SEEKA: SELF-EVOLVING SKILL OPTIMIZATION FOR ENTITY DISAMBIGUATION OVER KNOWLEDGE BASE

Anonymous authors
Paper under double-blind review

ABSTRACT

Entity disambiguation links an ambiguous mention to the entity it denotes in a knowledge base (KB). Existing learned rerankers require labeled data and retraining as domains or entity inventories change, while zero-shot large language model (LLM) rerankers avoid training but cannot improve from recurring errors when their instructions remain fixed. We introduce SEEKA, a nonparametric adaptation framework that co-evolves a KB-grounded synthetic curriculum and the external skill of a frozen LLM reranker (also called Solver). A Proposer generates challenging cases from target entities and their nearest confusers, while an audit verifies factual grounding and unique identification. The Solver's errors drive validation-gated updates to its instructions, memory, and decision workflow, while a learnability objective adapts the curriculum to the evolving Solver. Across the official held-out test sets of four benchmarks and four Solver families, SEEKA achieves higher top-1 accuracy than the corresponding zero-shot baseline in all 16 model–dataset comparisons, without modifying model parameters or candidate pools.

1 INTRODUCTION

Entity disambiguation links an ambiguous mention in text to the entity it denotes in a knowledge base (KB). For example, "Mercury" may refer to a planet, an element, a newspaper, or a record label, and the surrounding context must identify the intended entity. Reliable disambiguation supports search, question answering, and knowledge-graph construction, where an incorrect link propagates to downstream tasks.

A typical system first uses a *retriever* to select candidates from a large KB and then applies a *reranker* to choose the entity that best matches the mention and its context (Wu et al., 2020; Xu et al., 2023). This pipeline faces several forms of distribution shift: the intended entity may be rare or newly introduced, a more popular entity may share its name, and the KB may change over time. Sometimes none of candidates is correct, requiring the system to return NIL rather than force an incorrect link.

Existing approaches leave an adaptation gap. Learned rerankers capture fine-grained matching patterns but need labeled examples and parameter updates when the domain or entity inventory changes (Wu et al., 2020; Xu et al., 2023). Listwise large language model (LLM) rerankers compare candidate descriptions without task-specific updates (Sun et al., 2023; Haffoudhi et al., 2026), but a zero-shot reranker executes the same fixed instructions and cannot learn from recurring errors. It may rely on surface similarity, popularity, or stale parametric knowledge for rare and emerging entities.

Synthetic data offers an alternative source of supervision, but prior work uses generated examples as static benchmarks or augmentation data (Karpov et al., 2026; Remaki et al., 2026) that cannot follow the changing weaknesses of a particular reranker. Prompt optimizers revise instructions from execution feedback (Pryzant et al., 2023; Agrawal et al., 2026), skill-learning methods store reusable procedures outside model parameters (Nottingham et al., 2024; Yang et al., 2026), and SIPDO connects synthetic-data generation with prompt optimization (Yu et al., 2026). None of these jointly exploits KB structure to construct confusable entity sets, audits whether generated evidence uniquely identifies the target, and optimizes persistent memory and decision workflows beyond prompt text.

The KB itself provides the structure needed to close this gap. Entity descriptions supply factual evidence for generating mention contexts, while nearby entities provide realistic confusers. The generated examples should also evolve with the reranker: cases that are initially difficult may later

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.

become trivial, whereas cases that are too difficult provide little useful feedback. A useful curriculum should therefore target the reranker's evolving *competence frontier*, emphasizing examples that are challenging but learnable.

We introduce SEEKA, a Proposer–Solver co-evolution framework that adapts a frozen LLM reranker without updating its parameters. The **Solver** performs entity disambiguation by ranking retrieved candidates for a mention in context and selecting the top-ranked entity. The **Proposer** constructs a KB-grounded curriculum by selecting target entities and their nearest confusers and generating contexts that distinguish the targets. A separate LLM audit accepts an example only when its clues are supported by the target description, the mention directly denotes the target, and the evidence uniquely favors the intended decision.

Errors made by the Solver on synthetic examples and a small labeled development set guide an Optimizer that revises the Solver's external skill, including its instructions, retrieved memory, gates, and decision workflow. Candidate revisions are accepted only when they improve a protected validation fold. In the reverse direction, a learnability objective redirects the Proposer toward examples of intermediate difficulty for the current Solver. The curriculum and external skill thus evolve together while the LLM parameters and retrieved candidate pools remain fixed.

We evaluate SEEKA on the official held-out test sets of four benchmarks covering three distinct challenges: NIL prediction in NEL (Zhu et al., 2023), popularity shift in ShadowLink (Provatorova et al., 2021), and temporal shift in TempEL (Zaporojets et al., 2022) and DynamicER (Kim et al., 2024). The evaluation spans Grok 4.3 (xAI, 2026), GPT-5.6 Terra (OpenAI, 2026b), GPT-5.5 (OpenAI, 2026a), and the open-weight Llama 4 Maverick (Meta, n.d.). Within each comparison, the Solver, reasoning effort, evaluation examples, and candidate pool are fixed. SEEKA achieves higher top-1 accuracy than the corresponding zero-shot reranker in all 16 model–dataset comparisons.

Our experiments address four questions: whether the complete framework improves frozen LLM rerankers across the three forms of shift (**RQ1**), whether auditing improves curriculum validity and the resulting skill (**RQ2**), how curriculum scale, repeated cycles, and auxiliary-model choice affect optimization (**RQ3**), and whether a skill learned for one Solver transfers to another without re-optimization (**RQ4**).

