EVE-AGENT: EVIDENCE-VERIFIABLE SELF-EVOLVING AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-evolving search agents learn from the questions and answers they generate themselves; however, existing approaches typically use question difficulty as the primary training signal, providing little incentive to produce evidence that actually supports an answer. We ask whether self-generated evidence can serve as useful feedback for learning to answer questions both correctly and with justification. We introduce EVE-Agent, a self-evolving search agent that generates a candidate evidence span along with each question and answer. The proposer is rewarded based on how much the span increases a fixed solver's agreement with the generated answer when the span is provided versus when it is withheld, with search disabled in both conditions. This model-relative utility signal requires no external answer labels and directly measures whether the generated evidence is useful to the solver. The same span is subsequently reused as an evidence target for solver training, turning evidence generation into a reusable source of supervision. Across seven question-answering benchmarks, EVE-Agent consistently outperforms Dr. Zero in joint answer-and-evidence performance under the same backbone and retrieval pipeline. It also improves both average answer accuracy and judged evidence support. These results show that self-generated evidence can provide an effective training signal for self-evolving search agents rather than serving merely as an explanation appended to the final answer.

1 INTRODUCTION

A search agent should return an answer that a reader can check against supporting evidence. Retrieval-augmented generation and tool-using language models provide access to relevant information (Lewis et al., 2020; Yao et al., 2023), while evidence-annotated question-answering benchmarks make support an explicit evaluation target (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022). These developments distinguish two abilities: finding an answer and identifying the text that justifies it. Learning the former does not automatically provide a training signal for the latter.

This distinction becomes important when agents generate their own training tasks. In a self-evolving proposer–solver framework, a *proposer* creates questions and answer candidates, and a *solver* learns from the resulting curriculum (Zhao et al., 2025; Huang et al., 2026b). Dr. Zero extends this approach to search agents by rewarding questions with intermediate solver success rates (Yue et al., 2026). Difficulty determines which questions are worth practicing, but does not directly evaluate the usefulness of the evidence accompanying them. An evidence field can be present even when its contents do not support the answer. We therefore ask:

> *Can the utility of self-generated evidence provide training feedback that improves the joint production of correct answers and supporting evidence?*

We introduce EVE-Agent, which uses a generated evidence span both to evaluate a proposed task and to supervise the solver (Figure 1). The proposer generates a question, an answer candidate, and a span requested to be copied from available source text. A frozen solver then answers the question with and without that span. Both conditions disable search, so the difference in agreement with the generated answer measures the contribution of the span itself, rather than additional retrieval. This evidence-utility score supplements the inherited difficulty reward without requiring externally annotated answers or evidence for the generated questions.

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.

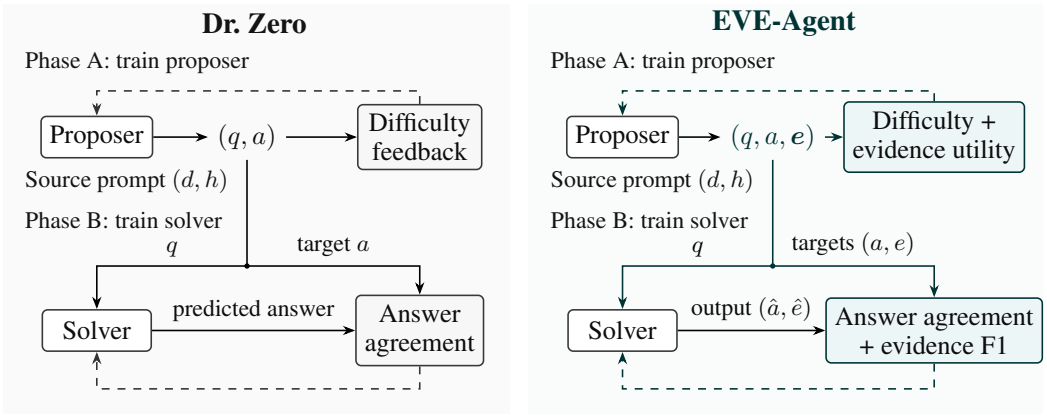


Figure 1: **One evidence span, two training roles.** EVE-Agent extends the difficulty-based proposer–solver loop of Dr. Zero. The fields (q, a, e) are a question, answer candidate, and evidence span. In Phase A, e is scored by its contribution to a frozen solver's answer agreement. In Phase B, spans generated by the trained proposer become evidence targets: the solver receives only q, while (a, e) are used to compute its reward. Evidence F1 measures token overlap. Solid arrows carry data; dashed arrows indicate policy updates.

Evidence serves a second role after proposer training. Spans generated by the trained proposer serve as pseudo-references for training the solver to return evidence alongside its answers. The solver must find the answer and evidence using only the question and its search tool; the generated targets are used for reward computation, not supplied as hints. This reuse connects the information rewarded during curriculum construction to the behavior learned during question answering. The search architecture and policy optimizers remain unchanged.

Across the evaluated benchmark suite, EVE-Agent improves joint answer-and-evidence performance over Dr. Zero on every benchmark, and achieves the highest average answer accuracy and judged evidence support among the compared systems. These results support a concrete answer to our research question: evidence can be a reusable source of self-training feedback, not merely an explanation appended to an answer. Our contribution is this connection between evidence utility, curriculum construction, and evidence-producing solver training.

2 Problem Setup and Background

2.1 Learning to answer questions with evidence

We study question answering with a finite, nonempty document corpus $\mathcal{D}$ and a search engine $\mathcal{R}$ that returns snippets from it. A *policy* is a model's conditional distribution over assistant actions, including text and search calls. A *rollout* τ comprises its generated messages and tool interactions. The training framework has two roles: a proposer policy π^{pro} constructs tasks, and a solver policy π^{sol} learns to answer them. Both can use the search engine during task generation or solving.

The proposer receives a source document $d \in \mathcal{D}$ and a prescribed hop label $h \in \mathcal{H}$, where $\mathcal{H}$ is a finite, nonempty set of positive integers. A hop denotes a step connecting facts toward an answer; h specifies the intended complexity of a question, not a verified count of reasoning steps. The generated task contains a question q and an answer candidate a. EVE-Agent additionally requests an evidence candidate e, a text span intended to support a. The solver receives q, may query $\mathcal{R}$, and returns an answer $\hat{a}$ and, under the evidence-output protocol, a span $\hat{e}$.

The targets (a, e) are generated by the proposer. Benchmark reference answers, denoted $a^\star$, are separate objects used only for downstream evaluation. The corpus and pretrained policies are given resources; the label-free property concerns the construction of the rewards. To compare answer strings, let NORM lowercase text, remove punctuation and the English articles "a", "an", and "the",

and collapse whitespace. Normalized exact match, $\mathrm{EM}(x, y)$, is one when $\textsc{Norm}(x) = \textsc{Norm}(y)$ and zero otherwise. We distinguish this comparison from unnormalized agreement where used below.

