StepGap: A Hybrid NLI-LLM Checker for Step-Level Evidence-Gap Detection in Multi-Hop Question Answering

Anonymous ACL submission

Abstract

We present **StepGap**, a hybrid NLI-LLM decision tree that detects step-level evidence gaps in multi-hop QA and emits one of three typed labels: CONTRADICTED CLAIM (CC), IRRELEVANT EVIDENCE (IE), or MISSING BRIDGE (MB), each tied to a concrete repair action. On 82 multi-hop questions (181 annotated steps, κ=0.704), StepGap reaches sF1=72.0, within the bootstrap confidence interval of an LLM-only baseline (70.1) but with a more decomposable structure: every StepGap stage *hurts* F1 when removed, while three of four LLM-only removals *improve* F1—a sign of *competing-error cancellation*, where internal stages mask each other's errors. We further expose a *Q-F1 trap*: question-level F1 is mechanically inflated by checkers that flag every step, making step-level F1 the necessary diagnostic. Used as a typed GRPO process reward, StepGap improves Qwen2.5-7B-Instruct Exact Match from 32.1±0.3 to 35.4±0.9 across three seeds, with the single-run comparison showing a +5.6 Avg EM gain over the matched Search-R1 GRPO reproduction.

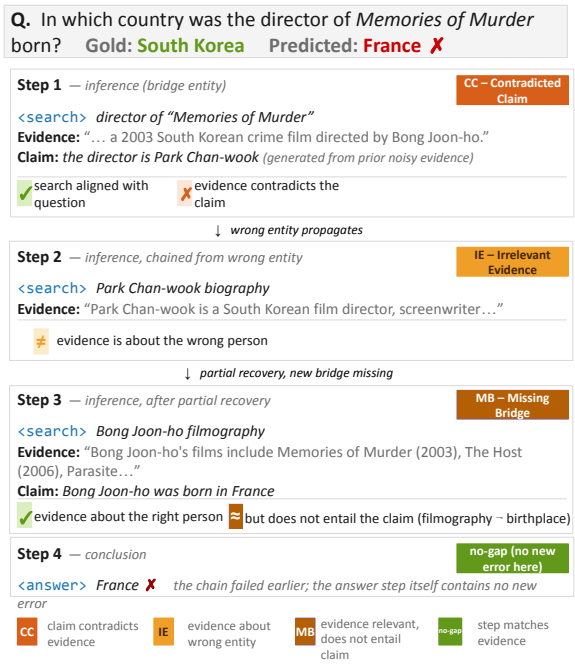

Figure 1: StepGap on a synthetic 4-step trace. Endpoint EM only sees the wrong final answer; StepGap typed-localizes CC, IE, and MB at the steps where each gap first emerges, with each gap type mapped to a distinct repair action.

1 Introduction

Multi-hop question answering (QA) requires retrieving, connecting, and verifying evidence across several intermediate steps (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022). Yet both evaluation and reinforcement-learning training in search-interleaved frameworks remain endpoint-oriented: systems are scored and rewarded on final-answer Exact Match (EM) alone.

Endpoint-only signals have two structural defects. *First*, EM cannot distinguish an evidence-grounded chain from a spuriously correct one: a policy can reach the right answer through shortcuts that bypass retrieval or fill missing facts from parametric memory (Jin et al., 2025; Min et al., 2023). *Second*, EM provides no localization on wrong answers: the policy receives zero reward with no signal about *which* step broke, so it cannot learn step-specific repair behavior (Lightman et al., 2023; Lu et al., 2025). We call such localized step-level failures *evidence gaps*, and distinguish three gap types. A CONTRADICTED CLAIM (CC) occurs when a step asserts a claim that conflicts with the retrieved evidence; IRRELEVANT EVIDENCE (IE) occurs when the evidence does not address the entity or relation needed by the step; and MISSING BRIDGE (MB) occurs when the evidence is rele-

Code, prompts, and diagnostics available at: https://anonymous.4open.science/r/step-checker-0925/

vant but does not yet entail the step's claim. While step-level error taxonomies have been explored before (Deshpande et al., 2025; Lu et al., 2025), our CC/IE/MB types are explicitly designed for *repair-action coupling*: each maps to a single concrete operation (retract / re-search / bridging search) that an RL policy can be trained to perform. Figure 1 illustrates this: three structurally distinct gap types can cascade into a single undifferentiated EM=0—a contradicted claim requiring retraction, irrelevant evidence requiring re-search, and a missing bridge requiring an additional bridging search.

Existing step-level checkers either collapse all subtasks into a single LLM call (Xiong et al., 2025b; Wang et al., 2024) or rely on a single NLI call (Tang et al., 2024; Laurer et al., 2024), neither of which simultaneously emits typed labels and grounds them in deterministic decisions. Post-hoc and claim-level verifiers (Gao et al., 2023; Min et al., 2023; Ru et al., 2024) operate after a complete answer and provide no training-time signal at all. **StepGap** is a five-stage hybrid decision tree: three LLM stages handle open-ended reading subtasks (alignment, abstention, entity consistency and quote extraction), and two NLI stages handle deterministic entailment (local and cross-step). The tree terminates at a typed gap label (CC/IE/MB) coupled to a concrete repair action, and the typed output enters GRPO as a per-step process reward, training the policy to perform the prescribed repair when a gap fires.

Our contribution. We make three claims. **First**, typed step-level gap labels are useful only when they are coupled to repair actions: CC maps to retraction, IE to re-search, and MB to bridging search.

Second, StepGap is more decomposable than an LLM-only checker. On the 181-step human benchmark, StepGap reaches sF1=72.0, within the bootstrap confidence interval of the LLM-only baseline (70.1), but every StepGap component hurts F1 when removed. In contrast, three of four LLM-only removals improve F1, revealing *competing-error cancellation*. We also expose a *Q-F1 trap*, showing why question-level detection must be paired with step-level F1 and balanced accuracy.

Third, the typed signal works as a process reward. A Typed+Shape GRPO reward built on StepGap improves Qwen2.5-7B EM from 32.1 ± 0.3 to 35.4 ± 0.9 across three seeds and raises answer-claim support to 71.5%; a distilled Qwen2.5-7B student retains 87% of teacher F1 with no API calls.

2 Related Work

Search-interleaved reasoning. IRCoT (Trivedi et al., 2023), ReAct (Yao et al., 2023), Search-R1 (Jin et al., 2025), and R1-Searcher (Song et al., 2025) train search policies with final-answer EM rewards and cannot attribute failure to specific steps; on a wrong answer, EM-only reward provides no gradient about which step broke. StepGap supplies a per-step typed signal during rollout, enabling the generator to learn gap-specific repair behaviors rather than a uniform "avoid errors" objective.

Post-hoc verification. RARR (Gao et al., 2023), CRITIC (Gou et al., 2023), and Chain-of-Verification (Dhuliawala et al., 2023) apply verification *after* a complete answer is produced, so they cannot localize mid-chain failures or provide training-time signal. StepGap is invoked inline at each step during rollout, redirecting the generator before the chain propagates a failure.

Claim-level faithfulness. FActScore (Min et al., 2023), MiniCheck (Tang et al., 2024), SelfCheckGPT (Manakul et al., 2023), RAGAS (Es et al., 2023), and RAGChecker (Ru et al., 2024) score whether generated output is supported by retrieved evidence, but do not capture reasoning-chain structure (e.g., whether an intermediate query targeted the right entity); a model trained on output-level faithfulness can still reach correct answers via ungrounded intermediate steps. StepGap evaluates each step against its own evidence, catching failures where they first occur.

Process rewards and agentic RAG. Process Reward Models (Lightman et al., 2023; Wang et al., 2024) provide step-wise scalar supervision, with ReARTeR (Sun et al., 2025) adding MCTS, and RAG-Gym (Xiong et al., 2025b) and ProRAG using LLM-judge process rewards. TRAIL (Deshpande et al., 2025) introduces a trace-level typed error ontology, and Doctor-RAG (Lu et al., 2025) localizes the earliest failing step for post-hoc repair. We extend this line on three axes simultaneously: *step granularity*, *NLI-anchored typed labels*, and *repair-action coupling*. This is a conjunction that is jointly necessary for a typed behavioral reward but has not previously been combined in a single system.

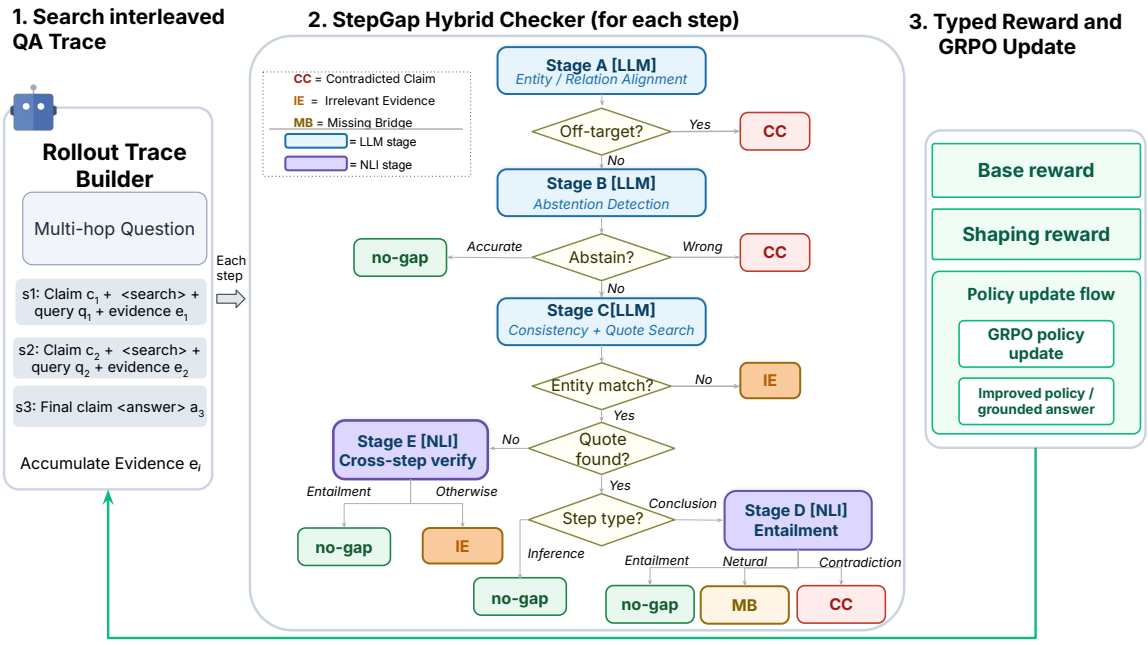

Figure 2: StepGap pipeline. **(1)** The generator produces a search-interleaved trace; each step $s_i = (c_i, q_i, e_i, a_i)$ is fed to the checker with accumulated evidence $e_{\leq i}$. **(2)** The hybrid checker traverses five typed stages (LLM: A alignment, B abstention, C entity+quote; NLI: D local entailment, E cross-step verification) and emits $\tau_i \in \{\text{no-gap}, \text{CC}, \text{IE}, \text{MB}\}$. **(3)** τ_i drives a typed GRPO reward ($r^{\text{base}} + r^{\text{shape}}$, §5), closing the loop back to the generator. CC = Contradicted Claim; IE = Irrelevant Evidence; MB = Missing Bridge.

