# When the Answer Extractor Is Part of RLVR Evaluation

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

## Abstract

In closed-set multiple-choice RLVR, one answer extractor often assigns reward and reports test accuracy. A score change under this shared reader can reflect a better answer choice, a more readable answer format, or both. We audit this coupling with frozen outputs and an independent output-contract intervention. On 1,270 common MedQA items, the shared parser reports a +5.62 point training gain, while two independent model readers report $-0.52$ points (DeepSeek; 95% CI $[-1.71, +0.66]$) and $-1.94$ points (Llama; $[-3.33, -0.63]$). With the same model weights and a 512-token budget, retaining the explanation but requiring a terminal `Answer: X` changes the shared-parser gain from +5.87 to +0.47 points; the training-by-prompt interaction is $-5.40$ points (95% CI $[-8.37, -2.10]$). The answer marker appears in 97.8–99.6% of completions, although exact final-line compliance varies by checkpoint. These results do not determine semantic truth from any single reader. They show that the reported RLVR gain depends materially on the interface that reads the answer, and that the shared reader should not be the sole validator.

## 1 Introduction

RLVR for multiple-choice tasks optimizes a generated completion through a verifier. Before the verifier can compare a response with the reference, an extractor must recover a finite answer label from free-form text. If the same extractor supplies the training reward and the reported evaluation, the metric is coupled to the optimization target. A higher score can therefore mean a better choice, a more reliable output contract, or a mixture. This is a measurement problem adjacent to reward hacking and verifier overoptimization (Gao et al., 2023; Chen et al., 2025; Chu et al., 2025): the reference and reward rule may be correct while the optimized interface remains an incomplete proxy for the intended task.

We separate these explanations without claiming that any reader is semantic ground truth. First, we freeze matched base and trained outputs and reread exactly the same text with the reward parser and two independent model readers. This changes only the reader. Second, we regenerate under the same weights and token budget with a prompt that preserves the explanation but requires a terminal `Answer: X`. This changes the output contract while holding the model checkpoint fixed within each condition. Together, the two tests ask whether the reported gain survives a different way of reading or a more explicit way of writing.

The main evidence is complementary. Frozen-output rereading changes the sign of the MedQA conclusion for two independent readers. The output-contract intervention shrinks the shared-parser gain from 5.87 to 0.47 points and yields a negative interaction with the answer field present in nearly all completions. We therefore make a measurement claim: the answer interface is part of the optimized system, so a shared extractor is insufficient as the sole validator of a capability gain.

**Scope of the claim.** We do not argue that model readers are ground truth, that RLVR cannot improve semantic accuracy, or that every parser-specific gain is spurious. We argue that a positive change measured only by the reader that supplied reward is underidentified. The strongest conclusion supported by our data is about invariance: the MedQA change is not stable across independent readings or a matched answer contract. This distinction keeps the audit useful even when all available evaluators are imperfect.



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.

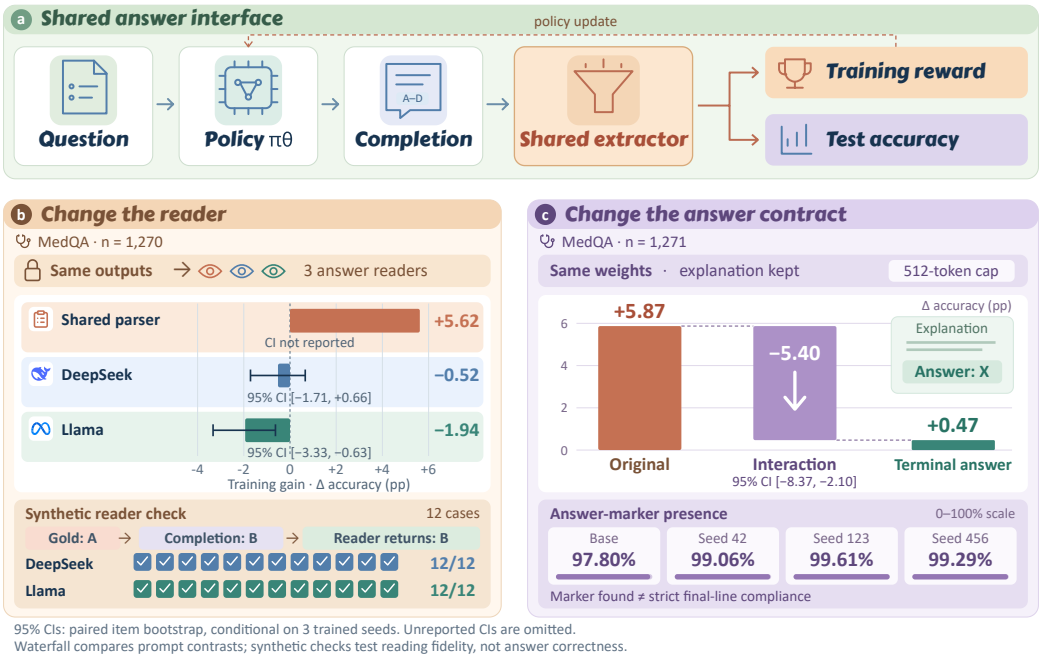


Figure 1: Auditing the shared answer interface. (a) One extractor supplies training reward and test accuracy. (b) Filled bars show paired training gains on identical frozen outputs; whiskers show reported 95% item-bootstrap intervals, conditional on three trained seeds. The 12-case synthetic check tests reading fidelity, not semantic correctness. (c) The waterfall compares prompt-specific training gains: $+5.87 - 5.40 = +0.47$ points; the interaction CI is $[-8.37, -2.10]$. Marker presence is shown per checkpoint and does not imply strict final-line compliance. Unreported intervals are omitted.

**Contributions.** (1) We formulate shared answer extraction as an optimization-coupled measurement problem. (2) We provide a fixed-output audit that separates parser-specific support from support that survives independent readers. (3) We show that a simple answer contract changes the measured gain while preserving the requested explanation and token budget.

## 2 Related Work

Modern post-training builds on preference-based reinforcement learning, instruction tuning, and direct preference optimization (Christiano et al., 2017; Ouyang et al., 2022; Schulman et al., 2017; Rafailov et al., 2023). Parameter-efficient adaptation and data-efficient alignment make these procedures practical across model scales (Hu et al., 2022; Dettmers et al., 2023; Zhou et al., 2023). Chain-of-thought prompting and self-training further establish generated reasoning traces as both supervision and evaluation objects (Wei et al., 2022; Singh et al., 2023; Yuan et al., 2024). Our study concerns a downstream measurement issue shared by these settings: a generated trace must eventually be mapped to a decision that can be scored.