2.2 Dr. Zero: learning from self-generated questions

Dr. Zero provides the difficulty-based proposer–solver framework on which EVE-Agent builds (Yue et al., 2026). Its curriculum is a collection of model-generated questions and answer candidates. Rather than assigning difficulty in advance, it tests each question against the solver: a question that the solver always answers as proposed offers little challenge, while one on which it never agrees provides no observed successful solution. The proposer is rewarded for generating questions between these extremes.

Constructing the curriculum. Within a proposer-training phase, keep the solver fixed. For a generated pair (q, a) and an integer $n \geq 2$, draw n independent search-enabled solver rollouts using the same answering protocol. Let $\hat{a}_j$ be the parsed answer on trial j. With $\mathbf{1}\{\cdot\}$ denoting the indicator, the difficulty component is

$$k = \sum_{j=1}^{n} \mathbf{1}\{\hat{a}_j = a\}, \qquad R^{\mathrm{DZ}}(q, a; k) = \mathbf{1}\{0 < k < n\} \frac{n - k}{n - 1}. \tag{1}$$

The reward is zero when $k = 0$ or $k = n$, and positive otherwise, with the largest realized reward at $k = 1$. It favors challenging questions with at least one observed successful trial, rather than difficulty alone. Here and in the utility test in Section 3.1, equality compares parsed answer strings with the generated target; a parse failure is a nonmatching outcome. Agreement is feedback relative to the proposed answer, not an external correctness judgment. Appendix A.2 derives the expected reward and its preferred agreement rate.

The proposer combines this difficulty component with a structural format reward, which scores whether a rollout follows the task-generation protocol. It is trained with *hop-grouped relative policy optimization* (HRPO): rewards are centered and scaled within groups sharing the same prescribed hop label. The resulting *advantage* measures how a rollout's reward compares with the other rewards in its group and supplies the signal for the policy update. Grouping by hop label compares tasks with similar intended complexity; it does not require knowing their true reasoning depth.

Training the solver. After proposer training, freeze the proposer and generate question–answer pairs for the solver. For each question, sample a group of $g \geq 2$ search-enabled solver responses, reward answer agreement with the generated target, and update the solver with *group-relative policy optimization* (GRPO) (Shao et al., 2024). Here the comparison group contains responses to the same question, rather than different questions with the same hop label. The generated answer is a reward target, not an input hint. Dr. Zero repeats these phases so that the updated solver guides the next curriculum. Algorithm 1 in Appendix B summarizes one such cycle.

The evidence signal that is missing. Dr. Zero asks the proposer to ground its reasoning steps and answer in source passages. Its rewards, however, do not directly measure a particular span's usefulness or reward the solver for recovering it. EVE-Agent retains the proposer–solver cycle and its policy optimizers, but attaches evidence feedback to both phases: it rewards the proposer's span for incremental utility and uses spans from the trained proposer as solver output targets. Figure 1 shows this change in the training signals.

3 EVE-Agent

EVE-Agent adds evidence feedback to both sides of this learning problem. The proposer is rewarded for spans that help a fixed solver recover a generated answer; the solver is then rewarded for recovering those spans together with the answer. We first define the utility test, then incorporate it into the two-phase training loop.

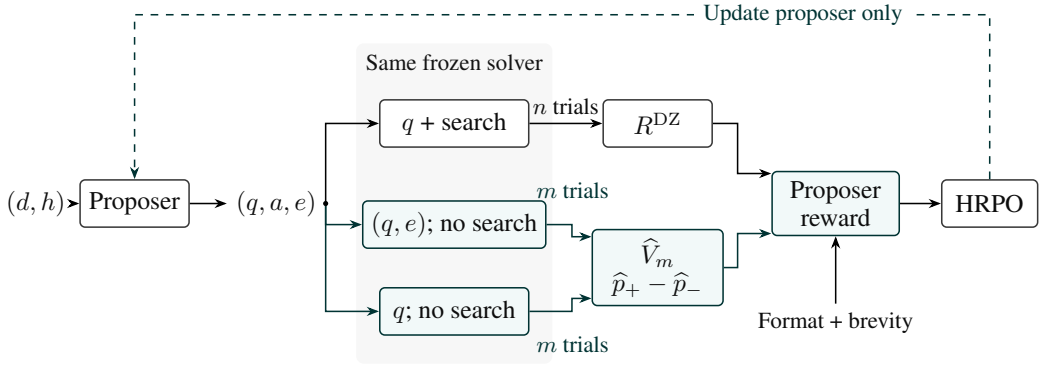


Figure 2: **Phase A separates difficulty from evidence utility.** All scoring branches use the same frozen solver. Search-enabled trials provide R^{DZ}; the two search-disabled conditions provide empirical agreement rates $\widehat{p}_+$ and $\widehat{p}_-$ with the generated answer a. Their difference is $\widehat{V}_m$. The combined reward updates only the proposer.

3.1 Measuring the contribution of evidence

With and without evidence. Fix a generated triple (q, a, e) and a solver checkpoint. Let $\widetilde{\pi}^{\mathrm{sol}}(\cdot \mid q, e)$ be its distribution over parsed answers under a single-turn, search-disabled protocol. The no-evidence condition uses the same checkpoint and decoding rule, with e withheld. A distinct failure outcome represents an unparseable response and never matches a. Define

$$\begin{aligned} p_+(q, e, a) &= \mathbb{P}_{\hat{a} \sim \widetilde{\pi}^{\mathrm{sol}}(\cdot \mid q, e)}[\hat{a} = a], \\ p_-(q, a) &= \mathbb{P}_{\hat{a} \sim \widetilde{\pi}^{\mathrm{sol}}(\cdot \mid q)}[\hat{a} = a], \\ V(q, e, a) &= p_+(q, e, a) - p_-(q, a). \end{aligned} \tag{2}$$

The with-evidence condition measures target agreement when the span is available. Subtracting the no-evidence condition discounts answers the solver can already recover without it. Disabling search prevents either condition from replacing the candidate span with newly retrieved information. Thus $V \in [-1, 1]$ measures the span's incremental usefulness to this solver.

We call this test the *evidence verifier*, using verification in the operational sense of model-relative utility. A positive score means that the span increases agreement with a; it is not a certificate that a is true or that e entails it. Training-time utility and evaluation-time evidence support have distinct roles: the former supplies feedback without external target labels, while the latter assesses outputs against benchmark references.

Why utility complements difficulty. The two signals ask different questions. Difficulty measures how often the solver recovers a while using search; utility measures how much a particular span changes recovery of a without search. For two spans paired with the same (q, a), the no-evidence probability p_- is identical, so their utility ordering is determined by their with-evidence agreement probabilities. Across different questions, subtracting p_- discounts targets that are already accessible to the solver. A fixed scoring checkpoint keeps these comparisons tied to a common solver throughout proposer training.

