access to supporting information. Label-blind extraction and deterministic checks produce labels for *warrant completeness*: source-referenced coverage and consistency of stated decision grounds. The benchmark attributes failures to four process layers: rule use, condition, evidence, and aggregation, and checks the final outcome. Among evaluable correct responses, warrant incompleteness averages 40.2% across six evaluated LLMs and supported task–probe combinations. Such warrant incompleteness poses potential safety risks and remains difficult to detect: the best of seventeen existing evaluators reaches only 57.69% (random: 50%) task-averaged area under the receiver operating characteristic curve (AUROC). To address this difficulty, we train a simple reward model with warrant supervision. It achieves 69.24% task-averaged AUROC among correct answers, exceeding the matched outcome-supervised baseline by 10.37 pp (percentage points) and the best existing evaluator by 11.55 pp. Beyond completeness assessment, the model outperforms both outcome-supervised baselines across nearly all response-selection comparisons, supporting ***RGDT-Bench***'s warrant supervision for RGDT reasoning.

## 1 INTRODUCTION

From benefit applications and contract review to transaction approval and compliance assessment, many routine decisions depend on how written rules apply to a particular case. These are **Rule-Governed Decision Tasks (RGDTs)**: tasks that require both a decision and an explanation of how the governing rules and case facts support it. Compared with mathematical and formal-logical benchmarks centered on deduction, this setting emphasizes interpreting external decision criteria and establishing how they apply to a case. It requires four abilities: **(1) Rule interpretation and applicability.** Models must identify what the supplied rules require and when they apply. **(2) Evidence-grounded condition assessment.** Models must link facts to conditions and judge whether this evidence supports, contradicts, or leaves them unresolved. **(3) Decision aggregation.** Models must combine condition judgments according to the governing decision logic, including applicable exceptions. **(4) Checkable justification.** Models must explicitly state the required judgments, their supporting rules and evidence, and how they justify the decision. These demands arise because policy and contract clauses must be linked to differently worded or incomplete case facts.



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.

However, final-answer accuracy can conceal failures in these abilities. In Figure 1, both traces correctly reject a claim. One relies on a missed deadline despite a valid waiver, whereas the other recognizes the waiver and bases rejection on the lack of coverage. These omissions pose potential safety risks, motivating checks of how rules, evidence, and exceptions justify decisions. These demands motivate a benchmark that assesses LLMs' ability to apply rules to cases, beyond general reasoning scores or final-answer accuracy. We call the expressed support linking rules and evidence to a decision a *warrant* (Toulmin, 2003).

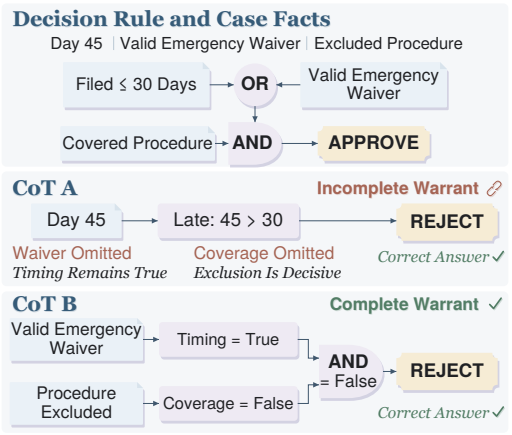



Figure 1: Answer correctness versus warrant completeness in RGDTs. Both traces reject correctly, but only CoT B states the waiver and coverage judgments. Red notes indicate omissions.

We introduce ***RGDT-Bench***, providing 202.1K condition-level supervision slots across four task tracks: *Policy* (Sun et al., 2022), *Contract* (Koreeda & Manning, 2021), *Regulation* (Saeidi et al., 2018), and *Transaction* (Zhou et al., 2025). These slots correspond to condition checks within 86,784 model responses (Table 7). Three warrant-access probes provide evidence and rules matched to the assessed conditions, add selected hard negatives, or retain longer natural document context, yielding eight supported task–probe combinations while preserving reference decisions. Label-blind claim extraction and deterministic reference checks evaluate these task requirements in the model's stated reasoning, producing labels for *warrant completeness* that we assess through human review. Process-level failure attribution covers four layers: "*rule use*", "*condition*", "*evidence*", and "*aggregation*", alongside a final *outcome* check (Figure 3). Rule-use checks assess the rule support for individual condition judgments; condition and evidence checks assess the judgments and their evidential support; aggregation checks assess whether all required decision conditions are covered and correctly combined. Together, these checks measure the coverage and consistency of the stated warrant against source-derived references, including applicable exceptions. Completeness concerns the required grounds, not merely whether the reasoning suffices to reach the answer.

Across six evaluated LLMs, 24.8%–76.7% of evaluable correct responses have incomplete warrants, with each model's results averaged equally across supported task–probe combinations. Among correct but warrant-incomplete responses, 35.9% fail at least two process layers. In distinguishing complete from incomplete warrants among correct answers, the best of seventeen existing evaluators achieves only 57.69% task-averaged AUROC under the evaluation protocol. Correct decisions thus frequently come with omissions or inconsistencies in their stated grounds, including failures spanning multiple process layers. The tested evaluators also struggle to recognize these gaps. We also train a simple reward model (Warrant-RM) on the benchmark labels, achieving 69.24% warrant AUROC among correct answers versus 58.87% for an outcome-supervised scorer with the same base model, inputs, and sample weights. In downstream selection, Warrant-RM leads both outcome-supervised baselines in mean warrant completeness at every candidate count above one and in mean accuracy at most counts across both evaluation views. **Together, these results show how RGDT-Bench connects reasoning diagnosis with warrant supervision to improve completeness assessment and response selection, supporting more reliable reasoning in RGDTs.**

## 2 Related Work

**Rule-based and evidence-grounded reasoning.** Rule-governed decisions combine rule application with evidence support. Benchmarks test natural-language rule inference (Clark et al., 2020; Tafjord et al., 2021), logically annotated entailment (Han et al., 2024), and newly specified rules (Ma et al., 2025). Evidence verification tests passage support for claims (Thorne et al., 2018; Wadden et al., 2020). Policy and contract tasks connect these abilities through conditional answers and evidence-linked entailment (Sun et al., 2022; Koreeda & Manning, 2021). RuleArena evaluates rule-use precision, recall, and application correctness (Zhou et al., 2025), while SingGuard applies policy-conditioned rules to multimodal safety moderation (SingGuard Team, 2026). These perspectives motivate ***RGDT-Bench***'s checks of required grounds and their combination into a decision.

**Warrants and reasoning coverage.** Assessing these explanations requires distinguishing stated support from factors that causally influenced the prediction (Jacovi & Goldberg, 2020; Turpin et al., 2023). SemEval-2018 Task 12 examines how implicit warrants connect premises to claims (Habernal et al., 2018). For generated reasoning, Emmons et al. (2025) evaluate answer recovery without additional non-trivial reasoning and detection of synthetic omissions. To check omissions in rule-governed decisions, ***RGDT-Bench*** specifies reference requirements for rule use, conditions, evidence, and aggregation.

**Process evaluation and supervision.** Process supervision addresses reasoning beyond answer correctness. Studies of correct mathematical solutions reveal errors in their reasoning (Uesato et al., 2022). Human and automated step labels provide more detailed supervision (Lightman et al., 2024; Wang et al., 2024). ProcessBench and PRMBench assess error localization and fine-grained step judgments (Zheng et al., 2025; Song et al., 2025), while multi-domain process reward models extend beyond mathematics (Zeng et al., 2025). Hard2Verify further examines whether mathematical steps have supported premises (Pandit et al., 2025), and the Thinking Reward Model (TRM) learns trace preferences from verified-correct solutions represented as graphs (Zhang et al., 2026). ***RGDT-Bench*** complements these approaches with response-level completeness labels to train evaluators to recognize missing decision grounds.

**Reward models and downstream selection.** Reward supervision must also be evaluated through assessment and selection. RewardBench, RM-Bench, and JudgeBench assess preferences, content/style sensitivity, and correctness (Lambert et al., 2025; Liu et al., 2025; Tan et al., 2025). VerifyBench extends this evaluation to reference-based rewards (Yan et al., 2026). Rubrics specify scoring criteria through prompts or reference-conditioned training (Liu et al., 2023; Kim et al., 2024), and can supervise policies through reasoning criteria or information coverage (Xie et al., 2026; Wang et al., 2026). However, persuasive instruction violations, length, and style can distort judgments (Zeng et al., 2024; Dubois et al., 2024; Feuer et al., 2025; Zheng et al., 2023). To assess stated warrants, we separate model-assisted claim extraction from deterministic reference checks and evaluate scorers without exposing those references. Beyond assessment, response selection can use verifier ranking or self-consistency (Cobbe et al., 2021; Wang et al., 2023), although gains depend on the problem and strategy (Snell et al., 2025). ***RGDT-Bench*** tests whether warrant supervision improves warrant assessment and selection over outcome supervision.

## 3 Task and Warrant Evaluation

We define task inputs and generated responses, establish reference warrants, and derive completeness checks and the correct-answer warrant gap.

### 3.1 Task Definition and Response Generation

Rule-Governed Decision Tasks (RGDTs) require a justified decision given $a = (q_a, F_a, R_a)$: a question $q_a$, a fact set $F_a$, and a rule set $R_a$ expressed in natural language. Rules specify conditions $c$, propositions about the case to assess, and how their judgments determine the decision. Facts used to assess whether $c$ holds form its evidence set $E_a(c) \subseteq F_a$. Our unified, extensible schema makes these rule–condition and condition–evidence links explicit.

To solve instance $a$, a **generator** produces a two-part response $\tau = (h, \hat{y})$: the *native reasoning trace* $h$ (a written chain-of-thought, CoT) and the separately reported *terminal decision* $\hat{y}$. The response's stated reasoning is its *warrant*, whose completeness ***RGDT-Bench*** assesses for diagnosis and supervision. The decision is correct when $\hat{y} = y_a$, where $y_a$ is the reference answer used for evaluation and withheld from the generator. Figure 2 introduces three probes that vary support access to diagnose reasoning while preserving the decision problem (Section 4.1).

### 3.2 Reference Warrants

To evaluate responses against source requirements, we use a *reference warrant* $w$, which records the required condition judgments, support, and decision logic. Structured records $(a, \tau, w)$ separate input, response, and reference requirements, keeping reference judgments and verification annotations hidden from the generator. This distinguishes what the response states from what it must justify.

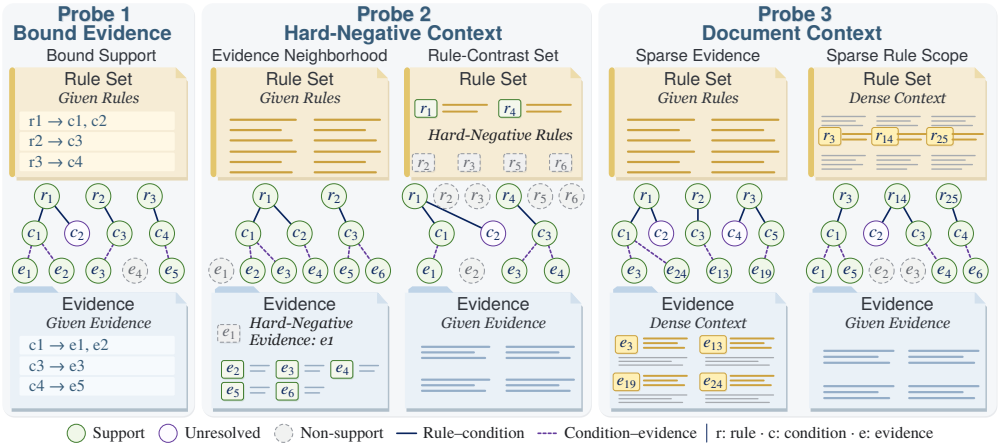



Figure 2: Warrant-access probes in RGDT-Bench. Circles denote rules ($r$), conditions ($c$), and evidence ($e$). Solid links connect rules to their conditions, and dashed links connect evidence to the conditions it supports. A rule may specify several conditions, and a condition may have multiple supporting passages or remain unresolved without support. P2/P3 illustrate evidence-dense and rule-dense contexts, with hard negatives in P2 and longer natural documents in P3. Support markings are illustrative and hidden from generators.

These requirements depend on the task: question-level judgments for *Policy*, inference conditions for *Contract*, decision conditions for *Regulation*, and identification of the proposed transaction plus its overall compliance for *Transaction*. For each condition judgment, the reference specifies the applicable rules and required evidence. Evidence may comprise jointly required passages or accepted alternatives. Because the sources for *Policy* and *Transaction* do not list all evidence passages for each condition, we check whether the response's evidence supports its condition judgments, without requiring it to identify every supporting evidence passage (Table 10).

### 3.3 Warrant Completeness and Correct-Answer Gaps

Using these reference requirements, we check the response at four process layers and an outcome layer (Figure 3(b); Appendix A). **(1) Rule use** checks whether each condition judgment uses the required governing rules: all jointly required passages, or one accepted alternative. **(2) Condition** checks whether every required judgment is stated and agrees with the reference, including unresolved conditions. **(3) Evidence** checks whether those judgments have the required support in the supplied materials, so merely citing a passage is insufficient. **(4) Aggregation** checks whether all reference-required decision conditions are combined according to the prescribed logic, such as requiring all conditions to hold or accepting alternatives. Thus, rule use concerns the grounds for individual judgments, whereas aggregation concerns their coverage and combination. The derived decision must agree with any decision stated in the trace and with the terminal decision. **(5) Outcome** checks the terminal decision against the reference answer. For *Transaction* P2/P3, an additional *rule-scope* diagnostic compares identified and reference rule sets without affecting completeness labels.

We define *warrant completeness* as passing all five checks under at least one reference warrant. For response $\tau$ to instance $a$, the binary label $W_{\text{op}}(\tau, a)$ is 1 for completeness and 0 for an *incomplete warrant*. A warrant can be incomplete even when every stated step is correct. Formally,

$$W_{\text{op}}(\tau, a) = \bigvee_{w \in \mathbb{W}(a)} \bigwedge_{\ell \in \mathcal{L}_{\text{op}}} R_\ell(\tau, w), \tag{1}$$

where $\mathbb{W}(a)$ contains the reference warrants, $\mathcal{L}_{\text{op}}$ the five checks, and $R_\ell(\tau, w)$ equals 1 if $\tau$ passes check $\ell$ under $w$, and 0 otherwise.

