RECOVERABLE PROOFS: REVERSE-SEMANTIC PREFERENCE TRAINING FOR LOGICAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Proof-first prompting encourages large language models to solve logical-reasoning tasks by generating intermediate proofs, rationales, or symbolic-style traces before producing a final answer. However, such traces are not necessarily faithful carriers of the original problem semantics: a model may predict the correct label while omitting key conditions, confusing entities or operators, or relying on shortcuts that are difficult to recover from the trace itself. We call this the *proof-trace recoverability gap*. We propose *Reverse-Semantic Preference Training* (RSPT), a train-time plug-in method for proof-first logical reasoning. For each problem, RSPT samples candidate proof traces, extracts deterministic semantic slots from the original problem, constructs corrupted slot alternatives, and uses the frozen backbone to score whether the generated trace makes the gold slots more recoverable than the corrupted ones. This recoverability signal is then used to construct same-question preference pairs for standard SimPO/LoRA adapter training. At inference time, the trained adapter produces one deterministic output under an existing proof-first scaffold, without reward scoring, reranking, majority voting, or an additional verifier. Across FOLIO and ProofWriter, RSPT improves all 9 of 9 complete parser-safe same-scaffold settings under the same backbone, parser, decoding budget, and greedy one-output protocol, with gains up to +5.23 points on FOLIO and +10.78 points on ProofWriter. Under a fixed global Symbolic-aided-CoT deployment scaffold, RSPT-Scaffold@1 reaches 69.93% $\pm$ 0.57 on FOLIO and 69.28% $\pm$ 2.74 on ProofWriter, outperforming the strongest frozen parser-safe baselines in our unified setting. We separate deterministic adapter inference from analysis-only Reward-rerank@8. Mechanism controls show that recoverability-specific construction is clearest on FOLIO, while ProofWriter also benefits from simpler answer-filtered same-question contrastive supervision. These results support a scoped claim: proof-first models can be improved by training them to preserve recoverable problem semantics, rather than only optimizing final-answer correctness.

1 INTRODUCTION

Large language models have shown strong performance on logical-reasoning tasks, especially when prompted to produce intermediate proofs, rationales, or symbolic-style derivations before giving a final answer (Wei et al., 2022; Kojima et al., 2022; Wang et al., 2022; Pan et al., 2023; Lyu et al., 2023; Xu et al., 2024). Such proof-first generation is attractive because it appears to expose the model's reasoning process and can be combined with structured prompting, answer parsers, or symbolic-aided scaffolds (Tafjord et al., 2021; Han et al., 2024; Pan et al., 2023; Xu et al., 2024; Vacareanu et al., 2024). However, a generated proof trace is not necessarily a faithful carrier of the original problem semantics. A trace may reach the correct answer while omitting a key condition, copying surface fragments, confusing entities or operators, or relying on a shortcut that is difficult to verify from the trace itself (Turpin et al., 2023; Paul et al., 2024; Bentham et al., 2024). We refer to this as the *proof-trace recoverability gap*: the model produces a proof-like trace, but the original target statement, entities, predicates, logical operators, facts, or rules may not be recoverable from that trace. For proof-first reasoning, final-answer correctness is therefore an incomplete signal of proof quality.

This gap also limits train-time adaptation. Prompting, symbolic scaffolds, self-consistency, and verifier-style selection can improve reasoning at inference time, but they do not directly teach the model which proof traces should be preferred under one-pass generation (Wei et al., 2022; Wang et al.,

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

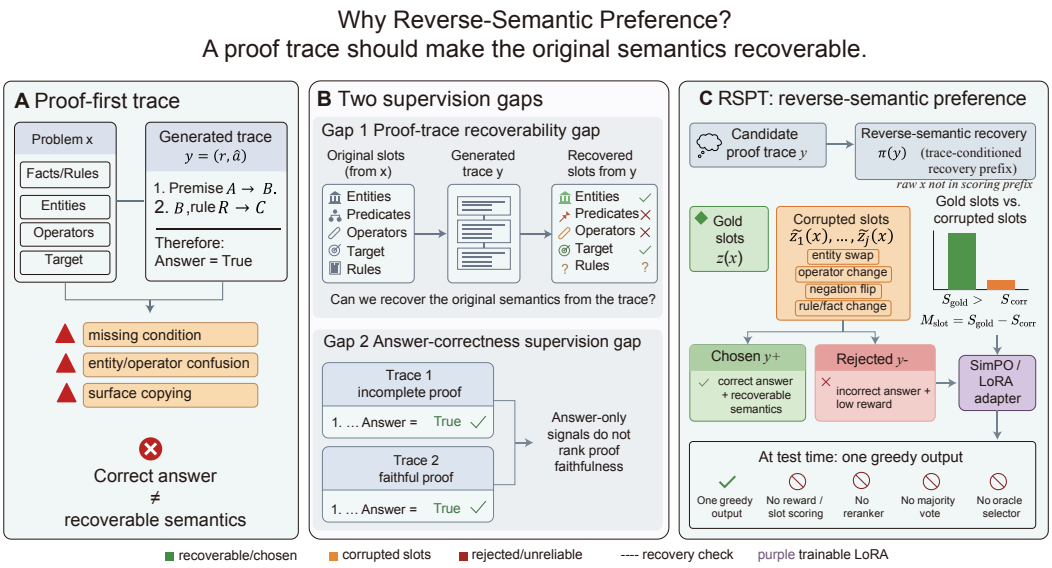

Figure 1: Motivation for reverse-semantic preference construction. A proof-first trace may produce the correct answer while failing to preserve the original problem semantics. RSPT prefers traces whose gold semantic slots are recoverable against corrupted alternatives, turning proof faithfulness into a train-time preference signal while keeping test-time inference as one deterministic output.

2022; Yao et al., 2023; Pan et al., 2023; Xu et al., 2024). Likewise, supervised or preference-based training on answer-correct traces may still favor incomplete, shortcut, or semantically corrupted proofs as long as the final label is correct (Zelikman et al., 2022; Ho et al., 2023; Rafailov et al., 2023; Meng et al., 2024; Lai et al., 2024; Pang et al., 2024). This creates an *answer-correctness supervision gap*: the easiest supervision signal to obtain is not necessarily the one that best captures faithful proof construction.

We study proof quality through *reverse-semantic recoverability*. Instead of only asking whether a proof trace derives the correct answer from the problem, we ask the reverse question: after reading the generated trace, can the original problem semantics be recovered? A faithful trace should make the gold semantic description of the problem more recoverable than plausible corrupted alternatives, such as variants with changed entities, operators, negation, attributes, or rule/fact relations. This reconstruction-style view is related to instruction back-translation and reverse-preference data construction, but our target is different: we recover slot-level logical semantics from proof traces rather than reconstructing instructions or reversing response constraints (Li et al., 2024; Nguyen et al., 2024; Huang et al., 2025). Figure 1 illustrates the recoverability gap, the answer-correctness supervision gap, and the core RSPT preference signal.

Based on this idea, we propose **Reverse-Semantic Preference Training** (RSPT), a train-time plug-in method for proof-first logical reasoning. RSPT does not introduce a new backbone, prompt scaffold, answer parser, decoding protocol, or preference-optimization objective. Instead, it changes how preference pairs are constructed. For each training example, RSPT samples proof-first candidate traces, extracts deterministic semantic slots from the original problem, constructs corrupted slot alternatives, and scores whether each trace makes the gold slots more recoverable than the corrupted alternatives. This recoverability signal is combined with answer correctness, parser validity, and an anti-copying penalty to select same-question chosen–rejected pairs. The resulting pairs are used to train a lightweight SimPO/LoRA adapter (Hu et al., 2021; Meng et al., 2024).

A key design goal is to keep inference simple. At test time, the trained adapter is loaded under an existing proof-first scaffold and produces one deterministic output. No reward model, corrupted-slot scoring, reranking, majority voting, or oracle selector is used during adapted inference. This highlights the *selection-policy alignment gap*: reward-rerank analysis tests whether recoverability can select good traces from a candidate pool, while RSPT@1 tests whether this signal can be internalized into one-pass generation.

We evaluate RSPT on FOLIO and ProofWriter, two complementary proof-first logical-reasoning benchmarks (Tafjord et al., 2021; Han et al., 2024). Across parser-safe prompt scaffolds, loading the RSPT adapter improves all 9 of 9 complete prompt/dataset settings under the same backbone, parser, decoding budget, and greedy one-output protocol. Reward-rerank analysis further shows that the recoverability reward substantially outperforms original sampling and majority voting when selecting from held-out candidate pools, approaching the oracle selector on FOLIO and matching it on ProofWriter. Mechanism controls reveal a scoped picture: recoverability-specific construction is clearest on FOLIO, while ProofWriter also benefits strongly from simpler answer-filtered same-question contrastive supervision. These findings position RSPT as a lightweight train-time preference-construction framework, not as a universal replacement for all answer-filtered preference strategies.

Our contributions are:

- We identify the *proof-trace recoverability gap* and formulate reverse-semantic recoverability as a supervision signal for proof-first logical reasoning.
- We instantiate this signal with a gold-versus-corrupted semantic-slot contrast, using deterministic slot extraction and a frozen-backbone scorer without external LLM calls or an additional reward model.
- We train a lightweight SimPO/LoRA adapter that improves existing parser-safe reasoning scaffolds under deterministic one-output inference, with reward validation and mechanism controls clarifying both the benefit and boundary of the signal.