For example, a question may ask which region an archive is located in. A candidate span could name the archive's city and then identify that city's region. The utility test asks whether supplying this information increases agreement with the proposed region, relative to the question alone. Merely naming the archive does not guarantee a positive score. The distinction is between text being present and text changing the solver's ability to recover the target. Appendix C follows a constructed task of this form through reward computation and solver training.

Sampling the utility signal. For an integer $m \geq 1$, independently draw m answers in each condition, with independence also between conditions. The difference of empirical agreement rates is

$$\widehat{V}_m(q, e, a) = \frac{1}{m} \sum_{j=1}^{m} \left[\mathbf{1}\{\hat{a}_+^{(j)} = a\} - \mathbf{1}\{\hat{a}_-^{(j)} = a\} \right]. \tag{3}$$

This requires $2m$ search-disabled decodes per eligible example. Figure 2 separates this test from the search-enabled difficulty branch.

Proposition 1 (What the utility estimate measures)**.** *Fix the triple, solver checkpoint, and decoding protocols in Eq. (3). If all $2m$ samples are independent, then, conditional on these fixed quantities,*

$$\mathbb{E}[\widehat{V}_m] = V, \qquad \mathrm{Var}(\widehat{V}_m) = \frac{p_+(1-p_+) + p_-(1-p_-)}{m} \leq \frac{1}{2m}. \tag{4}$$

Both V and $\widehat{V}_m$ belong to $[-1, 1]$.

The estimate therefore targets precisely the with–without difference, and its sampling variance scales as $1/m$. This result concerns the accuracy of the utility measurement, not the correctness of the generated target. Appendix A.1 gives the proof and distinguishes this conditional bound from variation across generated examples.

3.2 Rewarding useful evidence and reusing it

Generated outputs and eligibility. The proposer is asked to copy one contiguous evidence span from its source document or a snippet observed during its rollout. Using the normalization defined in Section 2, a rollout is eligible for content rewards when parsing succeeds, q and a are nonempty, and $\textsc{Norm}(a)$ is not a literal substring of $\textsc{Norm}(q)$. This filter removes degenerate question–answer pairs; source membership and semantic support are not additional eligibility conditions.

Let $F_{\mathrm{fmt}}(\tau, h) \in [0, 1]$ denote the inherited format score. It is a function of the complete rollout, including planning markers and tool interactions, not just the final fields. Appendix D.1 gives its components. For evidence length $|e|$ measured in proposer tokens and a length scale $L_{\max} > 0$, define the brevity bonus

$$B(e) = \max\left(0, 1 - \frac{|e|}{L_{\max}}\right). \tag{5}$$

The bonus favors concise spans without imposing a hard length constraint.

Proposer reward. For an eligible rollout, combine the signals as

$$\widehat{R}^{\mathrm{pro}}(\tau, h; k) = \tfrac{1}{2} F_{\mathrm{fmt}}(\tau, h) + R^{\mathrm{DZ}}(q, a; k) + \lambda_V \widehat{V}_m(q, e, a) + \lambda_B B(e), \tag{6}$$

where $\lambda_V, \lambda_B \geq 0$. The terms reward valid structure, useful difficulty, incremental evidence utility, and brevity, respectively. An ineligible rollout receives only $\frac{1}{2} F_{\mathrm{fmt}}$; failed parsing or an empty question or answer gives $F_{\mathrm{fmt}} = 0$. Such rollouts do not enter either solver-scoring branch.

The utility signal is additive rather than a hard acceptance gate. It lets evidence shape the curriculum alongside difficulty. Replacing $\widehat{V}_m$ by V gives its population-utility counterpart; setting $\lambda_V = \lambda_B = 0$ recovers the inherited proposer reward components. The latter is a statement about the reward, not an equivalence of complete training systems.

Solver reward. After proposer training, freeze the proposer and generate eligible triples. Store (a, e) as reward targets, treating e as a *pseudo-reference*: a generated span rather than human-annotated gold evidence. The solver receives only q and access to $\mathcal{R}$. For its output $(\hat{a}, \hat{e})$, use

$$\begin{aligned} R^{\mathrm{sol}}(\hat{a}, \hat{e}; a, e) &= \mathrm{EM}(\hat{a}, a) + \lambda_E R_{\mathrm{evidence}}(\hat{e}, e), \\ \mathrm{EM}(\hat{a}, a) &= \mathbf{1}\{\textsc{Norm}(\hat{a}) = \textsc{Norm}(a)\}, \\ R_{\mathrm{evidence}}(\hat{e}, e) &= \mathrm{F1}_{\mathrm{tok}}\big(\textsc{Norm}(\hat{e}), \textsc{Norm}(e)\big), \qquad \lambda_E \geq 0. \end{aligned} \tag{7}$$

Token F1 measures multiset word overlap after normalization; Appendix D.2 defines the score, including its empty-span boundary. The additional term rewards recovery of the evidence target, while exact match retains the answer objective. The evidence field links the phases: Phase A trains the proposer to generate useful spans, and Phase B uses its generated spans as targets for retrieval and reporting.

3.3 TWO-PHASE POLICY OPTIMIZATION

Phase A: learn an evidence-aware curriculum. Keep the solver fixed and train the proposer with HRPO (Yue et al., 2026). Sample one proposer rollout per source prompt, compute Eq. (6), and standardize rewards within each prescribed hop group. These relative rewards provide the advantages for the proposer update. Both scoring branches use the same frozen solver.

Phase B: learn to answer with evidence. Freeze the trained proposer and use its eligible triples to form a fixed solver-training dataset. Train the solver with GRPO (Shao et al., 2024): sample several responses to each question, evaluate Eq. (7), and compute advantages within that question's response group. At evaluation time the solver again receives only a question and its permitted tools. Thus training changes which outputs are rewarded, not the inputs available at inference.

What is compared within an update. Both phases center and scale rewards within a group, but they form those groups differently. In Phase A, a group contains proposer rollouts with the same prescribed hop label, potentially from different source documents. In Phase B, it contains solver rollouts for the same question. For a nonempty group $\mathcal{G}$ with rewards R_i, the shared normalization is

$$\bar{R}_{\mathcal{G}} = \frac{1}{|\mathcal{G}|}\sum_{i\in\mathcal{G}} R_i, \quad s^2_{\mathcal{G}} = \frac{1}{|\mathcal{G}|}\sum_{i\in\mathcal{G}}(R_i - \bar{R}_{\mathcal{G}})^2,$$
$$\widehat{A}_i = \frac{R_i - \bar{R}_{\mathcal{G}}}{s_{\mathcal{G}} + \delta_0}, \qquad \delta_0 > 0, \tag{8}$$

where $s_{\mathcal{G}}$ is the nonnegative square root. The optimization advantages $\widehat{A}_i$ are distinct from the evaluation indicators used below. A group with equal rewards, including a singleton, gives zero advantages; empty groups contribute no terms. These rewards and advantages are held fixed during each policy update. HRPO and GRPO retain their respective policy objectives (Yue et al., 2026; Shao et al., 2024).