Hybrid verifiers. NLI–LLM hybrids have been explored for output-level factuality (Xiong et al., 2025a; Liu et al., 2025; Tang et al., 2024; Ji et al., 2025) but not as process-reward signals. Scoring a final answer and scoring an intermediate step are structurally different; we show that an LLM-only checker, even a strong one, collapses to near-uniform IE labels when applied per-step (Appendix I), motivating the hybrid design. To our knowledge, prior work has not jointly demonstrated: (i) hybrid checkers exhibit a healthier ablation profile than LLM-only variants for step-level gap detection, and (ii) such checkers can serve as typed process rewards for search-interleaved RL training.

3 Problem Formulation

Reasoning traces. We model a search-interleaved multi-hop QA trace as an ordered sequence $\mathcal{T} = \{s_1, \ldots, s_n\}$. Each **step** s_i is a tuple $s_i = (c_i, q_i, e_i, a_i)$, where c_i is the model's intermediate reasoning text (a natural-language claim or sub-conclusion produced before the action), q_i is an optional <search> query, e_i is the evidence retrieved by q_i, and a_i is an optional <answer> block emitting the final answer $\hat{a}$. We call s_i an *inference step* when $q_i \neq \varnothing$ and $a_i = \varnothing$ (the model issues a search and writes a sub-claim), and a *conclusion step* when $a_i \neq \varnothing$ (the model commits to the final answer). The checker takes the entire tuple plus all earlier evidence as input, so its judgment depends jointly on the intermediate claim, the search query, the retrieved snippets, and the answer if present.

Throughout, **step** refers to a unit of the reasoning trace (s_i); **stage** refers to a node in the checker's decision tree (Stages A–E, indexed by letters to visually separate them from s_i). One checker invocation traverses up to five stages on a single reasoning step.

Step-level evidence gap detection. Step s_i is *gap-free* if its claim—c_i when s_i is an inference step, a_i when s_i is a conclusion step—is logically entailed by the accumulated evidence $\bigcup_{j \leq i} e_j$. Otherwise, the step has a *gap*, characterized by gap type $\tau \in \{\text{CC}, \text{IE}, \text{MB}\}$. The checker $\mathcal{C}$ maps each $(s_i, e_{\leq i})$ to a structured output $(\tau, \textit{confidence}, \textit{rationale})$. The five stages of $\mathcal{C}$ are described in Section 4.

4 Hybrid Step-Gap Checker: Decision Tree

For each step $s_i = (c_i, q_i, e_i, a_i)$ with accumulated evidence $e_{\leq i}$, the checker traverses up to five stages

Table 1: Stages of the StepGap decision tree, the gap type each can emit, and the NLI-grounded typed signal at that node. Stages D–E are NLI; A–C are LLM. Each gap type maps to a single repair action used downstream as the typed reward signal (§5).

Stage	Type	Signal	Repair
A (LLM)	CC	alignment fail	retract
B (LLM)	CC	wrong abstention	retract
C (LLM)	IE	entity mismatch	re-search
D (NLI)	CC	contradiction	retract
D (NLI)	MB	neutral, quote found	bridging search
E (NLI)	IE	no entailing prior	re-search

(Figure 2, panel 2). Each stage either emits a typed gap label and stops, or hands off to the next stage; if no stage fires, the step is labeled no-gap. We organize this section by traversal order (Stage A through Stage E) so that the tree's structure is visible end-to-end; Table 1 summarizes which stages can emit which gap type. Below, *the step's claim* means c_i when s_i is an inference step and a_i when s_i is a conclusion step. The search query q_i is consulted only at Stage A and is never used as an NLI hypothesis.

Stage A – Alignment. Stage A checks whether the step targets the correct entity and relation, *before* retrieval is consulted. This matters because a mis-targeted step cannot be rescued by retrieval—any fetched document is irrelevant by construction, and surface-token overlap can masquerade as support. We classify mis-targeting into three drift patterns—relation drift (wrong predicate), entity drift (wrong subject), and scope drift (sub-part vs. whole)— and emit CC on any drift.

Stage B – Abstention validity. Stage B handles steps that assert "cannot be determined," verifying that the abstention is grounded in the evidence. An unjustified abstention is functionally a contradicted claim: the step commits to "the answer is unknown" when the evidence rules that out. A grounded abstention exits to no-gap; a wrong abstention emits CC.

Stage C – Entity consistency and quote. Stage C verifies that the retrieved evidence is about the targeted entity and, if so, extracts a verbatim supporting quote. Without this filter, off-topic documents sharing lexical tokens with the claim would pass to NLI as spurious premises. If the page's title and lead sentence describe a different entity, Stage C emits IE directly; otherwise the checker finds a 5–20 word verbatim span on the same entity, relevant to the claim. A missing quote routes a conclusion step to Stage E and an inference step to no-gap (a search query is not a well-formed NLI hypothesis).

Stage D – Local entailment. Stage D decides whether the Stage C quote actually entails the step's claim. Entity match plus a returned quote is necessary but not sufficient: the quote can be on-topic yet fail to imply the claim (a missing-bridge case) or overtly contradict it. A single NLI call with the quote as premise and the claim as hypothesis resolves this: ENTAILMENT → no-gap; NEUTRAL → MB (*repair: bridging search*); CONTRADICTION → CC.

Stage E – Cross-step verification. Stage E handles conclusion steps where Stage C found no local quote, asking whether the claim is entailed by evidence retrieved at earlier inference steps. Multi-hop answers typically rest on facts established several steps earlier, so the effective evidence pool is the union of all prior retrievals. Stage E iterates over entity-matched prior snippets as candidate premises: an entailing snippet exits to no-gap; otherwise the step emits IE (*repair: re-search*). The entity-match constraint prevents off-topic earlier retrievals from producing spurious entailments.

5 Using the Typed Signal as a Process Reward

A checker that only *detects* gaps does not change policy behavior: the model can detect a gap at step i and still commit at step $i{+}1$. We inject the typed output into RL training as a per-step process reward so the policy is reinforced to execute the prescribed repair when a gap fires. The reward has two parts: a base scalar conditioned on gap type (§5.1) and an action-conditional shaping term that verifies the repair is performed (§5.2); both plug into GRPO unmodified (§5.3).

5.1 Base Reward by Gap Type

Let $\tau_i \in \{\text{no-gap}, \text{MB}, \text{IE}, \text{CC}\}$ be the gap type emitted by the checker for step s_i. The base reward is

$$r^{\text{base}}(\tau_i) = \begin{cases} +0.20 & \text{no-gap} \\ -0.05 & \text{MB} \\ -0.10 & \text{IE} \\ \mathbf{+0.05} & \text{CC detected} \end{cases} \tag{1}$$

Three choices are non-obvious. *No-gap is positive* (+0.20), following standard process-reward practice (Lightman et al., 2023; Wang et al., 2024; Xiong et al., 2025b; Wu et al., 2025b). *CC receives a small detection bonus* (+0.05): the reward is assigned after the checker detects a contradiction, not to the contradicted claim itself. Penalizing all detected CC labels would instead teach the generator to hide contradictions behind vaguer claims that fewer NLI entailment checks can reject (Jiang et al., 2025). The bonus is kept small so it does not compete with EM and only rewards exposing a contradiction that the next-step shaping term can require the policy to retract. *IE is penalized more than MB* (−0.10 vs. −0.05): IE requires reformulating the query, MB only a bridging search; the asymmetry pushes the generator toward better initial queries.

5.2 Behavioral Shaping Reward

We add a *shaping reward* conditional on the repair action at step s_i given the gap type at step s_{i-1}, following the action-conditional design of **RAG-Gym** (Xiong et al., 2025b) and the process-reward formulation of (Zhang et al., 2025).

$$r^{\text{shape}}(\tau_{i-1}, s_i) = \begin{cases} +0.10 & \text{IE/MB}_{i-1} \wedge \texttt{<search>}_i \wedge q_i \neq q_{<i} \\ +0.15 & \text{CC}_{i-1} \wedge \text{retract}_i \\ -0.15 & \text{IE/MB/CC}_{i-1} \wedge \texttt{<a>}_i \\ -0.05 & \text{IE/MB}_{i-1} \wedge \texttt{<search>}_i \wedge q_i \approx q_{<i} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Here q_i denotes the query string of a search step; we consider $q_i \approx q_{<i}$ if its token-F1 against any prior query exceeds 0.7—a threshold selected on the same 200-iteration pilot used for base reward weights (Appendix H). Retraction is detected by a small set of surface patterns ("actually", "wait", "correction", "I was wrong"), combined with a check that the new claim differs from the previous one.