Our contributions are:

- A KB-grounded curriculum procedure that generates disambiguation cases from target entities and their nearest confusers and audits them for factual support, direct denotation, and unique identification.
- A Proposer–Solver co-evolution framework that jointly adapts the curriculum and a persistent external skill, including instructions, memory, gates, and workflows, without updating LLM parameters.
- Empirical evidence that SEEKA improves frozen LLM rerankers across NIL prediction, popularity bias, and temporal shift, in all 16 comparisons spanning four benchmarks and four LLMs. Experiments show that learned skills transfer across Solvers without re-optimization.

2 METHOD

2.1 PROBLEM FORMULATION AND ROLES

Inference task. Let a knowledge-base catalog $\mathcal{K}$ contain an identifier, title, and description for each entity $e \in \mathcal{K}$. Given a mention-in-context x, a frozen retriever returns a candidate set $C_x \subset \mathcal{K}$. The decision set is $Y_x = C_x \cup \{\text{NIL}\}$ when abstention is allowed and $Y_x = C_x$ otherwise. A **Solver** f_s is a frozen LLM reranker that orders Y_x while following an external skill s; its prediction $\hat{y} = f_s(x, C_x)$ is the top-ranked decision, which should be the correct entity, or NIL when no candidate is correct.

Learning task. The LLM parameters, retriever, and candidate pools remain fixed, so the only learnable object is the skill $s = (p, M, G, W, c)$, where p is an instruction, M is retrieved memory such as an alias table, G contains evidence or confidence gates, W specifies a decision workflow, and c denotes optional deterministic executable components. Given $\mathcal{K}$, the frozen LLM, and a small labeled development set, the learning task is to find a skill s that maximizes the expected top-1

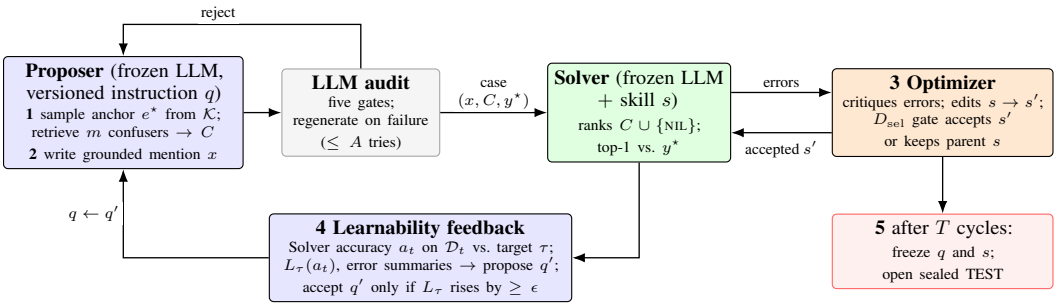

Figure 1: One SEEKA cycle. Steps 1–2: the Proposer generates KB-grounded cases and an LLM audit accepts or returns them (Section 2.2). Step 3: the frozen Solver ranks candidates and the Optimizer revises the external skill behind a selection gate (Section 2.3). Step 4: Solver accuracy and error summaries guide the Proposer update and the learnability reward gates its acceptance (Section 2.4). Step 5: all components are frozen before test evaluation.

accuracy $\mathbb{E}_{(x,y)\sim\mathcal{P}}\,\mathbf{1}[f_s(x, C_x) = y]$ on the target distribution $\mathcal{P}$. The training signal is therefore a *curriculum*: in each cycle t, a **Proposer** controlled by a versioned instruction q generates a labeled synthetic set $\mathcal{D}_t = \{(x_i, C_i, y_i^\star)\}$ from $\mathcal{K}$ (Section 2.2). The learning procedure is to search over skills: an **Optimizer** turns Solver errors on $\mathcal{D}_t$ and on development data into candidate revisions s', which replace s only when they improve a protected selection fold (Section 2.3). In turn, Solver accuracy on $\mathcal{D}_t$ and summaries of its errors update q, so that $\mathcal{D}_{t+1}$ targets the weaknesses of the current Solver (Section 2.4). Figure 1 shows one cycle.

Development data. When available, a small labeled development set is separated by normalized mention into a feedback fold D_{fb}, whose errors may be inspected by the Optimizer, and a selection fold D_{sel}, used to accept or reject candidate skills. Test examples and labels are not used to generate, revise, or select a skill.

2.2 KB-GROUNDED CURRICULUM AND AUDIT

For each case, the Proposer selects an intended entity $e^\star$ and retrieves its nearest KB neighbors as confusers. For a linkable case,

$$C = \{e^\star\} \cup \rho_m(e^\star), \qquad y^\star = e^\star, \tag{1}$$

where $\rho_m(e^\star)$ denotes the m nearest entities under the frozen retriever. For a NIL case, the intended entity is deliberately excluded:

$$C = \rho_m(e^\star), \qquad y^\star = \text{NIL}. \tag{2}$$

The number of confusers matches the candidate-set size of the benchmark.

After fixing $e^\star$ and C, the Proposer writes a natural mention and context whose identifying evidence is supported by the description of $e^\star$; Appendix E.3 shows a complete accepted case. Retriever scores, candidate positions, and the gold decision are not exposed in the generated text.