Algorithm 2 in Appendix B makes the phase transition explicit: the fixed solver-training dataset is constructed *after* proposer training. The trained proposer supplies reward targets, and the solver recovers them through its own search. The evidence mechanism does not depend on a particular source-document sampler.

4 EXPERIMENTS

The experiments connect the research question to two observable outcomes. The first analysis asks whether emitting an evidence span is a sufficient proxy for support. The main evaluation then asks whether EVE-Agent improves the production of correct answers *with* supporting evidence relative to difficulty-based self-training. Both analyses use the same evaluation outputs.

4.1 SHARED EXPERIMENTAL SETUP

Systems and retrieval. All systems use Qwen2.5-3B-Instruct (Qwen et al., 2024). *Initial (no search)* and *Initial (search)* evaluate the pretrained instruction-tuned backbone without the two-phase training. *Dr. Zero* is the difficulty-based proposer–solver baseline (Yue et al., 2026). *EVE-Agent* adds the evidence-generation protocol, proposer utility and brevity rewards, and solver evidence reward. Search-enabled systems share the FlashRAG Wikipedia-2018 corpus (Jin et al., 2025b), E5-base-v2 text embeddings (Wang et al., 2022), and an inverted-file (IVF) index in the FAISS similarity-search library. Each query returns three passages, and rollouts permit at most five assistant turns. Only the explicitly named no-search baseline lacks the search tool.

Training. Each phase uses one node with eight B200 GPUs, a global batch size of 256, and 50 policy-gradient steps. We set $(\lambda_V, \lambda_B, \lambda_E) = (0.5, 0.1, 0.3)$, $L_{\max} = 256$ tokens, and $m = 5$ samples per verifier condition. Hop labels $1, 2, 3, 4$ are sampled in the ratio 4:3:2:1. Proposer prompts use the FlashRAG NQ–HotpotQA mixture. Appendix E collects the remaining training settings and the construction of the solver-training data. The comparison holds the backbone, retrieval pipeline, and nominal training schedule fixed. EVE-Agent additionally performs ten search-disabled decodes per eligible proposer rollout, so these settings are not a compute-matched comparison.

Benchmarks and decoding. We evaluate on Natural Questions (NQ) (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023), HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (2Wiki) (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). Each benchmark contributes 1,000 evaluation instances, except Bamboogle, for which all 125 instances are used. Each system produces one greedy output per instance and is asked to return both an answer and evidence.

Answer, support, and their joint outcome. For instance i, let $a_i^\star$ be the benchmark reference and $(\hat{a}_i, \hat{e}_i)$ the output. Define $A_i = \mathrm{EM}(\hat{a}_i, a_i^\star)$. An external GPT-4.1 judge receives the question, reference answer, and emitted span, and returns $S_i \in \{0, 1\}$ indicating whether that span supports recovery of the reference answer. The judge does not see the full retrieval trace and is not used in the training rewards. For $M \geq 1$ instances, report

$$\text{Answer} = \frac{1}{M}\sum_{i=1}^{M} A_i, \quad \text{Support} = \frac{1}{M}\sum_{i=1}^{M} S_i, \quad \text{Joint} = \frac{1}{M}\sum_{i=1}^{M} A_i S_i. \tag{9}$$

The primary metric is Joint: correctness and judged support must occur on the *same instance*. A correct answer with insufficient evidence does not count, nor does supporting evidence accompanying an incorrect answer. This distinguishes the desired output from success in either component alone. Joint is not the product of the two aggregate rates. All three metrics use the same evaluation instances, and Avg. is an unweighted mean over benchmarks.

Separating the learning signal from the assessment. Training measures agreement with proposer-generated targets (a, e), whereas evaluation uses benchmark references $a^\star$ and judges the emitted text. In particular, support is not scored by lexical overlap with the proposer's pseudo-reference. The judge receives the benchmark answer rather than the model's predicted answer, so the support question is anchored to the same external target as answer exact match. It assesses the text supplied in the evidence field, not its provenance in the retrieval trace. This separation tests whether evidence-oriented self-training transfers to the joint behavior in our research question.

4.2 Does evidence presence imply support?

This diagnostic tests whether producing an evidence field is enough to measure the behavior sought in our research question. It requires no additional training or decoding. Table 1 reports the five benchmarks with recorded evidence-presence rates. Within each system, its paired columns distinguish outputs containing a parsed span from outputs whose spans receive a positive support judgment. Appendix E.1 includes the remaining initial baseline.

Table 1: **Evidence presence and support are distinct.** Presence counts outputs with a parsed span; Support counts positive external judgments. All entries are rates. Appendix E.1 includes the remaining initial baseline.

	Initial (no search)		Dr. Zero		**EVE-Agent**	
Benchmark	Presence	Support	Presence	Support	Presence	Support
NQ	0.996	0.274	0.991	0.242	0.999	0.484
TriviaQA	0.996	0.424	0.977	0.289	0.991	0.582
HotpotQA	0.996	0.247	0.971	0.209	0.998	0.332
2Wiki	0.996	0.199	0.933	0.205	0.997	0.166
MuSiQue	0.993	0.093	0.911	0.086	0.996	0.101

Evidence presence is high for both trained systems, but presence and support remain distinct. Dr. Zero's frequent evidence outputs do not translate into comparably high judged support. EVE-Agent preserves high presence while improving support on most of these benchmarks. The diagnostic therefore establishes why the main evaluation must assess what the evidence supports, not merely whether an evidence field is returned.

4.3 DOES EVIDENCE FEEDBACK IMPROVE THE JOINT OUTCOME?

The main evaluation tests whether the two-phase evidence-aware loop improves correct answers accompanied by supporting evidence. It uses all seven benchmarks with the shared protocol unchanged. Table 2 places the joint metric first; the answer and support panels show its two components. The relevant comparison for the research question is EVE-Agent versus Dr. Zero, while the initial models provide context for performance before self-training.

Table 2: **EVE-Agent improves joint answer-and-evidence performance over Dr. Zero on every benchmark.** All values are rates; higher is better. Joint requires answer exact match and judged support on the same instance. Bold marks the best value within each column and panel. Avg. is the unweighted benchmark mean.