The shaping reward r^{shape} defined above implements the principle "detection without repair is failure": ignoring the checker after a gap costs −0.15 (a step-level analogue of Reflexion's self-correction (Shinn et al., 2023)), executing the repair earns +0.10 (re-search) or +0.15 (retraction; CC's higher shaping bonus reflects the larger behavioral cost of retracting a contradicted claim), and lazy near-duplicate queries earn −0.05 to prevent gaming the shaping signal (cf. (Wu et al., 2025a)).

5.3 Plugging the Typed Reward into GRPO

The checker enters GRPO (Shao et al., 2024) only at the return-computation step. During rollout, $\mathcal{C}$ is invoked once per emitted step s_i, returning τ_i, which is cached with the step's token span $[\ell_i, r_i)$. The trajectory return is

$$R = r^{\text{EM}} + \lambda \sum_{i=1}^{n} \left[r^{\text{base}}(\tau_i) + r^{\text{shape}}(\tau_{i-1}, s_i) \right], \tag{3}$$

with $r^{\text{EM}} \in \{0, 1\}$, $\lambda = 1.0$, and $\tau_0 =$ no-gap at the boundary. We use the *dense* variant: each step's bracket is assigned only to tokens in $[\ell_i, r_i)$, and the group-relative standardization $\hat{A} = (R - \mu_R)/\sigma_R$ is computed token-wise. The clip ratio, KL penalty, and standardization are kept at GRPO defaults; only the return R in Eq. (3) differs across our reward ablations, so any EM gain attributes to the typed checker signal and shaping rather than to optimizer re-tuning.

6 Checker Evaluation

6.1 Human-Annotated Evaluation Set

We sample 82 questions from HotpotQA, 2WikiMultiHopQA, and MuSiQue, run Qwen2.5-7B-Instruct to produce full search- interleaved traces, and manually annotate 181 individual steps (see Appendix B for protocol). Each step is labeled as no-gap or gap (with type ∈ {CC, IE, MB}). Two annotators independently label the test subset. Inter-annotator agreement is Cohen's $\kappa = 0.704$, which we interpret as moderate agreement per McHugh (2012), consistent with the inherently subjective nature of distinguishing MB from IE near the decision boundary. Disagreements were resolved through discussion between the two annotators until consensus was reached on each disputed step; the post-discussion consensus serves as the ground-truth label set used in all evaluations (full protocol in Appendix B).

Per-type κ (reported in Appendix B) is highest for IE and CC, and lower for MB, where MB's low frequency (n=9) inflates sampling variance. Table 2 reports dataset composition.

Silver-standard scale check. To complement the 181-step gold evaluation set, we measure StepGap's agreement with the GPT-4.1-mini+NLI teacher checker on the 1,000-step distillation pool (Appendix C, disjoint from the human benchmark). Treating the teacher checker's labels as a *silver-*

Table 2: Human-annotated evaluation set statistics. Gap% = fraction of steps labeled as gap; Acc% = question-level accuracy. GRPO training uses 169K NQ+HotpotQA questions (`PeterJinGo/nq_hotpotqa_train`); standard dev splits: HotpotQA 7,405 Q, 2Wiki 12,576 Q, MuSiQue 2,417 Q (Appendix P.1).

Dataset	#Q	#Steps	Gap%	Acc%
2WikiMultiHopQA	29	66	62	21
HotpotQA	23	53	57	17
MuSiQue	30	62	58	10
Total	**82**	**181**	**59**	**16**

standard reference—i.e., a high-quality but model-generated label set that substitutes for human gold labels when annotation scale is impractical—StepGap reaches F1=0.88 on the binary `has_gap` dimension and F1=0.81 on typed CC/IE/MB. This high agreement at $\sim 5\times$ the gold-evaluation-set scale indicates that StepGap's behavior on the 181-step human evaluation set is representative of its behavior at scale, not a sampling artifact.

We compare StepGap against four internal variants **LLM-Strict** and **LLM-Only**: GPT-4.1-mini at every stage, differing only in the gap-flagging threshold (**LLM-Strict**: any single stage-level gap signal suffices; **LLM-Only**: requires the checker's overall confidence ≥ 0.5); **Single-LLM-XL**: GPT-5-mini at every stage, no NLI; **NER-Quote**: spaCy NER replacing Stage C's LLM quote search) and two external baselines (**MiniCheck** (Tang et al., 2024); **NLI-only** (Laurer et al., 2024): StepGap's DeBERTa-v3 applied directly to (full evidence, claim) without LLM stages). We also considered answer-level faithfulness checkers (RAGAS (Es et al., 2023), FActScore (Min et al., 2023), RAGChecker (Ru et al., 2024)), but these score *complete answers* rather than intermediate steps and are therefore not directly comparable as step-level checkers; among them MiniCheck is the closest step-applicable representative and is included as a baseline. Table 3 reports performance with bootstrap 95% CIs. We report both step-level F1 (sF1: per step, does `has_gap` match the human label?) and question-level F1 (Q-F1: on each wrong-answer question, did the checker detect any gap?), sF1 paired with balanced accuracy (BA) is the more diagnostic of the two: Q-F1 fires whenever *any* gap is detected on a wrong-answer question, so a checker that flags every step trivially achieves near-ceiling Q-F1 without distinguishing real gaps from spurious ones. sF1, in contrast, penalizes per-step false positives, and BA further corrects for the $\sim 60\%$ gap-rate class imbalance—together they capture both detection coverage and precision.

Statistical interpretation. *(1)* StepGap and LLM-Only are statistically indistinguishable on top-line sF1 (CIs overlap at n=181). Our argument for StepGap instead rests on *compositional* ablation properties (Appendix D): every StepGap component removal *hurts* F1, meaning each stage contributes real signal that the rest of the system cannot replace—the checker is *decomposable*. By contrast, three of four LLM-only component removals *improve* F1, meaning those components were actively introducing errors that other components had been masking; this is *competing-error cancellation*, and the checker is not decomposable. *(2)* Single-LLM-XL and NER-Quote expose what we call a **Q-F1 trap**: Q-F1 fires whenever *any* step is flagged on a wrong-answer question, so a checker flagging every step achieves near-ceiling Q-F1 mechanically. Both variants flag $\sim$95% of steps and reach Q-F1=91.4 while sF1 collapses to ≤ 69 and BA to chance (50%). A flag-everything checker reaches Q-F1= $2w/(1+w)$ in closed form, giving $0.67/0.78/0.91$ at w=$0.5/0.65/0.84$; a dummy reproduces this within ± 0.02 on four benchmarks (Appendix L). Q-F1 alone cannot distinguish a precise checker from one that always fires. *(3)* The three metrics must be read jointly: Q-F1 is inflated by flag-everything (point 2), sF1 can mask class imbalance (a no-gap predictor scores ~ 60 at our 59% gap rate), and BA alone ignores precision. StepGap is the only variant competitive on all three (sF1=72.0, Q-F1=81.2, BA=60.6%); Single-LLM-XL wins Q-F1 but loses sF1 and BA, falling into the trap.

Our claim is not that StepGap outperforms LLM-only on sF1 alone, but that its component ablations are *monotone*: every removal hurts F1. This matters because a system whose components consistently add signal admits modular debugging and distillation (Appendix C). LLM-only's ablations swing oppositely ($+1.6/+0.8/+1.7$ vs -6.5), so stage effects are entangled.

6.2 Category Distribution as a Health Indicator

The distribution of emitted gap types diagnoses *checker health*, orthogonal to sF1: a checker can score moderately on sF1 yet be useless as a

Table 3: Step-level performance on the 181-step evaluation set. sF1/sP/sR: step-level F1, precision, recall. Q-F1: a question is TP iff the checker detects ≥ 1 gap on a wrong-answer chain. BA: balanced accuracy. † degenerate variants whose Q-F1 is mechanically inflated by flagging nearly every step as IE (§6.2). ‡ binary checkers without typed CC/IE/MB output. Brackets: 2000-iter percentile bootstrap 95% CIs.

Variant	sF1 [95% CI]	sP	sR	Q-F1	BA	Note
Internal variants (full typed CC/IE/MB output)						
LLM-Strict	67.2 [60.4, 73.8]	60.3	76.0	89.2	47.8	high FP, recall-heavy
LLM-Only	70.1 [63.5, 76.2]	69.2	71.2	81.8	44.4	competing-error baseline
StepGap (ours)	**72.0** [65.6, 77.8]	**71.0**	73.1	81.2	60.6	best P/R balance, healthy ablation
Degenerate alternatives (full output but collapsed onto IE)						
Single-LLM-XL	69.2 [63.0, 75.1]	57.2	87.5	91.4†	50.0†	all flagged; IE=87.8%, CC=0%
NER-Quote	68.0 [61.5, 74.2]	56.8	84.6	91.4†	50.0†	NLI fires on noun phrases
Binary-only baselines (no typed gap output)‡						
NLI-only (Laurer et al., 2024)	61.7 [54.3, 68.2]	63.3	53.5	75.2	49.7	no LLM stages; pure entailment
MiniCheck (Tang et al., 2024)	67.3 [58.3, 73.2]	61.2	72.1	90.4	51.5	external published checker

typed reward source if it cannot reliably distinguish CC/IE/MB. LLM-Only and StepGap both produce IE in the 38–41% range, consistent with retrieval failure being the dominant mode in annotation. We treat this 38–41% range as a benchmark-specific sanity check rather than a universal target distribution. The two degenerate variants leave this band entirely. Single-LLM-XL collapses to 87.8% IE with zero CC and zero MB: the strongest single LLM yields a *less* discriminating checker. NER-Quote reaches 74% IE because spaCy noun-phrase queries are not well-formed NLI hypotheses, so Stage D over-reports retrieval failure. Both pathologies are threshold-independent (Appendix N), confirming the collapse is architectural. A confusion-matrix view (Appendix J) shows StepGap flags a gap on $56/69$ wrong-answer questions and $12/13$ correct-answer questions, exposing the *lucky-correct* regime endpoint EM cannot see; the full first-gap distribution is in Appendix K.