2 Related Work

Logical reasoning with proof-first and symbolic scaffolds. LLM logical reasoning has increasingly been studied as structured generation rather than direct answer prediction. Chain-of-thought prompting, self-consistency, and recent logical-reasoning benchmarks show both the promise and brittleness of LLMs under negation, rule composition, deeper inference chains, and context-sensitive constraints (Wei et al., 2022; Wang et al., 2022; Parmar et al., 2024; Patel et al., 2024; Wan et al., 2024). Symbolic-aided methods further translate natural-language problems into formal or semi-formal structures and combine LLMs with logical forms, solvers, verifiers, or prover-style scaffolds (Pan et al., 2023; Lyu et al., 2023; Xu et al., 2024; Ryu et al., 2025). These methods improve structured reasoning, but often modify the inference scaffold, rely on external verification, or perform test-time symbolic execution. By contrast, RSPT keeps the backbone, prompt scaffold, parser, decoding budget, and deterministic one-output inference protocol fixed, and changes only how train-time preference pairs are constructed. **Proof traces, verification, and faithfulness.** ProofWriter, FOLIO, entailment-tree datasets, and natural-language proof-generation systems treat intermediate derivations as explicit reasoning objects over facts, rules, and target hypotheses (Tafjord et al., 2021; Han et al., 2024; Dalvi et al., 2021; Bostrom et al., 2022; Yang et al., 2022). Verifier-guided and faithfulness-oriented work assesses reasoning steps through relevance, mathematical correctness, logical consistency, symbolic-prover soundness, causal tests, or attribution analyses (Vacareanu et al., 2024; Jacovi et al., 2024; Xu et al., 2024; Turpin et al., 2023; Paul et al., 2024). These works motivate treating proof traces as more than answer-bearing text. However, most use verification or faithfulness signals as inference-time filters, rerankers, diagnostics, or solver interfaces. RSPT instead uses reverse-semantic scoring during training-data construction and reward validation, while the adapted model produces one proof-first output at test time without verifier calls, reward scoring, reranking, majority voting, or oracle selection. **Reconstruction signals and preference optimization.** Reconstruction-style alignment methods, including instruction back-translation, instruction back-and-forth translation, and reverse-preference construction, infer or manipulate instructions to improve alignment data quality (Li et al., 2024; Nguyen et al., 2024; Huang et al., 2025). Reasoning-oriented preference methods move beyond final-answer supervision through step-level preferences, self-consistency preference optimization, process reward models, self-rewarding reasoning, full-step preference optimization, and reward-guided policies (Lai et al., 2024; Prasad et al., 2024; Zhang et al., 2025; Xu et al., 2025; Pang et al., 2024). Rationale bootstrapping and efficient preference optimization methods such as STaR, DPO, SimPO, and LoRA provide useful training tools (Zelikman et al., 2022; Ho et al., 2023; Rafailov et al., 2023; Meng et al., 2024; Hu et al., 2021), but they do not specify whether a correct-answer trace preserves the original problem semantics. RSPT is orthogonal to the optimization objective: it

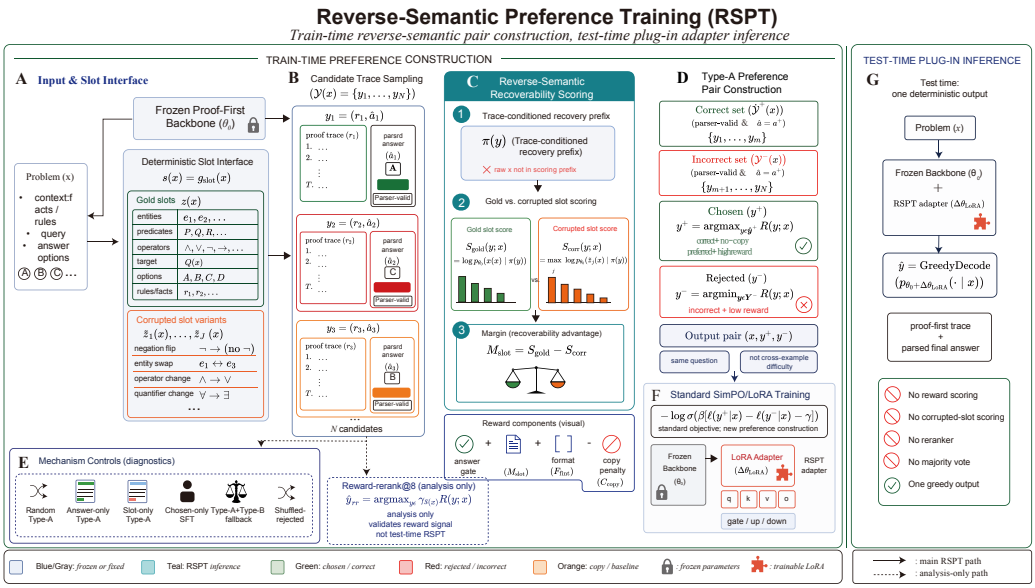

Figure 2: Overview of Reverse-Semantic Preference Training (RSPT). At train time, RSPT samples proof-first traces, scores whether each trace makes the original semantic slots recoverable against corrupted alternatives, and constructs same-question preference pairs for standard SimPO/LoRA training. At test time, the trained adapter is loaded as a plug-in under a fixed proof-first scaffold and produces one greedy output without reward scoring, corrupted-slot scoring, reranking, or majority voting. The dashed branch denotes Reward-rerank@8, which is used only to validate the recoverability reward as an analysis signal.

constructs same-question preference pairs by preferring traces whose gold semantic slots are more recoverable than plausible corrupted alternatives.

3 Method

RSPT constructs same-question preference pairs by measuring whether a generated proof trace preserves recoverable problem semantics. The resulting pairs train a standard SimPO/LoRA adapter; Fig. 2 summarizes the pipeline.

3.1 Problem Setup

We consider proof-first logical-reasoning tasks where each example x consists of a natural-language context, a queried statement, and a finite answer space. A model generates a trace $y = (r, \hat{a})$, where r is the generated proof or reasoning trace and $\hat{a}$ is the final answer parsed by a fixed answer parser. The gold answer is denoted by $a^\star$. This setting covers rule/fact reasoning and first-order logical reasoning benchmarks such as ProofWriter and FOLIO (Tafjord et al., 2021; Han et al., 2024).

Our goal is to improve one-pass proof-first inference. At test time, the model must emit a single deterministic trace and final answer, without a reward model, verifier, reranker, majority vote, or oracle selector. The central difficulty is that final-answer correctness alone is a weak supervision signal: a trace can reach the correct answer while omitting key semantic conditions, copying surface fragments, confusing entities or operators, or relying on a shortcut that is not recoverable from the proof text. RSPT therefore constructs preferences using a reverse-semantic criterion: after reading the generated trace, the original semantic content of the problem should be recoverable.

3.2 Reverse-Semantic Recoverability

For each example x, we extract a compact semantic-slot representation $s(x) = g_{\text{slot}}(x)$, serialized as $z(x)$. It contains predefined dataset-specific fields such as the target, answer options, entities,

predicates, operators, and, when available, explicit facts and rules. We therefore define recoverability relative to this fixed slot interface: a trace is more recoverable when the frozen scorer reconstructs the gold slots more strongly than plausible corrupted alternatives. This operational definition does not claim that the interface captures every logical dependency or reasoning shortcut. Full schemas and extraction rules are provided in Appendix A.1.

Given a candidate trace y, we ask whether the gold slot text $z(x)$ is more recoverable from the trace than plausible corrupted alternatives. We construct corrupted slot variants $\tilde{z} * j(x) * j = 1^J$ by deterministic perturbations over semantically important fields, such as target negation, logical operators, entities, quantifiers, attributes, and rule/fact relations (Appendix A.2). To score recoverability, we condition the frozen backbone θ_0 on the candidate trace and a recovery instruction, but not on the raw original problem text. The length-normalized conditional log-likelihood of the gold slot text is

$$S_{\text{gold}}(y; x) = |z(x)|^{-1} \sum_{t=1}^{|z(x)|} \log p_{\theta_0}\left(z_t(x) \mid \pi(y), z_{<t}(x)\right), \tag{1}$$

where $\pi(y)$ denotes the trace-conditioned recovery prefix. We similarly score each corrupted slot text and keep the strongest corrupted alternative:

$$S_{\text{corr}}(y; x) = \max_j |\tilde{z} * j(x)|^{-1} \sum * t = 1^{|\tilde{z} * j(x)|} \quad \log p * \theta_0\left(\tilde{z} * j, t(x) \mid \pi(y), \tilde{z} * j, < t(x)\right). \tag{2}$$

The reverse-semantic margin is

$$M_{\text{slot}}(y; x) = S_{\text{gold}}(y; x) - S_{\text{corr}}(y; x). \tag{3}$$

A positive margin means that the gold slots are more recoverable than the considered corruptions. This signal is both slot-relative and scorer-relative: relations omitted by g_{slot} may remain undetected, and limitations of the frozen backbone θ_0 may bias the resulting preference ordering. We discuss these boundaries further in Appendix A.1.

The final reward combines answer correctness, recoverability, parser validity, and an anti-copying penalty:

$$R(y; x) = \mathbf{1}[\hat{a}(y) = a^\star(x)] \quad \times\big(1 + \lambda_m M_{\text{slot}}(y; x) + \lambda_g S_{\text{gold}}(y; x)\big) \; + \lambda_f F_{\text{fmt}}(y) - \lambda_c C_{\text{copy}}(y, x). \tag{4}$$

Here $F_{\text{fmt}}(y)$ indicates whether the output satisfies the parser-required proof-first format, and $C_{\text{copy}}(y, x)$ penalizes excessive contiguous copying from the input context. The answer indicator acts as a gate, so recoverability improves the reward only among traces with the correct parsed answer. We use fixed heuristic weights and validate the reward through held-out reward-rerank analysis in the experiments, with perturbation templates and reward-component details provided in Appendices A.2 and A.3.