Generation and auditing are separate LLM calls (Figure 2). The auditor checks that (i) every identity-bearing clue is grounded in the target description, (ii) the mention directly denotes the intended entity, (iii) the evidence uniquely supports the target when present or NIL when absent, (iv) the context is natural prose without benchmark artifacts, and (v) at least one incorrect candidate is a plausible confuser. A case is accepted only if all five criteria pass. Otherwise, the failed criteria are returned to the Proposer for another attempt, up to a budget of A attempts, after which a reserve anchor replaces $e^\star$. The auditor cannot change the intended label or candidate set. For example, an accepted NEL case describes Bob Dylan's *New Morning* while excluding that album from the candidate set and including similarly titled albums as confusers; contextual clues identify the absent Dylan album, so the correct decision is NIL rather than the strongest title match (Appendix E.3).

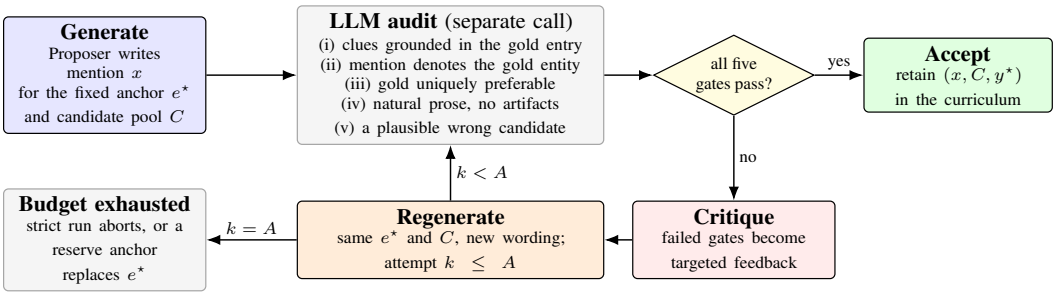

Figure 2: A generated case enters the curriculum only if it passes all audit criteria. Otherwise, the failed criteria are returned to the Proposer, which rewrites the context while keeping the intended entity and candidate set fixed.

2.3 Validation-gated skill optimization

The Solver ranks the decision set of each case in $\mathcal{D}_t$. The Optimizer receives the Solver's errors and representative successes on $\mathcal{D}_t$, together with Solver errors on D_{fb}. It identifies recurring problems—such as type mismatch, incorrect scope, temporal inconsistency, popularity bias (choosing the most prominent entity over the intended rarer one when they share the same mention name), or poor NIL calibration—and proposes a fixed number of skill revisions.

A candidate skill s' may change only the components declared for the experiment. It replaces the current skill s only when it improves the protected selection fold D_{sel}; otherwise, s is retained. When a component such as a gate or fusion weight is fitted on development examples, those examples are kept separate from the examples used for selection. The gate lets the Optimizer learn from specific errors without retaining revisions that do not generalize.

2.4 Learnability-driven Proposer adaptation

The curriculum should expose current weaknesses without becoming trivial or unsolvable. Let $a_t \in [0, 1]$ be the Solver's top-1 accuracy on $\mathcal{D}_t$, and let $\tau \in (0, 1)$ be a predeclared target accuracy. We define

$$L_\tau(a_t) = \begin{cases} a_t/\tau, & a_t \leq \tau, \\ (1 - a_t)/(1 - \tau), & a_t > \tau. \end{cases} \tag{3}$$

The reward is maximal at $a_t = \tau$ and decreases when the curriculum is too difficult or too easy. We use $\tau = 0.65$ for NEL and $\tau = 0.75$ for the other benchmarks, selected on development data and fixed within each run.

Because the reward is symmetric, the two signals play different roles. The sign of $a_t - \tau$ sets the direction of the update: when $a_t < \tau$, the revised instruction asks for clearer evidence and more separable confusers; when $a_t > \tau$, it asks for subtler evidence and harder confusers. The reward gates acceptance: a candidate instruction q' replaces q only if the curriculum it generates raises L_τ by at least a small margin ϵ. The Proposer receives a_t, τ, $L_\tau(a_t)$, and aggregate error summaries rather than raw test or development examples (Step 4 in Figure 1).

2.5 Complete loop and optional consolidation

Each cycle couples two updates: Solver errors revise the skill s, and Solver accuracy on $\mathcal{D}_t$ with error summaries revise q; the LLM and retriever stay frozen. Algorithm 1 in Appendix D summarizes one complete run. After the loop, an optional consolidation stage may fuse several Solver *views*, each a complete candidate ranking from one fixed Solver configuration, through reciprocal-rank fusion or a small deterministic gate fitted on synthetic and development data only (Appendix D). We report the number of views for each system, and all components are frozen before test evaluation.

3 EXPERIMENTS

3.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate SEEKA on four benchmarks covering three challenges in entity disambiguation. NEL (Zhu et al., 2023) tests NIL prediction: the system must abstain when none of the retrieved candidates is correct. ShadowLink (Provatorova et al., 2021) tests popularity bias by pairing less prominent entities with more popular entities that share the same name. TempEL (Zaporojets et al., 2022) and DynamicER (Kim et al., 2024) test temporal shift by linking mentions against KB snapshots whose entity inventories and descriptions change over time.

We use each benchmark's complete official held-out test split only for final evaluation. These splits contain 993 mentions for NEL, 2,709 for ShadowLink, 192,860 for TempEL, and 32,331 for DynamicER. No test labels are used to construct curricula, revise skills, select checkpoints, or tune hyperparameters.