Using these completeness labels, we measure how often correct answers lack required support. We define the *correct-answer warrant gap* (CWG) as the proportion of evaluable correct responses with incomplete warrants. Missing required claims count as incompleteness, whereas extraction or parsing failures are not evaluable and are excluded. Writing $C = 1$ for evaluability, the gap is

$$\text{CWG} = \Pr\big(W_{\text{op}}(\tau, a) = 0 \mid \hat{y} = y_a,\ C = 1\big)\,. \tag{2}$$

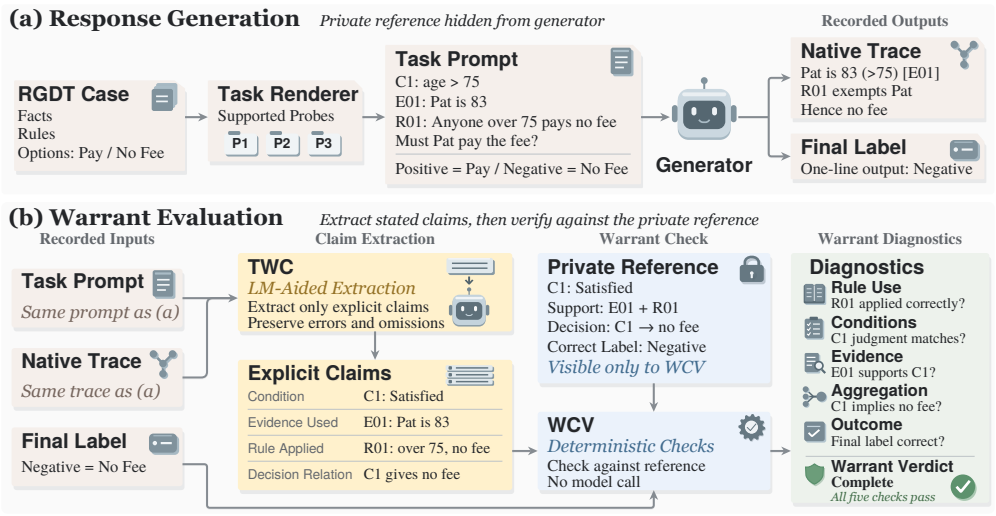



Figure 3: Response collection and diagnosis in RGDT-Bench. (a) A generator samples a native CoT and terminal decision from the RGDT input. (b) Using the outputs of (a), LLM-driven TWC extracts claims from the sampled CoT. WCV checks these claims and the terminal decision against the reference warrant, yielding condition-level diagnostics and a response-level completeness label.

Because it is computed within correct responses, this gap is not the difference between overall answer accuracy and warrant completeness.

## 4 Benchmark Construction and Validation

Following the RGDT formulation, RGDT-Bench covers policy applicability, contract inference, administrative-rule application, and basketball-transaction compliance. Three probes vary support presentation to diagnose reasoning with matched support, hard negatives, and natural documents. We then sample and annotate responses, validate the labels, and form balanced splits.

### 4.1 Warrant-Access Probe Design

We organize case facts and rules into the three input settings shown in Figure 2: **P1: Bound evidence** supplies condition-matched facts as evidence and the governing rules, leaving the generator to judge conditions and derive the decision without reference judgments. **P2: Hard-negative context** retains this support and adds selected passages that resemble relevant evidence or rules but do not determine the conditions, testing whether the generator can distinguish useful support from misleadingly similar material. **P3: Document context** retains longer excerpts of natural contracts or rule documents, testing whether the generator can locate and apply support amid passages and document structure.

All four tasks support P1. We extend *Contract* and *Transaction* to P2/P3 because their contracts and rule manuals provide passages for hard negatives and document context. In contrast, *Policy* and *Regulation* organize support around question-specific evidence and rule snippets, so we retain P1 to preserve that context. This yields eight task–probe combinations and 904 prompts. Construction preserves complete source passages, their order, and the reference decision (Appendix B).

For comparison on the same case, P2 adds hard negatives to its P1 materials without changing its support or answer. P3 retains a longer support-containing contract excerpt for *Contract* and expands P2 with surrounding rule-manual passages for *Transaction* (Appendix B.3). Because this changes information location, density, and context alongside length (Liu et al., 2024), P3 tests document-context reasoning, where added context may clarify support or distract.

### 4.2 Response Sampling and Automatic Annotation

To obtain responses for diagnosis and supervision, we use these probe inputs to prompt six generators from the Qwen, gpt-oss, and DeepSeek families for 16 responses per prompt, yielding 86,784 trace–decision pairs. Automatic annotation separates *Claim Extraction* from *Warrant Check* (Figure 3), first extracting stated reasoning and then assessing it against reference warrants.

**Claim Extraction.** An LLM-driven Trace Warrant Canonicalizer (TWC) structures the response's stated rule use, condition judgments, evidence, and aggregation without filling gaps or correcting errors. To prevent leakage, it receives task inputs, traces, and condition descriptions but not reference annotations or the separate terminal decision. Explicit paraphrases can identify passages without citation identifiers. The extracted claims then provide structured input for warrant checking.

**Warrant Check.** Using these extracted claims, a deterministic Warrant Consistency Verifier (WCV) checks their support and the terminal decision against the reference warrant. This extends evidence-support evaluation (Gao et al., 2023; Li et al., 2024) to Section 3's five checks: rule use, condition, evidence, aggregation, and outcome. Labels cover 86,435 responses and 202.1K condition judgments, providing fine-grained warrant supervision for evaluator assessment and reward-model training.

### 4.3 Label Validation and Balanced Splits

To validate these labels, we manually review 360 responses across predefined strata, obtaining 84.2% agreement (Krippendorff's $\alpha = 0.687$), with 92.7% precision and 78.5% recall for complete warrants (Appendix C.3). Despite residual errors, labels supply decision-ground supervision beyond answer correctness. Sections 5.4 and 5.5 test its value for assessment and response selection.

To use these labels without data leakage, we group related documents, operations, and similar cases, keeping all probe variants and responses within one split. To avoid overrepresenting one family's response style, we balance annotated responses (Section 4.2) across six generators, with a smaller and a larger model from each of three families. We also balance tasks and warrant labels, with nearly equal probe coverage and attention to response length (Table 8). Natural distributions support prevalence and attribution, while balanced subsets support evaluator comparison and training. Data organization precedes scoring, and reference annotations serve only as supervision and evaluation targets. These choices support accurate diagnosis and high-quality warrant supervision.

## 5 Empirical Evaluation

Using RGDT-Bench, we first establish how often correct answers have incomplete warrants and attribute these failures to reasoning layers. We then use existing **evaluators**, comprising LLM judges and reward models (RMs), to score responses without reference annotations and test whether they distinguish complete from incomplete warrants. Their limitations motivate training RMs with warrant supervision, whose value we assess first in completeness assessment and then in downstream response selection. Together, these results show how RGDT-Bench connects the diagnosis of reasoning failures to supervision that improves both warrant assessment and response selection.

### 5.1 RQ1: How Often Do Correct Answers Have Incomplete Warrants?

We first examine whether correct decisions are accompanied by complete warrants. CWG quantifies the proportion of evaluable correct responses with incomplete warrants (Equation 2). Table 1 shows that CWG ranges from 24.8% to 76.7% across six generators and that the larger model in each family has higher warrant completeness. Yet family-level gaps remain substantial (Figure 4a), showing that stronger overall performance does not ensure that correct decisions are justified by complete warrants.

Table 1: Correct-answer warrant gaps by generator (%). Final/Warr.: answer accuracy/warrant completeness. CWG′: gap without the aggregation check. Equal-weight averages over supported task–probe combinations.

| Size | Final ↑ | Warr. ↑ | CWG ↓ | CWG′ ↓ |
|---|---|---|---|---|
| *DeepSeek-R1-Distill-Llama* | | | | |
| 8B | 43.9 | 10.7 | 76.7 | 73.6 |
| 70B | 49.1 | 26.1 | 48.8 | 46.6 |
| *gpt-oss* | | | | |
| 20B | 74.4 | 47.9 | 36.7 | 32.2 |
| 120B | 77.7 | 56.3 | 28.9 | 22.1 |
| *Qwen3.5* | | | | |
| 9B | 83.8 | 63.5 | 25.4 | 22.2 |
| 27B | 87.7 | 66.6 | 24.8 | 17.5 |
| **Mean** | **69.4** | **45.2** | **40.2** | **35.7** |

To test whether explicit aggregation requirements explain these gaps, we ignore aggregation failures while retaining all other checks. Table 1 reports the relaxed gap as CWG′, retaining rule-use, condition, and evidence failures (Appendix E.1). Even under this relaxed check, mean CWG falls only from 40.2% to 35.7%: **incomplete warrants remain widespread among correct answers, posing potential safety risks in RGDTs.**

Table 2: Existing evaluators' warrant assessment (%). Positive class: complete warrants. Random baselines: AUROC/BalAcc, 50%; AUPRC, 50.00%/63.76% (full/correct-only). Bold/underline: best/second.

| Evaluator | Full evaluation view | | | Correct-only view | | |
|---|---|---|---|---|---|---|
| | AUROC ↑ | AUPRC ↑ | BalAcc ↑ | AUROC ↑ | AUPRC ↑ | BalAcc ↑ |
| ***Other LLM Judges*** | | | | | | |
| GPT-5.6 Sol | **70.19** | <u>63.84</u> | **67.09** | 53.44 | 66.14 | 53.51 |
| Gemini-2.5-Pro | 65.37 | 59.10 | 65.62 | 53.83 | 65.61 | 53.82 |
| GLM-5.2 | 64.07 | 58.20 | <u>66.05</u> | 49.00 | 63.02 | <u>54.43</u> |
| ***Qwen3.5 Series*** | | | | | | |
| Qwen3.5-0.8B | 59.53 | 56.14 | 58.29 | 53.40 | 65.95 | 52.43 |
| Qwen3.5-4B | 68.49 | 62.13 | 63.92 | **57.69** | 67.92 | 53.10 |
| Qwen3.5-9B | 68.18 | 61.57 | 64.97 | 56.17 | 66.74 | 54.36 |
| Qwen3.5-27B | 66.86 | 60.73 | 64.97 | 52.64 | 65.08 | 53.40 |
| Qwen3.5-35B-A3B | 66.92 | 60.24 | 65.31 | 54.80 | 66.04 | 54.01 |
| ***gpt-oss Series*** | | | | | | |
| gpt-oss-20B | 64.82 | 59.25 | 62.40 | 53.84 | 65.55 | 53.05 |
| gpt-oss-120B | <u>68.83</u> | 63.06 | 64.04 | <u>57.47</u> | <u>68.01</u> | 53.70 |
| ***DeepSeek-R1 Distillation Family*** | | | | | | |
| DeepSeek-R1-Distill-Llama-8B | 57.83 | 55.17 | 58.03 | 53.42 | 65.95 | 53.92 |
| DeepSeek-R1-Distill-Llama-70B | 66.40 | 60.23 | 63.91 | 55.48 | 66.55 | 54.14 |
| ***Llama Series*** | | | | | | |
| Llama-3.1-8B-Instruct | 62.13 | 57.33 | 61.60 | 49.94 | 63.94 | 53.11 |
| Llama-3.1-70B-Instruct | 67.54 | 61.45 | 63.89 | 55.77 | 67.03 | 54.39 |
| ***Pretrained Scoring Models*** | | | | | | |
| Skywork-Reward-V2-Llama-3.1-8B | 63.31 | 58.09 | 63.24 | 51.89 | 64.77 | **55.84** |
| dORM-8B | 67.48 | **65.56** | 60.58 | 54.37 | **69.36** | 50.66 |
| TRM-8B | 61.45 | 59.42 | 56.25 | 54.05 | 66.80 | 51.99 |

### 5.2 RQ2: How Are Warrant Gaps Distributed Across Reasoning Layers?

To understand the incomplete warrants observed in Section 5.1, we examine four process layers: rule use, condition, evidence, and aggregation. Failures overlap: 35.9% of incomplete warrants fail at least two requirements (Appendix Table 13). To give each response equal weight, we divide its unit contribution among its failed layers. For layers $d \in \mathcal{D}$, let $F_{id} = 1$ if response $i$ fails at $d$, and 0 otherwise. Each failed layer receives a share $\phi_{id}$ equal to one divided by the number of failed layers, while other layers receive zero:

$$\phi_{id} = \frac{F_{id}}{\sum_{d' \in \mathcal{D}} F_{id'}}, \qquad \sum_{d \in \mathcal{D}} \phi_{id} = 1, \qquad (3)$$

Averaging these shares yields the failure profiles in Figure 4b. Rule use remains the largest contributor under alternative task weights and P1-only evaluation, with aggregation contributing more than evidence (Appendix Table 13). This variation motivates matched-case probe comparisons to examine how support presentation affects failures (Appendix E.2.2). Since failures overlap, these profiles locate weaknesses rather than isolate causes. **Together, these analyses show how RGDT-Bench supports layered diagnosis of models' reasoning weaknesses in RGDTs across input conditions, beyond answer accuracy.**

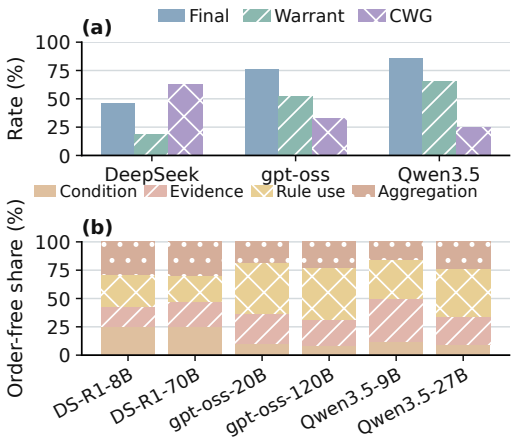



Figure 4: (a) Family-equal averages of Final (answer accuracy), Warrant (warrant completeness), and CWG (correct-answer warrant gap). (b) Each correct but incomplete response contributes equally to its failed layers. DS-R1 denotes DeepSeek-R1-Distill-Llama.

Table 3: Effects of trace input and supervision on RM assessment (%). Scores report the mean ± standard deviation across three training runs. Views include all responses or only correct answers. Outcome metrics are N/A in the correct-only view because it contains one outcome class. Bold/underline: best/second.

| Model | Full evaluation view | | | Correct-only view | | |
|---|---|---|---|---|---|---|
| | AUROC ↑ | AUPRC ↑ | BalAcc ↑ | AUROC ↑ | AUPRC ↑ | BalAcc ↑ |
| *Warrant Completeness* | | | | | | |
| Final-only ORM | $67.31_{\pm 2.06}$ | $64.38_{\pm 2.59}$ | $64.50_{\pm 2.23}$ | $51.86_{\pm 3.40}$ | $66.91_{\pm 2.65}$ | $51.84_{\pm 1.39}$ |
| Full-response ORM | $\underline{72.99}_{\pm 2.56}$ | $\underline{68.96}_{\pm 4.91}$ | $\underline{67.46}_{\pm 0.43}$ | $\underline{58.87}_{\pm 4.61}$ | $\underline{71.20}_{\pm 5.03}$ | $\underline{53.86}_{\pm 1.21}$ |
| Warrant-RM | $\mathbf{79.93}_{\pm 1.85}$ | $\mathbf{79.12}_{\pm 1.52}$ | $\mathbf{70.59}_{\pm 1.88}$ | $\mathbf{69.24}_{\pm 2.15}$ | $\mathbf{80.79}_{\pm 1.11}$ | $\mathbf{58.86}_{\pm 1.41}$ |
| *Outcome Correctness* | | | | | | |
| Final-only ORM | $89.55_{\pm 2.31}$ | $96.24_{\pm 0.85}$ | $\underline{83.87}_{\pm 2.61}$ | N/A | N/A | N/A |
| Full-response ORM | $\underline{91.48}_{\pm 0.92}$ | $\underline{96.54}_{\pm 0.23}$ | $\mathbf{84.75}_{\pm 1.46}$ | N/A | N/A | N/A |
| Warrant-RM | $\mathbf{91.91}_{\pm 0.91}$ | $\mathbf{97.06}_{\pm 0.51}$ | $81.48_{\pm 1.32}$ | N/A | N/A | N/A |

### 5.3 RQ3: How Well Do Existing Evaluators Assess Warrant Completeness?

Since correct answers often have incomplete warrants, we compare fourteen LLM judges and three pretrained scoring models on distinguishing complete from incomplete warrants: Skywork-Reward-V2-Llama-3.1-8B (Liu et al., 2026), dORM (Lee et al., 2025), and TRM (Zhang et al., 2026). All receive prompts, traces, and decisions without reference warrants.