Verifiers support outcome- and process-supervised reasoning on GSM8K and MATH (Cobbe et al., 2021; Hendrycks et al., 2021; Lightman et al., 2024; Wang et al., 2024). Recent RLVR systems scale this recipe through open training and serving stacks (Shao et al., 2024; Guo et al., 2025; Yu et al., 2025b; Kwon et al., 2023; Sheng et al., 2024). The approach now extends beyond mathematical reasoning to medicine, event extraction, and relation extraction (Zhang et al., 2025; Gao et al., 2024; Dai et al., 2026). Studies of data selection and trajectory use show that the observed training effect also depends on difficulty filtering and post-training data construction (Bae et al., 2026; Ding et al., 2026; Kang et al., 2025). We isolate a complementary dependency at evaluation time: how the generated answer is read.

Reward overoptimization studies what policies learn when a reward is easier to optimize than the intended objective (Gao et al., 2023; Chen et al., 2025; Mahmoud et al., 2026; Egashira et al., 2026; Chu et al., 2025). Multiple-choice evaluation exhibits related interface effects: scores vary with option order, label position, prompt formatting, tokenization, and distractor construction (Pezeshkpour & Hruschka, 2024; Zheng et al., 2024; Sclar et al., 2024; Sanz-Guerrero et al., 2025; Jayaram et al., 2024; Guo et al., 2026). Several systems learn to extract answers more reliably, while evaluation frameworks necessarily define regular expressions, fallback rules, and abstention behavior (Contributors, 2023; Gao et al., 2021; Yu et al., 2025a; Molfese et al., 2025). Our audit asks whether a gain produced and reported through one generated-answer extractor survives an independent reader applied to the same text.

The measurement problem spans benchmark families with different answer distributions. MedQA and clinical-knowledge evaluations emphasize medical multiple choice (Jin et al., 2021; Singhal et al., 2022; Nori et al., 2023); ARC, MMLU, WinoGrande, and LegalBench cover science, broad knowledge, commonsense, and legal reasoning (Clark et al., 2018; Hendrycks et al., 2020; Sakaguchi et al., 2020; Guha et al., 2023). Widely used open model families such as Qwen and Mistral make exact prompt, decoding, and extraction specifications especially important for comparisons across implementations (Bai et al., 2023; Yang et al., 2025; Jiang et al., 2023). The readers serve as controls; no baseline ranking is implied. The methodological contribution is the before-and-after comparison at frozen checkpoints, paired with a prompt intervention that makes the desired answer contract explicit: the reader is held fixed while outputs change in one audit, and outputs are held fixed while the reader changes in the other.

## 3 Answer Interfaces and Audit Framework

Let $D = \{(x_i, y_i, \mathcal{C}_i)\}_{i=1}^{n}$ contain questions, reference labels, and valid option sets. A policy at checkpoint $\theta$ produces completion $g_i^{(p)}(\theta)$ under prompt $p$. An extractor $e$ maps this text to a label in $\mathcal{C}_i$ or an abstention $\perp$. Define correctness and accuracy as

$$z_{i,e,p}(\theta) = \mathbf{1}\{e(g_i^{(p)}(\theta)) = y_i\}, \qquad A_{e,p}(\theta) = \frac{1}{n}\sum_{i=1}^{n} z_{i,e,p}(\theta). \tag{1}$$

Abstentions count as incorrect. When reward uses $e_r$, the policy is optimized for outputs accepted by that extractor. Evaluation with $e_r$ measures success of the resulting policy–reader pair. It does not by itself distinguish a changed answer choice from a changed expression of the same choice.

For base checkpoint $\theta_0$ and trained seeds $\theta_s$, $s \in S = \{42, 123, 456\}$, our primary estimand averages paired item changes across the selected seeds:

$$\Delta_{e,p} = \frac{1}{n}\sum_{i}\left[\frac{1}{|S|}\sum_{s\in S} z_{i,e,p}(\theta_s) - z_{i,e,p}(\theta_0)\right]. \tag{2}$$

All reported accuracy differences are percentage points. The same base output is shared across the three seed comparisons; the three copies are not independent observations.

The frozen-output audit holds $g_i(\theta)$ fixed and varies $e$. For a control reader $e_c$, define the reader gap

$$G_c = \Delta_{e_r} - \Delta_{e_c} = \big[A_{e_c}(\theta_0) - A_{e_r}(\theta_0)\big] - \big[\overline{A}_{e_c}(\theta_s) - \overline{A}_{e_r}(\theta_s)\big]. \tag{3}$$

Thus, a gap can arise when a control reader's baseline accuracy advantage over the parser contracts after training. This identity is a measurement decomposition, not an attribution of individual errors to formatting. It is useful because post-training reader agreement alone can hide a large baseline difference.

The contract audit instead fixes the reader and changes the generation prompt from original $o$ to explicit answer field $c$. Its paired interaction is

$$I = \Delta_{e_r,c} - \Delta_{e_r,o}. \tag{4}$$

A negative $I$ means that the training gain is smaller under the explicit contract. Unlike frozen-output rereading, this intervention can change the generated answer and the explanation as well as the format.

Table 1: Two complementary audits. Each tests an observable invariance, with a different remaining ambiguity.

| Audit | Held fixed | Changed | Identifies |
|---|---|---|---|
| Frozen outputs | Item, checkpoint, saved completion | Answer reader | Dependence on readout |
| Answer contract | Item, checkpoint, reader, token cap | Generation prompt | Dependence on output contract |

Table 1 summarizes the design. A positive shared-parser gain with no corresponding positive control-reader gain is *parser-specific support*. A gain that survives both readers and the contract test has broader measurement support. Neither outcome converts a model reader into a semantic gold standard. The audits assess whether an observed improvement is stable under alternative measurements of the intended answer.

To expose what a net gain conceals, we also count paired correctness transitions. For each reader and seed, let $N_{ab}^{(e,s)}$ count items with base correctness $a$ and trained correctness $b$. Then

$$\Delta_{e,s} = \frac{N_{01}^{(e,s)} - N_{10}^{(e,s)}}{n}, \qquad \sum_{a,b\in\{0,1\}} N_{ab}^{(e,s)} = n. \tag{5}$$

The transition counts describe changes under a particular reader. They distinguish offsetting gains and losses from genuinely stable predictions, but do not adjudicate which reading of a completion is right.