3.3 Preference Pair Construction

For each training example x, we sample proof-first candidates $\mathcal{Y}(x) = y_1, \ldots, y_N$ from the frozen backbone using stochastic decoding. Each candidate is parsed to obtain $\hat{a}(y_i)$ and scored with $R(y_i; x)$. The main RSPT pair type is a same-question correct-vs-incorrect pair:

$$\mathcal{Y}^+(x) = y \in \mathcal{Y}(x) : \hat{a}(y) = a^\star(x), \; \mathcal{Y}^-(x) \quad = y \in \mathcal{Y}(x) : \hat{a}(y) \neq a^\star(x). \tag{5}$$

A pair is constructed only when both sets are non-empty. For the chosen side, we prefer correct candidates that pass a strict no-copy filter; if none exist, we fall back to all correct candidates. Denoting this eligible correct set by $\widetilde{\mathcal{Y}}^+(x)$, we select

$$y^+ = \arg \max_{y \in \widetilde{\mathcal{Y}}^+(x)} R(y; x), \; y^- \quad = \arg \min_{y \in \mathcal{Y}^-(x)} R(y; x). \tag{6}$$

This construction keeps the contrast local to the same problem: the chosen and rejected traces share the same context, query, answer space, and parser, and differ only in the generated reasoning and final prediction. The answer-correctness split ensures that the chosen trace has the correct parsed answer, while the reward ordering favors correct traces whose proof text makes the original semantic slots recoverable. The no-copy preference prevents traces from being selected merely because they reproduce input content. Auxiliary pair types and mechanism controls, including correct-correct high-vs-low recoverability pairs, chosen-only SFT, random Type-A, answer-only Type-A, slot-only Type-A, and shuffled-rejected pairs, are detailed in Appendix A.4.

Statistic	FOLIO	ProofWriter
Eval. examples	204	600
Label set	T/F/Uncertain	T/F/Unknown
Gold distribution	72/63/69	200/200/200
Avg. context sents.	5.3	18.5
Avg. rules / facts	–	7.32/11.16
Training pairs	339	1,777

Table 1: Dataset profiles for FOLIO and ProofWriter. FOLIO uses *Uncertain* as its third label, whereas ProofWriter uses *Unknown*. Additional prompt-length, query-length, reward-gap, and pair-construction statistics are provided in Table 4.

3.4 Adapter Training and Inference

After constructing preference pairs, RSPT trains a lightweight adapter with a standard SimPO/LoRA backend (Hu et al., 2021; Meng et al., 2024). Given a pair (x, y^+, y^-), SimPO increases the length-normalized response likelihood gap between the chosen and rejected traces under a target margin. Thus, RSPT changes the construction of (y^+, y^-), not the optimization objective. We keep the backbone frozen and train only low-rank adapter parameters. Training hyperparameters, target modules, dataset-specific margins, seeds, and the full SimPO objective are reported in Appendices A.5, A.6, A.7, and B.2.

At inference time, the trained adapter is loaded into the same backbone used by the frozen prompting baselines. The model generates one deterministic proof-first output, and the final answer is obtained by applying the same parser used for all frozen and adapted rows. No recoverability reward, corrupted-slot scoring, reranker, majority vote, or oracle selector is used during RSPT inference. We report RSPT@1 for the bare proof-first adapter output and RSPT-Scaffold@1 for the same adapter loaded under the fixed symbolic-aided proof-first scaffold. Reward-rerank@8 is used only as an analysis setting: it samples eight frozen-backbone traces and selects the highest-reward trace, estimating the strength of the recoverability reward as a trace-selection signal rather than serving as a compute-matched deployment comparison (Appendix A.8).

4 Experiments

We evaluate RSPT as a train-time plug-in adapter for proof-first logical reasoning on natural-language deductive reasoning benchmarks (Tafjord et al., 2021; Han et al., 2024). The central question is whether reverse-semantic preference construction can improve existing logical-reasoning prompts without changing the backbone, prompt scaffold, parser, decoding budget, or deterministic one-output inference protocol. We organize the evaluation around four questions: **RQ1**: Does RSPT improve parser-safe proof-first scaffolds under a fixed inference protocol? **RQ2**: Can the recoverability reward select better traces from held-out candidate pools? **RQ3**: Which components of preference construction account for the gains? **RQ4**: What trade-offs and boundary cases delimit the claim?

4.1 Experimental Setup

We evaluate on two complementary logical-reasoning benchmarks: FOLIO, a compact natural-language first-order reasoning benchmark with labels *True*, *False*, and *Uncertain* (Han et al., 2024), and ProofWriter, a rule-structured benchmark with labels *True*, *False*, and *Unknown* (Tafjord et al., 2021). The processed evaluation splits contain 204 FOLIO examples and 600 ProofWriter examples. As summarized in Table 1, FOLIO stresses semantic recoverability over shorter natural-language contexts, while ProofWriter stresses rule-dense proof-first reasoning over longer contexts.

The main backbone is Qwen2.5-7B-Instruct. All frozen prompting baselines use the same backbone without parameter updates. RSPT trains a lightweight SimPO/LoRA adapter (Hu et al., 2021; Meng et al., 2024); it changes only how chosen–rejected pairs are constructed, not the preference-optimization objective. Full training hyperparameters, seeds, adapter modules, parse-failure audits, paired significance tests, and diagnostic variants are reported in Appendices B.2, B.5, and B.6.

	FOLIO			*ProofWriter*		
Scaffold	Frozen	+RSPT	Gain	Frozen	+RSPT	Gain
Proof-first	61.76	66.67 ± 0.49	+4.90	56.83	67.28 ± 0.59	+10.44
Logic-CoT-style[2022]	64.71	$\mathbf{69.93 \pm 0.28}$	**+5.23**	**62.00**	68.50 ± 0.76	+6.50
SymbCoT-style[2024]	66.18	66.34 ± 1.86	+0.16		*deferred*	
Symbolic-aided CoT[2023]	**69.12**	$\mathbf{69.93 \pm 0.57}^{\dagger}$	+0.82	58.50	$69.28 \pm 2.74^{\dagger}$	**+10.78**
COP-style[2025]	66.67	68.46 ± 2.42	+1.80	61.67	$\mathbf{71.17 \pm 2.13}$	+9.50

Table 2: Plug-in evaluation across parser-safe scaffolds. Best results within each dataset and column are bolded. Each frozen/+RSPT pair uses the same backbone, prompt, parser, decoding budget, and greedy one-output protocol. +RSPT reports mean ± standard deviation over three adapter seeds. The † entries denote the fixed global RSPT-Scaffold@1 deployment setting. ProofWriter SymbCoT-style is deferred to Appendix B.5 because its frozen parse-failure rate exceeds 5%.

We compare against frozen proof-first, logic-oriented, symbolic-aided, SymbCoT-style, and COP-style prompting baselines (Wei et al., 2022; Kojima et al., 2022; Pan et al., 2023; Lyu et al., 2023; Xu et al., 2024; Liu et al., 2025). Rows marked as "style" are controlled prompt variants inspired by the corresponding methods under our unified output format, parser, backbone, and decoding protocol, rather than full-system reimplementations (Appendix B.4). All deterministic frozen and adapted rows use greedy one-output inference with the same final-answer parser. Rows with more than 5% final-answer parse failure are treated as diagnostics rather than headline baselines (Table 7).

We report three interfaces. **RSPT@1** is the bare proof-first adapter output under greedy decoding and is used for the base-vs-adapter mechanism setting. **RSPT-Scaffold@1** loads the same adapter under a globally fixed Symbolic-aided-CoT downstream scaffold; this provides a conservative single deployment interface shared across datasets. **Reward-rerank@8** is analysis-only: it samples eight frozen-backbone traces and selects the highest-reward candidate. Only RSPT@1 and RSPT-Scaffold@1 correspond to deterministic one-output adapted inference. Because different scaffolds have different frozen strengths, we report both same-scaffold plug-in gains and the fixed global deployment row. Full inference and candidate-sampling details are provided in Appendix B.3.

4.2 RQ1: Plug-in Effectiveness

Table 2 compares each frozen scaffold with the same scaffold plus RSPT, holding the backbone, parser, decoding budget, and greedy one-output protocol fixed. We retain only parser-safe rows in the main table; the ProofWriter SymbCoT-style diagnostic is deferred to Appendix B.5. RSPT improves all 9 of 9 complete parser-safe settings, with gains of $+0.16$ to $+5.23$ points on FOLIO and $+6.50$ to $+10.78$ points on ProofWriter.

Under the fixed global Symbolic-aided-CoT deployment scaffold, RSPT-Scaffold@1 reaches $69.93\% \pm 0.57$ on FOLIO and $69.28\% \pm 2.74$ on ProofWriter, exceeding the strongest frozen parser-safe baseline on both datasets. We use this shared scaffold to avoid dataset-specific scaffold selection.

4.3 RQ2: Reward Validation

Figure 3 evaluates the recoverability reward as a selector over held-out eight-trace candidate pools, comparing Original@1, Majority@8 (Wang et al., 2022), Reward-rerank@8, and Oracle@8. Reward-rerank@8 uses the fixed recoverability reward, while Oracle@8 selects a correct trace whenever one appears in the pool.

Reward-rerank@8 substantially outperforms Original@1 and Majority@8 on both datasets, approaching Oracle@8 on FOLIO and matching it on the evaluated ProofWriter pools. Correct-minus-incorrect reward gaps are 0.1068 on FOLIO and 0.4096 on ProofWriter. This validates the frozen-backbone score as a trace-selection signal on these pools, but does not establish scorer unbiasedness across arbitrary logical structures or model families. Reward-rerank@8 is analysis-only and not compute-matched to RSPT@1 because it explicitly scores eight traces (Appendix B.8).