Evaluation metric. Our primary metric is top-1 accuracy: the proportion of mentions for which the highest-ranked decision is correct. For NEL, we report the official overall accuracy (OAC), which counts both correctly linked mentions and correct NIL decisions. We also report the frozen retriever's Recall@100, an upper bound for any reranker using that candidate set. The result tables also report an *Improvement* column, which expresses each gain as a fraction of the accuracy still reachable below that bound, $(\text{SEEKA} - \text{SEEKA-BASE})/(\text{ceiling} - \text{SEEKA-BASE})$. A value of 10% therefore means that SEEKA recovers one tenth of the errors that the retrieved candidates still allow a reranker to correct.

Zero-shot baseline. The baseline, denoted SEEKA-BASE, is a zero-shot listwise LLM reranker: given the mention, its context, and the titles and descriptions of the retrieved candidates, the Solver returns a ranking, with any omitted candidates appended in retrieval order. Its prompt is fixed, so it cannot retain information from earlier errors.

Candidates and Solver models. For ShadowLink, TempEL, and DynamicER, we freeze the top-100 candidate lists retrieved by Cohere Embed v4.0 (Cohere Team, 2025), which achieved the highest mean Recall@100 among the retrievers considered in our preliminary experiments. NEL uses its released candidate lists together with the NIL option. Within every SEEKA-BASE/SEEKA pair, the candidate pool, Solver model, reasoning effort, and evaluation examples are identical.

We evaluate four Solver families: Grok 4.3, GPT-5.6 Terra, GPT-5.5, and the open-weight Llama 4 Maverick. The three primary closed-model experiments use MEDIUM reasoning effort. Llama 4 Maverick fills the Solver, Proposer, Optimizer, and auditor roles in its corresponding experiments. Complete model and component assignments are provided in Appendix B.

Development data and skill selection. All adaptation uses development mentions disjoint from the held-out test splits: NEL's validation split, TempEL's 2017–2018 validation snapshots, DynamicER's pre-evaluation periods, and, since ShadowLink has no validation split, its Neutral subset, which is disjoint from the scored Tail, Shadow, and Top subsets. Counts range from none, for two ShadowLink systems optimized on synthesized cases alone, to a few thousand; Table 8 in Appendix B lists them per system and explains how the budgets were set.

Statistical testing. Because SEEKA-BASE and SEEKA predict on exactly the same examples, we use the exact McNemar test for paired comparisons. Published task-trained systems are shown as reference points only, since they may use different candidate generators or training resources.

3.2 OVERALL EFFECTIVENESS

RQ1: Does SEEKA improve frozen LLM rerankers? Tables 1–4 report the per-benchmark results for the three closed Solvers. Table 6 in Appendix A summarizes the matched results across four benchmarks and four Solver families, including Llama 4 Maverick. SEEKA achieves higher top-1 accuracy than SEEKA-BASE in all 16 model–dataset comparisons. The consistency of these gains is important: the same framework improves decisions involving missing candidates, popularity bias,

Table 1: NEL TEST results (top-1 accuracy, %; OAC pools 660 linked and 333 NIL mentions). The released candidate lists contain every linked gold entity, so the ceiling is 100%. *task-trained. Bold marks the better value within a matched SEEKA-BASE/SEEKA pair. Improvement: gain of SEEKA over SEEKA-BASE relative to the remaining headroom, (SEEKA − SEEKA-BASE)/(ceiling − SEEKA-BASE) in %, with R@100 as the ceiling elsewhere (all tables).

Method	Model	Pipeline	Non-NIL	NIL	OAC	Improvement
Reference points						
Bi-Encoder (R@1)	Cohere Embed v4	Retrieval	70.61	70.27	70.49	–
CLINK (Zhu et al., 2023)	BERT-large*	+Rerank	86.97	89.19	87.71	–
SEEKA-BASE	Grok 4.3	+Rerank	85.30	**91.89**	87.51	–
SEEKA	Grok 4.3	+Rerank	**87.27**	90.99	**88.52**	8.1%
SEEKA-BASE	GPT-5.6 Terra	+Rerank	90.15	89.49	89.93	–
SEEKA	GPT-5.6 Terra	+Rerank	**91.06**	**90.69**	**90.94**	10.0%
SEEKA-BASE	GPT-5.5	+Rerank	90.30	89.79	90.13	–
SEEKA	GPT-5.5	+Rerank	**90.45**	**92.19**	**91.04**	9.2%

Table 2: ShadowLink TEST results (top-1 accuracy, %); the All column pools 2,709 mentions. *task-trained; †mean of the three published subsets. Bold marks the better value within a matched SEEKA-BASE/SEEKA pair. Improvement as in Table 1.

Method	Model	Pipeline	Tail	Shadow	Top	All	Improvement
Reference points							
Bi-Encoder (R@1)	Cohere Embed v4	Retrieval	90.68	59.85	51.99	67.48	–
Bi-Encoder (R@100)	Cohere Embed v4	Retrieval	99.78	95.24	94.14	96.38	–
VERBALIZED (Rücker & Akbik, 2025)	BERT-base*	+Rerank	98.20	66.30	65.90	76.80†	–
SEEKA-BASE	Grok 4.3	+Rerank	98.56	83.63	85.07	89.07	–
SEEKA	Grok 4.3	+Rerank	**98.78**	**85.62**	**85.95**	**90.11**	14.1%
SEEKA-BASE	GPT-5.6 Terra	+Rerank	98.67	84.62	85.95	89.74	–
SEEKA	GPT-5.6 Terra	+Rerank	**99.22**	**86.95**	**86.06**	**90.73**	15.0%
SEEKA-BASE	GPT-5.5	+Rerank	98.56	86.95	87.28	90.92	–
SEEKA	GPT-5.5	+Rerank	98.56	**88.83**	**87.61**	**91.66**	13.5%

and temporally changing entity inventories without modifying either the Solver parameters or the candidate pools. Measured against the retrieval ceiling, the three closed Solvers recover between 8% and 21% of the remaining correctable errors (Improvement column in Tables 1–4).