## 4 Experimental Design

We center the audit on MedQA (Jin et al., 2021), with ARC-Challenge (Clark et al., 2018) as a secondary boundary check. The frozen MedQA audit uses 1,270 items with complete labels for the base and all three trained seeds under every reader. One incomplete item is excluded without imputation. The separate contract collection contains 1,271 matched items. These are distinct generation audits; their original-prompt baseline accuracies differ and their estimates must not be pooled.

The trained checkpoints were selected before the audit using the canonical development-set criterion. Candidate steps were 250 and 375; the checkpoint with higher development accuracy was selected within each seed and learning rate, with exact ties resolved toward the earlier step. The learning-rate rule compares mean seed-level development accuracy, choosing $10^{-6}$ for differences no larger than 0.002. The selected medicine learning rate is $2 \times 10^{-5}$. Our inference is conditional on the resulting three checkpoints, rather than an estimate of performance across a population of training runs.

The canonical policy backbone is Qwen2.5-7B-Instruct (Hui et al., 2024). Its archived training setup uses GRPO with eight completions per prompt, KL coefficient 0.001, and rank-64 LoRA on all linear layers (Hu et al., 2022). The canonical runs use 2,000 training examples when available and 375 updates; all three selected medicine checkpoints are at step 375. These settings describe the upstream policy training. The present reader and contract audits reuse those checkpoints rather than retraining a model for each evaluator. The development selection manifest is retained with the audit artifacts so that the policy choice can be distinguished from the subsequent choice of measurement.

### 4.1 Frozen completions and independent readers

The shared parser is the archived reward/evaluation readout. We compare its saved labels with those from `deepseek-v4-pro` and `llama-3.3-70b-instruct`. Both controls are instructed to extract the answer expressed by the saved response, not solve the question or correct it. They receive the question and options to interpret references, together with the completion; the parser's answer is not supplied. The reader contract requests one JSON choice from A–D, `NO_ANSWER`, or `AMBIGUOUS`. Invalid or unresolved choices count as incorrect. Appendix C gives the exact prompt.

Table 2: Frozen MedQA outputs: 1,270 paired items, trained accuracy averaged across three selected seeds. Accuracies are percentages; changes and intervals are percentage points. The source does not report a parser interval.

| Reader | Base | Trained mean | Change | 95% CI |
|---|---|---|---|---|
| Shared parser | 53.15 | 58.77 | $+5.62$ | – |
| DeepSeek | 59.37 | 58.85 | $-0.52$ | $[-1.71, +0.66]$ |
| Llama | 62.52 | 60.58 | $-1.94$ | $[-3.33, -0.63]$ |

The analysis joins labels by item identifier before computing accuracy or uncertainty. This prevents differences in the available item subset from being mistaken for reader effects. Because the generated strings are fixed within this audit, any difference between reader scores arises at readout rather than from a new policy sample. Agreement is reported separately from accuracy: two readers can agree on the wrong answer, and an extractor can be consistently different without being consistently worse.

We add a 12-case synthetic faithfulness check in which the question's gold answer is A but the saved response explicitly chooses B. Following the completion means returning B; returning A would instead reveal re-solving in this controlled setting. Both readers return B on all 12 cases, with no A answers or invalid outputs. This checks the intended use of the completion on unambiguous cases. It does not validate medical expertise or ambiguous-response interpretation.

### 4.2 Answer contract and uncertainty

The prompt pair uses the same checkpoints and a 512-token generation cap. The contract preserves the request for an explanation and asks for a terminal `Answer: X` field. It does not force decoding or truncate explanations to insert the field. We track three separate properties: marker presence, placement on a separate final line, and acceptance by the strict extraction rule. For an output property $h$, its rate is

$$C_h(\theta, p) = \frac{1}{n} \sum_i \mathbf{1}\{h(g_i^{(p)}(\theta))\}, \quad h \in \{\text{marker, final line, strict acceptance}\}. \tag{6}$$

These predicates are not interchangeable. In particular, finding an answer field does not establish that it obeys a standalone-line requirement.

Reported 95% intervals use 4,000 paired item-bootstrap resamples. Each resampled item carries all checkpoints and, where relevant, both prompts; the full gain or interaction is recomputed. If $T^{(b)}$ denotes a bootstrap replicate, the interval is

$$\mathrm{CI}_{95}(T) = [q_{.025}(\{T^{(b)}\}_{b=1}^{4000}), q_{.975}(\{T^{(b)}\}_{b=1}^{4000})]. \tag{7}$$

The intervals quantify item uncertainty conditional on the selected seeds. They do not include new-training-run variability, reader-model variation, or uncertainty about the semantic validity of a reader.

## 5 Results

### 5.1 The gain changes with the reader

Table 2 shows that the shared parser's positive MedQA change is not reproduced by either control. DeepSeek's interval includes zero; Llama's lies below zero. These are different outcomes, so we do not describe both readers as establishing a decrease. Their common result is that neither supports the parser's positive increase on these exact outputs.

The baseline gap is central to this discrepancy. Before training, DeepSeek scores 6.22 points above the parser and Llama 9.37 points above it. After averaging the trained checkpoints, these differences shrink to 0.08 and 1.81 points. Equation 3 therefore gives reader gaps of 6.14 and 7.56 points. A high level of post-training agreement is compatible with a strongly reader-dependent training effect when the baseline was read differently.