Method	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	Avg.
Joint answer-and-evidence rate (primary)								
Initial (no search)	0.052	0.180	0.069	0.048	0.016	0.015	**0.184**	0.081
Initial (search)	0.024	0.093	0.042	0.023	0.008	0.008	0.024	0.032
Dr. Zero	0.021	0.098	0.059	0.035	0.035	0.023	0.040	0.044
EVE-Agent	**0.242**	**0.342**	**0.264**	**0.152**	**0.070**	**0.038**	0.064	**0.167**
Answer exact match								
Initial (no search)	0.068	0.219	0.081	0.057	0.034	0.015	**0.200**	0.096
Initial (search)	0.032	0.125	0.055	0.029	0.013	0.010	0.024	0.041
Dr. Zero	0.069	0.257	0.134	0.110	0.077	**0.055**	0.104	0.115
EVE-Agent	**0.289**	**0.437**	**0.300**	**0.209**	**0.176**	0.050	0.088	**0.221**
Judged evidence-support rate								
Initial (no search)	0.274	0.424	0.153	0.247	0.199	0.093	**0.376**	0.252
Initial (search)	0.374	0.373	0.298	0.199	0.173	**0.103**	0.176	0.242
Dr. Zero	0.242	0.289	0.208	0.209	**0.205**	0.086	0.128	0.195
EVE-Agent	**0.484**	**0.582**	**0.392**	**0.332**	0.166	0.101	0.136	**0.313**

EVE-Agent improves the joint answer-and-evidence rate over Dr. Zero on every benchmark. It also achieves the highest average joint rate and the best joint rate on six of the seven benchmarks among all four systems. The consistency of the comparison with Dr. Zero is the central result: more outputs satisfy the answer and evidence requirements together after evidence is incorporated into the self-training loop.

The component panels show that this joint improvement accompanies the highest average answer accuracy and judged evidence support. Their benchmark-level rankings vary, whereas the joint improvement over Dr. Zero is consistent. These observations are compatible because the joint metric measures the overlap of correct answers and supporting spans on individual examples, rather than either marginal rate alone.

Taken together, the diagnostic and main evaluation support the intended role of evidence feedback. Evidence presence by itself is insufficient, but a curriculum that rewards evidence utility and reuses its spans yields better joint outputs under the evaluated setup. This is a comparison of complete training systems; it does not isolate the contribution of each added reward term.

5 Related work

Self-generated curricula and verification. Absolute Zero validates self-generated reasoning tasks with a code executor (Zhao et al., 2025), while R-Zero develops a curriculum through interaction between task generation and solving (Huang et al., 2026b). Dr. Zero brings proposer–solver self-evolution and hop-grouped optimization to search agents (Yue et al., 2026). Other frameworks enrich self-training with additional verification: SAGE uses specialized agent roles (Peng et al., 2026), EigenData generates interactive tool-use dialogues with executable checkers (Gao et al., 2026), and K2V combines knowledge-intensive data synthesis with reasoning verification (Yuan et al., 2026). EVE-Agent retains the Dr. Zero foundation and changes the feedback attached to each proposed task: the evidence span is evaluated for incremental utility and then reused for solver supervision.

Learning from informative hints. G-Zero uses a with–without hint log-probability shift to train a proposer, then forms preference pairs for generator training (Huang et al., 2026a). EVE-Agent instead estimates answer-agreement utility in search-disabled trials and reuses the generated evidence span as an *output target* for a search-enabled solver. This reuse makes evidence production, not only hint-conditioned response improvement, an explicit training objective.

Learning to retrieve and use information. Retrieval-augmented generation (Lewis et al., 2020), ReAct (Yao et al., 2023), and interleaved retrieval and reasoning (Trivedi et al., 2023) provide mechanisms for incorporating external information into generation. Self-RAG learns retrieval and reflection behavior (Asai et al., 2024); Search-R1 and R1-Searcher optimize search through reinforcement learning (Jin et al., 2025a; Song et al., 2025). A related principle appears in REPLUG, where a frozen language model supplies feedback for training a retriever (Shi et al., 2024). EVE-Agent instead uses a with–without evidence agreement difference to train the *task proposer*, and carries the scored span into solver training. The retriever itself remains fixed.

Evidence as an evaluation target. Answer verifiers (Cobbe et al., 2021; Shao et al., 2024) assess predictions, while evidence-annotated QA benchmarks (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022) and ALCE (Gao et al., 2023) also assess supporting content. Factual-consistency models can supervise citation generation (Aly et al., 2024). EVE-Agent instead learns from model-relative utility without a separate training-time semantic judge. Evaluation then tests answer correctness and judged evidence support jointly, keeping the learning signal distinct from the final assessment.

6 Discussion and limitations

EVE-Agent uses evidence to guide curriculum construction and to specify a solver output target. Its utility test measures agreement with a generated answer; downstream evaluation checks benchmark answers and separately judges evidence support. Increased target agreement is therefore not a certificate of truth or source attribution. Similarly, token overlap rewards recovery of one pseudo-reference, rather than every valid way to support an answer.

The results establish improved joint answer-and-evidence performance for the complete system under the reported backbone and retrieval setting, not the isolated effect of one reward term or a compute-efficiency advantage. Additional backbones, repeated training runs, and independent assessment of the evidence judge would test how broadly these gains hold. The supported conclusion is the consistent joint improvement over difficulty-based self-training in the reported comparisons.

7 Conclusion

EVE-Agent turns self-generated evidence into reusable training feedback. A frozen solver supplies evidence-utility feedback to the proposer; spans generated by the trained proposer then supervise evidence-producing question answering. This connects curriculum construction to the joint behavior posed in our research question without requiring external answer labels in the reward. Across the evaluated benchmarks, the resulting system consistently improves joint answer-and-evidence performance over Dr. Zero. The broader lesson is that self-training can learn from the information behind a generated answer, not only from the answer itself.

AI USE STATEMENT

The research question, core ideas, method, and experimental design were developed by the authors. Generative AI assisted drafting and editing, organization, notation, mathematical exposition and proof checking, presentation and interpretation of recorded results, and preparation of figures and references. It did not generate experimental measurements. Language models used in the method and GPT-4.1 used for evidence evaluation are described in Sections 3 and 4.1. The authors are responsible for the manuscript and its final verification.

REFERENCES

Rami Aly, Zhiqiang Tang, Samson Tan, and George Karypis. Learning to generate answers with citations via factual consistency models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11876–11896, 2024. doi: 10.18653/v1/2024.acl-long.641.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2310.11511.

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

Jiaxuan Gao, Jiaao Chen, Chuyi He, Shusheng Xu, Di Jin, and Yi Wu. From self-evolving synthetic data to verifiable-reward RL: Post-training multi-turn interactive tool-using agents. *arXiv preprint arXiv:2601.22607*, 2026.

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

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, 2020. doi: 10.18653/v1/2020.coling-main.580.