NEL: recognizing when no candidate is correct. NEL evaluates whether the Solver can distinguish between selecting a retrieved entity and returning NIL. Table 1 shows that SEEKA improves OAC by 1.01 points for Grok 4.3, 1.01 points for GPT-5.6 Terra, and 0.91 points for GPT-5.5. These improvements correspond to nine or ten additional correct decisions among 993 mentions.

The models improve through different tradeoffs: Grok 4.3 recovers more linked mentions (563 to 576 correct links) while losing three correct NIL decisions, whereas GPT-5.6 Terra and GPT-5.5 improve on both linked and NIL cases. The learned skill is thus not merely more willing to abstain or to link; it can improve the balance between the two decisions.

ShadowLink: resisting popularity bias. ShadowLink separates Tail mentions, Shadow mentions that share their surface form with a more popular entity, and Top mentions referring to the popular entity itself. As shown in Table 2, SEEKA improves overall accuracy by 1.04 points for Grok 4.3, 0.99 for GPT-5.6 Terra, and 0.74 for GPT-5.5. Llama 4 Maverick improves by 1.07 points.

The largest gains occur on the Shadow subset: +1.99 points for Grok 4.3, +2.33 for GPT-5.6 Terra, and +1.88 for GPT-5.5. Tail accuracy is already close to its retrieval ceiling, leaving little room for improvement. The concentration of gains on Shadow mentions indicates that the learned skills help the Solver prioritize contextual evidence over the popularity of a competing entity. The improvements for Grok 4.3, GPT-5.6 Terra, and Llama 4 Maverick are significant under the exact McNemar test ($p < 10^{-3}$, $p < 10^{-3}$, and $p = .011$, respectively).

Table 3: TempEL TEST results (top-1 accuracy, %) by entity category and snapshot year; the All column pools 192,860 mentions. *task-trained; †reconstructed from published subset values (CYCLE uses its own candidates). Bold marks the better value within a matched SEEKA-BASE/SEEKA pair; Improvement as in Table 1; Recall@100 (R@100) of the frozen retriever bounds every reranker row.

Method	Model	Pipeline	Continual Entities				New Entities				All	Improvement
			2019	2020	2021	2022	2019	2020	2021	2022		
Reference points												
Bi-Encoder (R@1)	Cohere Embed v4	Retrieval	1.49	1.48	1.46	1.51	5.45	4.83	2.21	4.48	1.57	–
Bi-Encoder (R@100)	Cohere Embed v4	Retrieval	28.77	28.69	28.24	27.97	54.76	49.10	43.10	52.34	29.06	–
CYCLE (Zhang et al., 2024)	BERT-base*	+Rerank	28.60	28.90	29.70	30.10	18.40	19.10	19.20	22.20	29.04†	–
SEEKA-BASE	Grok 4.3	+Rerank	21.22	21.27	20.86	20.58	41.93	36.69	33.31	41.24	21.50	–
SEEKA	Grok 4.3	+Rerank	**22.09**	**22.21**	**21.78**	**21.55**	**43.72**	**38.55**	**34.28**	**42.90**	**22.45**	12.5%
SEEKA-BASE	GPT-5.6 Terra	+Rerank	21.01	21.04	20.77	20.50	42.28	36.07	32.76	41.38	21.35	–
SEEKA	GPT-5.6 Terra	+Rerank	**21.68**	**21.68**	**21.27**	**21.12**	**43.86**	**39.03**	**33.86**	**43.66**	**22.00**	8.4%
SEEKA-BASE	GPT-5.5	+Rerank	23.27	23.33	22.88	22.68	48.55	42.21	36.62	46.69	23.65	–
SEEKA	GPT-5.5	+Rerank	**24.29**	**24.30**	**23.83**	**23.64**	**49.45**	**43.59**	**38.28**	**48.69**	**24.65**	18.4%

Table 4: DynamicER TEST results (top-1 accuracy, %) by domain and two-month period; the All column pools 32,331 mentions. *task-trained; †reconstructed from published totals. Bold marks the better value within a matched SEEKA-BASE/SEEKA pair (ties unmarked). Improvement as in Table 1.

Method	Model	Pipeline	Baseball			Soccer			All	Improvement
			1112	0102	0304	1112	0102	0304		
Reference points										
Bi-Encoder (R@1)	Cohere Embed v4	Retrieval	33.67	37.47	32.50	32.20	32.17	31.44	32.26	–
Bi-Encoder (R@100)	Cohere Embed v4	Retrieval	87.60	94.04	89.46	91.24	91.19	93.18	91.48	–
TempCCA (Kim et al., 2024)	BERT-base*	+Rerank	88.26	88.07	89.20	88.96	86.19	87.00	87.60†	–
SEEKA-BASE	Grok 4.3	+Rerank	**85.49**	92.73	88.32	89.07	88.84	90.61	89.30	–
SEEKA	Grok 4.3	+Rerank	85.27	92.73	**88.43**	**89.58**	**89.13**	**91.05**	**89.64**	15.7%
SEEKA-BASE	GPT-5.6 Terra	+Rerank	85.83	92.92	88.43	89.43	89.26	91.05	89.66	–
SEEKA	GPT-5.6 Terra	+Rerank	**86.05**	**93.20**	**88.52**	**89.71**	**89.48**	**91.33**	**89.90**	13.0%
SEEKA-BASE	GPT-5.5	+Rerank	85.83	93.01	88.35	89.68	89.76	91.29	89.94	–
SEEKA	GPT-5.5	+Rerank	**85.94**	**93.29**	**88.63**	**90.14**	**89.97**	**91.66**	**90.27**	21.3%