7 GRPO Training with Typed Reward

7.1 Setup

A high-quality checker does not by itself train a policy to fix the gaps it detects. We therefore test whether StepGap's typed outputs work as a per-step process reward, holding backbone, training mix, and hyperparameters constant. Only checkers emitting typed CC/IE/MB qualify: StepGap (ours) and LLM-Only; binary support-checkers (MiniCheck, NLI-only) are excluded by design and remain evaluated on checker quality alone (§6).

We train Qwen2.5-7B-Instruct with GRPO (Shao et al., 2024) on NQ+HotpotQA (Table 2); retrieval uses E5-base-v2 (Wang et al., 2022) over Wikipedia 2018 (top-k=5). The teacher checker (GPT-4.1-mini) is called ∼72K times per 3,000-iteration run at ∼$22; the distilled 7B student (Appendix C) eliminates this cost. We compare four reward configurations: **Search-Only** (EM only), **Binary Gap** (+0.2 no-gap / 0 gap; prior design), **Typed Base**, and **Typed+Shape (ours)**.

7.2 Main Results

Table 4 reports per-dataset and Avg EM. The headline result is Typed+Shape's 35.4 Avg EM, a +5.6 gain over our directly comparable Search-R1 GRPO baseline (29.8, same backbone and training mix), with the gain holding across all three datasets. Three-seed mean±std and the Llama-3.1-8B backbone-transfer study are in Appendix A; auxiliary grounding metrics in Appendix M.

Reward ablation. We note that 2Wiki and MuSiQue are out-of-domain w.r.t. our NQ+HotpotQA training mix while HotpotQA is in-domain; the consistent Typed+Shape gain on both suggests the typed signal generalizes beyond the training distribution.

Concretely, in the single-run macro-average setting, Typed+Shape improves EM by +2.4 over Binary Gap (typed labels matter) and by +1.2 over Typed Base (behavioral shaping matters). Without shaping, the generator can detect a gap yet still commit to an answer; with shaping, mean search count rises from 2.5 to 2.8, consistent with the policy learning to re-search after IE/MB detections. The three-seed Qwen setting (Appendix A, Table 5) confirms the same monotone ordering, with Typed+Shape improving over Search-Only by +3.3 EM. On strictly out-of-domain MuSiQue

Table 4: GRPO results on standard dev splits. **Top**: published baselines comparable to ours (same Qwen2.5-7B-Instruct backbone). **Middle**: external systems with non-comparable training setup (different base or different training mix). **Bottom**: our reward ablations on the same NQ+HotpotQA training mix and Qwen2.5-7B-Instruct backbone as Search-R1. The directly comparable row is Search-R1 (GRPO); **Typed+Shape gains** $+5.6$ **Avg EM over it** ($29.8 \rightarrow 35.4$). Bottom-panel "Search-Only" is the single-seed Search-R1 GRPO reproduction; three-seed mean$\pm$std for all our ablations (where Search-Only reaches 32.1 ± 0.3) is in Appendix A, Table 5.

Method	Backbone	Train mix	HQA	2Wiki	MuSiQue	Avg
Same backbone (Qwen2.5-7B-Inst), comparable to ours						
Naive RAG	7B-Inst	—	29.9	23.5	5.8	19.7
IRCoT	7B-Inst	—	13.3	14.9	7.2	11.8
SFT	7B-Inst	NQ+HQA	21.7	25.9	6.6	18.1
Search-R1 (PPO)	7B-Inst	NQ+HQA	37.0	41.4	14.6	31.0
Search-R1 (GRPO)	7B-Inst	NQ+HQA	38.6	34.6	16.2	29.8
Different setup (non-comparable backbone or training mix)						
R1-Searcher[‡]	7B-**Base**	**HQA+2Wiki**	44.2	51.3	15.8	37.1
StepSearch[†]	7B-Inst	MuSiQue	34.8	36.3	13.4	28.2
GlobalRAG[†]	7B-Inst	2Wiki+HQA	38.2	47.8	14.9	33.6
Our reward ablations: same setup as Search-R1 (Qwen2.5-7B-Inst, NQ+HQA)						
Search-Only	7B-Inst	NQ+HQA	38.6	34.6	16.2	29.8
Binary Gap	7B-Inst	NQ+HQA	36.6	39.0	24.6	33.4
Typed Base	7B-Inst	NQ+HQA	37.8	40.5	25.5	34.6
Typed+Shape (ours)	7B-Inst	NQ+HQA	**39.1**	**41.5**	**26.7**	**35.4**

[†] Reproduction numbers as reported in (Luo et al., 2025) on the standard Search-R1 dev splits. [‡] R1-Searcher reports Cover-EM in its own paper; the strict-EM numbers shown here are an independent reproduction by (Tan et al., 2025). R1-Searcher trains on HotpotQA+2Wiki, so HotpotQA and 2Wiki are in-domain for it; for our NQ+HotpotQA mix, 2Wiki and MuSiQue are out-of-domain.

(neither system trained on it), Typed+Shape outperforms R1-Searcher by $+10.9$ EM (26.7 vs. 15.8); the latter's higher Avg comes from its HotpotQA+2Wiki in-domain training data rather than a stronger reward design.

Two-phase CC dynamics. The CC detection bonus produces a characteristic non-monotone rate during training (Figure 3): CC rises early as the generator surfaces contradictions for the bonus, then falls as it learns to avoid making contradicted claims altogether. A straight-negative CC reward decreases CC monotonically but achieves it through vaguer, harder-to-check claims that hurt support rate; the positive bonus prevents this contradiction-suppression failure mode.

Llama backbone transfer. On Llama-3.1-8B under identical hyperparameters, all checker-augmented rewards underperform Search-Only (28.1): Binary Gap collapses to 19.2, Typed+Shape reaches 25.4. We attribute this to backbone-specific interaction with our Qwen-tuned reward magnitudes (see Limitations) and treat Qwen as the headline; the Untyped-Dense control is in Appendix P.2.

8 Conclusion

We presented **StepGap**, a hybrid NLI-LLM checker that emits typed step-level evidence-gap labels (CC/IE/MB), each anchored at a specific NLI decision and tied to a concrete repair action. Used as a typed GRPO process reward, StepGap improves Qwen2.5-7B-Instruct EM by $+3.3$ across three seeds. Typed+Shape obtains 71.5% answer-claim support, the highest among our reward ablations; a distilled Qwen2.5-7B student preserves 87% of the teacher's step-F1 without API calls at inference. Methodologically, we hope to leave the reader with the view that two checkers within each other's F1 CIs can still expose qualitatively different process signals—one through monotone components, the other through competing-error cancellation—and only the former is usable as a typed RL reward.

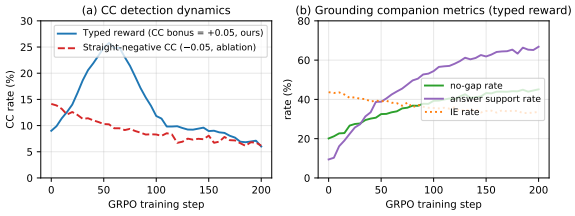

Figure 3: Two-phase CC dynamics during 200 GRPO iterations. **(a)** CC rate under our typed reward (blue, +0.05 bonus) vs. a straight-negative CC ablation (red dashed). **(b)** Companion grounding metrics (no-gap rate, answer support rate, IE rate) under the typed reward.

Limitations

Sample size and statistical power. Our human-annotated evaluation set covers 181 steps across 82 questions; bootstrap 95% CIs on sF1 overlap between StepGap, LLM-Only, and several baselines (Table 3). We therefore frame our contribution on the architectural and diagnostic axes (*healthy ablation pattern*, *Q-F1 trap*, *typed signal*) rather than as a top-line F1 win, and we report seed-level variance in Appendix A (Table 5). As partial mitigation of the small gold sample, we additionally report StepGap-vs-teacher agreement on the 1,000-step distillation pool (§6, binary F1=0.88, typed F1=0.81), showing that StepGap's binary gap-detection behavior is consistent at $\sim 5\times$ the gold-benchmark scale. Low-frequency gap types (MB, n=9) remain the main power-limited case, and a larger MB-targeted gold evaluation set is left to future work. We note, however, that our 181-step scale is comparable to step-level annotation in the agentic-RAG literature, and the silver-standard 1,000-step check addresses the architectural-level concern at $\sim 5\times$ scale.

Backbone-specific reward tuning. The reward magnitudes (§5) were tuned on Qwen2.5-7B and do not transfer zero-shot to Llama-3.1-8B (Table 5); we hypothesize interaction between dense per-step shaping and Llama's instruction-following distribution under a fixed KL budget, and leave per-backbone retuning to future work.

Other limitations. The shaping term relies on surface pattern matching for retraction detection (e.g., "actually", "wait"), which may miss subtler rewordings and, conversely, is in principle vulnerable to reward hacking via spurious retraction-pattern usage. We did not observe such hacking in our 200-iteration runs, but longer training could enable this failure mode and motivates a learned retraction classifier in future work.

Ethics Statement

StepGap is a diagnostic checker for multi-hop QA traces; it produces typed gap labels (CC/IE/MB) that can be consumed either as evaluation diagnostics or as a process reward during RL training. Like any automatic verifier, StepGap can be wrong: false negatives may let ungrounded reasoning slip through, and false positives may incorrectly penalize correct chains. We therefore recommend StepGap as a debugging and training aid rather than as a standalone correctness filter for downstream deployment.

All datasets used for evaluation and GRPO training (HotpotQA, 2WikiMultiHopQA, MuSiQue, NaturalQuestions) are publicly released for research use under their original licenses (see Appendix B). The Wikipedia 2018 retrieval corpus follows the Wikipedia text license. No private or personally identifiable content is processed.

Our human annotation (Appendix B) was performed by two graduate-student co-authors as part of standard research duties; no external crowdsourcing was used. We release the StepGap checker code, the distilled student checkpoint, and the 181-step evaluation benchmark under CC BY 4.0 for non-commercial research use.

As a process-reward source, StepGap could in principle be gamed by a generator that learns to emit surface patterns (e.g., the retraction keywords in r^{shape}) without genuinely repairing the underlying reasoning gap. We did not observe this failure mode in our 200-iteration GRPO runs, but flag it as a concern for longer or larger-scale training; a learned retraction classifier is a natural next step.