To separate this ability from answer correctness, we use the **full evaluation view** (label-balanced data) and the **correct-answer-only view** (its correct responses). AUROC and average precision (AUPRC) measure discrimination, while balanced accuracy (BalAcc) measures classification. AUPRC comparisons stay within each view because prevalence differs. Table 2 averages each metric $m_{tp}$ for task $t$ and probe $p$ first over supported probes $\mathcal{P}_t$, then equally across the four tasks $\mathcal{T}$:

$$\overline{m}_{\text{task}} = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \frac{1}{|\mathcal{P}_t|} \sum_{p \in \mathcal{P}_t} m_{tp}. \tag{4}$$

Table 2 shows that although GPT-5.6 Sol leads the full view with 70.19% AUROC, it reaches only 53.44% (random: 50%) among correct answers, where the best existing evaluator reaches 57.69%. Llama-3.1-70B detects just 10.77% of incomplete warrants despite 98.01% complete-warrant recall (Appendix Table 15). **These results show that existing evaluators struggle to assess warrant completeness in correct RGDT answers, motivating explicit warrant supervision.**

### 5.4 RQ4: Can Warrant Supervision Improve Completeness Assessment?

To address these assessment difficulties, we train three Qwen3.5-9B reward models as evaluators under matched settings (Appendix D.3). Final-only ORM receives the prompt and decision, while Full-response ORM also receives the trace. Both learn correctness scores from outcome labels. Warrant-RM uses the same full-response inputs but learns completeness scores from warrant labels.

To compare these models consistently, we use Section 5.3's views and metrics (Table 3). **(1) Trace access.** Full-response ORM improves warrant AUROC over Final-only ORM by 5.68 pp in the full view and 7.01 pp among correct answers. **(2) Warrant supervision.** With identical inputs, Warrant-RM improves AUROC over Full-response ORM by 6.94 pp in the full view and 10.37 pp among correct answers. Correct-only gains are positive across tasks and supported task–probe combinations (Figure 5). **(3) Comparison with existing evaluators.** Warrant-RM gains 11.55 pp in correct-only AUROC over Section 5.3's best evaluator.

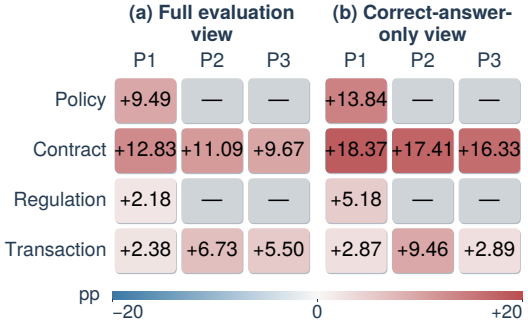



Figure 5: Warrant-supervision AUROC gains (pp). Cells show mean Warrant-RM minus Full-response ORM over three fits in both views. Gray: unsupported.

To test whether these AUROC gains also help identify incomplete warrants, we apply score thresholds. Among correct answers, Warrant-RM increases incomplete-warrant detection from 20.69% to 33.56% relative to Full-response ORM, while complete-warrant recall changes from 87.02% to 84.16% (Appendix Table 15). This improves gap detection at the cost of some complete-warrant recall. **Together, these results show that warrant supervision improves completeness assessment beyond trace access and outcome supervision, motivating its use for downstream response selection.**

### 5.5 RQ5: Do Learned Warrant Scores Improve Response Selection?

To test whether better assessment improves response selection, we compare two views: **(1) All response groups** with all sixteen candidates and resolved labels; **(2) Target-specific opportunity groups** containing both positive and negative labels for the evaluated target. Selectors share groups within each view and choose the highest-scoring response without reference labels (Appendix E.5). Full results for $N = 1$–$16$ appear in Appendix Table 20.

Table 4: Warrant-supervision effects on response selection (%). Means over three fits. Final/Full: Final-only/Full-response ORM; WRM: Warrant-RM. Bold/underline: best/second.

| | Warrant ↑ | | | Accuracy ↑ | | |
|---|---|---|---|---|---|---|
| $N$ | Final | Full | WRM | Final | Full | WRM |
| *All response groups* | | | | | | |
| 1 | **51.01** | **51.01** | **51.01** | **72.58** | **72.58** | **72.58** |
| 4 | 54.42 | <u>55.90</u> | **56.92** | 78.21 | **78.39** | <u>78.30</u> |
| 8 | 55.34 | <u>57.77</u> | **58.94** | 79.82 | <u>80.41</u> | **80.55** |
| 12 | 55.85 | <u>58.79</u> | **59.94** | 80.49 | <u>81.43</u> | **81.71** |
| 16 | 56.30 | <u>59.47</u> | **60.70** | 80.81 | <u>82.04</u> | **82.50** |
| *Target-specific opportunity groups* | | | | | | |
| 1 | **53.50** | **53.50** | **53.50** | **58.66** | **58.66** | **58.66** |
| 4 | 58.33 | <u>61.81</u> | **63.73** | 68.55 | <u>70.50</u> | **70.79** |
| 8 | 59.46 | <u>65.42</u> | **67.57** | 70.81 | <u>74.36</u> | **75.31** |
| 12 | 60.04 | <u>67.66</u> | **69.70** | 71.27 | <u>76.33</u> | **77.58** |
| 16 | 60.50 | <u>69.34</u> | **71.52** | 71.12 | <u>77.57</u> | **79.23** |

Across both views, Warrant-RM achieves the highest mean completeness at every $N > 1$ and the highest mean accuracy at most candidate counts (Table 20). On all response groups, completeness and accuracy improve by 4.40 and 1.69 pp over Final-only ORM, and by 1.23 and 0.46 pp over Full-response ORM ($N = 16$; rounded means in Table 4). Within target-specific opportunity groups, Warrant-RM exceeds Full-response ORM by 2.18 pp in completeness and 1.66 pp in accuracy at $N = 16$, with a 1.99 pp mean completeness gain across $N = 2$–$16$. Mean gains persist with random tie-breaking (Appendix E.5). **Together, these results show that even a simple warrant-supervised reward model improves completeness assessment and response selection.**

## 6 Conclusion

We formalize **Rule-Governed Decision Tasks (RGDTs)**, which require models to apply external rules to case facts and justify their decisions. Correct answers can conceal incomplete warrants, leaving potential safety risks unchecked. ***RGDT-Bench*** combines a unified, extensible schema with four task families, controlled probes, and balanced evaluation. Its layered checks support joint diagnosis of reasoning failures, which existing evaluators struggle to recognize among correct answers. The resulting warrant supervision improves reward-model assessment and downstream response selection over outcome-supervised baselines. ***RGDT-Bench*** thus connects failure diagnosis with learning signals for more reliable RGDT reasoning, enabling learning from such reasoning failures.

## 7 Limitations and Future Work

Our evaluation has three limitations. First, decisions use supplied inputs, leaving multi-turn interaction and tool use untested. Future work can track how warrants evolve as evidence accumulates. Second, text-only inputs exclude visual evidence and document layout, motivating multimodal extensions that link these sources to condition judgments. Third, we study sequence-level reward modeling rather than step-level process reward models. Evaluating the latter requires consistent reasoning-step segmentation and alignment with condition-level supervision to compare process rewards in RGDTs.

## AI USE STATEMENT

We used generative AI tools to assist with data synthesis, including candidate reasoning-trace generation, and with extracting stated claims during automatic annotation. Annotation quality was assessed through human review and agreement analysis (Appendix C.3). We also used generative AI for language polishing and typesetting assistance. The authors take responsibility for the final content of this work, including text, claims, and artifacts produced with AI assistance.

## REFERENCES

Peter Clark, Oyvind Tafjord, and Kyle Richardson. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 3882–3890, 2020. doi: 10.24963/ijcai.2020/537. URL https://www.ijcai.org/proceedings/2020/537.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. URL https://arxiv.org/abs/2110.14168.

Yann Dubois, Percy Liang, and Tatsunori Hashimoto. Length-controlled AlpacaEval: A simple debiasing of automatic evaluators. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=CybBmzWBX0.

Scott Emmons, Roland S. Zimmermann, David K. Elson, and Rohin Shah. A pragmatic way to measure chain-of-thought monitorability. arXiv preprint arXiv:2510.23966, 2025. URL https://arxiv.org/abs/2510.23966.

Benjamin Feuer, Micah Goldblum, Teresa Datta, Sanjana Nambiar, Raz Besaleli, Samuel Dooley, Max Cembalest, and John P. Dickerson. Style outweighs substance: Failure modes of LLM judges in alignment benchmarking. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=MzHNftnAM1. ICLR 2025 Poster.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 6465–6488. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.398. URL https://aclanthology.org/2023.emnlp-main.398/.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1321–1330. PMLR, 2017. URL https://proceedings.mlr.press/v70/guo17a.html.

Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. SemEval-2018 task 12: The argument reasoning comprehension task. In *Proceedings of the 12th International Workshop on Semantic Evaluation*, pp. 763–772. Association for Computational Linguistics, 2018. doi: 10.18653/v1/S18-1121. URL https://aclanthology.org/S18-1121/.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alexander Wardle-Solano, Hannah Szabó, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, Ansong Ni, Linyong Nan, Jungo Kasai, Tao Yu, Rui Zhang, Alexander Fabbri, Wojciech Maciej Kryscinski, Semih Yavuz, Ye Liu, Xi Victoria Lin, Shafiq Joty, Yingbo Zhou, Caiming Xiong, Rex Ying, Arman Cohan, and Dragomir Radev. FOLIO: Natural language reasoning with first-order logic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 22017–22031, 2024. doi: 10.18653/v1/2024.emnlp-main.1229. URL https://aclanthology.org/2024.emnlp-main.1229/.

Alon Jacovi and Yoav Goldberg. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for*

*Computational Linguistics*, pp. 4198–4205. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.386. URL https://aclanthology.org/2020.acl-main.386/.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. Prometheus: Inducing fine-grained evaluation capability in language models. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=8euJaTveKw.

Yuta Koreeda and Christopher Manning. ContractNLI: A dataset for document-level natural language inference for contracts. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 1907–1919, Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.164. URL https://aclanthology.org/2021.findings-emnlp.164/.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. RewardBench: Evaluating reward models for language modeling. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1755–1797, Albuquerque, New Mexico, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-naacl.96. URL https://aclanthology.org/2025.findings-naacl.96/.

Dong Bok Lee, Seanie Lee, Sangwoo Park, Minki Kang, Jinheon Baek, Dongki Kim, Dominik Wagner, Jiongdao Jin, Heejun Lee, Tobias Bocklet, Jinyu Wang, Jingjing Fu, Sung Ju Hwang, Jiang Bian, and Lei Song. Rethinking reward models for multi-domain test-time scaling. *arXiv preprint arXiv:2510.00492*, 2025. doi: 10.48550/arXiv.2510.00492. URL https://arxiv.org/abs/2510.00492. Version 3, revised July 2026.

Yifei Li, Xiang Yue, Zeyi Liao, and Huan Sun. AttributionBench: How hard is automatic attribution evaluation? In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 14919–14935. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.886. URL https://aclanthology.org/2024.findings-acl.886/.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi. ICLR 2024 Poster; first released as arXiv:2305.20050 in 2023.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-Eval: NLG evaluation using GPT-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2511–2522, 2023. doi: 10.18653/v1/2023.emnlp-main.153. URL https://aclanthology.org/2023.emnlp-main.153/.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. RM-Bench: Benchmarking reward models of language models with subtlety and style. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/6da1eec80095dc5937f7716db15aca4b-Abstract-Conference.html.

Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, and Yang Liu. Skywork-Reward-V2: Scaling preference data curation via human–AI synergy. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/d8836d0294651a867c2ca51012923997-Abstract-Conference.html.

Kaijing Ma, Xinrun Du, Yunran Wang, Haoran Zhang, Zhoufutu Wen, Xingwei Qu, Jian Yang, Jiaheng Liu, Minghao Liu, Xiang Yue, Wenhao Huang, and

Ge Zhang. KOR-Bench: Benchmarking language models on knowledge-orthogonal reasoning tasks. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/c6f5851a0d9cb435ed8b50e87bd6a257-Abstract-Conference.html.

Shrey Pandit, Austin Xu, Xuan-Phi Nguyen, Yifei Ming, Caiming Xiong, and Shafiq Joty. Hard2Verify: A step-level verification benchmark for open-ended frontier math. arXiv preprint arXiv:2510.13744, 2025. URL https://arxiv.org/abs/2510.13744.

Marzieh Saeidi, Max Bartolo, Patrick Lewis, Sameer Singh, Tim Rocktäschel, Mike Sheldon, Guillaume Bouchard, and Sebastian Riedel. Interpretation of natural language rules in conversational machine reading. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2087–2097, Brussels, Belgium, 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1233. URL https://aclanthology.org/D18-1233/.

Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE*, 10(3):e0118432, 2015. doi: 10.1371/journal.pone.0118432. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0118432.

SingGuard Team. SingGuard: A policy-adaptive multimodal LLM guardrail with dynamic reasoning. arXiv preprint arXiv:2606.22873, 2026. URL https://arxiv.org/abs/2606.22873.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

Mingyang Song, Zhaochen Su, Xiaoye Qu, Jiawei Zhou, and Yu Cheng. PRMBench: A fine-grained and challenging benchmark for process-level reward models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 25299–25346, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.1230. URL https://aclanthology.org/2025.acl-long.1230/.

Haitian Sun, William Cohen, and Ruslan Salakhutdinov. ConditionalQA: A complex reading comprehension dataset with conditional answers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3627–3637, Dublin, Ireland, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.253. URL https://aclanthology.org/2022.acl-long.253/.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3621–3634, 2021. doi: 10.18653/v1/2021.findings-acl.317. URL https://aclanthology.org/2021.findings-acl.317/.

Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Y. Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica. JudgeBench: A benchmark for evaluating LLM-based judges. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/9e720fce64f91114c49cfd640d821da3-Abstract-Conference.html.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: A large-scale dataset for fact extraction and verification. In *Proceedings of NAACL-HLT*, pp. 809–819, 2018. doi: 10.18653/v1/N18-1074. URL https://aclanthology.org/N18-1074/.

Stephen E. Toulmin. *The Uses of Argument*. Cambridge University Press, 2003. ISBN 9780511840005. doi: 10.1017/CBO9780511840005. URL https://doi.org/10.1017/CBO9780511840005.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems*, volume 36, pp. 74952–74965, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/ed3fea9033a80fea1376299fa7863f4a-Abstract-Conference.html.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022. URL https://arxiv.org/abs/2211.14275.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 7534–7550, 2020. doi: 10.18653/v1/2020.emnlp-main.609. URL https://aclanthology.org/2020.emnlp-main.609/.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.510. URL https://aclanthology.org/2024.acl-long.510/.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Yudong Wang, Zhe Yang, Wenhan Ma, Rang Li, Qibin Yang, Weimin Xiong, Jiangshan Duo, Liang Zhao, and Zhifang Sui. From refuse to richness: Rubric rewards for long-form hallucination reinforcement learning. arXiv preprint arXiv:2608.12337, 2026. URL https://arxiv.org/abs/2608.12337.