TempEL: linking against evolving KB snapshots. TempEL distinguishes Continual entities, which persist across KB snapshots, from New entities, which first appear in a given year. Table 3 shows that SEEKA improves pooled accuracy by 0.95 points for Grok 4.3, 0.65 for GPT-5.6 Terra, and 1.00 for GPT-5.5. For these three Solvers, the improvement holds in every year and for both entity categories. The paired gains are statistically significant, with $p \leq 2 \times 10^{-64}$.

The pooled score is dominated by 187,060 Continual-entity mentions, for which Recall@100 is only 28.0–28.8%, so retrieval rather than reranking is the primary bottleneck; New entities have much higher candidate recall, and there the LLM rerankers exceed the task-trained CYCLE reference. Llama 4 Maverick also improves in aggregate by 0.12 points ($p = .0013$), although not on every New-entity slice.

DynamicER: resolving rapidly changing aliases. DynamicER links baseball and soccer mentions against KB snapshots updated every two months. Correct decisions often depend on knowing which aliases identify an entity during a particular period. Table 4 shows improvements of 0.34 points for Grok 4.3, 0.24 for GPT-5.6 Terra, and 0.33 for GPT-5.5. Each improvement is significant under the exact McNemar test ($p \leq 2 \times 10^{-9}$).

Unlike TempEL, DynamicER has high Recall@100, so most remaining errors occur during reranking rather than retrieval. The improvements therefore show that external instructions, temporal alias memory, and decision gates can recover additional correct entities even when the zero-shot Solver is already close to the candidate-recall ceiling.

Taken together, the results answer RQ1 affirmatively: SEEKA improves every tested Solver–benchmark combination while holding the Solver parameters, reasoning effort, evaluation examples, and candidate sets fixed.

Table 5: Controlled ablations (%). Panels (a)–(c) use the TempEL 2022 New Entities TEST subset; panel (d) uses the complete DynamicER TEST.

Audit	Cycle	Raw pass	Refined	TEST
On	1	95.32	100.00	45.59
On	2	95.22	100.00	**46.00**
Off	1	95.32	95.32	45.59
Off	2	96.72	96.72	45.59

(a) LLM audit.

Prop./Opt. model	TEST
GPT-5.5	**48.69**
Grok 4.3	47.10
GPT-5.6 Terra	46.76

(b) Proposer/Optimizer/audit model.

Cases/cycle	Raw pass	VALID	TEST
100	92.67	86.00	45.93
500	86.87	87.25	46.69
1,000	82.57	87.00	47.10
5,000	83.65	86.25	**48.69**
10,000	83.68	86.00	45.93

(c) Curriculum size (refined pass 100%).

Terra skill (DynamicER)	TEST
SEEKA-BASE	89.66
GPT-5.5 transfer	89.90
Terra target-specific	**89.97**

(d) Cross-model skill transfer.

3.3 CONTROLLED ANALYSES

Unless otherwise specified, these analyses use the TempEL 2022 New Entities test subset (1,450 mentions) with a GPT-5.5 Solver at MEDIUM reasoning effort, frozen candidate lists, 99 nearest KB neighbors per synthesized anchor, and one shared random seed; the baseline accuracy is 46.69. The DynamicER transfer analysis uses its complete held-out test split.

RQ2: What does the LLM audit contribute? Table 5(a) compares two-cycle runs with and without the audit while holding the models, seed, candidate sets, and synthesis budget fixed. Before refinement, approximately 95% of generated cases satisfy all audit criteria. With auditing enabled, the Proposer revises rejected cases until 100% of the admitted curriculum satisfies the factual-support, direct-denotation, and unique-identification checks. Without the audit, the pass rate remains between 95.3% and 96.7%.

The audit also raises the second-cycle test checkpoint from 45.59 to 46.00, corresponding to six additional correct predictions. This controlled configuration does not exceed the zero-shot baseline, so its strongest evidence concerns curriculum validity; its downstream accuracy benefit is positive but modest within the two-cycle budget.

RQ3(a): How sensitive is optimization to the auxiliary models? Table 5(b) fixes the GPT-5.5 Solver and varies the model used for the Proposer, Optimizer, and auditor roles. GPT-5.5 obtains 48.69, Grok 4.3 obtains 47.10, and GPT-5.6 Terra obtains 46.76, compared with the shared zero-shot score of 46.69. Because all three auxiliary roles change together, this experiment measures their combined effect rather than attributing the difference to any single role.

RQ3(b): How much synthetic data is useful? Table 5(c) varies the number of generated cases per cycle from 100 to 10,000. Accuracy rises from 45.93 with 100 cases to 48.69 with 5,000 cases, but returns to 45.93 with 10,000 cases. The 100- and 10,000-case conditions select the same skill and consequently obtain identical validation and test scores.

The relationship is not monotonic: more accepted examples do not guarantee a better skill under a fixed Optimizer budget, which supports adapting the curriculum to its usefulness rather than maximizing its size.