Acknowledgments

Use of Generative AI. GPT-4.1-mini is used as a methodological component of this work, serving as the LLM backbone for the alignment, abstention, and entity-consistency/quote stages (Stages A–C) of the StepGap checker and as the teacher in our distillation experiments; the full mapping of models to pipeline stages is documented in Section 4 and Appendix O. We also used general-purpose LLM assistants for light editing and polishing of prose; all technical content, experimental design, results, and analyses are the authors' own.

References

Darshan Deshpande, Varun Gangal, Hersh Mehta, Jitin Krishnan, Anand Kannappan, and Rebecca Qian. 2025. Trail: Trace reasoning and agentic issue localization. *arXiv preprint arXiv:2505.08638*.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2023. Chain-of-verification reduces hallucination in large language models. *arXiv preprint arXiv:2309.11495*.

Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. 2023. RAGAS: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217*.

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Y. Zhao, Ni Lao, Honglak Lee, Da-Cheng Juan, and Kelvin Guu. 2023. RARR: Researching and revising what language models say, using language models. In *Proceedings of ACL*.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. CRITIC: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of COLING*.

Yuelyu Ji, Wuwei Lan, and Patrick NG. 2025. Mragsuite: A diagnostic evaluation platform for visual retrieval-augmented generation. *arXiv preprint arXiv:2509.24253*.

Jinhao Jiang, Jiayi Chen, Junyi Li, Ruiyang Ren, Shijie Wang, Wayne Xin Zhao, Yang Song, and Tao Zhang. 2025. Rag-star: Enhancing deliberative reasoning with retrieval augmented verification and refinement. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7064–7074.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Moritz Laurer, Wouter van Atteveldt, Andreu Casas, and Kasper Welbers. 2024. Less annotating, more classifying: Addressing the data scarcity issue of supervised machine learning with deep transfer learning and bert-nli. *Political Analysis*.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Weisi Liu, Guangzeng Han, and Xiaolei Huang. 2025. Examining and adapting time for multilingual classification via mixture of temporal experts. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6151–6166.

Yuxing Lu, Gecheng Fu, Wei Wu, Xukai Zhao, Sin Yee Goi, and Jinzhuo Wang. 2025. Doctorrag: Medical rag fusing knowledge with patient analogy through textual gradients. *arXiv preprint arXiv:2505.19538*.

Jinchang Luo, Mingquan Cheng, Fan Wan, Ni Li, Xiaoling Xia, Shuangshuang Tian, Tingcheng Bian, Haiwei Wang, Haohuan Fu, and Yan Tao. 2025. Globalrag: Enhancing global reasoning in multi-hop question answering via reinforcement learning. *arXiv preprint arXiv:2510.20548*.

Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. 2023. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*.

Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica*, 22(3):276–282.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of EMNLP*.

Dongyu Ru, Lin Qiu, Xiangkun Gu, Tianhao Zhao, Shuaichen Ding, Xinyu Jia, Pengfei Shi, Yike Yu, Qipeng Zheng, Mingye Geng, and 1 others. 2024. RAGChecker: A fine-grained framework for diagnosing retrieval-augmented generation. In *Advances in NeurIPS Datasets and Benchmarks*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*.

Zhongxiang Sun, Qipeng Wang, Weijie Yu, Xiaoxue Zang, Kai Zheng, Jun Xu, Xiao Zhang, Yang Song, and Han Li. 2025. Rearter: Retrieval-augmented reasoning with trustworthy process rewarding. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1251–1261.

Zhiwen Tan, Jiaming Huang, Qintong Wu, Hongxuan Zhang, Chenyi Zhuang, and Jinjie Gu. 2025. Rag-r1: Incentivizing the search and reasoning capabilities of llms through multi-query parallelism. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 33187–33195.

Liyan Tang, Philippe Laban, and Greg Durrett. 2024. Minicheck: Efficient fact-checking of llms on grounding documents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8818–8847.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 10014–10037.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. 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)*, pages 9426–9439, Bangkok, Thailand. Association for Computational Linguistics.

Jinming Wu, Zihao Deng, Wei Li, Yiding Liu, Bo You, Bo Li, Zejun Ma, and Ziwei Liu. 2025a. Mmsearch-r1: Incentivizing lmms to search. *arXiv preprint arXiv:2506.20670*.

Peilin Wu, Mian Zhang, Kun Wan, Wentian Zhao, Kaiyu He, Xinya Du, and Zhiyu Chen. 2025b. Hiprag: hierarchical process rewards for efficient agentic retrieval augmented generation. *arXiv preprint arXiv:2510.07794*.

Guangzhi Xiong, Zhenghao He, Bohan Liu, Sanchit Sinha, and Aidong Zhang. 2025a. Toward faithful retrieval-augmented generation with sparse autoencoders. *arXiv preprint arXiv:2512.08892*.

Guangzhi Xiong, Qiao Jin, Xiao Wang, Yin Fang, Haolin Liu, Yifan Yang, Fangyuan Chen, Zhixing Song, Dengyu Wang, Minjia Zhang, and 1 others. 2025b. Rag-gym: Optimizing reasoning and search agents with process supervision. *arXiv e-prints*, pages arXiv–2502.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Wenlin Zhang, Xiangyang Li, Kuicai Dong, Yichao Wang, Pengyue Jia, Xiaopeng Li, Yingyi Zhang, Derong Xu, Zhaocheng Du, Huifeng Guo, and 1 others. 2025. Process vs. outcome reward: Which is better for agentic rag reinforcement learning. *arXiv preprint arXiv:2505.14069*.

A GRPO Robustness: Seeds, Backbones, Untyped Control

Table 5 reports GRPO EM (mean±std over 3 seeds) on Qwen2.5-7B-Instruct and Llama-3.1-8B-Instruct under identical hyperparameters. On Qwen, the ordering Search-Only $<$ Binary Gap $<$ Typed Base $<$ Typed+Shape is monotone with all seed-level std ≤ 1.2 EM, well below the $+3.3$ headline gain. On Llama all checker-augmented rewards underperform Search-Only; we discuss this backbone-specific failure in §7.2 and explicitly limit our generalization claim to the Qwen backbone in the Limitations section. The *Untyped-Dense* control row (matched per-step density to Typed+Shape but no CC/IE/MB distinction) isolates the typing-vs-density confound on Qwen; the sparser Binary Gap row in Table 4 provides a complementary density-matched control.

Table 5: GRPO EM (mean±std over 3 seeds) across reward designs and two generator backbones. On Qwen2.5-7B, Typed+Shape gives a clear ordering Search-Only $<$ Typed Base $<$ Typed+Shape; on Llama-3.1-8B, all checker-augmented rewards *underperform* Search-Only, with Binary Gap collapsing most severely. We discuss this backbone- specific failure in §7.2 and treat the Qwen result as the primary headline result.

Reward	Qwen2.5-7B	Llama-3.1-8B
Search-Only	32.1 ± 0.3	$\mathbf{28.1} \pm 0.7$
Binary Gap	31.0 ± 0.4	19.2 ± 0.5
Typed Base	34.2 ± 1.2	28.3 ± 0.7
Typed+Shape (ours)	$\mathbf{35.4} \pm 0.9$	25.4 ± 0.4

B Annotation Protocol

Annotators. Two annotators with graduate-level NLP training labeled the test subset of 82 questions / 181 steps. Both were familiar with multi-hop QA conventions but were naive to the StepGap taxonomy at the start of training.

Training. Annotators completed a 30-minute calibration session on a held-out set of 20 steps, with discussion of edge cases (in particular, the boundary between MISSING BRIDGE and IRRELEVANT EVIDENCE). Decision rules were finalized as: a

step is IE if the retrieved evidence is about a different entity than the step's claim addresses; MB if the evidence is about the right entity but does not entail the claim; CC if the evidence contradicts the claim or the step targets the wrong relation.

Disagreement resolution. Steps where the two annotators disagreed were resolved through discussion between them until consensus was reached on each disputed case. The post-discussion consensus serves as the final ground-truth label set used in all evaluations.

Per-type agreement. Cohen's κ on the binary has_gap decision is 0.704. Per-type κ values are: IE 0.71, CC 0.68, MB 0.43. The lower κ on MB is driven by its low frequency (n=9 true MB cases) and inflated sampling variance, not by systematic disagreement.

Annotation interface. Annotators saw the question, the full reasoning trace, the retrieved evidence per step, and the final answer. They were asked to assign one of {no-gap, CC, IE, MB} per step plus a brief free-form justification.

Licenses and artifact use. HotpotQA, 2WikiMultiHopQA, and MuSiQue are released under their original research licenses (CC BY-SA 4.0, Apache 2.0, and CC BY 4.0 respectively); we use them only for academic evaluation, consistent with the original intended use. The Wikipedia 2018 dump used for retrieval follows the Wikipedia text license (CC BY-SA 3.0). E5-base-v2 and DeBERTa-v3 are released under MIT; Qwen2.5-7B-Instruct under the Tongyi Qianwen license; Llama-3.1-8B under the Llama 3.1 Community License. We release the StepGap checker code, the distilled student checkpoint, and the 181-step evaluation benchmark under CC BY 4.0 for non-commercial research use.

Recruitment and compensation. Both annotators are co-authors and members of the authors' research lab. No external recruitment or crowdsourcing platform was used; annotation work was conducted as part of standard research duties without separate per-task compensation. Both annotators are graduate students with prior NLP research experience.

C Knowledge Distillation to Local Student

Setup. We distill the hybrid teacher (GPT-4.1-mini + DeBERTa-v3-NLI) into a fully local Qwen2.5-7B student via supervised fine-tuning on 1,000 teacher-labeled steps (4 datasets excluding the test benchmark). Training uses LoRA ($r = 16$) on a single H100 GPU; total cost is 2 hours + $8 teacher API.

Results. Table 6 compares teacher, student, and an ablated student without NLI input features.

Table 6: Distillation results. F1 and κ vs. the human-annotated set.