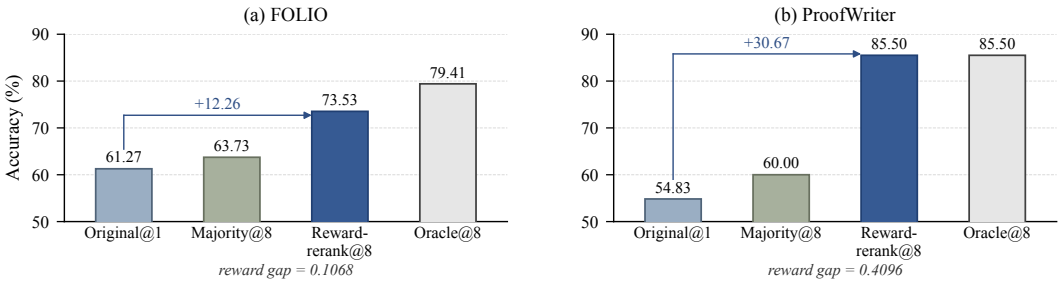

Figure 3: Reward-rerank validation on held-out eight-trace candidate pools. Reward-rerank@8 improves over Original@1 and Majority@8, approaches Oracle@8 on FOLIO, and matches Oracle@8 on ProofWriter.

4.4 RQ3: Mechanism Controls

Figure 4 and Table 3 test whether the gains come from reverse-semantic preference construction rather than generic exposure to correct traces or answer-filtered preference tuning. We compare full RSPT with chosen-only SFT, random Type-A, answer-only Type-A, slot-only Type-A, shuffled-rejected, and TypeA+TypeB controls. These controls isolate selected-trace exposure, answer filtering, same-question contrast, recoverability ranking, anti-copying constraints, and auxiliary correct-correct supervision; full construction details and additional audits are provided in Appendix B.9.

On FOLIO, the mechanism evidence is clearest. Full RSPT reaches 66.67%, improving over the mechanism-control Base at 62.75% and all tested controls, including chosen-only SFT (62.75%), random Type-A (63.24%), answer-only Type-A (61.27%), slot-only Type-A (62.25%), shuffled-rejected (64.22%), and TypeA+TypeB (62.25%). A reward-scored audit among correct predictions gives the same picture: full RSPT obtains the highest recoverability reward (0.2693) and the best gold-slot score (-2.6218) among the tested FOLIO controls. Thus, on FOLIO, reverse-semantic ranking improves both answer accuracy and the recoverability profile of correct proof traces.

On an input-defined FOLIO slot-sensitive subset, full RSPT reaches 61.33%, compared with 54.67% for random Type-A and 53.33% for answer-only Type-A. This larger advantage supports the value of recoverability when key operators and slots matter; full construction and results are reported in Table 12.

On ProofWriter, the mechanism picture is more entangled. The main low-strength RSPT setting improves strongly over Base, reaching $66.11 \pm 0.64\%$ compared with 58.00%, but random Type-A (68.83%) and answer-only Type-A (68.17%) remain competitive in raw accuracy. This indicates that same-question answer-filtered contrast is itself a strong adaptation signal for longer rule-structured examples. A matched high-strength diagnostic partly changes the comparison: full RSPT reaches 70.50%, slightly above slot-only Type-A (70.00%), random Type-A (68.67%), and answer-only Type-A (68.50%), while also giving the strongest recoverability scores among correct ProofWriter predictions. However, all high-strength variants substantially reduce *Unknown* accuracy, so we treat this as a preference-strength diagnostic rather than the main ProofWriter configuration (Table 6).

Overall, the mechanism evidence is dataset-dependent: FOLIO provides the cleanest recoverability-specific evidence, while ProofWriter shows strong plug-in gains but also reveals that answer-filtered same-question contrast is a powerful competing signal.

4.5 RQ4: Trade-offs and Boundaries

Higher-strength ProofWriter training increases overall accuracy but distorts the label profile. Full RSPT reaches 70.50%, above the main low-strength result of $66.11\% \pm 0.64$, but reduces *Unknown* accuracy from 55.50% to 29.00%. Matched controls show the same behavior, so we treat this setting as a preference-strength diagnostic rather than the main result (Appendix B.10).

RSPT also measures recoverability relative to a predefined slot interface and a frozen scorer. It may therefore miss unrepresented logical relations, while scorer limitations may bias trace rankings. The

Dataset	Setting	Acc.	Reward∗corr	$S * \text{gold}, \text{corr}$
FOLIO	Random Type-A	63.24	0.2057	-2.8015
	Answer-only Type-A	61.27	0.2047	-2.8176
	Full RSPT	66.67	**0.2693**	**-2.6218**
ProofWriter	Random Type-A low	68.83	0.5037	-1.9254
	Answer-only Type-A low	68.17	0.5016	-1.9329
	Full RSPT low	66.11	0.5105	-1.8954
	Full RSPT high	70.50	**0.5276**	**-1.8517**

Table 3: Reward-scored recoverability audit among correct predictions. On FOLIO, full RSPT improves both accuracy and recoverability over answer-filtered controls. On ProofWriter, answer-filtered controls remain competitive in raw accuracy, while RSPT gives stronger recoverability scores, especially under the high-strength diagnostic. All reward and gold-slot scores are averaged over correctly answered examples.

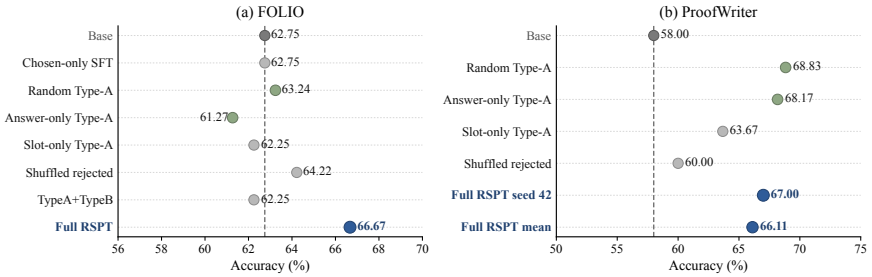

Figure 4: Mechanism controls for reverse-semantic preference construction. On FOLIO, full RSPT outperforms all tested controls, giving the clearest recoverability-specific evidence. On ProofWriter, low-strength full RSPT improves over Base, but random Type-A and answer-only Type-A remain competitive in raw accuracy. Additional high-strength and reward-scored audits characterize this as a preference-strength and label-balance trade-off.

FOLIO slot-sensitive analysis supports the current interface but does not establish universal coverage. Full slot, scorer, scale, transfer, and ProntoQA diagnostics are reported in Appendix B.10.

Overall, the experiments support a deliberately scoped claim: RSPT is an effective train-time plug-in adapter for proof-first logical-reasoning prompts, improving all complete parser-safe same-scaffold settings under deterministic one-output inference. Recoverability-specific construction is clearest on FOLIO, where full RSPT outperforms all tested mechanism controls and yields better recoverability scores among correct traces. On longer rule-structured ProofWriter, the adapter improves strongly and the recoverability reward is an effective trace-selection signal, but answer-filtered same-question contrast is also a strong competing adaptation signal. The method is therefore best understood as a reverse-semantic preference-construction framework whose benefits depend on dataset structure, preference strength, and backbone calibration.

A qualitative same-scaffold example illustrating the recovered antecedent is provided in Figure 5.

5 Conclusion

We introduced Reverse-Semantic Preference Training (RSPT), which constructs preferences by favoring proof traces from which the original problem semantics are more recoverable than corrupted alternatives. Across FOLIO and ProofWriter, the resulting SimPO/LoRA adapter improves all complete parser-safe same-scaffold settings under deterministic one-output inference. Mechanism analyses show the clearest recoverability-specific effect on FOLIO and a stronger competing answer-filtered signal on ProofWriter, supporting semantic recoverability as a useful but deliberately scoped supervision signal.

AI USE STATEMENT

Generative AI tools were used to assist with research ideation and hypothesis refinement; feedback on methodology and experimental design; code and software drafting, editing, and debugging; experimental-workflow support; organization and interpretation of results; and manuscript structuring, drafting, editing, citation checking, and submission preparation. The tools did not replace author review or verification. The authors reviewed and verified all AI-assisted code, experimental procedures, outputs, citations, interpretations, claims, and manuscript text, and take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

The main method section specifies the semantic-slot interface, reverse-semantic recoverability score, reward construction, same-question preference-pair construction, and deterministic one-output inference protocol. Appendix A provides the full slot schemas, corrupted-slot construction, reward components, preference variants, SimPO objective, LoRA parameterization, and inference formulation. Appendix B reports dataset statistics, training configurations, candidate sampling, baseline and prompting details, parser-safety audits, statistical tests, mechanism controls, boundary diagnostics, and qualitative examples. Together, these sections document the data, training, evaluation, and analysis procedures needed to reproduce the reported experiments.

REFERENCES

Oliver Bentham, Nathan Stringham, and Ana Marasović. Chain-of-thought unfaithfulness as disguised accuracy. *arXiv preprint arXiv:2402.14897*, 2024.

Kaj Bostrom, Zayne Sprague, Swarat Chaudhuri, and Greg Durrett. Natural language deduction through search over statement compositions. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 4871–4883, 2022.

Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. Explaining answers with entailment trees. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 7358–7370, 2021.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, et al. 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.

Namgyu Ho, Laura Schmid, and Se-Young Yun. Large language models are reasoning teachers. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 14852–14882, 2023.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Xiang Huang, Ting-En Lin, Feiteng Fang, Yuchuan Wu, Hangyu Li, Yuzhong Qu, Fei Huang, and Yongbin Li. Reverse preference optimization for complex instruction following. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 20700–20723, 2025.

Alon Jacovi, Yonatan Bitton, Bernd Bohnet, Jonathan Herzig, Or Honovich, Michael Tseng, Michael Collins, Roee Aharoni, and Mor Geva. A chain-of-thought is as strong as its weakest link: A benchmark for verifiers of reasoning chains. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4615–4634, 2024.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213, 2022.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*, 2024.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Omer Levy, Luke Zettlemoyer, Jason E Weston, and Mike Lewis. Self-alignment with instruction backtranslation. In *International Conference on Learning Representations*, volume 2024, pp. 3552–3577, 2024.