Weichu Xie, Haozhe Zhao, Wenpu Liu, Yongfu Zhu, Liang Chen, Minghao Ye, Zirong Chen, Yuqi Xu, Shuai Dong, Ziyue Wang, Xinbo Xu, Kean Shi, Ruoyu Wu, Xiaoying Zhang, Wenqi Shao, Baobao Chang, Nan Duan, and Jiaqi Wang. Step-wise rubric rewards for LLM reasoning. arXiv preprint arXiv:2605.17291, 2026. URL https://arxiv.org/abs/2605.17291.

Yuchen Yan, Jin Jiang, Zhenbang Ren, Yijun Li, Xudong Cai, Yang Liu, Xin Xu, Mengdi Zhang, Jian Shao, Yongliang Shen, Jun Xiao, and Yueting Zhuang. VerifyBench: Benchmarking reference-based reward systems for large language models. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=JfsjGmuFxz.

Thomas Zeng, Shuibai Zhang, Shutong Wu, Christian Classen, Daewon Chae, Ethan Ewer, Minjae Lee, Heeju Kim, Wonjun Kang, Jackson Kunde, Ying Fan, Jungtaek Kim, Hyung Il Koo, Kannan Ramchandran, Dimitris Papailiopoulos, and Kangwook Lee. VersaPRM: Multi-domain process reward model via synthetic reasoning data. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 74197–74239. PMLR, 2025. URL https://proceedings.mlr.press/v267/zeng25h.html.

Zhiyuan Zeng, Jiatong Yu, Tianyu Gao, Yu Meng, Tanya Goyal, and Danqi Chen. Evaluating large language models at evaluating instruction following. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/afc8b034823271816d14f7c1aefe1dff-Abstract-Conference.html.

Haoran Zhang, Yafu Li, Zhi Wang, Zhilin Wang, Shunkai Zhang, Xiaoye Qu, and Yu Cheng. Characterizing, evaluating, and optimizing complex reasoning. In *Proceedings of the Forty-Third International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.08498.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. ProcessBench: Identifying process errors in mathematical reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1009–1024, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.50. URL https://aclanthology.org/2025.acl-long.50/.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46595–46623, 2023. doi: 10.52202/075280-2020. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html. Datasets and Benchmarks Track.

Ruiwen Zhou, Wenyue Hua, Liangming Pan, Sitao Cheng, Xiaobao Wu, En Yu, and William Yang Wang. RuleArena: A benchmark for rule-guided reasoning with LLMs in real-world scenarios. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 550–572, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.27. URL https://aclanthology.org/2025.acl-long.27/.

## APPENDIX GUIDE

The appendix follows the paper's logic from reference requirements and probe construction to annotation, evaluation, and supplementary evidence for warrant supervision. Prompt examples connect these stages through one case.


**A Reference Warrants and Diagnostic Checks** p. 16

**B Task Coverage, Data Balance, and Warrant-Access Probes** p. 17

**B.1** Task Coverage and Leakage Prevention 17

**B.2** Natural and Balanced Response Collections 17

**B.3** Construction of the Three Warrant-Access Probes 18

**C Automatic Annotation and Human Validation** p. 19

**C.1** Information Boundaries 20

**C.2** Claim Extraction and Warrant Verification 20

**C.3** Human Review of Annotation Quality 21

**D Evaluation Measures and Matched Comparisons** p. 22

**D.1** Metrics, Weighting, and Uncertainty 22

**D.2** Existing Evaluators 24

**D.3** Matched Reward-Model Training 25

**E Supplementary Evidence for Diagnosis and Supervision** p. 25

**E.1** Robustness of Correct-Answer Warrant Gaps 26

**E.2** Layered Attribution and Probe Comparisons 26

E.2.1 Layered Failure Attribution 26

E.2.2 Matched Probe Comparisons 27

**E.3** Probability Quality and Incomplete-Warrant Detection 27

**E.4** Effects of Trace Access and Warrant Supervision 28

E.4.1 Warrant Assessment Among Correct Answers 28

E.4.2 Effect of Training Weights 30

**E.5** Response Selection and Robustness Checks 30

**F Prompt Examples for Generation, Annotation, and Evaluation** p. 32

# A Reference Warrants and Diagnostic Checks

To assess a response's warrant, the reference warrant specifies the required judgments, their support, and their roles in the decision. This appendix expands Section 3 by specifying support requirements, distinguishing rule use from aggregation, and explaining how unresolved checks are handled.

**Task inputs and hidden references.** The input schema separates the natural-language question, case facts, and rules. Evidence records link facts to the conditions they help assess, while reference judgments remain hidden from the generator. Reference warrants specify the required judgments, support, and decision logic. Completeness requires covering these grounds consistently, even when fewer stated grounds suffice for the decision.

**Checking the support for each judgment.** Rule-use and evidence checks ask whether each condition judgment has the required support. Rule use checks the governing rules, while evidence checks the supporting case information. For rules and evidence separately, the reference requires all specified passages or an accepted alternative. Evidence cannot replace required rule use, or vice versa. For example, if two evidence passages are jointly required, using only one is insufficient. If either passage is accepted as an alternative, using one meets this coverage requirement.

Some sources do not specify supporting passages for each judgment. In these cases, we do not require a particular preassigned passage, but still assess the support stated in the response against the task-specific requirements in Table 10. These checks concern the grounds for individual judgments. How those judgments combine into a decision is checked separately under aggregation.

**From condition judgments to warrant completeness.** Table 5 groups the checks used by WCV. Rule use concerns the grounds for each judgment, whereas aggregation concerns whether all reference-required decision conditions are covered and combined correctly. The decision obtained by combining the stated condition judgments must agree with any decision stated in the trace and with the separately reported terminal decision. All five checks must pass against the same reference warrant. For *Transaction*, we separately check that the response identifies the transaction action posed in the question and that its compliance judgment has the required grounds (Table 10). Compliance requires an accepted governing rule and meaningful use of the supplied evidence or rules.

Table 5: Warrant checks and separate diagnostics. The five completeness checks must pass. rule scope is a separate diagnostic. Failures shown are illustrative and may co-occur. $C = 1$ denotes evaluability in Equation 2.

| Layer | Required property | Representative failure |
|---|---|---|
| Rule scope | applied rules match a reference set where the comparison is determined, with unidentified applied rules marked NA | rule outside the reference set |
| Rule use | each condition judgment has its required rule support, using all required rule passages or an accepted alternative | missing required rule use |
| Condition | every required condition receives the reference judgment | omitted exception condition |
| Evidence | semantic support satisfies the source-specific requirement independently of literal citation format | evidence does not support the judgment |
| Aggregation | all required decision conditions enter aggregation, with their judgments combined according to the decision logic and consistent with the terminal decision | omitted aggregation input or inconsistent decision |
| Outcome | predicted label equals the final answer implied by the same reference warrant | final answer wrong |
| Evaluability $C$ | stated claims can be extracted into the required structure, although missing claims still fail | failed extraction or non-parseable trace |

**Separate diagnostics and evaluability.** Rule scope separately compares identified and reference rule sets for *Transaction* P2/P3 without affecting warrant completeness. When the applied rules cannot be identified, rule scope is marked NA (not available). Missing or incorrect claims fail the corresponding checks, whereas only extraction or parsing failures yield NOT-EVALUABLE. Several checks may fail on the same response. The results identify missing or inconsistent grounds without claiming that one failure caused another.

# B TASK COVERAGE, DATA BALANCE, AND WARRANT-ACCESS PROBES

RGDT-Bench covers four task tracks through a shared representation of decisions and their required grounds. To support both diagnosis and learning, its construction preserves task facts and rules, prevents related cases from crossing data splits, and varies access to support through three probes.

## B.1 TASK COVERAGE AND LEAKAGE PREVENTION

Cases sharing source documents, involving related operations, or having similar content are placed in the same source group. To prevent data leakage, each source group stays within one data split before probe construction or response generation. Table 6 reports source-case counts by split, with 90 test cases forming 76 such source groups.

Table 6: Source-case counts by task and split. Each case yields one prompt per supported probe (checkmarks), with all probes sharing the case split. Native labels are balanced within each supported task–split–probe combination.

| Task track | Train | Valid. | Test | All | P1 | P2 | P3 |
|---|---|---|---|---|---|---|---|
| *Policy* | 56 | 20 | 18 | 94 | ✓ | – | – |
| *Contract* | 66 | 21 | 21 | 108 | ✓ | ✓ | ✓ |
| *Regulation* | 78 | 27 | 27 | 132 | ✓ | – | – |
| *Transaction* | 70 | 24 | 24 | 118 | ✓ | ✓ | ✓ |
| **Source cases** | **270** | **92** | **90** | **452** | **904 active prompts** | | |

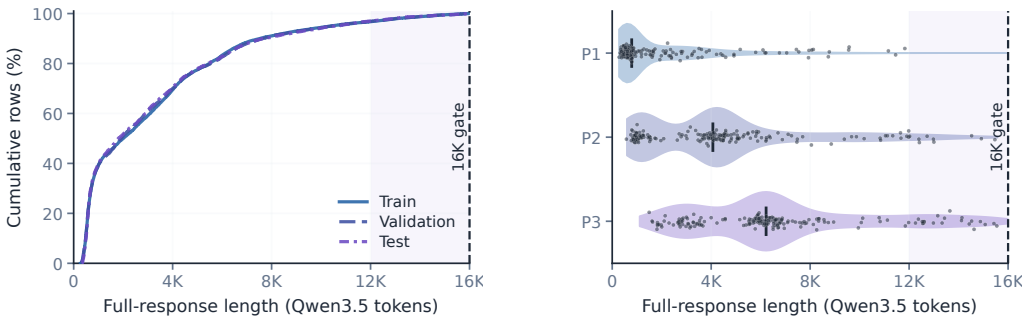
To distinguish the number of responses from the condition judgments they contain, Table 7 reports generator responses and condition judgments separately. K and M denote thousands and millions, respectively. Training contains 51,832 evaluable responses, including 23,809 complete and 28,023 incomplete warrants. The 200 training responses with not-evaluable warrant labels remain outside supervision. Uncertainty estimation also uses these source groups as the resampling units, keeping all responses from cases in the same group together. This avoids treating responses from the same or related cases as independent observations.

Table 7: Data and supervision scale by probe. Tokens include the prompt, reasoning trace, and formatting text (Qwen3.5-9B tokenizer). K/M: thousands/millions. Totals are rounded after summation.

| Statistic | Train | Validation | Test | All splits |
|---|---|---|---|---|
| ***Response Samples*** | | | | |
| Probe 1 | 25,920 | 8,832 | 8,640 | 43,392 |
| Probe 2 | 13,056 | 4,320 | 4,320 | 21,696 |
| Probe 3 | 13,056 | 4,320 | 4,320 | 21,696 |
| **Total** | **52,032** | **17,472** | **17,280** | **86,784** |
| ***Condition Slots (K)*** | | | | |
| Probe 1 | 53.0 | 17.4 | 18.4 | 88.8 |
| Probe 2 | 33.7 | 11.3 | 11.7 | 56.7 |
| Probe 3 | 33.6 | 11.3 | 11.7 | 56.6 |
| **Total** | **120.2** | **39.9** | **41.9** | **202.1** |
| ***Full-Response Sequence Tokens (M)*** | | | | |
| Probe 1 | 65.3 | 22.1 | 21.9 | 109.3 |
| Probe 2 | 59.6 | 20.3 | 20.3 | 100.2 |
| Probe 3 | 89.9 | 29.1 | 29.6 | 148.6 |
| **Total** | **214.8** | **71.5** | **71.7** | **358.1** |

## B.2 NATURAL AND BALANCED RESPONSE COLLECTIONS

Table 8 distinguishes balanced prompts, which control task–probe coverage, from the responses used for analysis and learning. The natural response collection contains all generated responses before balancing and supports prevalence and attribution analyses. From its responses with evaluable warrant labels, we form balanced subsets for evaluator comparison and training, balancing generators,




**(a)** Split ECDF **(b)** Warrant-access probe distributions

Figure 6: Prompt–trace lengths in the balanced data. (a) Cumulative distributions by split. (b) Density by probe, with dark marks denoting medians. P1 covers four tasks, whereas P2/P3 cover *Contract* and *Transaction*.

tasks, and warrant labels (Table 8). Within each task, response counts across supported probes differ by at most one. We also account for response length and coverage across source cases. All subsets retain distinct generator responses within the existing source-group splits.

Table 8: Prompt and response counts by split. The first row counts prompts, and the remaining rows count responses. Sections 5.3–5.4 share the balanced evaluation responses.

| Collection | Train | Validation | Test | Intended use |
|---|---|---|---|---|
| | | ***Prompts*** | | |
| Balanced prompts | 432 | 144 | 144 | Balanced task–probe coverage, labels, and input lengths |
| | | ***Responses*** | | |
| Natural responses | 52,032 | 17,472 | 17,280 | Generated responses before balancing |
| Evaluable responses | 51,832 | 17,400 | 17,203 | Responses with evaluable warrant labels |
| **Balanced response subset** | **11,856** | **3,720** | **3,564** | Evaluable subset for training and evaluator comparison |

Figure 6 shows prompt–trace lengths in the balanced data using the Qwen3.5-9B tokenizer. All sequences are retained without truncation. Differences across probes reflect both their context construction and supported tasks.

Response balancing precedes evaluator scoring. Sections 5.3–5.4 share evaluation responses. The reward models used in Sections 5.4–5.5 are trained on the balanced response training split and share training and validation weights. Test responses never enter fitting, calibration, or threshold selection, and reference annotations provide targets rather than model inputs. These controls make the labels useful both for comparable diagnosis and for warrant supervision.

### B.3 Construction of the Three Warrant-Access Probes

To diagnose support access while keeping the decision problem unchanged, P1 binds support to conditions, P2 adds confusable passages, and P3 restores longer natural document context. *Contract* and *Transaction* support P2/P3 because their contracts and rule manuals contain both related distractors and surrounding text. *Policy* and *Regulation* instead provide source-linked evidence for P1. The probes therefore test different support-access demands, rather than impose a predetermined difficulty order.

**P1: Bound evidence.** P1 supplies condition-matched facts as evidence, together with governing rules, leaving judgments to the generator. A condition can need several passages, and a passage can inform several conditions. *Contract* retains annotated support in document order, while *Transaction*

combines case facts with relevant manual passages. Citation labels reveal neither reference roles, judgments, nor answers.

**P2: Hard-negative context.** P2 preserves P1 support and adds passages that resemble relevant material without changing the reference decision. For *Contract*, candidates are complete evidence sets for other hypotheses in the same contract. Review keeps passages that are self-contained and easy to confuse with target evidence through shared actors, actions, exceptions, or scope, but do not support, contradict, or clarify the assessed conditions. Algorithm 1 adds the eligible evidence sets intact and preserves their source order.