Checker	F1	κ	Cost/200 steps
Teacher (4.1-mini+NLI)	.720	.284	$0.06
Student 7B (v4)	.626	.264	$0
Student 7B (no NLI)	.582	.198	$0
Student 8B (v5)	.603	.060	$0

We report two student variants. The Qwen3-8B v5 row exhibits near-zero κ (.060), which we attribute to thinking-mode KV-cache behavior at our LoRA rank (r=16) producing unstable Stage-C outputs; we therefore adopt Qwen2.5-7B v4 as the primary student.

The 7B student retains 87% of teacher F1 *and* removes teacher API calls at inference time, making the checker suitable for integration into long RL runs without per-step API expense. Cohen's κ drops from .284 (teacher) to .264 (student), a small absolute decline of 0.02. We note that both values are low in absolute terms; this reflects the inherent difficulty of typed step-level annotation – already evident in the human κ=0.704 on n=181 – rather than student degradation. The retained signal is sufficient for downstream typed reward: substituting the student for the teacher in the GRPO loop yields +1.7 EM over Search-Only (vs. +3.3 with the teacher, Table 4) – about half the teacher's gain at zero API cost, which we view as a favorable cost-quality trade-off for long RL runs.

StepGap-vs-teacher agreement on the distillation pool. The 1,000-step distillation pool also serves as a silver-standard validation set for StepGap itself. We measure agreement between StepGap (the deployed system, GPT-4.1-mini Stages A–C + DeBERTa-v3 Stages D–E) and a held-out teacher run on the same 1,000 steps, reporting binary F1=0.88 on has_gap and typed F1=0.81 on

CC/IE/MB. The gap between the silver typed F1 (0.81) and student typed F1 (.626, Table 6) localizes student degradation to capacity rather than supervision noise: the teacher signal that the student is being trained on is itself internally consistent at scale.

Student failure analysis. Manual inspection of 47 student false negatives reveals one dominant pattern: **`found_quote` → `inference` → `no-gap` over-triggering**. The student reports `found_quote=True` at Stage C even when the "quote" is from the wrong entity or does not address the claim. At this point the pipeline short-circuits to no-gap, bypassing the NLI entailment check that the teacher performs. 23 of 47 FN cases (49%) match this pattern. The bottleneck is not model capacity (the 7B student matches teacher F1 on explicit-gap cases) but a *calibration bias* in the student's Stage C judgment. Hard-negative augmentation on 26 teacher-confirmed FN cases raises student F1 by 3.7 points (Appendix P.3).

D Hybrid Architecture Rationale

Why hybrid, not LLM-only. Step-level gap detection requires several distinct judgments: entity-relation alignment, abstention validity, entity consistency, verbatim-quote extraction, logical entailment, and cross-step verification. Conflating these into one LLM prompt invites *prompt context contamination*: framing introduced by an early node biases later judgments. The Whitehorse case in Appendix F shows this in isolation – Stage A's "airline vs. terminal" framing pushes a shared-context Stage D away from accepting an otherwise entailing quote. The hybrid breaks this by making Stage D and Stage E pure NLI calls that receive only the `(premise, hypothesis)` pair.

Hybrid ablation pattern. Table 7 reports parallel component-removal ablations on the LLM-Only and StepGap checkers. The two architectures show qualitatively different ablation profiles: *(i) LLM-Only shows competing-error cancellation*: three of four component removals *improve* F1 (+1.6, +0.8, +1.7), indicating internal components actively cancel each other's errors. Only one component (Global Prior) is a net positive contributor. *(ii) StepGap is healthy*: every component removal hurts F1 (−6.6 for Stage A, −4.4 for Stage E cross-verification), indicating each stage is a positive contributor to the final decision. This compositional property is the architectural signal that motivates StepGap over LLM-Only at equal top-line F1.

Why only Stages A and E. Stage B (abstention validity) fires on too few steps in our evaluation set (n<10) to support a meaningful component-wise ablation. Stages C (entity consistency) and D (local entailment) form a tightly coupled LLM→NLI pipeline: Stage D consumes Stage C's verbatim quote as its NLI premise, so removing either alone collapses the decision tree to a different topology rather than yielding a clean ablation. We therefore ablate the two stages with structurally independent input pathways—Stage A (pre-retrieval alignment) and Stage E (cross-step verification)—both of which show −F1, confirming the monotone pattern.

Table 7: Component-removal ablations on the 181-step evaluation set, reported as step-level F1 and its change (ΔF1) relative to the full system. **LLM-Only** (top): three of four removals improve F1, a competing-error cancellation pattern in which internal sub-prompts mutually offset each other's errors. **StepGap** (bottom): every removal hurts F1, the healthy additive pattern that motivates the hybrid design. Common-Knowledge / Semantic-Leap / Global Prior denote the LLM-Only checker's internal sub-prompts for general-knowledge inference, paraphrase tolerance, and prior probability of gap, respectively.

Configuration	F1	ΔF1
LLM-Only checker		
Full system	70.0	—
− Stage A (alignment)	71.6	**+1.6**
− Common-Knowledge	70.8	**+0.8**
− Semantic-Leap	71.7	**+1.7**
− Global Prior	63.5	−6.5
StepGap (ours)		
Full system	72.0	—
− Stage A (alignment)	65.4	−6.6
− Stage E (cross-verify)	67.6	−4.4

Auditability. Each decision is unit-testable. NLI outputs are deterministic given model weights, and the label mapping (ENTAILMENT → no-gap, NEUTRAL → MB, CONTRADICTION → CC) is closed-form. Debugging a misclassification reduces to isolating quote extraction (Stage C, LLM) from entailment (Stage D, NLI). An LLM-only checker offers no such isolation.

Distillability and reward grounding. The hybrid is a prerequisite for two downstream uses. (i) Local deployment: the NLI sub-system is already a 435M model, and Stage A–C produce

structured JSON that a 7B student can imitate (Appendix C). (ii) Typed reward: the CC/IE/MB labels are grounded in the deterministic NLI output, which an LLM-only checker cannot match because its categorical judgments shift across reruns and prompt perturbations.

E Failure Mode Taxonomy of the Full System

Inspection of 13 checker false negatives and 12 false positives reveals four recurrent failure modes. We describe them using the typed taxonomy.

FM1: Parametric shortcut (FN source). Generator retrieves partially relevant evidence; NLI assigns entailment because a surface token overlap exists; generator fills in the remaining fact from parametric memory, producing a wrong answer. No CC/IE/MB is flagged.

FM2: Granularity mismatch (FN source, 6/13). Generator retrieves a related but lower-granularity fact (year instead of birth date, district instead of municipality); NLI sees the entity at some granularity and judges entailment. The answer is wrong but the step passes. This is the most common FN.

FM3: Comparison logic (FP source). Conclusion steps for comparison questions ("who died later?") require combining evidence from multiple prior steps; NLI judges each step against its direct evidence pool in isolation, often producing NEUTRAL and spurious MB flags.

FM4: Answer string normalization (FP source). Generator produces a correct answer with different surface form from the gold string ("La Vida Alrededor was released earlier" vs. gold "La Vida Alrededor"); the question counts as wrong under EM, but the checker correctly detects no evidence gap. This is a limitation of EM-based ground truth, not of the checker.

F Case Studies

We illustrate how StepGap handles each gap type with two representative examples drawn from our human-annotated evaluation set. Two further cases (a student-checker failure mode and a Single-LLM-XL failure mode) are deferred to Appendix G.

Case 1: NLI breaks prompt contamination (Whitehorse, 2WikiMultiHopQA). *Question*: "The airline operating in Whitehorse, Yukon handled how many passengers in 2012?" *Retrieved evidence*: "The terminal handled 294,000 passengers in 2012." *Gold*: 294,000.

Stage	LLM-Only	StepGap (ours)
A: alignment	on-target → pass	on-target → pass
C: quote search	finds "terminal handled 294,000 passengers"	finds "terminal handled 294,000 passengers"
D: entailment	LLM judges CONTRADICTION ("terminal ≠ airline")	NLI judges ENTAILMENT
Result	**CC (false positive)**	**no-gap (correct)**

The two systems run on identical inputs but disagree at Stage D. LLM-Only's Stage A introduced an "airline vs. terminal" framing into the shared prompt, which biased its Stage D judgment away from accepting the quote. StepGap's NLI Stage D receives only the `(premise, hypothesis)` pair, with no upstream framing, and judges entailment correctly. Air North is the sole commercial airline operating from the Whitehorse terminal, so the quote does support the claim. This is the context-contamination mechanism we discuss in Appendix D, isolated to a single decision.

Case 2: Missing-Bridge detection (Kuhn vs. Pertramer, 2WikiMultiHopQA). *Question*: "Were Heinrich Gerhard Kuhn and Elfie Pertramer from the same country?" *Gold*: yes (both German). *Predicted answer*: no (the model concluded "not from same country").

Step	StepGap decision
1. "Kuhn is a British physicist"	Stage D: NLI=ENTAILMENT (quote found, supported) → no-gap
2. "Pertramer is a German actress"	Stage D: NLI=ENTAILMENT → no-gap
3. "Britain ≠ Germany"	common-knowledge inference, no quote needed → no-gap
4. "Therefore not from same country"	**Stage D: NLI=NEUTRAL; quote found but does not entail ("British physicist" specifies nationality, not country of birth) → MB**

StepGap correctly identifies the conclusion step as MISSING BRIDGE: the retrieved evidence supports the sub-claim "Kuhn is British" but *not* the unstated bridging premise "Kuhn was born in Britain." The actual bridge is that "British physicist" refers to academic affiliation, not birthplace; Kuhn was born in Germany. This is the kind of unstated-premise

failure that endpoint EM cannot catch (the answer is wrong, but no claim is overtly contradicted) and that a binary support-checker would also miss (the quote is supported, just incomplete). The repair action prescribed by an MB tag ("issue a bridging search") is exactly what the generator would need to do to recover.

G Additional Case Studies

We supplement the two case studies in §F with two further examples that illustrate failure modes of alternative checkers, motivating specific design choices in StepGap.