Chengsong Huang, Haolin Liu, Tong Zheng, Runpeng Dai, Langlin Huang, Jinyuan Li, Zongxia Li, Zhepei Wei, Yu Meng, and Jiaxin Huang. G-Zero: Self-play for open-ended generation from zero data. *arXiv preprint arXiv:2605.09959*, 2026a.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-Zero: Self-evolving reasoning LLM from zero data. In *International Conference on Learning Representations (ICLR)*, 2026b. arXiv:2508.05004.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025a.

Jiajie Jin, Yutao Zhu, Zhicheng Dou, Guanting Dong, Xinyu Yang, Chenghao Zhang, Tong Zhao, Zhao Yang, and Ji-Rong Wen. FlashRAG: A modular toolkit for efficient retrieval-augmented generation research. In *Companion Proceedings of the ACM on Web Conference 2025*, pp. 737–740, 2025b. doi: 10.1145/3701716.3715313.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, 2017. doi: 10.18653/v1/P17-1147.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, 2020.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9802–9822, 2023. doi: 10.18653/v1/2023.acl-long.546.

Yulin Peng, Xinxin Zhu, Chenxing Wei, Nianbo Zeng, Leilei Wang, Ying Tiffany He, and F. Richard Yu. SAGE: Multi-agent self-evolution for LLM reasoning. *arXiv preprint arXiv:2603.15255*, 2026.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, 2023. doi: 10.18653/v1/2023.findings-emnlp.378.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. REPLUG: Retrieval-augmented black-box language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8371–8384, 2024. doi: 10.18653/v1/2024.naacl-long.463.

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

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022. doi: 10.1162/tacl_a_00475.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 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)*, pp. 10014–10037, 2023. doi: 10.18653/v1/2023.acl-long.557.

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

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, 2018. doi: 10.18653/v1/D18-1259.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2210.03629.

Zhonghang Yuan, Zhefan Wang, Fang Hu, Zihong Chen, Jinzhe Li, Gang Li, Jie Ying, Huanjun Kong, Songyang Zhang, and Nanqing Dong. Knowledge-to-verification: Exploring RLVR for LLMs in knowledge-intensive domains. *arXiv preprint arXiv:2605.18261*, 2026.

Zhenrui Yue, Kartikeya Upasani, Xianjun Yang, Suyu Ge, Shaoliang Nie, Yuning Mao, Zhe Liu, and Dong Wang. Dr. Zero: Self-evolving search agents without training data. *arXiv preprint arXiv:2601.07055*, 2026.

Andrew Zhao, Yiran Wu, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute Zero: Reinforced self-play reasoning with zero data. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, 2025. doi: 10.52202/085713-3534.

A PROPERTIES OF THE REWARD SIGNALS

These results characterize agreement with a fixed generated target. They describe the reward signals, not the factual correctness of the target or convergence of policy optimization.

A.1 PROOF OF PROPOSITION 1

Proof. Fix the triple, solver checkpoint, and decoding protocols. All quantities in this proof are conditional on them. For $j = 1, \dots, m$, put $X_j = \mathbf{1}\{\hat{a}_+^{(j)} = a\}$ and $Y_j = \mathbf{1}\{\hat{a}_-^{(j)} = a\}$. These are Bernoulli variables with means p_+ and p_-, respectively. Thus

$$\mathbb{E}[\widehat{V}_m] = \frac{1}{m}\sum_{j=1}^{m}\big(\mathbb{E}[X_j] - \mathbb{E}[Y_j]\big) = p_+ - p_- = V,$$

$$\mathrm{Var}(\widehat{V}_m) = \frac{1}{m^2}\sum_{j=1}^{m}\big(\mathrm{Var}(X_j) + \mathrm{Var}(Y_j)\big) = \frac{p_+(1-p_+) + p_-(1-p_-)}{m} \le \frac{1}{2m}.$$

The variance equality uses independence of all $2m$ variables, and the bound uses $u(1-u) \le 1/4$ for $u \in [0,1]$. Both the population score and its estimate are differences of quantities in $[0,1]$, so they lie in $[-1,1]$. □

Sampling noise and variation across examples. For a random triple $Z = (q, a, e)$, keep the solver checkpoint and protocols fixed and draw samples conditionally as in Proposition 1. The law of total variance gives

$$\mathrm{Var}(\widehat{V}_m) = \mathbb{E}\Big[\mathrm{Var}(\widehat{V}_m \mid Z)\Big] + \mathrm{Var}(V(q,e,a)). \tag{10}$$

Only the first term is bounded by $1/(2m)$. Correlated draws require the corresponding covariance terms. A near-zero utility can reflect equally high or equally low agreement in the two conditions; it does not identify a unique cause.

A.2 EXPECTED DIFFICULTY REWARD

Lemma 1 (Expected difficulty and its maximizer)**.** *Let $n \ge 2$ be an integer and $k \sim \mathrm{Bin}(n,p)$, the binomial distribution with n trials and agreement probability $p \in [0,1]$. The expectation of Eq. (1) is*

$$\phi_n(p) = \frac{n}{n-1}(1-p)\left[1 - (1-p)^{n-1}\right]. \tag{11}$$

It is continuous, takes values in $[0,1]$, and vanishes at both endpoints. Its unique maximizer and maximum value are

$$p^\star = 1 - n^{-1/(n-1)} \in (0,1), \qquad \phi_n(p^\star) = n^{-1/(n-1)}. \tag{12}$$

Proof. For $k \in \{0, \dots, n\}$, the identity $\mathbf{1}\{0 < k < n\}(n-k) = (n-k) - n\mathbf{1}\{k = 0\}$ gives

$$\mathbb{E}[R^{\mathrm{DZ}}] = \frac{\mathbb{E}[n-k] - n\mathbb{P}(k=0)}{n-1} = \frac{n(1-p) - n(1-p)^n}{n-1} = \phi_n(p).$$

This polynomial is continuous and zero at the endpoints, with derivative

$$\phi_n'(p) = \frac{n}{n-1}\big[n(1-p)^{n-1} - 1\big].$$

The bracket decreases strictly from $n-1 > 0$ to -1 on $[0,1]$, with unique zero $p^\star$. Hence ϕ_n increases before $p^\star$ and decreases afterward. Substitution gives the maximum value. The range follows because every realized reward lies in $[0,1]$. □

This preferred rate concerns agreement with a, not its correctness. Holding the question, target, and solver outcomes fixed, changing the evidence leaves this reward unchanged. Evidence utility supplies the missing span-dependent signal.

B TRAINING ALGORITHMS

The two algorithms make the shared learning cycle and the evidence-specific changes explicit. Both freeze the solver during proposer training, then train the solver on tasks generated by the frozen proposer. The generated targets are used for reward computation, never as solver input hints. The algorithms summarize the learning flow; HRPO and GRPO retain the policy objectives of their respective references.