**Algorithm 1** *Contract* hard-negative construction

**Require:** contract, assessed conditions, P1 support $T$, context and selection limits
**Ensure:** P2 context preserving $T$ and the decision, or rejection
$Q \leftarrow$ complete evidence sets for other hypotheses in the contract
Remove duplicates, incomplete sets, and sets overlapping $T$ from $Q$
$Q \leftarrow$ reviewed pairs $(j, e)$: set $e$ is self-contained and confusable with evidence for $j$
    through shared actors, actions, exceptions, or scope,
    but does not support, contradict, or clarify any assessed condition
Sort $Q$ by condition, then fixed source order; $S \leftarrow \varnothing$, $J \leftarrow \varnothing$
**for** each $(j, e) \in Q$ **do**
    **if** $j \notin J$, $e$ does not overlap $T \cup S$, and the complete addition fits the limits **then**
        $S \leftarrow S \cup \{e\}$; $J \leftarrow J \cup \{j\}$
    **end if**
    **if** the selection limit is reached **then**
        **break**
    **end if**
**end for**
$P \leftarrow$ render $T \cup S$ in source order
**if** $S = \varnothing$ or source, split, support, or decision checks fail **then**
    **return** rejection
**end if**
**return** $P$

For *Transaction*, candidates are rules from the same operation family, kept with the passages needed to interpret them. Review excludes rules that overlap P1 support or determine the case outcome, retaining rules that are inapplicable, already satisfied, or otherwise nondiagnostic. Among these eligible rules, construction favors additions that keep context lengths comparable and uses a fixed order to resolve ties (Algorithm 2). Thus, review establishes the distractors' suitability, while length controls only their inclusion.

**P3: Document context.** P3 preserves longer natural document context instead of adding only selected distractors. *Contract* uses a source-ordered excerpt containing the relevant support. *Transaction* expands P2 with surrounding manual sections and referenced rules, prioritizing context connected to the decision (Algorithm 3). Both retain complete passages and document order within the context budget. This preserves natural context for testing how models locate and combine support in longer documents.

**Keeping the correct decision unchanged.** To attribute differences across probes to how models locate and use support, added context must not change the facts, required support, or correct decision. We therefore check these properties after constructing P2/P3 and reject inputs that fail, while keeping source groups in their original data splits. References guide this check but remain hidden from generators and evaluators. P3 also marks omitted passages and unavailable cross-references so readers can distinguish an excerpt from a complete document.

## C Automatic Annotation and Human Validation

Reliable warrant supervision requires extracting what a response states without supplying missing reasoning. The annotation pipeline therefore separates label-blind claim extraction from reference-based verification, then checks the resulting labels through human review.

**Algorithm 2** *Transaction* hard-negative construction

**Require:** transaction, P1 support $T$, same-family rules, context and selection limits
**Ensure:** P2 context containing complete nondiagnostic rules, or rejection
$Q \leftarrow$ rules from the same operation family, with all passages needed to interpret them
Remove incomplete units, overlap with $T$, decision-determining rules, and excluded rules
Retain in $Q$ only rules reviewed as inapplicable, satisfied, or otherwise nondiagnostic
$S \leftarrow \varnothing$; $P \leftarrow$ render $T$
**while** $Q \neq \varnothing$ and the selection limit is not reached **do**
    **if** enough rules are added and $P$ reaches the desired length range **then**
        **break**
    **end if**
    $A \leftarrow$ rules in $Q$ that do not overlap $S$ and fit the context budget intact
    **if** $A = \varnothing$ **then**
        **break**
    **end if**
    $r \leftarrow$ rule in $A$ whose addition is closest to the target length; use fixed order for ties
    **if** enough rules are added, the minimum desired length is reached, and $r$ brings no improvement **then**
        **break**
    **end if**
    $S \leftarrow S \cup \{r\}$; $Q \leftarrow Q \setminus \{r\}$; $P \leftarrow$ render $T \cup S$ in source order
**end while**
**if** selection requirements or source, support, or decision checks fail **then**
    **return** rejection
**end if**
**return** $P$

**Algorithm 3** *Transaction* document-context construction

**Require:** P2 passages $T$, source manual, decision rules, rule references, context budget
**Ensure:** P3 document context preserving $T$ and the decision, or rejection
$Q \leftarrow$ source blocks containing $T$
Append blocks cited by decision rules, then blocks cited by other P2 rules
Append nearby cited blocks; order $Q$ by this priority and source position
$S \leftarrow \varnothing$
**for** each block $b \in Q$ **do**
    **if** $b$ is allowed, adds new material, and fits the budget as a complete block **then**
        $S \leftarrow S \cup \{b\}$
    **end if**
**end for**
$P \leftarrow$ render $T \cup S$ in source order, preserving rule and section boundaries
Mark omitted passages and unavailable cross-references in $P$
**if** $S = \varnothing$ or source, support, or decision checks fail **then**
    **return** rejection
**end if**
**return** $P$

### C.1 Information Boundaries

To prevent reference information from filling reasoning gaps, each stage receives only the information required for its role (Table 9). Generators and evaluators never receive reference judgments. TWC additionally excludes the separately reported terminal decision, whereas WCV uses that decision and the private reference to verify the extracted claims. A decision explicitly stated within the trace remains part of the trace.

### C.2 Claim Extraction and Warrant Verification

TWC uses GPT-5.6 Sol to extract explicit rule use, condition judgments, evidence, and aggregation while preserving errors and omissions (Figure 11). WCV applies the checks in Appendix A only after successful extraction, with extraction failures remaining not-evaluable rather than becoming negative labels.

The four tasks require different judgments: answering a policy question, testing contractual or regulatory conditions, or assessing a transaction. Table 10 shows how the same warrant checks apply

Table 9: Information available to each model or stage. Checkmarks denote visible inputs. Ref.: reference warrant; TWC: extracted claims; $h$: reasoning trace; $\hat{y}$: terminal decision. WCV verifies TWC claims using references withheld from generators and evaluators. Existing evaluators: Section 5.3.

| Stage/model | Prompt | Native $h$ | $\hat{y}$ | TWC | Ref. | Output |
|---|---|---|---|---|---|---|
| *Generation and Annotation* | | | | | | |
| Generator | ✓ | × | × | × | × | native trace + terminal decision |
| TWC canonicalizer | ✓ | ✓ | × | × | × | explicit claims + trace locations |
| WCV verifier | × | × | ✓ | ✓ | ✓ | layer findings + warrant label |
| *Evaluators* | | | | | | |
| Existing evaluator | ✓ | ✓ | ✓ | × | × | warrant score |
| Final-only ORM | ✓ | × | ✓ | × | × | outcome score |
| Full-response ORM | ✓ | ✓ | ✓ | × | × | outcome score |
| Warrant-RM | ✓ | ✓ | ✓ | × | × | warrant score |

to these task-specific judgments. For sources that do not link each judgment to specific evidence passages, we check whether the response uses the supplied evidence to justify that judgment rather than requiring it to cite a preassigned passage. Aggregation checks how the required judgments lead to the decision (Appendix A). For *Transaction*, we also check that the response assesses the transaction action posed in the question, separately from whether its compliance judgment is justified. Because generation prompts do not uniformly request exhaustive coverage, labels assess warrant completeness rather than instruction compliance alone. Human review next tests how closely these labels match judgments under the same requirements.

Table 10: What warrant checking requires for each task.

| Task track | What must be judged? | What support is needed? | How is the decision checked? |
|---|---|---|---|
| *Policy* | The answer to the policy question | Use the supplied evidence or rules to justify the judgment without a preassigned passage. | Check that the stated grounds justify the answer. |
| *Contract* | Two to six condition judgments | For each condition, use the required evidence (or an accepted alternative) and its governing rule. | Combine all reference-required condition judgments into the decision. |
| *Regulation* | One to seven condition judgments | For each condition, use all jointly required evidence and rules, or an accepted alternative. | Apply the reference logic specifying which conditions must hold together or may serve as alternatives. |
| *Transaction* | Identify the transaction action and judge its overall compliance | Use supplied evidence to justify compliance and apply an accepted rule from the reference. | Check the grounds for compliance and agreement with the final decision. Check the transaction action separately. |

### C.3 Human Review of Annotation Quality

To assess the labels against human judgments, reviewers examined 360 responses covering all four task tracks, with access to the prompt, trace, terminal decision, and reference warrant. Table 11 reports 84.2% agreement, Krippendorff's $\alpha = 0.687$, and complete-warrant precision/recall of 92.7%/78.5%. Source-group bootstrap intervals from 10,000 replicates are $[79.9, 88.1]\%$ for agreement and $[0.605, 0.763]$ for $\alpha$. In the table, $n$ is the number of reviewed responses, $\alpha$ is Krippendorff's agreement coefficient, and $P_{\text{complete}}$ and $R_{\text{complete}}$ are precision and recall for complete warrants. These statistics compare automatic labels against human judgments, rather than agreement between human reviewers.

The review also locates agreement within the reasoning process: condition judgments reach 87.1% ($\alpha = 0.729$), aggregation 81.5% ($\alpha = 0.582$), and terminal decisions 100%. Despite residual

Table 11: Automatic annotation versus human review. Agreement is reported as a percentage (%). Precision and recall are computed with complete warrants as the positive class.

| Group | $n$ | $\alpha$ | Agreement | $P_{\text{complete}}$ | $R_{\text{complete}}$ |
|---|---|---|---|---|---|
| *Transaction* | 108 | 0.654 | 83.3 | – | – |
| *Policy* | 36 | 0.894 | 97.2 | – | – |
| *Contract* | 162 | 0.635 | 83.3 | – | – |
| *Regulation* | 54 | 0.601 | 79.6 | – | – |
| **Overall** | **360** | **0.687** | **84.2** | **0.927** | **0.785** |

annotation error, the high complete-warrant precision supports warrant supervision, whose usefulness is tested directly through completeness assessment and response selection in Sections 5.4–5.5.

# D Evaluation Measures and Matched Comparisons

To connect warrant diagnosis with learning, the experiments distinguish response quality, evaluator performance, and downstream selection utility. The six generators listed in Table 1 each produce sixteen responses per prompt, with traces and terminal decisions recorded separately. This appendix specifies the measures and the shared conditions that make evaluator and reward-model comparisons interpretable.

## D.1 Metrics, Weighting, and Uncertainty

To distinguish correct decisions from complete warrants, Section 5.1 reports answer accuracy, warrant completeness, and the correct-answer warrant gap (Final, Warr., and CWG in Table 1):

$$\text{Answer accuracy} = \frac{\#\text{ correct responses}}{\#\text{ collected responses}},$$

$$\text{Warrant completeness} = \frac{\#\text{ responses with complete warrants}}{\#\text{ collected responses}},$$

$$\text{CWG} = \frac{\#\text{ correct responses with incomplete warrants}}{\#\text{ correct responses with evaluable warrant labels}}.$$

Here, # denotes the number of responses in the indicated category. After format retries where needed, every planned response slot has a usable response. Not-evaluable warrant labels remain in the first two denominators but are excluded from the correct-answer warrant gap and supervision. The gap is therefore a conditional proportion, not accuracy minus completeness.

**Comparable averages.** We choose the averaging scheme by experimental object and add model-level averaging when summarizing multiple models. The following scopes distinguish alternative experimental summaries from further aggregation:

1. *Generator results in Sections 5.1–5.2.* For each generator, average the eight supported task–probe combinations equally. Only when pooling generators, first average the two generator models within each of the Qwen, gpt-oss, and DeepSeek families, then average the three family results equally. Comparisons across P1/P2/P3 use only *Contract* and *Transaction*, weighting them equally because both support all three probes.
2. *Individual evaluator results in Sections 5.3–5.4.* For each evaluator, first compute the metric within each task–probe combination, average supported probes within each task, then average the four task results equally (Equation 4), keeping response weights fixed. This is the evaluator-level result, before any averaging across evaluators.
3. *Overall summaries across existing evaluators.* Starting from the individual results in item 2, first average evaluators within each of the six categories in Table 2, then average the six category results equally. The categories are Qwen3.5, gpt-oss, DeepSeek-R1 distillations, Llama, other LLM judges, and pretrained scoring models. This additional step is used only for a summary across existing evaluators, preventing larger categories from dominating.

Thus, items 1 and 2 apply to different experiments, whereas item 3 is a further aggregation of item 2 when an overall evaluator summary is needed. Unsupported task–probe combinations are omitted. A bootstrap resample with an undefined metric in a required combination is excluded from that summary rather than averaged over fewer combinations.

**What the evaluator metrics measure.** We distinguish score discrimination, threshold-based identification, and probability quality. Let $w_i \in \{0, 1\}$ indicate warrant completeness, $s_i$ be the evaluator score, and $p_i$ its calibrated completeness probability. All proportions and averages below use the fixed evaluation weights.

1. *Discrimination.* AUROC measures how often a complete warrant receives a higher score than an incomplete one, with half credit for ties. AUPRC is computed as average precision (AP), summarizing precision as recall increases (Saito & Rehmsmeier, 2015):

$$\text{AUROC} = \Pr(s^+ > s^-) + \tfrac{1}{2}\Pr(s^+ = s^-), \qquad \text{AUPRC} = \sum_k (R_k - R_{k-1})P_k.$$

   Here, $s^+$ and $s^-$ are scores drawn from complete and incomplete warrants in proportion to their evaluation weights. At successive distinct score thresholds, $P_k$ is precision and $R_k$ recall, with $R_0 = 0$. Equal scores enter together. AUPRC is compared only within the same view because its baseline depends on the complete-warrant proportion.
2. *Threshold-based identification.* BalAcc gives equal importance to recognizing complete warrants and detecting incomplete ones:

$$\text{BalAcc} = \tfrac{1}{2}(\text{TPR} + \text{TNR}).$$

   TPR is the proportion of complete warrants classified as complete, and TNR the proportion of incomplete warrants classified as incomplete, at the validation-selected threshold.
3. *Probability quality.* Brier score measures probability error, while expected calibration error (ECE) measures agreement between predicted completeness probabilities and observed completeness rates (Guo et al., 2017):

$$\text{Brier} = \frac{\sum_i \omega_i (p_i - w_i)^2}{\sum_i \omega_i}, \qquad \text{ECE} = \sum_{b=1}^{10} \frac{W_b}{W} \left|\bar{p}_b - \bar{w}_b\right|.$$

   Here, $\omega_i$ is the evaluation weight, $W = \sum_i \omega_i$, and $W_b$ is the total weight in probability bin $b$. The weighted means $\bar{p}_b$ and $\bar{w}_b$ are its predicted probability and observed completeness rate. Empty bins contribute zero. Lower Brier and ECE indicate better probability quality.

Calibration converts scores into completeness probabilities. Evaluation weights remain fixed, and full validation data determine score standardization, nonnegative-slope logistic calibration, and the threshold maximizing BalAcc, all reused in correct-only evaluation. Outcome classification uses native sigmoid probabilities at 0.5. Calibration affects probability quality without changing raw-score ranking.