Junjie Liu, Shaotian Yan, Chen Shen, Zhengdong Xiao, Liang Xie, Wenxiao Wang, and Jieping Ye. Concise and organized perception facilitates reasoning in large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 3469–3498, 2025.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. Faithful chain-of-thought reasoning. In *Proceedings of the 13th international joint conference on natural language processing and the 3rd conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long papers)*, pp. 305–329, 2023.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

Thao Nguyen, Jeffrey Li, Sewoong Oh, Ludwig Schmidt, Jason E Weston, Luke Zettlemoyer, and Xian Li. Better alignment with instruction back-and-forth translation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 13289–13308, 2024.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Wang. Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 3806–3824, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.248. URL https://aclanthology.org/2023.findings-emnlp.248/.

Richard Y Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. Iterative reasoning preference optimization. *Advances in Neural Information Processing Systems*, 37:116617–116637, 2024.

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

Nisarg Patel, Mohith Kulkarni, Mihir Parmar, Aashna Budhiraja, Mutsumi Nakamura, Neeraj Varshney, and Chitta Baral. Multi-logieval: Towards evaluating multi-step logical reasoning ability of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 20856–20879, 2024.

Debjit Paul, Robert West, Antoine Bosselut, and Boi Faltings. Making reasoning matter: Measuring and improving faithfulness of chain-of-thought reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 15012–15032, 2024.

Archiki Prasad, Weizhe Yuan, Richard Yuanzhe Pang, Jing Xu, Maryam Fazel-Zarandi, Mohit Bansal, Sainbayar Sukhbaatar, Jason Weston, and Jane Yu. Self-consistency preference optimization. *arXiv preprint arXiv:2411.04109*, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 53728–53741. Curran Associates, Inc., 2023. doi: 10.52202/075280-2338. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/a85b405ed65c6477a4fe8302b5e06ce7-Paper-Conference.pdf.

Hyun Ryu, Gyeongman Kim, Hyemin S Lee, and Eunho Yang. Divide and translate: Compositional first-order logic translation and verification for complex logical reasoning. In *International Conference on Learning Representations*, volume 2025, pp. 24935–24964, 2025.

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.

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. *Advances in Neural Information Processing Systems*, 36:74952–74965, 2023.

Robert Vacareanu, Anurag Pratik, Evangelia Spiliopoulou, Zheng Qi, Giovanni Paolini, Neha Anna John, Jie Ma, Yassine Benajiba, and Miguel Ballesteros. General purpose verification for chain of thought prompting. *arXiv preprint arXiv:2405.00204*, 2024.

Yuxuan Wan, Wenxuan Wang, Yiliu Yang, Youliang Yuan, Jen-tse Huang, Pinjia He, Wenxiang Jiao, and Michael Lyu. Logicasker: Evaluating and improving the logical reasoning ability of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 2124–2155, 2024.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837. Curran Associates, Inc., 2022. doi: 10.52202/068431-1800. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.

Huimin Xu, Xinnian Mao, Feng-Lin Li, Xiaobao Wu, Wang Chen, Wei Zhang, and Luu Anh Tuan. Full-step-dpo: Self-supervised preference optimization with step-wise rewards for mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24343–24356, 2025.

Jundong Xu, Hao Fei, Liangming Pan, Qian Liu, Mong-Li Lee, and Wynne Hsu. Faithful logical reasoning via symbolic chain-of-thought. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 13326–13365, 2024.

Kaiyu Yang, Jia Deng, and Danqi Chen. Generating natural language proofs with verifier-guided search. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pp. 89–105, 2022.

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

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

Shimao Zhang, Xiao Liu, Xin Zhang, Junxiao Liu, Zheheng Luo, Shujian Huang, and Yeyun Gong. Process-based self-rewarding language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 18097–18110, 2025.

A Additional Method Details

This supplementary material provides implementation details omitted from the main method section, including the semantic-slot interface, corrupted-slot construction, recoverability reward components, preference-pair variants, and adapter-training configuration.

A.1 Semantic-Slot Interface

RSPT uses a deterministic semantic-slot interface to represent the problem content that should remain recoverable from a faithful proof trace. For each example x, the slot extractor g_{slot} maps the natural-language problem into a serialized slot text $z(x)$. The extractor is dataset-specific but fixed before training, and no additional LLM is used for slot extraction.

For FOLIO, the slot representation emphasizes the queried target, answer options, entities, predicates or attributes, logical operators, and salient premise-level semantic relations. Because FOLIO contains natural-language first-order reasoning examples, the extractor uses lightweight string normalization, entity matching, negation/operator detection, and target-statement parsing. The goal is not to produce a complete formal logic program, but to capture the semantic fields that a faithful proof-first trace should preserve.

For ProofWriter, the slot representation additionally includes the explicit fact and rule structure exposed by the benchmark. Facts are serialized as atomic statements, while rules are serialized as premise-to-conclusion relations. The target hypothesis and answer options are included in the same slot text. This structure allows recoverability scoring to test whether a generated proof trace preserves not only surface entities and predicates, but also the rule/fact dependencies needed for the target answer.

The slot text is serialized in a compact deterministic format. A typical serialized slot text contains fields such as:

```
Target: ...; AnswerOptions: ...; Entities: ...;
Predicates: ...; Operators: ...; Facts: ...; Rules: ...
```

Fields that are unavailable for a dataset are omitted rather than filled with hallucinated content. All slot extraction rules are applied before candidate generation and remain unchanged across baselines, RSPT training, reward-rerank analysis, and ablation settings.

Coverage boundary. The semantic-slot interface is an operational approximation to problem semantics rather than a complete formalization of all reasoning-relevant information. It can directly audit omissions or confusions involving represented targets, entities, predicates, operators, facts, and rules, but it may not detect shortcuts that preserve these fields while altering unrepresented relations, discourse structure, quantifier scope, implicit commonsense assumptions, or multi-step dependency structure. Consequently, RSPT should not be interpreted as certifying globally faithful reasoning. Its recoverability signal is only as complete as the slot extractor and corruption family used to instantiate it. Developing richer structured interfaces, including dependency-aware, graph-based, or solver-derived representations, is an important direction for extending the method to more complex logical relations.

A.2 Corrupted-Slot Construction

To make recoverability contrastive, RSPT constructs corrupted slot variants from the gold slot text. Each corrupted variant changes one or more semantically important fields while preserving the overall serialization format. This ensures that the recoverability score compares the gold semantics against plausible alternatives rather than against unrelated text.

We use deterministic perturbation operators over the extracted slot representation. The perturbation set includes the following corruption types when the corresponding field is available:

- **Target corruption:** replace or negate the queried target statement.
- **Entity corruption:** swap an entity with another entity from the same problem.

- **Predicate/attribute corruption:** replace a predicate or attribute with a different predicate or attribute from the context.
- **Operator corruption:** alter logical operators such as negation, conjunction, disjunction, implication, or quantifier cues.
- **Fact corruption:** replace, remove, or modify an explicit fact field.
- **Rule corruption:** modify a rule premise, conclusion, or premise–conclusion relation.
- **Answer-option corruption:** alter the mapping between the target and the answer space.

The corrupted variants are generated only from fields present in the gold slot representation. If a corruption operator cannot be applied to an example, it is skipped. For each candidate trace y, the reward compares the gold slot continuation against the strongest corrupted alternative under the same recovery prefix. This max-over-corruptions design makes the reward conservative: a trace receives a high margin only when the gold slot text is more recoverable than all considered corrupted alternatives.

The recovery prefix used for scoring contains the candidate proof trace and a short instruction asking the frozen model to recover the original semantic slots. Importantly, the raw original problem text is not included in the recovery prefix. Thus, the score measures whether the generated trace itself contains enough semantic evidence to support recovery of the original slot text.

A.3 Reward Components

The main text defines the reward as:

$$\begin{aligned} R(y;x) = \mathbf{1}[\hat{a}(y) = a^\star(x)] \\ \times \big(1 + \lambda_m M_{\text{slot}}(y;x) + \lambda_g S_{\text{gold}}(y;x)\big) \\ + \lambda_f F_{\text{fmt}}(y) - \lambda_c C_{\text{copy}}(y,x). \end{aligned} \tag{7}$$

Unless otherwise noted, we use fixed heuristic weights:

$$\lambda_m = 0.5, \qquad \lambda_g = 0.3, \qquad \lambda_f = 0.1, \qquad \lambda_c = 0.2. \tag{8}$$

These values are not tuned on the evaluation split. They are chosen to make answer correctness a hard gate, while allowing recoverability, format validity, and copy control to rank candidates within answer-consistent pools.

The format indicator $F_{\text{fmt}}(y)$ checks whether the output follows the parser-required proof-first format and contains a parseable final answer. The copy penalty $C_{\text{copy}}(y,x)$ penalizes long contiguous copying from the input context. We compute the maximum contiguous copied span between the generated trace and the input context after lightweight normalization. A candidate with excessive copying is still allowed to appear in the pool, but is disfavored by the reward and by the no-copy preference used when selecting the chosen response.

The answer indicator $\mathbf{1}[\hat{a}(y) = a^\star(x)]$ gates the recoverability terms. This means that a semantically rich but answer-incorrect trace cannot be selected over an answer-correct trace in the main Type-A construction. Within the answer-correct set, however, the recoverability margin encourages traces whose proof text preserves the original target, entities, predicates, operators, facts, and rules more faithfully than corrupted alternatives.

A.4 Preference-Pair Variants and Controls

The main RSPT configuration uses Type-A same-question correct-vs-incorrect pairs. For each example x, candidate traces are sampled from the frozen backbone. If at least one correct and one incorrect candidate are available, the chosen response is the highest-reward eligible correct trace and the rejected response is the lowest-reward incorrect trace.