Case 3: Student over-triggers `found_quote` (Peter and Paul Fortress, MuSiQue). *Question*: "Where did the designer of Peter and Paul Fortress die?" *Gold*: Saint Petersburg. *Evidence*: "Peter and Paul Fortress is the original citadel of St. Petersburg, Russia, built to Domenico Trezzini's designs."

Stage	Teacher (GPT-4.1-mini + NLI)	Student (Qwen2.5-7B distilled)
C: quote	finds "citadel of St. Petersburg"	also finds it; sets `found_quote=True`
D: entailment	NLI=NEUTRAL (citadel location $\neq$ designer's place of death)	*(short-circuited; never invoked)*
Result	**MB (correct)**	**no-gap (false negative)**

The student's failure is structural: it learns during distillation to map "`found_quote=True` $\wedge$ inference step" directly to no-gap, bypassing the NLI entailment check the teacher performs. This pattern accounts for 23 of 47 (49%) student false negatives (§C). The fix is not more capacity but *stronger Stage C calibration*: the student must learn that "found a quote" is necessary but not sufficient for no-gap. We address this via hard-negative augmentation (briefly described in Appendix C).

Case 4: Single-LLM-XL flags everything (HotpotQA). *Question*: "Which 2017 film was directed by the director of *Whiplash*?" *Gold*: *La La Land*. *Evidence at step 1*: "*Whiplash* is a 2014 film directed by Damien Chazelle."
A single LLM, even a strong one, applies its quote search and entailment criteria too uniformly, treating any trace of paraphrasing or compositional inference as unsupported. Across the 181-step evaluation set this collapses into the 87.8% IE rate reported in §6.2—and produces the same 91.4% Q-F1 that we identify in §6 as the Q-F1 trap. The decomposed pipeline in StepGap, by contrast, accepts paraphrastic matches at Stage C and only enforces strict entailment at Stage D, where the inputs are proper propositions.

Stage	StepGap (ours)	Single-LLM-XL
A: alignment	on-target	on-target
C: quote	finds "directed by Damien Chazelle"	"no exact-match quote" (over-strict)
D/E	NLI=ENTAILMENT $\rightarrow$ no-gap	— (skipped)
Result	**no-gap (correct)**	**IE (false positive)**

H Reward Hyperparameters

Base reward weights. The base reward magnitudes $\{+0.20, -0.05, -0.10, +0.05\}$ for {no-gap, MB, IE, CC} were selected on a 200-iteration pilot, optimizing answer-claim support rate at fixed EM. The relative ordering (no-gap $\gg$ CC, MB $>$ IE) was held fixed; only the absolute scale was tuned. For the CC detection bonus specifically, we tested $\{+0.02, +0.05, +0.10\}$ at single seed: $+0.02$ produced the same monotone CC-suppression as the negative ablation (Figure 3), $+0.10$ caused CC rate to plateau without the second-phase decline, and $+0.05$ exhibited the two-phase rise-then-fall dynamics we report as the intended design behavior.

Shaping weights. The shaping reward magnitudes $\{+0.10, +0.15, -0.15, -0.05\}$ were chosen so that the total per-step reward magnitude does not exceed the EM reward magnitude (which is 1.0 binary).

Near-duplicate query threshold. The token-F1 threshold 0.7 used in r^{shape} to detect near-duplicate queries (§5.2) was set on the same 200-iteration pilot as the base reward weights: lower values incorrectly flagged paraphrastic reformulations as duplicates and dampened re-search behavior, while higher values failed to flag near-identical retries differing only in stop words or token order. We leave a fine-grained sweep of this threshold to future work.

Lambda. The reward mixing coefficient λ defaults to 1.0. Sensitivity to $\lambda \in \{0.5, 1.0, 2.0\}$ shows EM varies within ± 0.6 points; we report $\lambda = 1.0$ as the main configuration.

Limitations. A learned reward weighting (e.g., inverse-propensity per gap type) is left to future work.

I Step-Category Distribution

Table 8 reports the per-variant distribution of assigned gap types over the 181-step evaluation set. A well-calibrated checker should produce IE in the 38–41% range (the rate observed on LLM-Only and StepGap); deviations from this band reflect systemic biases that sF1 alone may not penalize.

Table 8: Step-category distribution across checker variants on the 181-step evaluation set. Healthy IE band: 38–41%. † marks degenerate variants whose IE rate exceeds 70% with collapsed CC and MB.

Variant	no-gap	IE	CC	MB
LLM-Strict	27.6	40.9	17.1	14.4
LLM-Only	40.9	39.2	14.9	5.0
StepGap (ours)	**40.9**	**38.1**	**14.9**	**6.1**
Single-LLM-XL†	12.2	87.8	0.0	0.0
NER-Quote†	14.4	74.0	10.5	1.1

The two pathologies sF1 alone cannot make visible: *Single-LLM-XL* assigns zero steps to CC and zero to MB, reducing the three-way taxonomy to a binary (no-gap vs. IE) decision – a stronger single LLM produces a checker that *cannot discriminate between gap types*. This is precisely the property that disqualifies it from typed-reward use: typed reward needs all three categories reliably emitted. *NER-Quote* replaces Stage C's LLM quote search with spaCy NER plus token overlap; the result is 74% IE because Stage D's NLI call fires on noun-phrase queries that are not well-formed propositions, so the checker over-reports retrieval failure. The hybrid works because LLM and NLI handle *different factor types*, not because NLI is smaller or because an LLM is larger.

J Answer-Gap Correlation

Table 9 cross-tabulates final-answer correctness with gap detection on the 82-question subset. Of 69 wrong-answer questions, 56 (81%) have at least one gap detected. Of 13 correct-answer questions, 12 still have a gap detected – the *lucky-correct* regime: a substantial fraction of correct answers in multi-hop QA are not fully grounded in retrieved evidence.

Table 9: Answer correctness vs. gap detection ($n = 82$ questions).

	Gap Detected	No Gap Detected
Wrong Answer	56 (TP)	13 (FN)
Correct Answer	12 (FP)	1 (TN)

K First-Gap-Type Localization

For each wrong-answer question we examine the first step the checker flags as a gap and record its type (Table 10). StepGap and LLM-Only both identify IE (retrieval failure) as the first broken step in roughly 80% of wrong-answer chains, confirming that retrieval, not reasoning, is the dominant root cause of failure in our multi-hop evaluation set. StepGap differs from LLM-Only in striking a more balanced first-gap distribution between IE (79%), MB (11%), and CC (11%), giving downstream repair logic a richer typed signal. Degenerate variants attribute essentially every first failure to IE, producing no usable signal for routing repair actions.

Table 10: For each wrong-answer question, type of the first step flagged by the checker as containing a gap. Entries are percentages of wrong-answer questions. Degenerate variants trivially attribute nearly all first-failures to IE.

Variant	First-IE	First-CC	First-MB
LLM-Strict	62	12	26
LLM-Only	80	12	7
StepGap (ours)	79	11	11
Single-LLM-XL	100 (deg.)	—	—
NER-Quote	97 (deg.)	—	—

L Q-F1 Trap: Empirical Confirmation

We complement the closed-form argument in Section 6 with two empirical checks (Table 11). *First*, on four public QA datasets at their natural wrong-answer rate w (estimated from a Qwen2.5-7B-Instruct Search-Only baseline), a flag-everything dummy checker reaches a Q-F1 within ± 0.02 of the analytic prediction $2w/(1 + w)$. *Second*, on stratified resamples of our own evaluation set with w controlled at 0.50, 0.65, and 0.84, empirical Q-F1 reproduces the analytic curve, confirming that the effect is driven by w and the metric, not by the specific question content.

Table 11: Empirical Q-F1 of a flag-everything checker on public benchmarks (top) and stratified resamples of our evaluation set (bottom). Analytic prediction: $2w/(1+w)$.

Dataset / stratum	w	Q-F1	(analytic)
Cross-dataset, natural w			
NQ	0.55	0.70	(0.71)
HotpotQA	0.68	0.80	(0.81)
2Wiki	0.76	0.85	(0.86)
MuSiQue	0.90	0.94	(0.95)
Stratified resamples, w controlled			
Stratum-50	0.50	0.66	(0.67)
Stratum-65	0.65	0.79	(0.78)
Stratum-84	0.84	0.91	(0.91)

M GRPO Auxiliary Grounding Metrics

Table 12 reports the auxiliary grounding metrics that accompany the EM numbers in Table 4: token-F1, answer support rate (fraction of answer claims NLI-entailed by a retrieved quote), and mean search calls per question. Across reward configurations, support rate rises monotonically ($0\% \rightarrow 38.2\% \rightarrow 56.1\% \rightarrow 71.5\%$), and search count increases under typed shaping ($2.1 \rightarrow 2.8$), indicating that the policy learns to issue additional searches in response to detected gaps rather than answering through.

Table 12: Auxiliary grounding metrics for each reward ablation on the 82-Q subset (aggregate over the three datasets). *Supp.* = fraction of answer claims NLI-entailed by a retrieved quote; |<s>| = mean search calls per question.

Configuration	F1	Supp.	|<s>|
Search-Only	39.6	—	2.1
Binary Gap	40.4	38.2	2.3
Typed Base	41.6	56.1	2.5
Typed+Shape (ours)	**43.1**	**71.5**	**2.8**
w/ Student checker	41.2	48.3	2.6
w/ Teacher checker	43.1	71.5	2.8

N Threshold-Sweep Diagnostic

To verify that the degenerate classification of Single-LLM-XL and NER-Quote (§6.2) is not threshold-dependent, we sweep confidence thresholds for these variants and the StepGap reference. Per-variant confidence is the geometric mean of stage-level confidences: for LLM stages, the softmax probability of the chosen output token; for NLI stages, the softmax probability of the chosen entailment label.