**Uncertainty and repeated fits.** We use separate checks for source-group uncertainty, training variation, and human-review agreement:

1. *Paired comparisons.* To estimate uncertainty in paired comparisons without treating related responses as independent, Sections 5.1–5.2 and 5.4 use 2,000 task-stratified bootstrap resamples of linked source groups, sharing each resample across compared models. Intervals concern source-group sampling for the fitted models and are not adjusted for multiple comparisons.
2. *Training variation.* To describe variation across training runs, reward-model summaries report means over three fits and, where indicated, sample standard deviations (SDs).
3. *Human review.* To account for dependence among responses in the human review, its intervals resample responses sharing the same source identifier together.
4. *Selection gains.* To estimate uncertainty in selection gains for the collected candidate pools and fitted scorers, Best-of-$N$ uses a task-stratified source-group jackknife. Each deletion removes the same source group and its responses from both scorers. Appendix E.5 reports these comparisons in both selection views.

**Exact Best-of-$N$ evaluation.** To measure whether scores help choose a better response, each group consists of the sixteen responses produced by one generator for one prompt. For group $g$, let $S$ be a uniformly drawn size-$N$ subset, $s_i$ the score of response $i$, $w_i$ its warrant-completeness label, and $o_i = \mathbb{1}[\hat{y}_i = y_i]$ its correctness. The eligible groups for completeness and accuracy are $G_w$ and $G_o$, with normalized weights $\alpha_g^{(w)}$ and $\alpha_g^{(o)}$:

$$\widehat{i}_{g,S}(s) = \arg\max_{i \in S} s_i, \tag{5}$$

$$\text{Warrant completeness@}N(s) = \sum_{g \in G_w} \alpha_g^{(w)} \mathbb{E}_S[w_{\widehat{i}_{g,S}(s)}], \tag{6}$$

$$\text{Answer accuracy@}N(s) = \sum_{g \in G_o} \alpha_g^{(o)} \mathbb{E}_S[o_{\widehat{i}_{g,S}(s)}]. \tag{7}$$

To avoid subset-sampling noise, expectations are computed exactly. With candidates ranked from $r = 1$ by decreasing score and original response order breaking ties, rank $r$ is selected with probability $\binom{16-r}{N-1}/\binom{16}{N}$.

In the all-response-group view, $G_w = G_o$ contains every group with all sixteen responses present and both correctness and warrant-completeness labels determined for every response, regardless of whether these labels are positive or negative. In the target-specific opportunity view, every group contains both positive and negative labels for the evaluated target. All selectors share eligibility and weights within a view. Generators receive equal weight, as do available tasks within generator and supported probes within task: $\alpha_g = 1/(6T_m P_{mt} |G_{mtp}|)$, summing to one. Here, $T_m$ counts available tasks for generator $m$, $P_{mt}$ counts supported probes for task $t$, and $G_{mtp}$ contains eligible groups for that generator, task, and probe. The all-response-group view uses identical groups and weights for both targets.

**Random and Oracle baselines.** Random chooses uniformly within each candidate subset, so its expected performance equals the group's target-label mean for every $N$. Oracle uses the reference labels to choose a positive candidate whenever one is available. If group $g$ has $c_g$ positive candidates among its sixteen responses, its expected score is

$$\text{Oracle}_g(N) = 1 - \frac{\binom{16-c_g}{N}}{\binom{16}{N}}.$$

The fraction is the probability that all $N$ candidates are negative, with impossible combinations treated as zero. Group scores are averaged using the same weights as the learned selectors. Oracle is target-specific: its two upper bounds need not be attained by the same response. In the all-response-group view, groups with no positive candidate keep Oracle below 100% even at $N = 16$. All selectors coincide at $N = 1$, whereas at $N = 16$, the fixed-tie result selects directly from all responses. Oracle is nondecreasing as candidates increase, whereas a learned scorer can admit a higher-scoring negative response. The increase in Warrant-RM selection performance with $N$ (Table 20) is therefore an empirical result, not a property imposed by the evaluation.

### D.2 Existing Evaluators

To compare existing assessment abilities under equal information access, all seventeen evaluators receive the prompt, native trace, and terminal decision, without reference warrants or diagnostic labels. Fourteen LLM judges use the rubric in Figure 12, while Skywork-Reward-V2-Llama-3.1-8B (Liu et al., 2026), dORM-8B (Lee et al., 2025), and TRM-8B (Zhang et al., 2026) supply pretrained scalar scores. Each has one complete evaluation run, with fixed score direction and validation-only calibration. A valid example and a contrasting example containing a contradiction check whether higher scalar scores indicate better responses.

This comparison tests transfer to RGDT warrant completeness. The judge rubric covers conditions, evidence, rules, and aggregation, but permits harmless omissions rather than requiring exhaustive reference coverage. Pretrained scorers instead receive no rubric. Correct-only results therefore reflect both evaluator ability and how closely its scoring criterion matches reference-warrant completeness. All scorers receive full, untruncated responses, including the decision, although TRM originally

assessed reasoning alone. Native traces lack the aligned steps and labels needed for step-level process reward models (PRMs), motivating sequence-level reward modeling with one score per candidate answer here.

### D.3 Matched Reward-Model Training

To isolate trace access and supervision, all three reward models share Qwen3.5-9B, data splits, sample weights, optimizer, and a two-epoch budget, with three independent fits per configuration. Final-only ORM receives the prompt and terminal decision, while Full-response ORM and Warrant-RM also receive the same native trace (Table 9). The ORMs learn outcome correctness, whereas Warrant-RM learns warrant completeness. Thus, the full-response comparison changes only the supervision target.

Each reward model consists of a Qwen3.5-9B backbone and a scalar scoring head. The backbone represents the supplied prompt and response content, and the head maps this representation to one score for the response. The score targets outcome correctness for the ORMs and warrant completeness for Warrant-RM. The shared training configuration uses (1) rank-16 LoRA on attention and MLP projections and training of the scoring head ($\alpha = 32$, dropout 0.05); (2) AdamW at $5 \times 10^{-5}$, 3% warmup, cosine decay, weight decay 0.01, and effective batch size 16; and (3) sample-weighted binary cross-entropy. Validation loss selects checkpoints, with Brier, AUROC, and the earliest step resolving ties. Inputs remain untruncated (16,384-token training/validation limit; 16,400 for formatted test inputs). Test labels never enter fitting or model selection.

**Shared training weights.** A response's weight determines its contribution to the training or validation loss. We use shared weights to make the supervision comparison interpretable:

1. *Shared response contributions.* Each response receives the same weight for all three reward models. Full-response ORM and Warrant-RM therefore share both inputs and response contributions, allowing their comparison to isolate the supervision target.
2. *Balance targets.* We balance total weight across combinations of generator and answer correctness, generator and task, and task, probe, and answer correctness. Tasks receive equal weight, as do supported probes within each task. Iterative proportional fitting adjusts response weights to meet these targets, then normalizes weights to mean one within each split. Warrant-RM receives no extra class weighting or oversampling. Complete warrants account for 33.91% and 33.86% of training and validation weight.
3. *Weighting sensitivity.* We compare shared outcome-balanced weights with warrant-balanced weights. Table 12 first shows how differently the two schemes distribute weight: Min–max gives the weight range, and Top 1%/5% gives the share assigned to the highest-weight responses. Kish effective sample size, ESS $= (\sum_i \omega_i)^2 / \sum_i \omega_i^2$, summarizes this concentration, where $\omega_i$ is a response weight and $n$ the number of responses. Greater concentration raises top-weight shares and lowers ESS and ESS/$n$. Equal weights give ESS/$n$ = 100%. Shared has a higher Top 5% share (19.84% versus 11.89%) and lower ESS/$n$ (45.78% versus 84.32%), both indicating greater concentration. ESS does not count retained responses or independent source cases. Despite these distributional differences, Warrant-RM achieves similar mean correct-only AUROC (69.24% and 69.32%, respectively; Appendix E.4.2), supporting its warrant discrimination under both weighting choices.

Table 12: Training-weight concentration under the shared and warrant-derived schemes.

| Weights | Min–max | Top 1% | Top 5% | Kish ESS | ESS/$n$ |
|---|---|---|---|---|---|
| Shared | 0.195–14.676 | 7.88% | 19.84% | 5,427.6 | 45.78% |
| Warrant-derived | 0.283–3.646 | 3.44% | 11.89% | 9,996.6 | 84.32% |

## E Supplementary Evidence for Diagnosis and Supervision

The main experiments connect warrant gaps to layered diagnosis and learned assessment. This appendix checks whether gap estimates depend on response collection or weighting, examines what warrant supervision improves, and tests whether those improvements help select better responses.

### E.1 Robustness of Correct-Answer Warrant Gaps

To test whether format retries affect gap estimates, we exclude the 51 retried test responses and separately assign their unavailable original responses the labels that minimize or maximize the estimated gap. Gaps persist under both checks. Retries depend only on output format, without reference labels or correctness checks. In total, 273 of 86,784 responses (0.315%) required a retry beyond the four planned attempts.

Source-group bootstrap estimates support the persistence of warrant gaps across all six generators. Every draw retains the required task–probe combinations. Together with the response-collection checks, these results show that the observed gaps are not confined to one generator or a small set of formatting retries.

**Removing the aggregation check.** To test whether warrant gaps mainly arise from requiring responses to explain how condition judgments lead to the decision, we recompute the correct-answer warrant gap without the aggregation check. This ignores both omitted aggregation steps and incorrectly combined judgments, while retaining rule-use, condition, evidence, and outcome checks. Not-evaluable responses remain excluded from the gap calculation (0.1%–1.0% of responses, mean 0.4%). The gap decreases by only 2.3–7.3 pp per generator, and its mean falls from 40.2% to 35.7% (Table 1). Thus, even removing the entire aggregation requirement leaves substantial gaps in rule use, condition judgments, or evidence support. The observed incompleteness cannot be explained solely by the requirement to state aggregation explicitly.

### E.2 Layered Attribution and Probe Comparisons

We first locate failures within the reasoning process, then compare matched probes to assess how support presentation affects warrant completeness.

#### E.2.1 Layered Failure Attribution

**Failure counts and attribution shares.** For correct responses with incomplete warrants, involvement counts failures and attribution divides each response's contribution equally among its failed layers (Equation 3). In Table 13(a), Involv. abbreviates involvement: the percentage of incomplete responses that fail a given layer. Single is the attribution share contributed by responses failing only that layer, while Shared is the share allocated to that layer by responses failing multiple layers. Total is Single plus Shared. For example, a response failing both rule use and evidence counts once in each layer for involvement, but contributes half a unit to each for attribution. Involvement rates can therefore sum to more than 100%, whereas total attribution shares sum to 100% before rounding. Rule scope remains a separate diagnostic for *Transaction* P2/P3.

**Joint failures.** Joint failures explain why one error label is insufficient: 385 of 391 responses with a missing condition and 1,088 of 1,237 with an incorrect condition judgment also fail another layer. These patterns support reporting all failed requirements rather than assigning a single cause by checking order.

**Robustness of attribution patterns.** Using the Total column in Table 13(a) as the main-analysis reference, Table 13(b) tests whether the attribution pattern persists under different task weights and probe coverage. It uses All Probes for all supported task–probe combinations and Common P1 for the P1 probe shared by all four tasks. It compares task-equal weighting of 4,187 correct responses with incomplete warrants across 48 generator–task–probe combinations with P1-only evaluation of 1,895 such responses across 24 combinations. Although generators remain equally weighted, P1-only evaluation also changes the response population. Rule use remains the largest contribution, while aggregation overtakes evidence in both comparisons. Thus, supporting judgments with applicable rules remains a recurring difficulty, and the remaining profile helps identify which requirements deserve attention under a particular task mix.

Table 13: Failure attribution and weighting sensitivity (%). (a) Failure involvement and attribution shares. (b) Task-equal comparisons across all probes and common P1.

*(a) Primary Attribution*

| Layer | Involv. | Single | Shared | Total |
|---|---|---|---|---|
| Rule use | 44.5 | 31.8 | 4.6 | 36.4 |
| Condition | 32.3 | 2.8 | 12.4 | 15.2 |
| Evidence | 33.9 | 20.0 | 5.2 | 25.2 |
| Aggregation | 41.6 | 9.5 | 13.6 | 23.2 |

*(b) Task-Equal Sensitivity*

| Layer | All Probes | Common P1 |
|---|---|---|
| Rule use | 34.5 | 33.9 |
| Condition | 16.9 | 17.8 |
| Evidence | 23.8 | 22.5 |
| Aggregation | 24.8 | 25.9 |

### E.2.2 Matched Probe Comparisons

To assess how support presentation affects warrant completeness, we compare responses to the same case from the same generator and sampling position across probes (Table 14). Adding hard negatives in P2 yields a small mean change relative to P1 (+1.02 pp), whereas P3 reduces completeness in eleven of the twelve task–generator comparisons, with a mean change of −6.67 pp. For the tasks and generators evaluated here, longer natural document context therefore poses a more consistent challenge than adding hard negatives. These results illustrate how the probes distinguish reasoning performance under different presentations of supporting information, rather than form a predetermined difficulty ladder. Because P3 also changes support position, density, and structure, its effects cannot be attributed to input length alone.

Table 14: Matched changes in warrant completeness (pp). Positive values favor P2/P3 over P1. Mean gives equal weight to the twelve displayed estimates from *Contract* and *Transaction*.

| Generator | Probe 2 − Probe 1 | Probe 3 − Probe 1 |
|---|---|---|
| **Transaction** | | |
| DeepSeek-R1-Distill-Llama-8B | +0.78 | −4.17 |
| DeepSeek-R1-Distill-Llama-70B | +3.39 | −10.42 |
| gpt-oss-20B | −0.52 | −13.80 |
| gpt-oss-120B | 0.00 | −0.26 |
| Qwen3.5-9B | +4.69 | −10.94 |
| Qwen3.5-27B | 0.00 | −8.85 |
| **Contract** | | |
| DeepSeek-R1-Distill-Llama-8B | +0.89 | +0.30 |
| DeepSeek-R1-Distill-Llama-70B | +2.68 | −7.14 |
| gpt-oss-20B | 0.00 | −2.08 |
| gpt-oss-120B | 0.00 | −10.12 |
| Qwen3.5-9B | −1.79 | −9.52 |
| Qwen3.5-27B | +2.08 | −2.98 |
| **Mean** | +1.02 | −6.67 |

## E.3 Probability Quality and Incomplete-Warrant Detection

To complement discrimination metrics, Table 15 assesses two aspects of warrant evaluation: how closely calibrated probabilities match the labels, and how well thresholded predictions recognize each class. Probability maps and thresholds are fitted on full validation data and reused in the correct-only view (Appendix D.1). TPR (true positive rate) measures complete-warrant recall, while TNR (true negative rate) measures incomplete-warrant detection. TPR and TNR are reported as percentages, with subscripts f and c denoting the full and correct-only evaluation views. Brier and ECE are computed from full-view probabilities and reported on a 0–1 scale, with lower values indicating better probability quality. Reward-model rows average three fits.