RQ3(c): Do additional Proposer–Solver cycles help? We further follow a ten-cycle trajectory with 1,000 newly synthesized cases per cycle, in which the initial skill and four candidates per cycle yield 41 evaluated skills. Validation-gated optimization changes the selected skill in four of the ten cycles, and the test accuracy of the selected checkpoints is not monotonic (47.10, 47.24, 46.97, and 47.52 at those cycles). A separate protected selection over the complete set of 41 frozen skills

identifies a skill scoring 47.93, an improvement of 1.24 points, or 18 additional correct predictions, over the zero-shot baseline.

Later cycles can thus discover stronger skills, but more cycles do not ensure monotonic improvement, and protected validation is needed to separate useful revisions from ones that fit the current feedback batch.

RQ4: Does a learned skill transfer between Solvers? Table 5(d) transfers a skill learned with GPT-5.5 to a GPT-5.6 Terra Solver on DynamicER without further optimization. The GPT-5.6 Terra zero-shot baseline obtains 89.66. Applying the GPT-5.5-developed skill raises accuracy to 89.90, while a skill optimized specifically for GPT-5.6 Terra reaches 89.97. The transferred skill therefore retains 77% of the gain achieved by target-specific optimization.

This result answers RQ4 affirmatively: most of the improvement transfers across the two Solver families without re-optimization, so the external skill captures reusable disambiguation strategies rather than model-specific prompt adjustments.

4 Related Work

Classical and neural entity linking. Early systems ranked KB candidates by mention context, entity popularity, and document-level coherence (Cucerzan, 2007; Ratinov et al., 2011), later extended to multilingual (Sil et al., 2018) and multimodal (Yao et al., 2024) settings. Neural models learned contextual representations and compatibility functions from labeled data (Ganea & Hofmann, 2017; Kolitsas et al., 2018), and dense retrieval and reranking scaled them to large inventories (Wu et al., 2020; Xu et al., 2023). Adapting their parameters to new domains or KBs requires further supervision.

LLMs for entity linking. LLMs can compare candidate descriptions through prompting, enabling zero-shot listwise reranking without parameter updates (Sun et al., 2023; Haffoudhi et al., 2026). A fixed prompt, however, cannot retain lessons from recurring errors, and parametric knowledge is unreliable for rare or newly introduced entities. SEEKA keeps the LLM frozen and adapts an external skill instead.

Synthetic supervision and adaptive curricula. Synthetic data has expanded entity-linking coverage and built evaluation examples (Karpov et al., 2026; Remaki et al., 2026), but such datasets are generated once and do not respond to the weaknesses of a particular reranker. Our Proposer builds cases from target entities and their nearest KB confusers, audits their grounding and unique identification, and steers generation toward the Solver's competence frontier.

Prompt and skill optimization. Automatic prompt optimizers revise instructions from execution feedback (Pryzant et al., 2023; Agrawal et al., 2026), and skill-learning methods keep reusable procedures outside model parameters (Nottingham et al., 2024; Yang et al., 2026). SIPDO jointly evolves synthetic data and prompts (Yu et al., 2026) but neither exploits KB structure nor optimizes persistent memory and decision workflows. SEEKA combines these directions: a KB-grounded curriculum adapts to the Solver, while validation-gated updates improve the Solver's external skill without changing its parameters or candidate pool.

5 Conclusion

We introduced SEEKA, a Proposer–Solver co-evolution framework in which a KB-grounded synthetic curriculum and the external skill of a frozen LLM reranker improve together: a KB-grounded, audited curriculum exposes Solver errors that drive validation-gated updates to instructions, memory, gates, and workflows, without parameter updates or changes to the candidate pools. Across four benchmarks and four LLM families, SEEKA improves over the zero-shot reranker in all 16 model–dataset comparisons, spanning NIL prediction, popularity bias, and temporal KB shift. More broadly, when a task provides structured knowledge and verifiable feedback, that structure can generate its own curriculum, and a persistent external skill can turn a zero-shot reranker into an adaptive Solver. Future work includes relaxing retrieval ceiling and reducing reliance on labeled development data.

ETHICS STATEMENT

This work uses only publicly released entity-linking benchmarks (NEL, ShadowLink, TempEL, and DynamicER) and public knowledge-base snapshots; no human subjects, private data, or new annotation were involved. The synthetic curriculum is generated from knowledge-base descriptions and audited for factual grounding, but generated text and learned skills can inherit biases of the underlying language models and catalogs, including uneven coverage of long-tail entities and non-English names.

REPRODUCIBILITY STATEMENT

Section 3.1 specifies the benchmarks, official test splits, frozen retriever and candidate pools, Solver models, and reasoning efforts used in every comparison, and Section 3.3 states the fixed configuration of the controlled ablations. Appendix B lists the composition of every reported system, including the skill components that were revised, the number of fused views, and the development data used for feedback and selection. Appendix C gives the audit implementation, Appendix D states the complete algorithm and consolidation procedure, and Appendix E reproduces the role prompt templates, examples of learned skills, and an accepted curriculum case.

AI USE STATEMENT

In this work, large language models are the object of study: hosted LLM APIs act as the Proposer, Solver, Optimizer, and auditor exactly as described in the paper, which includes generating the synthetic curriculum. Additionally, we used generative AI tools to edit the manuscript's language. We have reviewed all AI-assisted work: all experimental design, code, analyses, and claims were made and verified by the authors, and every reported number was checked against the underlying run records. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Lakshya A. Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J. Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=RQm2KQTM5r.