B.1 DR. ZERO: DIFFICULTY-BASED SELF-TRAINING

Algorithm 1 summarizes the baseline cycle described in Section 2 (Yue et al., 2026). Its proposer receives format and difficulty feedback; its solver receives answer-agreement feedback. The original paper denotes the format reward by r^{f}, which is left generic here. Valid pairs are those accepted by its task-generation protocol. Repeating this cycle lets the updated solver guide subsequent proposer training. EVE-Agent adds the evidence-utility and evidence-overlap terms shown in Algorithm 2.

Algorithm 1 Dr. Zero: difficulty-based proposer–solver learning cycle

Require: Proposer π^{pro}, solver π^{sol}, search engine $\mathcal{R}$, source sampler, hop distribution, training budgets, integers $n \geq 2$, $g \geq 2$, inherited format reward r^{f}

Train the proposer; freeze the solver
for each proposer-training batch **do**
  Sample source documents and hop labels; generate question–answer rollouts
  for each rollout τ_i with parsed pair (q_i, a_i) and hop label h_i **do**
    Set $R_i \leftarrow r^{\mathrm{f}}(\tau_i, h_i)$
    if the rollout yields a valid question–answer pair **then**
      Sample n independent search-enabled solver answers to q_i
      Compute k_i and add $R^{\mathrm{DZ}}(q_i, a_i; k_i)$ using Eq. (1)
    end if
  end for
  Compute advantages within hop groups; update the proposer with HRPO
end for
Train the solver; freeze the trained proposer
Generate valid (q, a) pairs and store a as a separate reward target
for each batch from the generated dataset **do**
  Sample g search-enabled solver rollouts per question, using only q as input
  Reward agreement of the parsed answer $\hat{a}$ with the generated target a
  Compute within-question advantages; update the solver with GRPO
end for
return Updated proposer and solver for the next self-evolution cycle

B.2 EVE-AGENT: EVIDENCE FEEDBACK AND REUSE

Algorithm 2 retains that cycle while adding the span e to the generated task. The two utility conditions share one frozen solver and disable search. After Phase A, the trained proposer generates the fixed dataset used in Phase B. Neither generated target is visible to the solver when it answers a training question.

Algorithm 2 EVE-Agent: learn an evidence-aware curriculum, then train the solver

Require: Proposer π^{pro}, solver π^{sol}, search engine $\mathcal{R}$, source sampler, hop distribution, training budgets, integers $n \geq 2$, $m \geq 1$, $g \geq 2$, reward parameters

Phase A: train the proposer; freeze the solver
for each proposer-training batch **do**
  Sample source documents and hop labels; generate one rollout per prompt
  for each rollout τ_i with parsed fields (q_i, a_i, e_i) and hop label h_i **do**
    Initialize $R_i \leftarrow \frac{1}{2} F_{\text{fmt}}(\tau_i, h_i)$
    if the rollout is eligible under Section 3.2 **then**
      Use n search-enabled solver trials to compute R_i^{DZ}
      Independently sample m search-disabled answers with e_i and m without it
      $R_i \leftarrow R_i + R_i^{\text{DZ}} + \lambda_V \widehat{V}_{m,i} + \lambda_B B(e_i)$
    end if
  end for
  Compute advantages within hop groups and update the proposer with HRPO
end for
Phase B: train the solver; freeze the trained proposer
Generate eligible triples with the proposer and store (a, e) as reward targets
for each batch from this fixed generated dataset **do**
  Sample g search-enabled solver rollouts per question, with only q as input
  Score each output $(\hat{a}, \hat{e})$ against (a, e) using Eq. (7)
  Compute within-question advantages and update the solver with GRPO
end for
return Trained solver

C A WORKED EXAMPLE OF EVIDENCE FEEDBACK AND REUSE

The following example is *constructed solely to illustrate the algorithm*. Its source text, trial outcomes, and trial count are not experimental observations or additional reported hyperparameters.

Generate a task. Suppose the source contains the following two sentences:

> The Lydian Archive is in Bellhaven. Bellhaven is in the Westreach region.

With a prescribed hop label of two, the proposer asks "In which region is the Lydian Archive located?" and proposes "Westreach" as the answer. Its evidence is the two source sentences, copied as one contiguous span. The question and answer are nonempty, and the normalized answer does not occur in the normalized question, so the triple is eligible for content rewards.

Measure difficulty and utility. For illustration, suppose one of four search-enabled solver trials agrees with "Westreach". The difficulty reward is then $(4-1)/(4-1) = 1$. Separately, suppose four of five search-disabled trials agree when given the evidence, and one of five agrees without it. The estimated utility is $4/5 - 1/5 = 3/5$. Its role is specific: it rewards the increase attributable to supplying these sentences to the fixed solver. The proposer's total reward is

$$\tfrac{1}{2}F_{\mathrm{fmt}} + 1 + \tfrac{3}{5}\lambda_V + \lambda_B B(e),$$

where the complete rollout determines F_{fmt} and the proposer tokenizer determines $B(e)$. The solver remains frozen while the proposer is updated.

Reuse the evidence. After Phase A, the trained proposer generates triples for the solver-training dataset. For a triple of this form, the solver sees only the question and its search tool. "Westreach" and the source span are stored separately for reward computation. A rollout that returns the target answer and the same normalized evidence receives $1 + \lambda_E$. Rewards are compared within the group of solver rollouts for this question, and only the solver is updated. At evaluation, no generated target is provided to the solver.

How a partial span is scored. The full evidence contains ten words after the normalization in Appendix D.2:

> lydian archive is in bellhaven bellhaven is in westreach region

Suppose the solver returns the correct answer but emits only the first source sentence. Its normalized evidence contains five words, all present in the pseudo-reference with the required multiplicities. The overlap count is therefore five, and

$$R_{\mathrm{evidence}} = \frac{2 \cdot 5}{5 + 10} = \frac{2}{3}, \qquad R^{\mathrm{sol}} = 1 + \frac{2}{3}\lambda_E.$$

Returning both sentences gives full token overlap and reward $1 + \lambda_E$. This calculation illustrates the distinction between the two reward terms: the answer term evaluates the answer string, while the evidence term grades recovery of the generated span. Whether an emitted span supports the benchmark answer is a separate evaluation-time judgment, not the token-overlap calculation used in training.