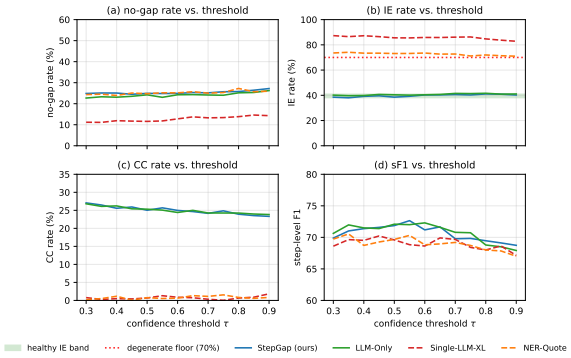

Figure 4: Threshold sweep for StepGap (healthy reference) vs. Single-LLM-XL and NER-Quote (degenerate variants). Top-right panel: green band marks the healthy IE range (38–41%) observed across LLM-Only and StepGap; red dashed line marks the 70% degenerate floor. Single-LLM-XL and NER-Quote never enter the healthy band at any threshold $\tau \in [0.3, 0.9]$.

O Checker Implementation Prototype

This appendix provides a minimal reference implementation of the StepGap hybrid checker. The full production code handles batching, caching, retrieval-server integration, and robust JSON parsing; here we present the core decision logic in stripped-down form. The five-stage pipeline (Figure 2) is implemented as a linear sequence of early-return checks.

O.1 Structured-Output Schema

The LLM (GPT-4.1-mini in the teacher; Qwen2.5-7B in the student) returns a single structured JSON object per step, containing the judgments for Stage A, Stage B, and Stage C in one pass. Structured-output mode is enforced via `response_format={"type":"json_schema",...}` with `strict=true`, which guarantees parseable JSON. The schema is:

```
{
  "step_minus1_alignment": {
    "is_off_target":       bool,
    "drift_type":  // one of:
      "none" | "entity_drift" |
      "relation_drift" | "scope_drift",
    "alignment_reasoning": str
  },
  "step_0_abstention_check": {
    "is_abstention_step":     bool,
    "abstention_is_accurate": bool,
    "abstention_reasoning":   str
  },
  "step_1_quote_search": {
    "entity_match":           bool,
    "entity_match_reasoning": str,
    "found_quote":            bool,
    "evidence_quote":         str,
```

```
    "quote_search_reasoning": str
  }
}
```

O.2 LLM Prompt Templates

Stages A–C are invoked via a single prompt that returns the schema above. The prompt has a fixed three-section structure (alignment / abstention / entity-and-quote) and switches between an *inference* variant and a *conclusion* variant of Section A based on the step type. We reproduce the core text below; whitespace and formatting follow the actual production prompt.

Section A: alignment (inference variant).

```
STEP -1: SUBQUESTION ALIGNMENT
Does this step target the CORRECT entity and
relation?
CRITICAL: retrieval failure != off-target.
Only flag if step_text ITSELF names the wrong
entity.

EC quick test - is_off_target = FALSE if:
 - Entity in step_text appears verbatim in
   the question
 - Entity is the correct intermediate result
 - step_text names correct film/person but
   retrieval returned wrong doc

TRUE drift: Q asks "AUTHOR of book" -> step
  searches "DIRECTOR of film adaptation"
NOT drift: retrieval failed but query is
  correct

If is_off_target = TRUE -> has_gap = TRUE,
gap_type = "unsupported_claim", STOP.
```

Section A: alignment (conclusion variant).

```
STEP -1: CONCLUSION ALIGNMENT (mandatory)
RULE 0: answer contains N/A or unknown
  -> is_off_target = FALSE, bypass to Node 0.

Otherwise C-1 -> C-2 -> C-3:
  C-1: target attribute type
       (PLACE / BIRTH_YEAR / SPOUSE /
        AGENT / COUNTRY / YES_NO)
  C-2: actual type of answer value
  C-3: TYPE mismatch  = relation_drift.
       Wrong value but same type
                        = NOT drift.

DRIFT:    Q "Where born?" -> "Naresh Kumar"
          (person != place) -> TRUE
NO DRIFT: Q "Who did X?"  -> "Viktor Belash"
          (person  = person) -> FALSE
```

Section B: abstention.

```
STEP 0: ABSTENTION CHECK
(skip if Step -1 triggered)
Is this step saying N/A / cannot determine?
  Evidence lacks it
    -> GROUNDED ABSTENTION
       is_abstention_step = True,
       abstention_is_accurate = True
  Evidence has it
    -> WRONG ABSTENTION
       is_abstention_step = True,
       abstention_is_accurate = False
```

For conclusion steps with prior inference evidence available, the prompt prepends a `GLOBAL EVIDENCE` block listing the snippets retrieved at all earlier steps; Stage B uses this pool to decide whether an asserted abstention is justified.

Section C: entity consistency + verbatim quote.

```
STEP 1: ENTITY CONSISTENCY + QUOTE SEARCH
(skip if Step -1 or Step 0 triggered)

Part A - Entity consistency check:
  EC-1: What entity/topic does this step's
        query target?
  EC-2: What entity/topic is the retrieved
        evidence actually about?
        (check the document title and first
         sentence)
  EC-3: Same entity? -> entity_match = TRUE,
                        proceed to quote search
        Different    -> entity_match = FALSE,
                        found_quote  = FALSE
                        (stop here)

  *Quote from the WRONG document does NOT
   count.*
   Step:     "Sruthilayalu composer"
   Evidence: "M.G. Sreekumar" (different
              person) -> FALSE

Part B - Verbatim quote search
        (only if entity_match = TRUE):
  Find a 5-20 word EXACT span from evidence.
  Must be: (i) exact text,
           (ii) about the same entity,
           (iii) relevant to the step claim.
```

The prompt closes with the question, the previous-step summaries (truncated to the last three), the current step's text, and the evidence pool serialized as JSON.

NLI calls (Stage D and Stage E). The NLI stages take no prompt: they pass a `(premise, hypothesis)` pair directly to a DeBERTa-v3-NLI cross-encoder (Laurer et al., 2024). Stage D sets `premise` = the verbatim quote returned by Stage C and `hypothesis` = the step's claim text; Stage E iterates over entity-matched snippets from prior steps as the premise pool, with the same hypothesis. We softmax the three-class logits and take the argmax label (ENTAILMENT / NEUTRAL / CONTRADICTION); ties default to NEUTRAL.

O.3 NLI Label Calibration

NLI models from the `cross-encoder` family emit a three-dimensional logit vector whose label order varies by checkpoint (e.g. `[contradiction, entailment, neutral]` or `[entailment,`

neutral, contradiction]). At checker initialization we probe with a known entailment pair ("A cat is an animal.", "A cat is an animal.") and identify which output dimension scores highest; the label order is set accordingly. All subsequent calls use the calibrated mapping. Thresholds are ENTAILMENT=0.5 and CONTRADICTION=0.5 on the softmaxed scores; labels between these thresholds are treated as NEUTRAL.

O.4 Step Type Classification

A step is classified as inference if its text contains a <search> tag (or, in Hermes-style outputs, a <tool_call> JSON object with the search action), and as conclusion if it contains an <answer> tag. Ambiguous steps (neither tag) are treated as inference by default.

O.5 Pipeline Path Accounting

Every check returns a pipeline_path string encoding the exact sequence of stage decisions taken (e.g. stageC:entity_mismatch, stageD:neutral). We use this for two purposes: (i) distributional analysis, confirming each stage is actually exercised in practice and not starved of input, and identifying which branches dominate the error modes; (ii) unit testing, where we assert that specific annotated cases follow the expected path. The pipeline-path signature also underlies our distillation loss: the student is trained to reproduce the teacher's full path, not just the terminal label, which prevents the student from short-circuiting via surface heuristics (see Section C).

P Additional Experimental Details

This appendix documents experimental protocols and reproducibility details that complement the main paper.

P.1 Standard-Split Re-evaluation Protocol

For comparability with the published baselines in Table 4 (top panel), we follow the standard Search-R1 evaluation pipeline on the standard dev splits (sizes in Table 2): Qwen2.5-7B-Instruct generator with temperature 0 and max-new-tokens 1024; E5-base-v2 retrieval (Wang et al., 2022) over Wikipedia 2018 with top-k=5; GRPO-trained checkpoint at the best validation EM step, chosen on a held-out 500-Q dev pool disjoint from these splits. Each cell reports strict EM averaged over the full split. Our reward-ablation rows in Table 4 use the same pipeline.

P.2 GRPO Training Configuration

All GRPO runs hold the following hyperparameters constant: group size G=8, KL coefficient β=0.05, learning rate 1×10^{-6} for the policy with the checker frozen, 3,000 training iterations with 200-iteration evaluations, on the NQ+HotpotQA training mix specified in Section 7. The *Untyped-Dense* control reward referenced in Section 7.2 uses r^{base} that maps any gap (CC, IE, MB) to -0.10 and no-gap to $+0.20$, with no shaping term and no per-type distinction; this isolates the contribution of *typing* from per-step density.

Computing infrastructure. All GRPO training is run on a single NVIDIA H100 (80GB) GPU. A full 3,000-iteration GRPO run takes approximately 18 H100 GPU-hours for Qwen2.5-7B-Instruct and 22 H100 GPU-hours for Llama-3.1-8B-Instruct under our setup. Teacher-checker API usage (GPT-4.1-mini) is approximately 72K calls per run at $22; distillation of the student checker costs $8 in teacher API plus 2 H100 GPU-hours. The 181-step human evaluation set was processed once per checker variant, totalling under 0.5 H100 GPU-hours of additional inference. Total compute across all reported configurations (three seeds on Qwen, single seed on Llama, all reward ablations, distillation, and Appendix experiments) is approximately **350-400** H100 GPU-hours.

P.3 Hard-Negative Augmentation Protocol

The +3.7 F1 gain reported for the student in Appendix C comes from augmenting the student's training data with 26 hard-negative cases identified by manual inspection of teacher–student disagreement on the held-out distillation pool. A case is selected if the teacher's NLI label and the student's NLI label diverge *and* the teacher's prediction matches the human label on the 181-step evaluation set for an adjacent step from the same trace. Augmented training uses the same LoRA configuration (r=16) for 200 additional iterations with a learning rate of 5×10^{-7} (an order of magnitude smaller than initial distillation, so as not to overfit the small augmentation set). We hold out 13 of the 26 hard-negative examples as a within-augmentation check to confirm the gain is not attributable to memorization.