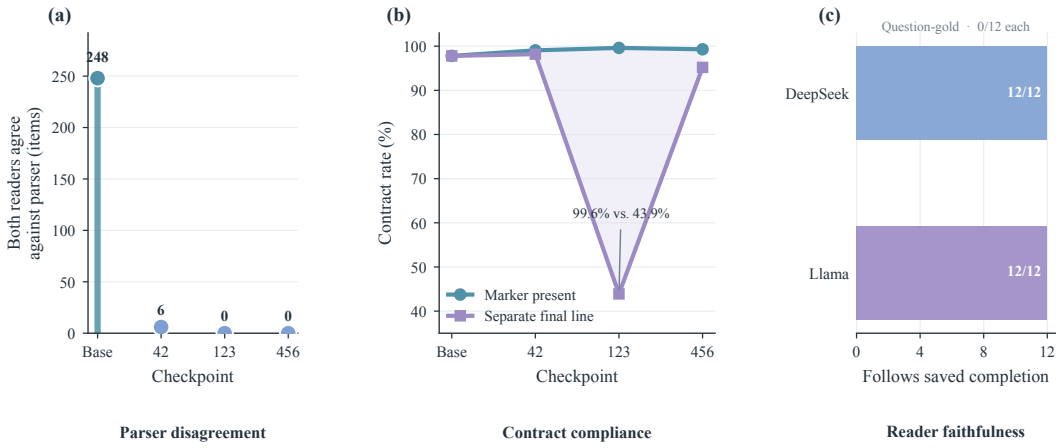


Figure 2: Diagnostics distinguish reader agreement, answer-field availability, and fidelity. The shared parser disagrees with both readers on 248 base items but at most six trained items. Marker presence does not imply final-line compliance. The synthetic 12-case check tests whether readers follow the saved answer, not whether that answer is correct.

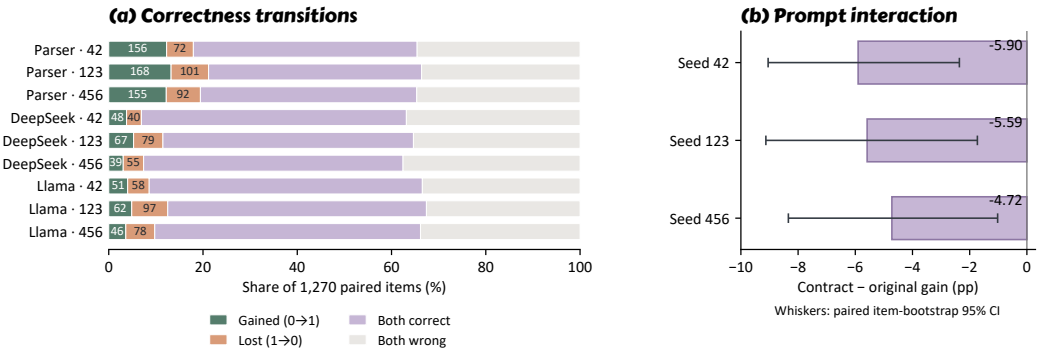


Figure 3: Paired evidence beyond a net score. (a) Correctness transitions for each reader and trained seed on the same 1,270 items; labels are counts in gained and lost segments. A transition is reader-specific, not a semantic adjudication. (b) Contract-minus-original training gains on the separate 1,271-item collection; whiskers are paired item-bootstrap 95% intervals conditional on each checkpoint.

The parser disagrees with both model readers on 248 base items, compared with 6, 0, and 0 items for seeds 42, 123, and 456 (Figure 2). Independent-reader agreement is 96.22% for the base and 98.03–98.66% after training. Together, these observations locate the largest readout mismatch in the baseline. They are consistent with a change in how choices are expressed or recovered, but do not identify a unique causal mechanism.

Figure 3a decomposes each change into item gains and losses. The parser records 156, 168, and 155 newly correct items against 72, 101, and 92 newly incorrect items. DeepSeek instead records gain/loss pairs of 48/40, 67/79, and 39/55. Llama records 51/58, 62/97, and 46/78. Thus, parser gains dominate losses in all three seeds, whereas Llama losses dominate gains in every seed and DeepSeek's balance changes sign. The mean result is not produced by one anomalous parser run.

The transition view also explains why net accuracy alone is incomplete. A near-zero change can contain substantial turnover: DeepSeek seed 123 changes correctness on 146 items but has a net loss of 12. This is distinct from a model making the same decisions before and after training. Reporting both directions makes the item pairing auditable without treating any reader's labels as latent truth.

Table 3: Prompt intervention on 1,271 matched MedQA items. Both prompts use the same checkpoints and 512-token cap. The interaction is paired directly; it is not inferred from overlap between the individual intervals.

| Prompt / contrast | Base | Trained mean | Gain (pp) | 95% CI (pp) |
|---|---|---|---|---|
| Original | 52.79 | 58.67 | $+5.87$ | $[+3.49, +8.18]$ |
| Explain + answer field | 62.00 | 62.47 | $+0.47$ | $[-1.49, +2.44]$ |
| Contract − original | – | – | $-5.40$ | $[-8.37, -2.10]$ |

Table 4: Observed answer forms under the two prompts. Rates are percentages. Original letter-only rates are from the original prompt; the remaining columns are from the contract prompt.

| Checkpoint | Original | Contract compliance | | | Strict |
|---|---|---|---|---|---|
| | letter-only | Marker | Final line | Strict coverage | accuracy |
| Base | 26.51 | 97.80 | 97.80 | 99.61 | 62.63 |
| Seed 42 | 45.95 | 99.06 | 98.19 | 99.13 | 63.49 |
| Seed 123 | 100.00 | 99.61 | 43.90 | 44.06 | 27.22 |
| Seed 456 | 59.64 | 99.29 | 95.20 | 95.59 | 61.29 |

### 5.2 An explicit answer field changes the training contrast

With an explicit answer field, the parser-reported gain falls from 5.87 to 0.47 points (Table 3). The paired interaction is $-5.40$ points, with its interval entirely below zero. The contract improves the base score by 9.21 points but the trained mean by only 3.80 points; the difference between these prompt effects is the same interaction. The interaction therefore measures how the prompt affects the base and trained checkpoints unequally, changing their accuracy contrast.

The pattern holds across all three selected checkpoints (Figure 3b). Original-prompt gains of 6.85, 5.59, and 5.19 points become 0.94, 0.00, and 0.47 points under the contract. The corresponding interactions are $-5.90$, $-5.59$, and $-4.72$ points, and every seed-specific interval excludes zero. These conditional checks establish consistency within the selected set; three seeds are not sufficient to characterize all possible training trajectories.

Table 4 shows why coverage requires several measurements. The contract marker is present in 97.80–99.61% of completions, but seed 123 places it on a separate final line in only 43.90%. Its strict extraction coverage is 44.06%, compared with 99.61% marker presence, and strict accuracy is 27.22%. The permissive parser still reports 62.00% accuracy. A claim of near-perfect contract compliance would therefore be false: answer-field availability is high, while exact placement is checkpoint-dependent.