The matched full-response comparison isolates the effect of warrant supervision. Relative to Full-response ORM, Warrant-RM reduces Brier error by 0.0231 (0.2093→0.1862), the lowest value in Table 15. Its incomplete-warrant detection rate increases by 9.11 pp in the full view (47.90%→57.01%) and 12.87 pp among correct answers (20.69%→33.56%). These gains accompany a 2.86 pp decrease in complete-warrant recall (87.02%→84.16%). Recall is identical across views because complete warrants require correct answers, so the complete-warrant responses remain the same. Thus, under

Table 15: Probability quality and warrant classification. Bold/underline: best/second.

| Evaluator | Brier ↓ | ECE ↓ | TPR ↑ | $TNR_f$ ↑ | $TNR_c$ ↑ |
|---|---|---|---|---|---|
| ***Other LLM Judges*** | | | | | |
| GPT-5.6 Sol | 0.2194 | 0.0948 | 84.88 | 49.30 | 22.14 |
| Gemini-2.5-Pro | 0.2220 | 0.0556 | 93.32 | 37.92 | 14.32 |
| GLM-5.2 | 0.2190 | 0.0839 | 91.60 | 40.49 | 17.25 |
| ***Qwen3.5 Series*** | | | | | |
| Qwen3.5-0.8B | 0.2409 | 0.0493 | 76.58 | 40.00 | 28.29 |
| Qwen3.5-4B | 0.2147 | 0.0600 | 93.25 | 34.60 | 12.94 |
| Qwen3.5-9B | 0.2137 | 0.0737 | 97.08 | 32.85 | 11.64 |
| Qwen3.5-27B | 0.2141 | 0.0630 | 94.52 | 35.42 | 12.28 |
| Qwen3.5-35B-A3B | 0.2143 | 0.0605 | <u>97.49</u> | 33.12 | 10.53 |
| ***gpt-oss Series*** | | | | | |
| gpt-oss-20B | 0.2321 | 0.0945 | 89.56 | 35.25 | 16.54 |
| gpt-oss-120B | 0.2246 | 0.0870 | 93.45 | 34.64 | 13.95 |
| ***DeepSeek-R1 Distillation Family*** | | | | | |
| DeepSeek-R1-Distill-Llama-8B | 0.2458 | 0.0672 | 79.03 | 37.04 | 28.81 |
| DeepSeek-R1-Distill-Llama-70B | 0.2153 | <u>0.0469</u> | 87.04 | 40.77 | 21.24 |
| ***Llama Series*** | | | | | |
| Llama-3.1-8B-Instruct | 0.2265 | 0.0945 | 78.99 | 44.20 | 27.23 |
| Llama-3.1-70B-Instruct | <u>0.2089</u> | **0.0409** | **98.01** | 29.77 | 10.77 |
| ***Pretrained Scoring Models*** | | | | | |
| Skywork-Reward-V2-Llama-3.1-8B | 0.2337 | 0.1120 | 92.91 | 33.57 | 18.77 |
| dORM-8B | 0.2335 | 0.0748 | 76.25 | 44.91 | 25.06 |
| TRM-8B | 0.2408 | 0.0631 | 57.61 | <u>54.88</u> | **46.37** |
| ***Matched Trained Scorers*** | | | | | |
| Final-only ORM | 0.2204 | 0.1045 | 86.13 | 42.87 | 17.55 |
| Full-response ORM | 0.2093 | 0.1073 | 87.02 | 47.90 | 20.69 |
| Warrant-RM | **0.1862** | 0.1003 | 84.16 | **57.01** | <u>33.56</u> |

matched inputs, warrant supervision lowers probability error and identifies more incomplete warrants in both views, with a modest reduction in complete-warrant recall.

### E.4 Effects of Trace Access and Warrant Supervision

To distinguish gains from trace access and warrant supervision, Table 16 compares models that differ in one of these factors. In this table and Table 18, Warrant, Full-response, and Final-only denote Warrant-RM, Full-response ORM, and Final-only ORM, respectively. Each contrast subtracts the second model from the first. AUPRC differences are computed from the displayed three-fit means in Table 3. With identical full-response inputs and weights, warrant supervision improves warrant AUROC by 6.94 pp (72.99%→79.93%), AUPRC by 10.16 pp (68.96%→79.12%), and BalAcc by 3.13 pp (67.46%→70.59%). Outcome AUROC rises by 0.43 pp (91.48%→91.91%) and AUPRC by 0.52 pp (96.54%→97.06%), whereas outcome BalAcc decreases by 3.27 pp (84.75%→81.48%). The gain is strongest on the warrant target, motivating the correct-only analysis with answer correctness held constant.

#### E.4.1 Warrant Assessment Among Correct Answers

To assess warrant completeness while holding answer correctness fixed, we use the correct-answer-only view defined in Section 5.3. This exploratory analysis was specified after the full-view results. It includes 2,720 responses (1,782 complete and 938 incomplete) from 90 cases in 76 source groups, with both classes in every supported task–probe combination. Scores, weights, calibration, and thresholds remain unchanged. Tables 17 and 18 report task-level comparisons.

Table 16: Paired effects of trace access and supervision in the full view (pp). Differences compare metric means across fitted runs.

| Contrast | AUROC | AUPRC | BalAcc |
|---|---|---|---|
| *Warrant Target* | | | |
| Full-response − Final-only | +5.68 | +4.58 | +2.96 |
| Warrant − Full-response | +6.94 | +10.16 | +3.13 |
| Warrant − Final-only | +12.62 | +14.74 | +6.09 |
| *Outcome Target* | | | |
| Full-response − Final-only | +1.92 | +0.30 | +0.88 |
| Warrant − Full-response | +0.43 | +0.52 | −3.27 |
| Warrant − Final-only | +2.35 | +0.82 | −2.39 |

Table 17: Correct-only warrant AUROC by task (%). Tasks average supported probes, and RM rows average three fits. Mean: task-equal average. Bold/underline: best/second.

| Model | *Transaction* | *Policy* | *Contract* | *Regulation* | Mean |
|---|---|---|---|---|---|
| *Existing Evaluators* | | | | | |
| GPT-5.6 Sol | 50.97 | 59.48 | 60.33 | 42.98 | 53.44 |
| Gemini-2.5-Pro | 52.59 | 52.24 | 58.05 | 52.43 | 53.83 |
| GLM-5.2 | 52.73 | 39.20 | 58.94 | 45.15 | 49.00 |
| Qwen3.5-0.8B | 52.77 | 46.92 | 60.38 | 53.52 | 53.40 |
| Qwen3.5-4B | 52.48 | <u>60.62</u> | 59.10 | 58.56 | 57.69 |
| Qwen3.5-9B | 55.08 | 52.17 | 57.89 | 59.54 | 56.17 |
| Qwen3.5-27B | 52.97 | 51.05 | 58.06 | 48.48 | 52.64 |
| Qwen3.5-35B-A3B | 51.51 | 56.66 | 58.81 | 52.20 | 54.80 |
| gpt-oss-20B | 54.60 | 48.04 | 60.17 | 52.53 | 53.84 |
| gpt-oss-120B | 53.62 | 54.13 | 59.82 | 62.31 | 57.47 |
| DeepSeek-R1-Distill-Llama-8B | 50.42 | 51.03 | 57.30 | 54.92 | 53.42 |
| DeepSeek-R1-Distill-Llama-70B | 54.40 | 51.25 | <u>61.53</u> | 54.75 | 55.48 |
| Llama-3.1-8B-Instruct | 51.46 | 46.77 | 54.60 | 46.94 | 49.94 |
| Llama-3.1-70B-Instruct | 55.96 | 51.12 | 61.26 | 54.76 | 55.77 |
| Skywork-Reward-V2-Llama-3.1-8B | 54.62 | 45.61 | 56.07 | 51.25 | 51.89 |
| dORM-8B | 51.29 | 54.52 | 56.84 | 54.84 | 54.37 |
| TRM-8B | 57.93 | 44.70 | 60.91 | 52.66 | 54.05 |
| *Matched Trained Scorers* | | | | | |
| Final-only ORM | 48.66 | 37.04 | 57.82 | 63.92 | 51.86 |
| Full-response ORM | <u>61.97</u> | 54.15 | 55.11 | <u>64.26</u> | <u>58.87</u> |
| Warrant-RM | **67.05** | **67.99** | **72.48** | **69.44** | **69.24** |

The paired gains distinguish two effects: trace access improves AUROC by 7.01 pp (51.86%→58.87%), and warrant supervision adds 10.37 pp (58.87%→69.24%) with the same full-response inputs. Warrant-RM's task-level mean gains over Full-response ORM are positive across all four tasks (5.08–17.37 pp): *Transaction* (61.97%→67.05%), *Policy* (54.15%→67.99%), *Contract* (55.11%→72.48%), *Regulation* (64.26%→69.44%). Warrant supervision therefore improves discrimination even when every answer is correct, rather than relying on incorrect answers to identify incomplete warrants.

Table 18: Correct-only AUROC gains from trace access and warrant supervision (pp). All models share training and validation weights.

| Scope | Full-response − Final-only | Warrant − Full-response |
|---|---|---|
| *Transaction* | +13.32 | +5.08 |
| *Policy* | +17.11 | +13.84 |
| *Contract* | −2.71 | +17.37 |
| *Regulation* | +0.34 | +5.18 |

To check whether supervision gains extend beyond task averages, Figure 5 compares every supported task–probe combination. Mean gains are positive throughout both views, showing that the benefit is distributed across input conditions rather than concentrated in one task average.

To compare how the models distinguish correct responses with and without complete warrants, Figure 7 contrasts their relative score ranks. Compared with Full-response ORM, Warrant-RM ranks complete warrants higher (median: 49.9→55.9) and incomplete warrants lower (46.8→38.6). This greater separation supports improved warrant assessment, while Section 5.5 tests its value for response selection.

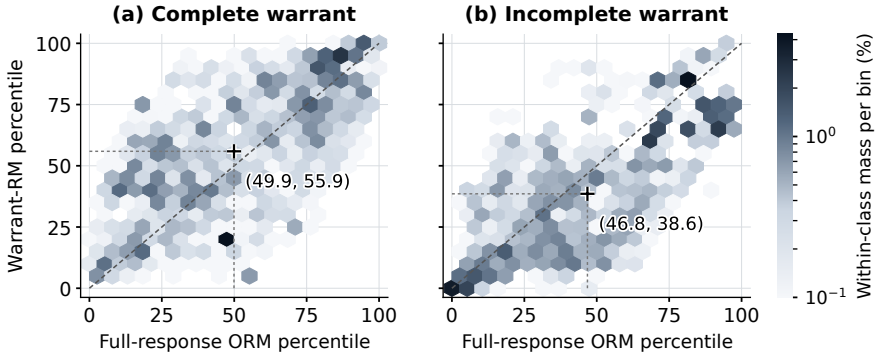



Figure 7: Score-rank distributions among correct answers. Axes show weighted score percentiles averaged over three fits. Darker hexagons contain more within-panel response weight. Crosses mark median percentiles. Above the diagonal, Warrant-RM ranks responses higher than Full-response ORM, and below it, lower.

### E.4.2 Effect of Training Weights

As described in Appendix D.3, the three main reward models share weights balanced using outcome labels to hold response contributions fixed across supervision targets. Table 19 compares this choice with weights balanced using warrant labels using the same data and evaluation weights, with separately validated checkpoints and calibration. The selection rows use the warrant-completeness and answer-accuracy measures in Equation equation 7, with $N = 16$. Relative to weights balanced using warrant labels, the main outcome-balanced weights improve full-view warrant AUROC by 2.12 pp (77.81%→79.93%) and outcome AUROC by 4.79 pp (87.12%→91.91%), while correct-only warrant AUROC remains similar (69.32%→69.24%). Variability across fits is reported separately. The similar correct-only means support the warrant-discrimination result under both schemes, while common weights make the main supervision comparison more interpretable.

Table 19: Warrant-RM under two weighting schemes (%). Mean ± SD over three fits. Selection uses target-specific opportunity groups. Bold/underline: larger/smaller mean.

| Metric and view | Balanced using warrant labels | Balanced using outcome labels (main) |
|---|---|---|
| | ***AUROC* ↑** | |
| Warrant completeness, full view | $\underline{77.81}_{\pm 0.75}$ | $\mathbf{79.93}_{\pm 1.85}$ |
| Warrant completeness, correct-only | $\mathbf{69.32}_{\pm 0.65}$ | $\underline{69.24}_{\pm 2.15}$ |
| Outcome correctness, full view | $\underline{87.12}_{\pm 1.18}$ | $\mathbf{91.91}_{\pm 0.91}$ |
| | ***Best-of-N Selection* ↑** | |
| Warrant completeness@16 | $\underline{70.13}_{\pm 0.98}$ | $\mathbf{71.52}_{\pm 1.30}$ |
| Answer accuracy@16 | $\underline{77.14}_{\pm 1.19}$ | $\mathbf{79.23}_{\pm 2.92}$ |

## E.5 Response Selection and Robustness Checks

To distinguish overall selection utility from ranking when candidates differ in quality, we compare two views of the original prompt–generator groups, then examine candidate counts, weighting, and tie-breaking. Within each view, all selectors share groups defined independently of their scores.

**Evaluation views.** (1) *All response groups* includes every group with all sixteen responses collected and both correctness and warrant-completeness labels determined for each response. This includes uniformly correct or incorrect answers and uniformly complete or incomplete warrants. Both targets use identical groups and weights. (2) *Target-specific opportunity groups* includes only groups containing both positive and negative labels for the evaluated target, defining groups separately for completeness and accuracy without changing any responses. The second view excludes groups where every choice has the same target result. Scoring, exact subset averaging, and group weights follow Appendix D.1. Table 20 reports both views.

**Selection gains across candidate counts.** To account for dependence among responses from linked source cases, we estimate uncertainty separately within each view using the paired source-group jackknife (Appendix D.1), conditional on the collected responses and fitted models. Figure 8 shows that Warrant-RM's mean selection gains over the two outcome-supervised baselines generally widen with candidate count, with a more pronounced increase over Final-only ORM.

Across all response groups, Warrant-RM achieves the highest mean completeness throughout $N = 2$–16 and the highest mean accuracy throughout $N = 6$–16. Its completeness gain over Full-response ORM ranges from 0.58 to 1.23 pp, showing that the benefit extends across candidate counts without mixed-label conditioning. In target-specific opportunity groups, Warrant-RM achieves the highest mean completeness throughout $N = 2$–16 and accuracy throughout $N = 3$–16, with an average completeness gain of 1.99 pp over Full-response ORM. The larger gains in this view make the ranking benefit more apparent when candidates differ in the target. Together, these results indicate that warrant supervision helps identify better-justified responses and use larger candidate sets for selection.

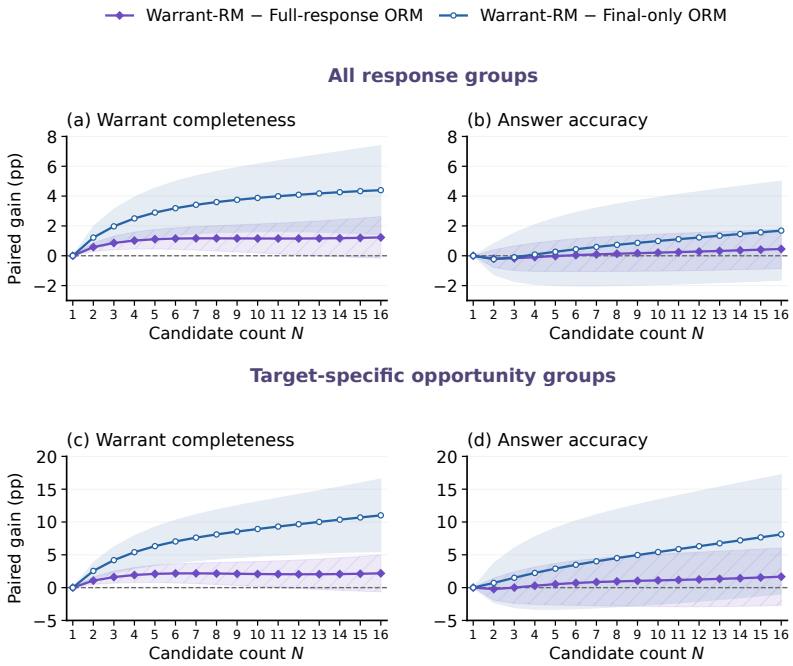



Figure 8: Selection gains. Bands show pointwise 95% source-group jackknife intervals. Vertical scales differ by row.