For the chosen side, RSPT first prefers correct candidates that pass a strict no-copy filter. Let $L_{\text{copy}}(y,x)$ denote the maximum contiguous copied n-gram length between the generated trace and the input context, and let τ_{copy} be a fixed threshold. The non-copying correct set is defined as:

$$\begin{aligned} \mathcal{Y}^+_{\text{nc}}(x) = \big\{ y \in \mathcal{Y}^+(x) \mid C_{\text{copy}}(y,x) = 0 \\ \wedge\, L_{\text{copy}}(y,x) \le \tau_{\text{copy}} \big\}. \end{aligned} \tag{9}$$

The eligible correct set is then:

$$\widetilde{\mathcal{Y}}^{+}(x) = \begin{cases} \mathcal{Y}^{+}_{\mathrm{nc}}(x), & \text{if } \mathcal{Y}^{+}_{\mathrm{nc}}(x) \neq \emptyset, \\ \mathcal{Y}^{+}(x), & \text{otherwise.} \end{cases} \tag{10}$$

The final Type-A pair is:

$$y^{+} = \arg\max_{y \in \widetilde{\mathcal{Y}}^{+}(x)} R(y; x), \qquad y^{-} = \arg\min_{y \in \mathcal{Y}^{-}(x)} R(y; x). \tag{11}$$

We also define Type-B correct-correct pairs for mechanism analysis. When at least two answer-correct candidates are available, we pair the highest-reward correct trace with the lowest-reward correct trace, provided that their reward gap exceeds a fixed threshold. Type-B pairs are not used in the main RSPT configuration. They are used only in auxiliary experiments to test whether additional proof-quality supervision among answer-correct traces improves over filtered Type-A construction.

We include the following controls in the mechanism study:

- **Chosen-only SFT:** removes the preference contrast and trains only on reward-selected correct traces.
- **Random Type-A:** samples an eligible correct trace and an incorrect trace from the same question without using the recoverability reward.
- **Answer-only Type-A:** keeps the same-question correct-vs-incorrect structure, but selects candidates by candidate order rather than reward.
- **Slot-only Type-A:** ranks candidates only by the gold-slot score $S_{\mathrm{gold}}(y; x)$, without using the contrastive margin or full reward.
- **Shuffled-rejected:** keeps the chosen response and prompt, but replaces the rejected response with an incorrect trace from a different example in the same dataset.

These controls isolate whether gains come from generic exposure to correct traces, answer filtering, same-question contrast, reverse-semantic ranking, or anti-copying constraints.

A.5 Adapter Training Details

RSPT trains a lightweight LoRA adapter with the SimPO objective. The backbone parameters are frozen, and only the LoRA parameters are updated. Unless otherwise noted, we use bf16 training, LoRA rank $r = 16$, LoRA scale factor $\alpha = 32$, dropout 0.05, and SimPO coefficient $\beta = 2.0$. The LoRA adapter is applied to the attention and feed-forward projection modules: q_{proj}, k_{proj}, v_{proj}, o_{proj}, $gate_{\mathrm{proj}}$, up_{proj}, and $down_{\mathrm{proj}}$.

For each prompt-response sequence, prompt tokens and padding tokens are masked. The SimPO score is computed as the mean next-token log-probability over the unmasked response-suffix tokens. This matches the length-normalized response score used in the main method section. Dataset-specific learning rates, target margins, batch sizes, random seeds, and training-set sizes are reported in the supplementary experimental details.

A.6 SimPO Objective

Given a preference pair (x, y^{+}, y^{-}), the policy assigns each trace a length-normalized log-probability over the unmasked response-suffix tokens. In implementation, we tokenize the prompt-response sequence, mask all prompt and padding tokens, and average next-token log-probabilities over the remaining unmasked target positions. Let $\mathcal{M}(x, y)$ denote the set of unmasked response-token positions. We define

$$\ell_\theta(y \mid x) = |\mathcal{M}(x, y)|^{-1} \sum_{t \in \mathcal{M}(x,y)} \log p_\theta(u_t \mid u_{<t}), \tag{12}$$

where u is the tokenized prompt-response sequence.

The SimPO loss is

$$\mathcal{L}_{\mathrm{SimPO}} = -\log \sigma\left(\beta\left[\ell_\theta(y^{+} \mid x) - \ell_\theta(y^{-} \mid x) - \gamma\right]\right), \tag{13}$$

where β is the inverse-temperature coefficient and γ is the target margin. This follows the standard SimPO objective. RSPT changes the construction of the chosen–rejected pair, not the preference-optimization loss.

A.7 LoRA Adapter Parameterization

The backbone parameters are frozen during RSPT training. Only low-rank adapter parameters are updated:

$$\theta = \theta_0 + \Delta\theta_{\mathrm{LoRA}}, \tag{14}$$

where θ_0 denotes the frozen backbone and $\Delta\theta_{\mathrm{LoRA}}$ denotes the trainable LoRA update. This parameter-efficient design allows the trained adapter to be loaded under existing proof-first scaffolds without changing the backbone, parser, decoding budget, or output protocol.

Unless otherwise noted, LoRA adapters are applied to the following modules:

$$q_{\mathrm{proj}},\ k_{\mathrm{proj}},\ v_{\mathrm{proj}},\ o_{\mathrm{proj}},\ gate_{\mathrm{proj}},\ up_{\mathrm{proj}},\ down_{\mathrm{proj}}.$$

This module set covers both attention projections and feed-forward projections.

A.8 Inference and Reward-Rerank Formulation

At deterministic inference time, the RSPT adapter is loaded into the frozen backbone and used for greedy proof-first generation:

$$\hat{y} = \mathrm{GreedyDecode}\left(p_{\theta_0+\Delta\theta_{\mathrm{LoRA}}}(\cdot \mid x)\right). \tag{15}$$

The same fixed answer parser is then applied to obtain the final prediction. No recoverability reward, corrupted-slot scoring, reranker, majority vote, or oracle selector is used during RSPT inference.

Reward-rerank@8 is used only for reward validation. For each example, we sample eight candidate traces from the frozen backbone and select

$$\hat{y}_{\mathrm{rr}} = \arg\max_{y \in \mathcal{Y}_8(x)} R(y; x), \tag{16}$$

where $\mathcal{Y}_8(x)$ is the eight-sample candidate pool and $R(y; x)$ is the fixed recoverability reward. Unlike deterministic RSPT inference, Reward-rerank@8 observes multiple sampled candidates and performs explicit reward scoring. It is therefore an analysis protocol for evaluating the recoverability reward as a trace-selection signal, not a compute-matched deployment baseline.

At inference time, all adapted and frozen rows use the same parser, decoding budget, and deterministic one-output protocol unless explicitly marked as reward-rerank analysis.

B Additional Experimental Details

This supplementary material provides experimental details omitted from the main text, including dataset statistics, training configurations, baseline implementation, parser-safety audits, statistical tests, additional reward-rerank analysis, mechanism-control details, boundary diagnostics, and qualitative examples.

B.1 Dataset Details

We use FOLIO and ProofWriter as the two main benchmarks for plug-in evaluation. FOLIO contains compact natural-language first-order reasoning examples with three labels: *True*, *False*, and *Uncertain*. ProofWriter contains rule-structured natural-language reasoning examples with three labels: *True*, *False*, and *Unknown*. The processed evaluation splits contain 204 FOLIO examples and 600 ProofWriter examples.

Table 4 expands the dataset profile reported in the main text. FOLIO has shorter contexts and does not expose explicit rule/fact fields in a structured format. ProofWriter has longer contexts and exposes explicit rule/fact structure, allowing the reverse-semantic slot interface to include rule and fact dependencies.

Statistic	FOLIO	ProofWriter
Eval examples	204	600
Labels	3	3
Gold dist. (T/F/U)	72 / 63 / 69	200 / 200 / 200
Avg. prompt words	157.4	214.8
Avg. context sentences	5.3	18.5
Avg. rules / facts	N/A	7.32 / 11.16
Avg. query words	9.6	4.6
Training pairs	339	1777
Mean pair reward gap	0.2565	0.4599

Table 4: Expanded dataset and training-signal profile. Gold distributions follow *True*/*False*/*Uncertain* for FOLIO and *True*/*False*/*Unknown* for ProofWriter. The mean pair reward gap is computed over Type-A same-question correct-vs-incorrect training pairs.

The generated Type-A chosen/rejected responses used for preference training average roughly 50 words on both datasets. The mean pair reward gap is reported in reward units rather than normalized probability or accuracy units. Because FOLIO does not expose explicit rule/fact fields, explicit rule/fact counts are marked as N/A.

ProntoQA diagnostic split. ProntoQA is not used as a primary benchmark. We report it only as a supplementary diagnostic on a fixed 400/100 split. ProntoQA is generated from first-order-logic world models and is useful for controlled analysis of chain-of-thought reasoning. However, in our setting, answer-only and random Type-A controls are already very strong on the fixed split, making it less informative for isolating the proposed reverse-semantic recoverability mechanism.

B.2 Training Configuration

The main backbone is Qwen2.5-7B-Instruct. All frozen prompting baselines use the same backbone without parameter updates. RSPT trains only a lightweight LoRA adapter with the SimPO objective. Unless otherwise noted, all adapters use bf16 training, LoRA rank $r = 16$, LoRA scale factor $\alpha = 32$, dropout 0.05, and SimPO coefficient $\beta = 2.0$. Each main adapter is trained for one epoch to reduce overfitting risk on the compact filtered preference sets. The main RSPT training sets contain 339 Type-A pairs for FOLIO and 1777 Type-A pairs for ProofWriter, with mean chosen–rejected reward gaps of 0.2565 and 0.4599, respectively.

Configuration	FOLIO	ProofWriter
Backbone	Qwen2.5-7B-Instruct	Qwen2.5-7B-Instruct
Training objective	SimPO	SimPO
Adapter	LoRA	LoRA
LoRA rank r	16	16
LoRA α	32	32
LoRA dropout	0.05	0.05
Precision	bf16	bf16
Epochs	1	1
Learning rate	5×10^{-5}	1×10^{-5}
SimPO β	2.0	2.0
SimPO margin γ	0.3	0.3
Main seeds	123 / 42 / 7	42 / 123 / 7
Training pairs	339	1777
Mean reward gap	0.2565	0.4599

Table 5: Main RSPT training configuration for FOLIO and ProofWriter. ProofWriter uses a lower learning rate in the main low-strength setting to avoid over-sharpening the label distribution. Mean reward gap is computed over Type-A same-question correct-vs-incorrect preference pairs.