No contract completion is letter-only, and every marker-bearing completion has nonempty explanatory text before the marker. By contrast, the original-prompt letter-only rate ranges from 26.51% in the base to 100% for seed 123. The contract is consequently a meaningful change in observed response form while retaining explanations. It does not establish equal reasoning content or depth; the same token cap controls the maximum budget, not how the model uses it.

The strict-coverage loss can be separated from accuracy among accepted responses. Let $v_i$ indicate whether strict extraction returns a valid label and let $a_i$ indicate a correct returned label, with $a_i = 0$ when $v_i = 0$. For nonzero coverage,

$$A_{\text{strict}} = \underbrace{\frac{\sum_i v_i}{n}}_{C_{\text{strict}}} \underbrace{\frac{\sum_i a_i}{\sum_i v_i}}_{A_{\text{accepted}}}. \tag{8}$$

This is an exact factorization of the observed metric, not an estimate of accuracy on rejected outputs. It explains why an evaluator can report a large accuracy drop even when the correctness rate among accepted answers changes little.

Table 5: Strict-extraction accounting for the contract prompt, derived from the 1,271-item metrics. Accepted-answer accuracy conditions on different subsets and is diagnostic, not a replacement benchmark score.

| Checkpoint | Accepted | Rejected | Correct | Accepted accuracy (%) |
|---|---|---|---|---|
| Base | 1,266 | 5 | 796 | 62.88 |
| Seed 42 | 1,260 | 11 | 807 | 64.05 |
| Seed 123 | 560 | 711 | 346 | 61.79 |
| Seed 456 | 1,215 | 56 | 779 | 64.12 |

Table 6: Task-level boundary check. ARC values are retained from the existing audit summary; the new item-level transition analysis uses only the bundled MedQA records. Changes and intervals are in percentage points.

| Task | Items | Parser gain | DeepSeek gain [95% CI] | Llama gain [95% CI] |
|---|---|---|---|---|
| MedQA | 1,270 | $+5.62$ | $-0.52\ [-1.71, +0.66]$ | $-1.94\ [-3.33, -0.63]$ |
| ARC-C | 1,169 | $+1.63$ | $+0.09\ [-0.09, +0.31]$ | $+0.09\ [-0.09, +0.29]$ |

Table 5 reconstructs integer counts from the recorded rates. Seed 123 has 346 correct accepted responses out of 560 accepted items: its conditional accuracy is 61.79%, while the full-set strict accuracy is only 27.22%. The latter follows from multiplying 44.06% coverage by 61.79% accepted accuracy. Other checkpoints retain at least 1,215 items, and their accepted-answer accuracies range from 62.88% to 64.12%. The strict score is therefore heavily affected by rejection for this seed.

This decomposition should not be used to rank the checkpoints by conditional accuracy. The accepted subsets differ, and a strict reader can reject difficult items preferentially. Nor does it imply that every rejected response expresses a correct answer. Instead, the diagnostic identifies which denominator is driving the headline score and makes a concrete follow-up possible: adjudicate rejected completions separately from accepted-but-incorrect ones. Reporting the unconditional score, acceptance count, and conditional score together prevents a formatting requirement from silently redefining the evaluation population.

### 5.3 Scope across tasks and complementary controls

ARC-Challenge offers a smaller boundary case (Table 6). Parser accuracy changes from 87.34% to 88.96%, while DeepSeek changes from 89.22% to 89.31% and Llama from 89.56% to 89.65%. Reader agreement stays above 99.5%. Both control intervals contain zero. The contrast is smaller than MedQA and should not be presented as an equally large replication; it shows why extraction dependence should be reported task by task.

The two main audits address different alternatives. Frozen-output rereading excludes different policy generations as the source of cross-reader disagreement. The contract intervention asks whether the training contrast survives an explicit answer boundary while retaining an explanation request. Model-reader error remains possible in the first audit; prompt-induced changes in reasoning remain possible in the second. Their convergence supports interface dependence without requiring either control to be a perfect measure of semantic accuracy.

## 6 Discussion

The main result concerns the unit of evaluation. When a generated answer is scored through a program, the effective object being measured is a policy–prompt–reader system. Reusing the reward extractor at test time is operationally reasonable, but does not provide an independent validator of the interpretation attached to its gain. Our data distinguish a directly observed parser improvement from the stronger claim of improved answer quality under alternative readings.

A practical audit can be staged without retraining. First preserve completions, reference labels, and extracted answers for the base and selected checkpoints. Reread the same strings with a control

whose implementation differs from the reward parser, keeping item pairing explicit. If the change does not survive, investigate baseline disagreement before expanding the seed sweep. Next run a matched output-contract prompt and report the interaction along with coverage. This second step determines whether an explicit deployment interface changes the training contrast.

For benchmark authors, the answer interface should be a versioned artifact alongside the item set. It includes the prompt, decoding budget, accepted output forms, fallback extraction paths, abstention behavior, and reference comparison. Two implementations can share questions and gold labels while evaluating different effective systems. Publishing these details enables an apparent model improvement to be checked against an extraction change or a shift in answer placement.

For trainers, the transition analysis provides a concrete diagnostic. Large net gains under one parser can coexist with balanced or negative transitions under another reader. Logging only aggregate reward loses this distinction. Saving item-level outputs makes post-training audit possible and allows uncertainty to preserve the dependence between a base response and all trained responses to the same question. Independent rereading incurs reader inference, but does not require another policy training run.

A deployment-facing evaluation should additionally specify whether rejection is itself a task failure. If a downstream application requires a valid answer field, unconditional accuracy remains the relevant end-to-end measure, even when rejected text contains an interpretable answer. If the scientific question concerns the answer expressed in prose, a strict-format score alone is incomplete. Keeping these objectives explicit makes the coverage decomposition useful without choosing whichever denominator favors a particular checkpoint. The same saved outputs can support both evaluations when the extraction rules and rejected cases are retained.

The reader-gap decomposition explains why endpoint agreement is insufficient for interpreting a training gain. DeepSeek's accuracy advantage over the parser falls from 6.22 points at the base checkpoint to 0.08 points at the trained mean; the corresponding Llama advantage falls from 9.37 to 1.81 points. These changes account for the difference between the parser and reader gains. An audit must therefore inspect both endpoints. Evaluating only the trained outputs would conceal the baseline readout gap that makes the reported changes diverge, even when the evaluators agree closely after training.