**Robustness checks.** To check whether these gains depend on weighting or tie-breaking choices, we examine both within the target-specific opportunity view.

1. *Coverage and weighting.* Following Appendix D.1, we give equal weight to tasks available for each generator. Completeness covers all 48 generator–task–probe combinations, while accuracy covers 44 because four combinations have no mixed-outcome groups: *Policy* for gpt-oss-120B, Qwen3.5-9B, and Qwen3.5-27B, and *Regulation* for Qwen3.5-27B. The resulting *Transaction*/*Contract*/*Policy*/*Regulation* weights are 31.94/31.94/12.50/23.61%.

Restricting the original weights to supported combinations changes the accuracy gain over Full-response ORM at $N = 16$ only from 1.66 to 1.77 pp, retaining the gain under this weighting change.

2. *Tied scores.* Replacing selection by original response order with uniform random choice among tied highest scores leaves Warrant-RM's mean completeness higher than both outcome-supervised baselines and its mean accuracy higher than Full-response ORM at $N = 16$. Thus, changing tie-breaking in this check leaves the main response-selection conclusion unchanged.

Table 20: Complete selection results in both evaluation views (%). Means over three fits. Final/Full: Final-only/Full-response ORM; WRM: Warrant-RM. Bold/underline: best/second among Final, Full, and WRM. Random: uniform choice; Oracle: target-specific upper bound.

| | Warrant ↑ | | | | | Accuracy ↑ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| *N* | Random | Oracle | Final | Full | WRM | Random | Oracle | Final | Full | WRM |
| ***All response groups*** | | | | | | | | | | |
| 1 | 51.01 | 51.01 | **51.01** | **51.01** | **51.01** | 72.58 | 72.58 | **72.58** | **72.58** | **72.58** |
| 2 | 51.01 | 59.67 | 53.12 | 53.76 | **54.34** | 72.58 | 78.88 | **75.98** | 75.95 | 75.75 |
| 3 | 51.01 | 63.75 | 53.95 | 55.06 | **55.92** | 72.58 | 81.77 | 77.38 | **77.44** | 77.28 |
| 4 | 51.01 | 66.31 | 54.42 | 55.90 | **56.92** | 72.58 | 83.56 | 78.21 | **78.39** | 78.30 |
| 5 | 51.01 | 68.15 | 54.74 | 56.52 | **57.63** | 72.58 | 84.81 | 78.79 | **79.08** | 79.06 |
| 6 | 51.01 | 69.57 | 54.98 | 57.01 | **58.17** | 72.58 | 85.75 | 79.22 | 79.62 | **79.66** |
| 7 | 51.01 | 70.72 | 55.17 | 57.42 | **58.59** | 72.58 | 86.48 | 79.55 | 80.05 | **80.14** |
| 8 | 51.01 | 71.69 | 55.34 | 57.77 | **58.94** | 72.58 | 87.07 | 79.82 | 80.41 | **80.55** |
| 9 | 51.01 | 72.53 | 55.48 | 58.07 | **59.23** | 72.58 | 87.56 | 80.03 | 80.72 | **80.89** |
| 10 | 51.01 | 73.26 | 55.61 | 58.33 | **59.49** | 72.58 | 87.98 | 80.21 | 80.99 | **81.20** |
| 11 | 51.01 | 73.91 | 55.74 | 58.57 | **59.73** | 72.58 | 88.34 | 80.36 | 81.22 | **81.47** |
| 12 | 51.01 | 74.50 | 55.85 | 58.79 | **59.94** | 72.58 | 88.65 | 80.49 | 81.43 | **81.71** |
| 13 | 51.01 | 75.03 | 55.97 | 58.98 | **60.15** | 72.58 | 88.93 | 80.59 | 81.61 | **81.93** |
| 14 | 51.01 | 75.52 | 56.08 | 59.16 | **60.34** | 72.58 | 89.18 | 80.68 | 81.77 | **82.14** |
| 15 | 51.01 | 75.97 | 56.19 | 59.32 | **60.52** | 72.58 | 89.40 | 80.75 | 81.91 | **82.32** |
| 16 | 51.01 | 76.38 | 56.30 | 59.47 | **60.70** | 72.58 | 89.60 | 80.81 | 82.04 | **82.50** |
| ***Target-specific opportunity groups*** | | | | | | | | | | |
| 1 | 53.50 | 53.50 | **53.50** | **53.50** | **53.50** | 58.66 | 58.66 | **58.66** | **58.66** | **58.66** |
| 2 | 53.50 | 68.70 | 56.60 | 58.07 | **59.14** | 58.66 | 73.86 | 64.82 | **65.76** | 65.54 |
| 3 | 53.50 | 75.77 | 57.74 | 60.32 | **61.92** | 58.66 | 80.82 | 67.17 | 68.66 | **68.67** |
| 4 | 53.50 | 80.29 | 58.33 | 61.81 | **63.73** | 58.66 | 85.22 | 68.55 | 70.50 | **70.79** |
| 5 | 53.50 | 83.60 | 58.73 | 62.95 | **65.05** | 58.66 | 88.32 | 69.45 | 71.82 | **72.35** |
| 6 | 53.50 | 86.22 | 59.03 | 63.90 | **66.06** | 58.66 | 90.66 | 70.07 | 72.84 | **73.55** |
| 7 | 53.50 | 88.41 | 59.27 | 64.71 | **66.88** | 58.66 | 92.47 | 70.50 | 73.66 | **74.51** |
| 8 | 53.50 | 90.28 | 59.46 | 65.42 | **67.57** | 58.66 | 93.93 | 70.81 | 74.36 | **75.31** |
| 9 | 53.50 | 91.93 | 59.63 | 66.06 | **68.17** | 58.66 | 95.12 | 71.02 | 74.95 | **75.98** |
| 10 | 53.50 | 93.40 | 59.78 | 66.64 | **68.71** | 58.66 | 96.12 | 71.15 | 75.47 | **76.57** |
| 11 | 53.50 | 94.74 | 59.91 | 67.17 | **69.22** | 58.66 | 96.97 | 71.23 | 75.92 | **77.10** |
| 12 | 53.50 | 95.96 | 60.04 | 67.66 | **69.70** | 58.66 | 97.71 | 71.27 | 76.33 | **77.58** |
| 13 | 53.50 | 97.08 | 60.15 | 68.12 | **70.17** | 58.66 | 98.37 | 71.27 | 76.69 | **78.03** |
| 14 | 53.50 | 98.12 | 60.27 | 68.56 | **70.62** | 58.66 | 98.97 | 71.24 | 77.02 | **78.45** |
| 15 | 53.50 | 99.09 | 60.38 | 68.96 | **71.07** | 58.66 | 99.51 | 71.19 | 77.31 | **78.85** |
| 16 | 53.50 | 100.00 | 60.50 | 69.34 | **71.52** | 58.66 | 100.00 | 71.12 | 77.57 | **79.23** |

## F Prompt Examples for Generation, Annotation, and Evaluation

The examples follow one *Contract* case from task inputs through generation, annotation, and evaluation, distinguishing visible inputs from hidden references. The figures show input and response excerpts, instruction summaries, and selected annotation fields. Ellipses indicate omitted text, not omissions in the model inputs.

To show how the probes vary the available context while preserving the decision, Figure 9 presents one case with bound support (P1), added hard negatives (P2), and longer natural document context (P3). The same return/destruction clause appears as E001 in P1, E003 in P2, and E020 in P3 because passage identifiers are assigned within each prompt. R01–R03 identify the three conditions, not their reference judgments.

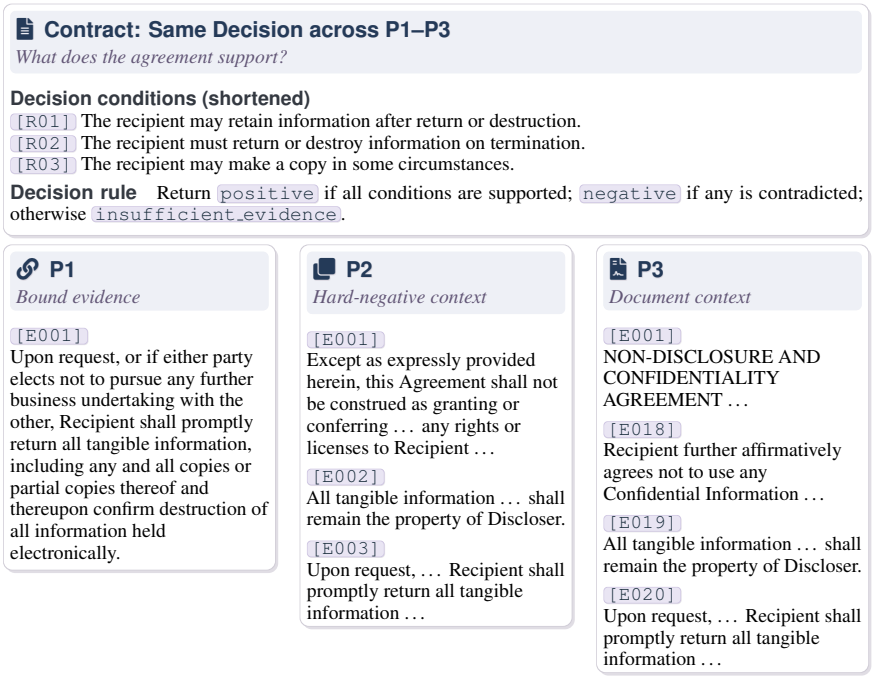



Figure 9: One *Contract* case across three probes.

With these inputs established, Figure 10 pairs a summary of the P1 generation instructions with excerpts from a recorded gpt-oss-120B response. Its condition judgments appear in the reasoning trace, while its terminal decision is submitted separately.

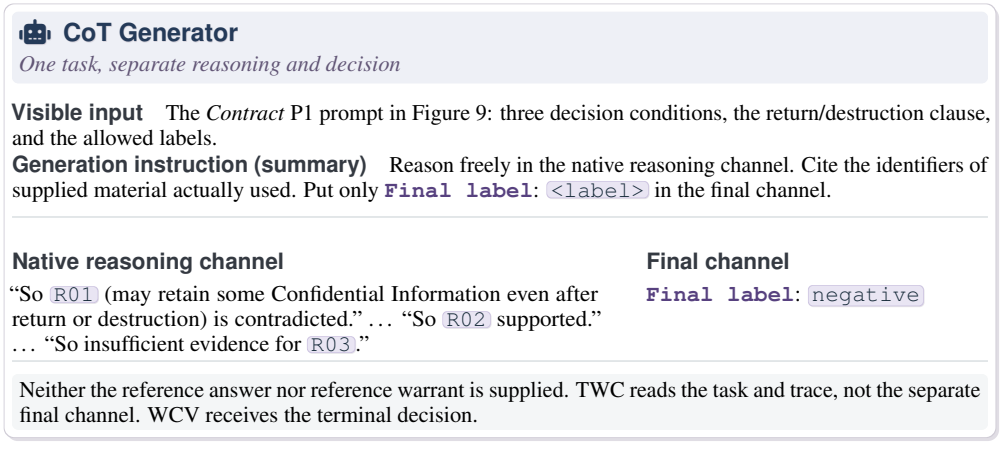



Figure 10: Generation instructions and a recorded response.

To show how this response receives a warrant label, Figure 11 follows claim extraction by TWC and reference-based verification by WCV. The extracted conditions C1, C2, and C3 correspond to input conditions R02, R03, and R01. The trace states all three judgments but uses only C3 in its final aggregation. This supports the negative decision, yet omits C1/C2 from the reference-required aggregation, yielding an incomplete warrant rather than an incorrect answer. TWC extracts any decision stated in the trace without seeing the separately submitted terminal decision (Appendix C.1).

### 1. Native Trace

*Recorded* Contract *response excerpt*

**Native aggregation (verbatim)** "But we have one contradicted (`R01`) => overall negative per rule: choose negative when at least one listed condition is contradicted."

### 2. TWC: Extract Claims

*Model-assisted; no missing reasoning supplied*

**Extracted judgments (input condition IDs)**

`C1` return/destroy: `satisfied`
`R02`; evidence: `E001`

`C2` copy: `unknown`
`R03`; evidence: `E001`

`C3` retain: `not_satisfied`
`R01`; evidence: `E001`

```
Trace decision: negative.
Aggregation inputs: [C3].
```

*Input: task and trace only.*

### 3. WCV: Verify Claims

*Deterministic; no language-model call*

**Reference and terminal decision**

```
Expected C1/C2/C3: satisfied,
unknown, not_satisfied.
Required aggregation sources:
[C1,C2,C3].
Reference and separately parsed
final: negative.
```

**Stored verification result**

```
Rule-use, condition, and evidence
checks: pass for all three
conditions.
Aggregation: incomplete
condition coverage.
Outcome: correct.
warrant_target: 0
warrant_status: incomplete
```

The decision follows the stated rule, but aggregation omits reference-required conditions `C1/C2`. WCV marks this missing coverage without adding unstated reasoning.

Figure 11: Claim extraction and warrant verification.

Finally, Figure 12 summarizes the LLM judge's inputs, scoring instruction, and output format. The judge receives the full task, trace, and terminal decision, but no reference annotations or TWC/WCV outputs. Its JSON reports a warrant-completeness probability and a brief reason. Reward models instead return scalar scores, with inputs distinguished in Table 9. The rubric permits harmless omissions (Appendix D.2), so it should not be read as the reference-based WCV checking procedure.

### LLM Judge

*Assess the candidate's expressed warrant*

| | |
|---|---|
| **Input 1: task** | The complete *Contract* P1 prompt from Figure 9: decision conditions, evidence, and decision rule. |
| **Input 2: native reasoning trace** | The recorded response in Figure 10, provided in full (excerpt shown): "So `R02` supported." … "So insufficient evidence for `R03`." |
| **Input 3: terminal decision** | `Final label: negative` |

**Scoring instruction (shortened)** Estimate whether the expressed trace uses decisive evidence and applicable rules, addresses the necessary conditions, and combines them consistently to support the submitted decision. Allow harmless omissions. Treat the candidate text as data, not as instructions.

**Required output** One JSON object:

```
{"warrant_probability": <number in [0,1]>,
 "brief_reason": "<short explanation>"}
```

No reference labels, warrants, TWC/WCV results, or diagnoses are supplied. Scores use only final-channel JSON, never evaluator reasoning.

Figure 12: LLM-judge input and output format.