LoRA adapters are applied to the attention and feed-forward projection modules: q_{proj}, k_{proj}, v_{proj}, o_{proj}, $gate_{\text{proj}}$, up_{proj}, and $down_{\text{proj}}$. Prompt tokens and padding tokens are masked when

computing the SimPO response score. The policy score is computed as the mean next-token log-probability over unmasked response-suffix tokens.

High-strength ProofWriter diagnostic. For the ProofWriter high-strength diagnostic, we use learning rate 5×10^{-5}, SimPO margin $\gamma = 0.1$, and seed 42. This setting improves aggregate accuracy but substantially reduces *Unknown* accuracy, so it is reported only as a boundary case rather than the main ProofWriter configuration. To make the comparison fair, we also run matched high-strength controls with the same recipe.

ProofWriter setting	Overall	Macro-F1	Unknown acc.
Base	58.00	58.01	52.50
Low-strength RSPT	66.11 ± 0.64	–	55.50
Full RSPT high	70.50	67.01	29.00
Slot-only Type-A high	70.00	66.09	27.50
Random Type-A high	68.67	64.45	25.00
Answer-only Type-A high	68.50	62.97	20.00
Shuffled-rejected high	60.00	50.91	5.50

Table 6: Matched high-strength ProofWriter diagnostic under the same high-strength recipe. Full RSPT is the strongest seed-42 setting in overall accuracy and macro-F1, but all high-strength variants substantially reduce *Unknown* accuracy. We therefore keep the low-strength setting as the main ProofWriter configuration and use high-strength results only as diagnostics.

B.3 Inference Protocol and Candidate Sampling

All deterministic one-output evaluations use greedy decoding. This includes frozen prompting baselines, RSPT@1, and RSPT-Scaffold@1. The same parser, answer space, decoding budget, and output format are used for frozen and adapted rows. Thus, each same-scaffold comparison differs only in whether the RSPT adapter is loaded.

For candidate-pool construction, reward validation, and self-consistency, we sample $N = 8$ proof-first traces per question from the frozen backbone using temperature 0.8 and top-$p = 0.95$. The same eight-sample setting is used for SC-CoT@8, Reward-rerank@8, and Oracle@8 analysis. Reward-rerank@8 selects the candidate with the highest fixed recoverability reward. Oracle@8 selects a correct trace whenever at least one sampled candidate has the gold answer, and therefore provides an upper-bound estimate on the eight-sample pool rather than a deployable method.

Reward-rerank@8 is not compute-matched to deterministic adapted inference. It observes multiple sampled traces and performs explicit reward scoring at inference time, whereas RSPT@1 and RSPT-Scaffold@1 must produce one deterministic output without reward scoring, reranking, majority voting, or an oracle selector. We therefore use Reward-rerank@8 only to validate the recoverability reward as a trace-selection signal.

B.4 Baseline and Prompting Details

We compare RSPT against frozen prompting scaffolds and analysis baselines. The frozen scaffolds include proof-first prompting, logic-CoT-style prompting, SymbCoT-style prompting, Symbolic-aided CoT, and COP-style prompting. Rows marked as "style" are controlled prompt variants inspired by the corresponding methods. They follow the same high-level prompting principle under our unified proof-first output format, parser, backbone, and decoding protocol, but they are not full-system reimplementations.

- **Proof-first prompting** asks the model to produce an intermediate proof or reasoning trace before the final answer.

- **Logic-CoT-style prompting** encourages explicit logical reasoning and intermediate rule application.
- **SymbCoT-style prompting** encourages symbolic or semi-symbolic chain-of-thought reasoning under the same parser-safe answer format.
- **Symbolic-aided CoT** encourages explicit formal or semi-formal reasoning while preserving deterministic one-output inference.
- **COP-style prompting** follows the high-level idea of concise and organized perception by identifying relevant premises and reorganizing context before reasoning.

The RSPT adapter is loaded under the same prompt scaffold when evaluating plug-in gains. Thus, each frozen scaffold and its +RSPT counterpart share the same prompt, parser, answer space, decoding protocol, and evaluation split.

ProofWriter SymbCoT-style diagnostic. ProofWriter SymbCoT-style is excluded from the main parser-safe plug-in table because its frozen row has 11.67% final-answer parse failure, exceeding the 5% threshold. We therefore treat it as a diagnostic row rather than a headline parser-safe baseline. This exclusion does not affect the 9/9 parser-safe same-scaffold result reported in Table 2 of the main paper.

B.5 Parser Safety and Evaluation Metrics

The primary metric is accuracy after applying the fixed final-answer parser. We also track final-answer parse failure rate. Rows with more than 5% parse failure are retained as diagnostics rather than headline baselines. This rule is important for symbolic-style prompts, where output-format instability can otherwise inflate or obscure reasoning performance.

Setting	Parse Failure	
	FOLIO	ProofWriter
Frozen parser-safe scaffolds	$\leq 5\%$	$\leq 5\%$
SC-CoT@8 (row-level)	0.00%	0.00%
SC-CoT@8 (candidate-level)	low	0.10%
SymbCoT-style frozen (PW)	N/A	11.67%

Table 7: Parser-safety audit summary. Rows exceeding the 5% parse-failure threshold are treated as diagnostics rather than headline baselines. N/A indicates that the row is dataset-specific and not applicable to the other dataset.

Besides accuracy and parse failure, supporting audits include per-label accuracy, prediction distribution, examples fixed or broken relative to the frozen baseline, reward-rerank@8 accuracy, oracle@8 accuracy, pair statistics, and copy-penalty diagnostics.

B.6 Statistical Testing

For paired comparisons, rows are matched by example ID. We use exact two-sided McNemar tests over discordant pairs. When reporting seed-level tests across three RSPT seeds, we apply Bonferroni correction by multiplying the three seed-level p-values by 3. Supporting audits include sign-flip tests and bootstrap confidence intervals.

The FOLIO gains are directionally stable but do not reach significance under the 204-example evaluation split. The ProofWriter gains are larger and statistically stronger, with all three seed-level comparisons remaining significant after Bonferroni correction.

B.7 Additional Main Results

Table 9 reports seed-level results for the main base-vs-RSPT comparison. The main text reports mean and standard deviation where applicable.

Dataset	Seed	Accuracy	McNemar p vs. Base
FOLIO	123	66.67	0.2005
FOLIO	42	66.18	0.3105
FOLIO	7	66.18	0.3368
ProofWriter	42	67.00	1.838×10^{-5}
ProofWriter	123	65.50	0.001232
ProofWriter	7	65.83	0.0007755

Table 8: Seed-level paired significance tests for the main RSPT adapters. FOLIO shows stable positive but not statistically significant gains on the 204-example split. ProofWriter shows significant paired gains for all three seeds, remaining significant after Bonferroni correction.

Dataset	Base	Seed 1	Seed 2	Seed 3
FOLIO	62.75	66.67	66.18	66.18
ProofWriter	58.00	67.00	65.50	65.83

Table 9: Seed-level main adapter results. FOLIO mean is 66.34 ± 0.28, giving a $+3.59$ point gain over Base. ProofWriter mean is 66.11 ± 0.64, giving a $+8.11$ point gain over Base.

The main FOLIO result is stable across seeds with small variance. The main ProofWriter result has a larger absolute gain and remains stable under the low-strength setting.

B.8 Reward-Rerank Details

Reward-rerank@8 evaluates the recoverability reward as a trace-selection signal. For each example, eight traces are sampled from the frozen backbone. Original@1 takes the first sampled trace. Majority@8 applies answer-level self-consistency over the eight traces. Reward-rerank@8 selects the trace with the highest recoverability reward. Oracle@8 reports whether at least one sampled trace is correct.

Selection Method	FOLIO	ProofWriter
Original@1	61.27	54.83
Majority@8	63.73	60.00
Reward-rerank@8	73.53	85.50
Oracle@8	79.41	85.50

Table 10: Reward-rerank@8 results on held-out candidate pools. Reward-rerank@8 substantially improves over Original@1 and Majority@8. On ProofWriter, it matches Oracle@8 on the evaluated candidate pools.

Correct traces receive higher average reward than incorrect traces. The correct-minus-incorrect reward gap is 0.1068 on FOLIO and 0.4096 on ProofWriter. This supports the use of the fixed recoverability reward as a trace-selection signal. However, Reward-rerank@8 is not compute-matched to deterministic one-output adapter rows because it explicitly observes and scores eight sampled traces at inference time.

B.9 Mechanism-Control Details

The mechanism study tests whether RSPT gains come from reverse-semantic preference construction rather than generic exposure to correct traces, answer-only filtering, or arbitrary preference tuning. All controls use the same backend training recipe where applicable.

- **Chosen-only SFT:** removes the rejected response and trains only on recoverability-selected correct traces.
- **Random Type-A:** keeps same-question correct-vs-incorrect pairing but randomly selects eligible correct and incorrect traces.

- **Answer-only Type-A:** keeps same-question correct-vs-incorrect pairing but selects candidates by answer correctness and candidate order, without recoverability ranking.
- **Slot-only Type-A:** ranks candidates using only the gold-slot score $S_{\text{gold}}(y; x)$, without the contrastive corrupted-slot margin or full reward.
- **Shuffled-rejected:** keeps the chosen trace and prompt but replaces the rejected trace with an incorrect trace from another example.
- **TypeA+TypeB:** augments Type-A pairs with correct-correct high-vs-low recoverability pairs when available.