The contract experiment provides a complementary operational lesson. An explicit answer field raises the base score from 52.79% to 62.00%, while the trained mean moves from 58.67% to 62.47%. The negative interaction summarizes these unequal responses to the same prompt intervention. For applications that can specify an answer contract before deployment, this comparison helps identify how much of the measured training advantage remains under that interface. It also motivates reporting prompt-specific base and trained scores together: an isolated trained score does not reveal whether the prompt benefits an already trained policy or primarily improves the starting point.

**Limitations.** Model readers can share errors, and prompts can affect generation. The audits establish interface dependence without assigning semantic ground truth or isolating a pure formatting effect. Estimates and bootstrap intervals are conditional on the evaluated tasks and selected checkpoints. Extending model and task coverage is a natural next step.

Future work can incorporate versioned readers and saved completions into routine evaluation, and use paired audits to test typed answer channels alongside model changes.

## 7 Conclusion

An answer extractor shared by reward and evaluation is part of the optimized system. On frozen MedQA outputs, its $+5.62$-point training gain is not reproduced by two independent readers. Under an explicit answer contract, the parser gain falls from $+5.87$ to $+0.47$ points, with a paired interaction of $-5.40$ points. Item transitions and checkpoint-level coverage show where these differences arise without treating reader agreement as semantic truth. RLVR reports should therefore include independent readings of saved outputs and matched answer-contract checks, with claim strength tied to the evidence that survives both.

## AI USE STATEMENT

AI assistance was used in drafting and editing this article and in preparing analysis code and figures from existing experimental records. All editorial judgement and conclusions are the authors' own. The authors take full responsibility for the research concepts, analyses, and content of this paper.

## ETHICS STATEMENT

This study audits the evaluation of language models on existing multiple-choice benchmarks. MedQA is used as an evaluation dataset, not as evidence of clinical readiness or a source of medical advice. Reader disagreement and parser accuracy should not be used to infer safety in patient-facing settings. The label-only analysis package avoids redistributing benchmark question text and generated medical explanations; reuse of benchmark data and external models remains subject to their applicable licenses and access conditions. The study does not establish the validity of automated readers as substitutes for expert medical judgment.

## REPRODUCIBILITY STATEMENT

The accompanying source includes label-only item records, aggregate audit metrics, the synthetic reader-faithfulness protocol, and scripts that recompute the MedQA accuracies and paired transitions and regenerate the added experimental figure. Section 4 specifies pairing, checkpoint selection, and uncertainty, while Appendix E distinguishes reproducible aggregation from regeneration requiring external data, checkpoints, and model access. These artifacts support inspection of the reported measurements; they are not a complete distribution of the original training and generation environment.

## REFERENCES


Sanghwan Bae, Jiwoo Hong, Min Young Lee, Hanbyul Kim, JeongYeon Nam, and Donghyun Kwak. Online difficulty filtering for reasoning oriented reinforcement learning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 700–719, 2026.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report, 2023.

Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *Advances in Neural Information Processing Systems*, 38:57654–57689, 2025.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training, 2025.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems, 2021.

OpenCompass Contributors. Opencompass: A universal evaluation platform for foundation models, 2023.

Runpeng Dai, Tong Zheng, Run Yang, Kaixian Yu, and Hongtu Zhu. R1-re: Cross-domain relation extraction with rlvr. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 34387–34401, 2026.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

Bowen Ding, Yuhan Chen, Jiayang Lyu, Jiyao Yuan, Qi Zhu, Shuangshuang Tian, Dantong Zhu, Futing Wang, Heyuan Deng, Fei Mi, et al. Rethinking expert trajectory utilization in llm post-training for mathematical reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 33081–33106, 2026.

Kazuki Egashira, Mark Vero, Jasper Dekoninck, Florian E Dorner, Robin Staab, and Martin Vechev. Delay, plateau, or collapse: Evaluating the impact of systematic verification error on rlvr, 2026.

Jun Gao, Huan Zhao, Wei Wang, Changlong Yu, and Ruifeng Xu. Eventrl: Enhancing event extraction with outcome supervision for large language models, 2024.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, et al. A framework for few-shot language model evaluation. *Zenodo*, 2021.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pp. 10835–10866. PMLR, 2023.

Neel Guha, Julian Nyarko, Daniel Ho, Christopher Ré, Adam Chilton, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel Rockmore, Diego Zambrano, et al. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. *Advances in neural information processing systems*, 36:44123–44279, 2023.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Xu Guo, Qiming Ge, Jian Tong, Kedi Chen, Jin Zhang, Xiaogui Yang, Xuan Gao, Haijun Lv, Zhihui Lu, Yicheng Zou, et al. Rethinking multiple-choice questions for rlvr: Unlocking potential via distractor design. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 20092–20113, 2026.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2020.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report, 2024.

Vivekananda Jayaram, Vishnu Ramineni, and Manjunatha Sughaturu Krishnappa. Mitigating order sensitivity in large language models for multiple-choice question tasks. *International Journal of Artificial Intelligence Research and Development (IJAIRD)*, 2024.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b, 2023.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021.

Feiyang Kang, Michael Kuchnik, Karthik Padthe, Marin Vlastelica, Ruoxi Jia, Carole-Jean Wu, and Newsha Ardalani. Quagmires in sft-rl post-training: When high sft scores mislead and what to use instead, 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

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*, volume 2024, pp. 39578–39601, 2024.

Anas Mahmoud, MohammadHossein Rezaei, Zihao Wang, Anisha Gunjal, Bing Liu, and Yunzhong He. Reward hacking in rubric-based reinforcement learning, 2026.

Francesco Maria Molfese, Luca Moroni, Luca Gioffré, Alessandro Scirè, Simone Conia, and Roberto Navigli. Right answer, wrong score: Uncovering the inconsistencies of LLM evaluation in multiple-choice question answering. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 18477–18494, 2025.

Harsha Nori, Yin Tat Lee, Sheng Zhang, Dean Carignan, Richard Edgar, Nicolo Fusi, Nicholas King, Jonathan Larson, Yuanzhi Li, Weishung Liu, et al. Can generalist foundation models outcompete special-purpose tuning? case study in medicine, 2023.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback, 2022.

Pouya Pezeshkpour and Estevam Hruschka. Large language models sensitivity to the order of options in multiple-choice questions. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 2006–2017, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2023.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 8732–8740, 2020.