D OUTPUT PROTOCOL AND REWARD DETAILS

D.1 FORMAT REWARD

If parsing fails or the question or answer is empty, set $F_{\text{fmt}} = 0$. Otherwise the inherited score contains four equally weighted terms:

$$F_{\text{fmt}}(\tau, h) = \tfrac{1}{4}\big(1 + F_{\text{think}} + F_{\text{tool}}(h) + F_{\text{ans}}\big). \tag{13}$$

The constant term records successful structural parsing. Let T_{ass} be the number of assistant turns, T_{think} the number that begin with a planning step, T_{tc} the number of syntactically valid tool-call emissions, and T_{resp} the number of returned tool responses. These are nonnegative integers, with $T_{\text{think}} \le T_{\text{ass}}$. Then

$$F_{\text{think}} = \frac{T_{\text{think}}}{\max(1, T_{\text{ass}})}, \qquad F_{\text{tool}}(h) = \begin{cases} 1, & h = 1, \\ \min\{(1 + T_{\text{tc}})/h, 1\}, & h > 1,\ T_{\text{tc}} = T_{\text{resp}}, \\ 0, & h > 1,\ T_{\text{tc}} \neq T_{\text{resp}}. \end{cases} \tag{14}$$

For the answer sub-score, put $\tilde{a} = \text{NORM}(a)$ and let w be its whitespace-delimited word count. Let Σ concatenate the source document and tool responses observed in the rollout, and let C indicate whether $\tilde{a}$ occurs as a literal substring of $\text{NORM}(\Sigma)$. Define

$$F_{\text{ans}} = \begin{cases} 1, & \tilde{a} \in \{\text{yes}, \text{no}\} \text{ or } (C = 1 \text{ and } w \le 5), \\ \tfrac{1}{2}, & \tilde{a} \notin \{\text{yes}, \text{no}\},\ C = 1,\ 5 < w \le 10, \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

The cases are disjoint and every component is in $[0, 1]$, which establishes the stated range of F_{fmt}. These are structural and literal-match tests, not semantic evidence judgments. Eligibility is applied separately: a rollout that discloses its answer in the question receives only the format term even if that score is positive. Under the usual substring convention, an empty normalized answer also fails eligibility, since the empty string is a substring of every string.

D.2 ANSWER MATCHING AND TOKEN F1

Difficulty and utility compare parsed answers with the generated string a without the normalization used by the solver reward. Solver answer rewards compare $\text{NORM}(\hat{a})$ with $\text{NORM}(a)$; evaluation compares with a benchmark reference instead. Normalization lowercases the text, removes punctuation and English articles, and collapses whitespace.

For finite normalized strings x, y, use whitespace tokenization, with v ranging over the word tokens present in either string. Let $c_x(v)$ be the multiplicity of word v in x, $\ell_x = \sum_v c_x(v)$, and $o(x, y) = \sum_v \min\{c_x(v), c_y(v)\}$. To expose the empty-pair boundary explicitly, denote its convention by a fixed $c_\varnothing \in [0, 1]$ and write

$$\text{F1}_{\text{tok}}(x, y) = \begin{cases} \dfrac{2o(x, y)}{\ell_x + \ell_y}, & \ell_x + \ell_y > 0, \\ c_\varnothing, & \ell_x = \ell_y = 0. \end{cases} \tag{16}$$

For positive lengths, multiset precision is $o(x, y)/\ell_x$ and recall is $o(x, y)/\ell_y$. With nonzero overlap, their harmonic mean is the first branch of Eq. (16). No overlap, including exactly one empty string, gives zero. Since $0 \le 2o(x, y) \le \ell_x + \ell_y$, the score lies in $[0, 1]$ for every stated convention. The bounds in Appendix D.3 hold for every $c_\varnothing \in [0, 1]$. Evidence length $|e|$ in the brevity bonus, by contrast, is measured using the proposer tokenizer, not these whitespace tokens.

D.3 RANGES OF THE COMBINED REWARDS

Since $F_{\text{fmt}}, R^{\text{DZ}}, B \in [0, 1]$, $\widehat{V}_m \in [-1, 1]$, and all reward weights are nonnegative,

$$-\lambda_V \le \widehat{R}^{\text{pro}} \le \tfrac{3}{2} + \lambda_V + \lambda_B, \qquad 0 \le R^{\text{sol}} \le 1 + \lambda_E. \tag{17}$$

The solver bound follows from exact match and token F1 in $[0, 1]$, including the empty-pair convention in Appendix D.2. Ineligible proposer rewards lie in $[0, 1/2]$ and satisfy the same bound. The endpoints need not be jointly attainable; mean training rewards need not lie in $[0, 1]$.

E ADDITIONAL EXPERIMENTAL SETTINGS

Table 3 collects the phase-specific settings supplementing Section 4.1. All policies and scoring conditions use the same backbone. The verifier uses the initial solver checkpoint, with one assistant turn and no search calls in either condition.

Table 3: **Training configuration.** A dash denotes a phase-specific quantity that is not used in the other phase.

Setting	Phase A: proposer	Phase B: solver
Optimized policy	Proposer (HRPO)	Solver (GRPO)
Frozen policy	Initial solver	Trained proposer
Hardware	8×B200, one node	8×B200, one node
Policy-gradient steps	50	50
Global batch size	256	256
Micro-batch per GPU	2	8
Learning rate	10^{-6}	10^{-6}
Warmup ratio	0.03	0.03
Gradient-norm clipping	0.1	0.1
Kullback–Leibler regularization	Disabled	Disabled
Nested proposer grouping	None	—
Verifier samples per condition m	5	—
GRPO responses per question g	—	5
(λ_V, λ_B)	$(0.5, 0.1)$	—
λ_E	—	0.3
Brevity length scale $L_{\max}$	256 proposer tokens	—
Hop labels and sampling weights	$\{1, 2, 3, 4\}$; 4:3:2:1	Inherited generated tasks

Constructing the solver-training data. After Phase A, the frozen proposer generates five samples per prompt from the FlashRAG NQ–HotpotQA mixture, with hop labels assigned in the ratio stated in Section 4.1. Triples satisfying the eligibility rule in Section 3.2 form the fixed Phase B dataset. Each record contains the question and the separate reward targets (a, e); only the question is presented to the solver. Evidence is stored as a pseudo-reference, not an additional input.

E.1 COMPLETE EVIDENCE-PRESENCE MEASUREMENTS

Table 4 reports every available evidence-presence measurement, including the search-enabled initial model omitted from Table 1. It uses the same evaluation outputs, not an additional experiment. Presence was recorded for the five benchmarks shown. The full answer, support, and joint results already appear in Table 2.

Table 4: **Complete evidence-present rates.** Each entry is the fraction of outputs containing a parsed evidence span, not a semantic-support judgment.

Benchmark	Initial (no search)	Initial (search)	Dr. Zero	EVE-Agent
NQ	0.996	0.785	0.991	0.999
TriviaQA	0.996	0.723	0.977	0.991
HotpotQA	0.996	0.567	0.971	0.998
2Wiki	0.996	0.555	0.933	0.997
MuSiQue	0.993	0.585	0.911	0.996