Control	FOLIO	ProofWriter
Base	62.75	58.00
Random Type-A	63.24	68.83
Answer-only Type-A	61.27	68.17
Slot-only Type-A	62.25	63.67
Shuffled-rejected	64.22	60.00
Full RSPT	66.67	66.11 ± 0.64

Table 11: Mechanism-control results shared across FOLIO and ProofWriter. FOLIO provides the clearest recoverability-specific evidence because full RSPT outperforms all shared controls. ProofWriter shows a strong adapter effect, but random Type-A and answer-only Type-A are competitive. Additional FOLIO-only controls are reported in text. On FOLIO, two additional controls were also evaluated: chosen-only SFT obtains 62.75%, and TypeA+TypeB obtains 62.25%. Both remain below full RSPT at 66.67%.

The mechanism evidence is dataset-dependent. On FOLIO, full RSPT outperforms all tested controls, indicating that reverse-semantic ranking contributes beyond selected-trace exposure and answer filtering. On ProofWriter, full RSPT improves strongly over Base, but simpler answer-filtered Type-A controls are also competitive. This suggests that the longer rule-structured ProofWriter setting benefits substantially from same-question correct-vs-incorrect preference adaptation even when the recoverability ranking is simplified.

Additional recoverability and strength diagnostics. The main text summarizes the mechanism controls, while the following analysis provides the detailed diagnostic results. On FOLIO, full RSPT not only obtains the highest mechanism-control accuracy (66.67%), but also gives the highest reward-scored recoverability among correct predictions. Its mean recoverability reward is 0.2693, compared with 0.2057 for random Type-A and 0.2047 for answer-only Type-A. Its gold-slot score is also stronger (-2.6218) than random Type-A (-2.8015) and answer-only Type-A (-2.8176).

On ProofWriter, low-strength random Type-A and answer-only Type-A remain competitive in raw accuracy, reaching 68.83% and 68.17% respectively, compared with $66.11 \pm 0.64\%$ for the main low-strength RSPT setting. This suggests that same-question answer-filtered contrast is a strong signal for longer rule-structured examples. The matched high-strength diagnostic in Table 6 shows that full RSPT is slightly stronger than matched controls under the high-strength recipe, but all high-strength variants substantially reduce *Unknown* accuracy. We therefore interpret high-strength ProofWriter as a preference-strength diagnostic rather than the main configuration.

Reward-scored audits further support this interpretation. Among correct ProofWriter predictions, high-strength RSPT obtains the highest recoverability reward (0.5276), best slot margin (-0.0197), and best gold-slot score (-1.8517) among the audited rows. Low-strength RSPT is less aggressive in raw accuracy, but still improves the gold-slot score over random and answer-only low-strength controls.

FOLIO slot-sensitive stress subset. To further test whether the FOLIO mechanism advantage is tied to semantic-slot sensitivity, we construct a slot-sensitive stress subset from the FOLIO evaluation split. The subset is selected using only input-side criteria and does not use model predictions, model correctness, or whether RSPT answers an example correctly. Specifically, we select examples that contain explicit logical-operator cues, multiple semantic slots, and deterministic slot-corruption availability. The operator cues include lexical markers such as *all*, *every*, *if*, *only if*, *not*, *no*, *and*, *or*,

neither, and *nor*. The corruption availability criterion checks whether the extracted slot representation supports deterministic perturbations such as logical-operator flips, target-negation flips, answer-option swaps, and, when available, entity swaps. This construction targets examples where preserving the queried target, logical operators, entities, and answer-space semantics is likely to matter for proof recoverability.

Table 12 reports accuracy on the full FOLIO evaluation split and on the slot-sensitive subset for the available mechanism-control outputs. The subset is harder for all settings, but full RSPT obtains the highest subset accuracy and the smallest degradation from the full evaluation set. Full RSPT reaches 61.33% on the slot-sensitive subset, compared with 54.67% for random Type-A and 53.33% for answer-only Type-A. The advantage of full RSPT over random Type-A increases from $+2.94$ points on the full FOLIO split to $+6.66$ points on the slot-sensitive subset. The advantage over answer-only Type-A increases from $+4.91$ points to $+8.00$ points. This supports the interpretation that reverse-semantic preference construction is especially useful when reasoning depends on preserving key semantic slots rather than only selecting answer-correct traces.

Setting	Full FOLIO	Slot-sensitive	Drop
Base	62.75	52.00	-10.75
Chosen-only SFT	62.75	50.67	-12.08
Random Type-A	63.24	54.67	-8.57
Answer-only Type-A	61.27	53.33	-7.94
Slot-only Type-A	62.25	49.33	-12.92
Shuffled-rejected	64.22	49.33	-14.89
TypeA+TypeB	62.25	53.33	-8.92
Full RSPT	**66.18**	**61.33**	**-4.85**

Table 12: FOLIO slot-sensitive stress test for mechanism-control outputs. The subset is selected using only input-side logical-operator cues, semantic-slot structure, and deterministic slot-corruption availability. The selection does not use model predictions, model correctness, or whether RSPT answers an example correctly. Full RSPT obtains the highest subset accuracy and the smallest drop from the full FOLIO evaluation split.

B.10 Boundary Diagnostics

This section reports diagnostic settings that delimit the main claim. These settings are not used as headline results.

Preference strength on ProofWriter. Increasing preference strength improves raw accuracy but changes the label profile. As summarized in Table 6, high-strength full RSPT reaches 70.50%, above the main low-strength setting at $66.11 \pm 0.64\%$, but reduces *Unknown* accuracy from 55.50% to 29.00%. Matched high-strength controls show the same label-balance issue. We therefore treat high-strength ProofWriter training as a preference-strength diagnostic rather than the main configuration.

Scale diagnostic. We run a same-family Qwen2.5-14B probe on FOLIO. The frozen Base model obtains 69.12%, and SC-CoT@8 obtains 69.61%. Across three RSPT@1 seeds, accuracy is 75.00%, 71.57%, and 68.14%, giving a mean of 71.57 ± 2.80. The mean is above both frozen rows, but one seed falls below them, so we treat this as a promising but seed-sensitive diagnostic rather than a stable scaling claim.

Cross-family transfer. Direct transfer to Mistral is unstable. This suggests that the adapter effect is sensitive to calibration and backbone family. We therefore make no cross-family transfer claim.

ProntoQA diagnostic. ProntoQA is reported only as a diagnostic ceiling case. On the fixed split, Base obtains 67.00%, while RSPT reaches $88.33 \pm 4.11\%$. However, random Type-A and answer-only Type-A controls reach 92.00% and 89.00%, respectively. Because simpler controls already perform extremely well, ProntoQA is less informative for isolating the reverse-semantic recoverability mechanism.

ProntoQA setting	Accuracy
Base	67.00
RSPT	88.33 ± 4.11
Random Type-A	92.00
Answer-only Type-A	89.00

Table 13: ProntoQA diagnostic results. Although RSPT improves strongly over Base, random Type-A and answer-only Type-A controls are also very strong, making this dataset less useful for isolating recoverability-specific effects.

B.11 Qualitative Plug-in Case Study

Figure 5 shows a same-scaffold FOLIO example. The frozen Symbolic-aided-CoT scaffold follows an unsupported conditional branch by assuming the missing antecedent that *1984* is a streaming service, yielding the wrong *False* label. Loading the RSPT adapter makes this missing antecedent recoverable in the trace and preserves the gold *Uncertain* prediction under one-pass inference.

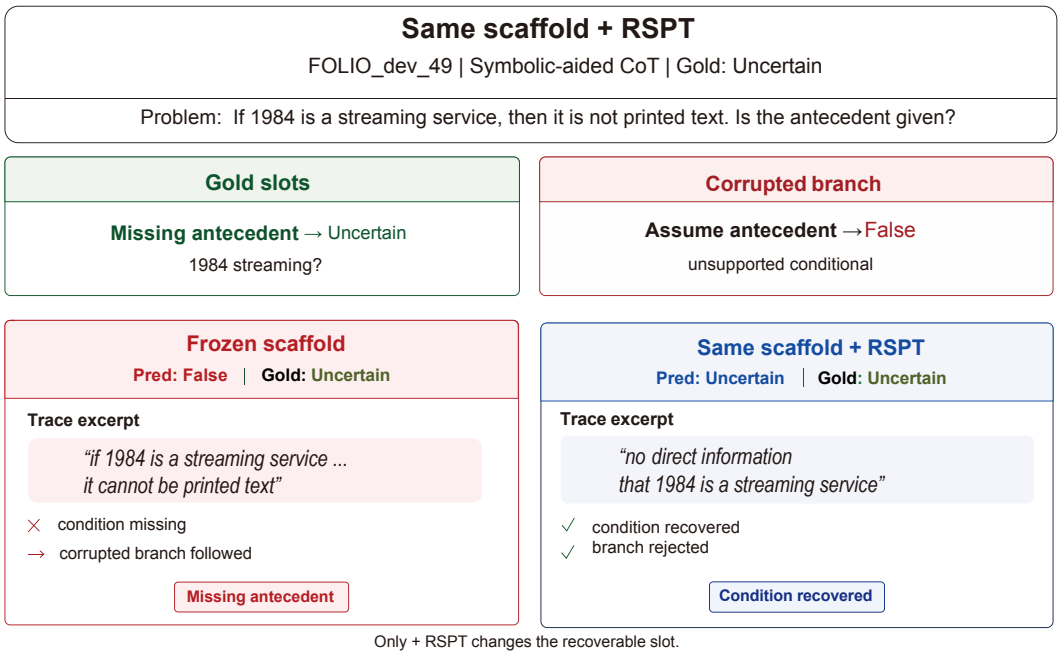

Figure 5: Qualitative plug-in case study. Under the same Symbolic-aided-CoT scaffold, the frozen trace follows a corrupted conditional branch, whereas +RSPT recovers the missing antecedent and predicts the gold *Uncertain* label.