Mario Sanz-Guerrero, Minh Duc Bui, and Katharina von der Wense. Mind the gap: A closer look at tokenization for multiple-choice question answering with LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 19573–19583, 2025.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *International Conference on Learning Representations*, volume 2024, pp. 25055–25083, 2024.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework, 2024.

Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, et al. Beyond human data: Scaling self-training for problem-solving with language models, 2023.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. Large language models encode clinical knowledge, 2022.

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, 2024.

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

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report, 2025.

Qingchen Yu, Zifan Zheng, Shichao Song, Zhiyu Li, Feiyu Xiong, Bo Tang, and Ding Chen. xFinder: Large language models as automated evaluators for reliable evaluation. In *International Conference on Learning Representations*, 2025a.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244, 2025b.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models, 2024.

Sheng Zhang, Qianchu Liu, Guanghui Qin, Tristan Naumann, and Hoifung Poon. Med-rlvr: Emerging medical reasoning from a 3b base model via reinforcement learning, 2025.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. Large language models are not robust multiple choice selectors. In *International Conference on Learning Representations*, volume 2024, pp. 19426–19454, 2024.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36:55006–55021, 2023.

## A Extended Frozen-Output Evidence

The common MedQA intersection contains 1,270 item identifiers across the base and three trained checkpoints. The incomplete item `medqa_usmle_4opt:2e3c375dc159ed4b355c` is excluded; no label is imputed. Table 7 retains the per-checkpoint accuracies behind Table 2. The parser disagrees with both readers on 248, 6, 0, and 0 items respectively. These disagreement counts are not semantic error labels. For example, two model readers can share the same incorrect interpretation. The paired transition CSV contains the four correctness states for each reader and seed; every row sums to 1,270. The gain reconstructed as $(N_{01} - N_{10})/n$ equals the corresponding difference of accuracies.

Table 7: Per-checkpoint frozen-output accuracies and independent-reader agreement, all in percent.

| Condition | Shared parser | DeepSeek | Llama | Reader agreement |
|---|---|---|---|---|
| Base | 53.15 | 59.37 | 62.52 | 96.22 |
| Seed 42 | 59.76 | 60.00 | 61.97 | 98.03 |
| Seed 123 | 58.43 | 58.43 | 59.76 | 98.66 |
| Seed 456 | 58.11 | 58.11 | 60.00 | 98.03 |
| Trained mean | 58.77 | 58.85 | 60.58 | – |

Table 8: Complete correctness transitions behind Figure 3a. States refer to base $\rightarrow$ trained correctness under the indicated reader. Each row contains the same 1,270 items; the gain is in percentage points.

| Reader | Seed | $0 \rightarrow 0$ | $0 \rightarrow 1$ | $1 \rightarrow 0$ | $1 \rightarrow 1$ | Gain |
|---|---|---|---|---|---|---|
| Parser | 42 | 439 | 156 | 72 | 603 | $+6.61$ |
| | 123 | 427 | 168 | 101 | 574 | $+5.28$ |
| | 456 | 440 | 155 | 92 | 583 | $+4.96$ |
| DeepSeek | 42 | 468 | 48 | 40 | 714 | $+0.63$ |
| | 123 | 449 | 67 | 79 | 675 | $-0.94$ |
| | 456 | 477 | 39 | 55 | 699 | $-1.26$ |
| Llama | 42 | 425 | 51 | 58 | 736 | $-0.55$ |
| | 123 | 414 | 62 | 97 | 697 | $-2.76$ |
| | 456 | 430 | 46 | 78 | 716 | $-2.52$ |

The table exposes offsetting changes hidden by net accuracy. For seed 42, the parser records 156 gains and 72 losses, whereas DeepSeek records 48 gains and 40 losses. Averaging counts across seeds before taking their difference gives 71.33 net parser gains, $-6.67$ DeepSeek gains, and $-24.67$ Llama gains; dividing by 1,270 recovers the reported three-seed effects. The $0 \rightarrow 0$ state denotes two incorrect readings, not necessarily an unchanged answer label.

## B Answer-Contract Details

The contract collection contains 1,271 matched items. Its original-prompt baseline is 52.79%, compared with 53.15% in the frozen-output audit. We treat these as separate generation collections and do not attribute the difference solely to the one-item intersection change. Table 9 reports per-seed interactions. Marker presence, final-line rate, and strict coverage are distinct fields in the archived metrics. The strict extraction rule loses coverage for seed 123, whose 99.61% marker presence coexists with 44.06% strict acceptance and 43.90% final-line placement. These close but unequal rates should not be relabeled as a single predicate. The strict accuracy figures count unaccepted responses as incorrect rather than conditioning on acceptance. All marker-bearing responses contain preceding explanatory text.

Table 9: Seed-specific parser gains and paired interactions, in percentage points.

| Seed | Original gain | Contract gain | Interaction | 95% CI |
|---|---|---|---|---|
| 42 | $+6.85$ | $+0.94$ | $-5.90$ | $[-9.05, -2.36]$ |
| 123 | $+5.59$ | $0.00$ | $-5.59$ | $[-9.13, -1.73]$ |
| 456 | $+5.19$ | $+0.47$ | $-4.72$ | $[-8.34, -1.02]$ |

**Operational format checks.** The matched scoring implementation defines a terminal marker with the case-sensitive expression `Answer:\s*([A-D])\s*$`. It need not start on a separate line. The final-line check uses `(?m)^\s*Answer:\s*[A-D]\s*$` and additionally requires no non-whitespace text after its last match. Thus an explanation ending with an inline `Answer: B` can satisfy the marker check without satisfying the separate-line check. Pre-marker explanation is measured by whether the prefix before the terminal match contains non-whitespace text; this is a text-presence check, not a measure of reasoning quality. Strict coverage instead counts non-null saved `strict_parsed_answer` labels. Its predicate must not be reconstructed from either format check. These definitions are preserved in the supplied 512-token scoring script; the separate 64-token pilot is not the contract collection reported here.

**Which items lose strict acceptance?** Table 10 pairs the acceptance indicators across base and trained completions under the contract. This extends the marginal counts in Table 5: for seed 123, 707 items accepted at base become rejected, whereas only one rejected base item becomes accepted. The coverage change is therefore $(1 - 707)/1271 = -55.55$ points. This identifies the items to inspect in the completion archive without assigning an answer to rejected text.

Table 10: Paired strict-acceptance transitions under the contract, on 1,271 items. Here 1 means accepted, independently of answer correctness.

| Seed | Both rejected | Newly accepted | Newly rejected | Both accepted | Coverage change |
|---|---|---|---|---|---|
| 42 | 0 | 5 | 11 | 1,255 | $-0.47$ pp |
| 123 | 4 | 1 | 707 | 559 | $-55.55$ pp |
| 456 | 2 | 3 | 54 | 1,212 | $-4.01$ pp |

## C Reader Specification and Fidelity

The recorded reader protocol uses the following system instruction:

> You extract the answer actually expressed by a saved multiple-choice response. Do not solve the question or correct the response. Use the question and options only to interpret references in the response. If the response expresses one final choice, return its letter. If it gives no choice, return NO_ANSWER. If multiple final choices remain unresolved, return AMBIGUOUS. Reply with exactly one JSON object, for example `{"choice":"B"}`. Valid values: A, B, C, D, NO_ANSWER, AMBIGUOUS.

The model identifiers are `deepseek-v4-pro` and `llama-3.3-70b-instruct`. The protocol requests a JSON choice, not a rationale. The 12 synthetic fidelity cases have question gold A and explicit completion choice B. Each reader returns B 12 times, A zero times, and invalid output zero times. The test checks fidelity on these cases and cannot calibrate reader correctness on ambiguous benchmark responses.

## D Estimands and Uncertainty

All core estimates pair records by item ID. The frozen-output gain averages the three trained correctness indicators within an item before subtracting the base indicator. The prompt interaction additionally subtracts this difference under the original prompt from the difference under the contract

prompt. Bootstrap resampling operates on complete item vectors, preserving these dependencies. The 4,000-resample percentile intervals are conditional on the selected checkpoints.

The aggregate contract metrics report intervals for both prompt gains and their interaction: original $[+3.49, +8.18]$, contract $[-1.49, +2.44]$, and interaction $[-8.37, -2.10]$ percentage points. These intervals should not be replaced by an independence approximation. The archived headline frozen-parser gain has no reported interval; it is shown descriptively rather than assigned an invented one.

**Paired resampling recipe.** For each contract item, form a vector of eight correctness indicators (four checkpoints under each of two prompts). With $z_{i,s,p}$ denoting these indicators, first compute

$$d_{i,p} = \tfrac{1}{3} \sum_{s \in S} z_{i,s,p} - z_{i,0,p}, \qquad h_i = d_{i,c} - d_{i,o}, \qquad I^{*(b)} = \tfrac{1}{n} \sum_{j=1}^{n} h_{J_j^{(b)}}. \tag{9}$$

For each replicate $b$, draw $n$ indices $J_j^{(b)}$ uniformly with replacement from $\{1, \ldots, n\}$. Reuse the same index draw for both prompt gains and all seed-specific interactions. Take the 2.5th and 97.5th percentiles of 4,000 replicate means and multiply by 100 for percentage points. The supplied matched scorer uses NumPy's `default_rng(20260925)` on sorted item IDs. The added offline appendix audit repeats this procedure directly from the bundled labels and checks the results against the archived intervals.

Resampling complete vectors retains the shared base and the covariance between prompts. Resampling each prompt separately would discard that covariance; treating the three copies of the base as separate examples would instead change the unit of analysis. For a single seed and reader, the transition identity follows directly from $A_0 = (N_{10} + N_{11})/n$ and $A_s = (N_{01} + N_{11})/n$: the stable-correct count cancels, leaving Equation 5. These arithmetic checks also detect inconsistent item joins.

## E Artifact Provenance and Reproduction Boundary

The analysis package includes `medqa_reader_item_scores.csv.gz`, `answer_contract_item_scores.csv.gz`, their aggregate JSON summaries, the recorded reader protocol and fidelity summary, and the analysis and plotting scripts. The new `paired_transitions.csv` is derived directly from the frozen MedQA labels. The manuscript's tables and the added transition figure can be checked without rerunning policy generation.

**Development-only checkpoint selection.** Table 11 expands the manifest behind Section 4. The candidate learning rates have mean selected development accuracies of 57.47% and 57.87%; their 0.40-point difference exceeds the 0.20-point tie tolerance. The selected rate is therefore $2 \times 10^{-5}$, with step 375 for all three seeds. Test-reader and answer-contract outcomes do not enter this selection.

Table 11: Recorded MedQA development accuracies (%) at the candidate checkpoints. The last column applies the within-seed step-selection rule, before comparing learning rates.

| Learning rate | Seed | Step 250 | Step 375 | Selected step |
|---|---|---|---|---|
| $10^{-6}$ | 42 | 56.60 | 57.20 | 375 |
| | 123 | 57.40 | 57.60 | 375 |
| | 456 | 57.60 | 57.40 | 250 |
| $2 \times 10^{-5}$ | 42 | 59.00 | 59.60 | 375 |
| | 123 | 56.00 | 56.20 | 375 |
| | 456 | 57.00 | 57.80 | 375 |

**Offline reproduction order.** From the source directory, run `python code/analyze_core_data.py` for accuracy aggregates, followed by `python code/audit_appendix.py` for the complete correctness and acceptance transitions, checkpoint-selection table, and paired contract intervals. The latter uses NumPy and writes `data/appendix_audit.json`. It checks unique condition–item keys, identical paired item sets, invariant gold labels, row totals, and agreement with stored summary values. Figure regeneration is a separate step using the plotting scripts. The scripts retained under `code/protocol_reference/` document the upstream 512-token generation and scoring definitions; they require the original cluster environment and are not needed for offline aggregation.

The core CSVs contain item identifiers, gold labels, conditions, and extracted outcomes; they do not contain the full questions, options, or generated explanations. Policy weights are also outside this package. Consequently, label aggregation can be rerun locally, while independent extraction or regeneration requires the upstream completion archive, exact extraction implementation, benchmark access, and the original model/checkpoint environment. This distinction prevents label-level reproducibility from being mistaken for complete end-to-end reproduction.

Before an end-to-end rerun, verify the original generation prompts, decoding settings, checkpoint identities and selection manifest, extraction and abstention rules, item joins, 512-token cap, and the definitions of marker presence and strict acceptance. The ARC values in the main text are retained from the prior summary and are not generated by the MedQA-only scripts. No new policy training or new benchmark evaluation was performed to produce the additional transition visualization.