Cohere Team. Introducing Embed 4: Multimodal search for business, 2025. URL https://cohere.com/blog/embed-4. Published April 15, 2025. Accessed: 2026-09-18.

Silviu Cucerzan. Large-scale named entity disambiguation based on Wikipedia data. In Jason Eisner (ed.), *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 708–716, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/D07-1074.

Octavian-Eugen Ganea and Thomas Hofmann. Deep joint entity disambiguation with local neural attention. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel (eds.), *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2619–2629, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1277. URL https://aclanthology.org/D17-1277.

Samy Haffoudhi, Fabian M. Suchanek, and Nils Holzenberger. LELA: An LLM-based entity linking approach with zero-shot domain adaptation. In *The Semantic Web – ISWC 2026: 25th International Semantic Web Conference*, Lecture Notes in Computer Science, Bari, Italy, 2026. Springer. To appear. Preprint: https://arxiv.org/abs/2601.05192.

Ilia Karpov, Alexander Kirillovich, Elisaveta Goncharova, Andrey Parinov, Alexander Chernyavskiy, Dmitry Ilvovsky, Natalia Semenova, Artyom Sosedka, Ekaterina Lisitsyna, and Mikhail Belkin. SynEL: A synthetic benchmark for entity linking. *PLOS ONE*, 21(1):e0339468, 2026. doi: 10.1371/journal.pone.0339468. URL https://doi.org/10.1371/journal.pone.0339468.

Jinyoung Kim, Dayoon Ko, and Gunhee Kim. DynamicER: Resolving emerging mentions to dynamic entities for RAG. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 13752–13770, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.762. URL https://aclanthology.org/2024.emnlp-main.762/.

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. End-to-end neural entity linking. In Anna Korhonen and Ivan Titov (eds.), *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pp. 519–529, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/K18-1050. URL https://aclanthology.org/K18-1050.

Meta. Unmatched performance and efficiency | Llama 4, n.d. URL https://developer.meta.com/ai/models/llama-4/. Accessed September 18, 2026.

Kolby Nottingham, Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Sameer Singh, Peter Clark, and Roy Fox. Skill set optimization: reinforcing language model behavior via transferable skills. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

OpenAI. GPT-5.5 system card, April 2026a. URL https://deploymentsafety.openai.com/gpt-5-5/gpt-5-5.pdf. Published April 23, 2026. Accessed September 18, 2026.

OpenAI. GPT-5.6 system card, 2026b. URL https://deploymentsafety.openai.com/gpt-5-6/gpt-5-6.pdf. Accessed: 2026-09-18.

Vera Provatorova, Samarth Bhargav, Svitlana Vakulenko, and Evangelos Kanoulas. Robustness evaluation of entity disambiguation using prior probes: the case of entity overshadowing. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 10501–10510, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.820. URL https://aclanthology.org/2021.emnlp-main.820.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7957–7968, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.494. URL https://aclanthology.org/2023.emnlp-main.494.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. Local and global algorithms for disambiguation to Wikipedia. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea (eds.), *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 1375–1384, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1138.

Adam Remaki, Christel Gérardin, Eulàlia Farré-Maduell, Martin Krallinger, and Xavier Tannier. SynCABEL: Synthetic contextualized augmentation for biomedical entity linking. In Diego Calvanese (ed.), *Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence, IJCAI-26*, pp. 6859–6867. International Joint Conferences on Artificial Intelligence Organization, August 2026. doi: 10.24963/ijcai.2026/763. URL https://doi.org/10.24963/ijcai.2026/763.

Susanna Rücker and Alan Akbik. Evaluating design decisions for dual encoder-based entity disambiguation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15685–15701, 2025.

Avi Sil, Heng Ji, Dan Roth, and Silviu-Petru Cucerzan. Multi-lingual entity discovery and linking. In Yoav Artzi and Jacob Eisenstein (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pp. 22–29, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-5008. URL https://aclanthology.org/P18-5008.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. Is ChatGPT good at search? investigating large language models as re-ranking agents. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 14918–14937, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.923. URL https://aclanthology.org/2023.emnlp-main.923.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. Scalable zero-shot entity linking with dense entity retrieval. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6397–6407, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.519. URL https://aclanthology.org/2020.emnlp-main.519.

xAI. Grok 4.3 Beta, April 2026. URL https://grok.com/release-notes/apr-17-2026. Released April 17, 2026. Accessed September 18, 2026.

Zhenran Xu, Yulin Chen, Baotian Hu, and Min Zhang. A read-and-select framework for zero-shot entity linking. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 13657–13666, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.912. URL https://aclanthology.org/2023.findings-emnlp.912.

Yifan Yang, Ziyang Gong, Weiquan Huang, Qihao Yang, Ziwei Zhou, Zisu Huang, Yan Li, Xuemei Gao, Qi Dai, Bei Liu, Kai Qiu, Yuqing Yang, Dongdong Chen, Xue Yang, and Chong Luo. SkillOpt: Executive strategy for self-evolving agent skills, 2026. URL https://arxiv.org/abs/2605.23904.

Barry Yao, Sijia Wang, Yu Chen, Qifan Wang, Minqian Liu, Zhiyang Xu, Licheng Yu, and Lifu Huang. Ameli: Enhancing multimodal entity linking with fine-grained attributes. In Yvette Graham and Matthew Purver (eds.), *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2816–2834, St. Julian's, Malta, March 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.eacl-long.172. URL https://aclanthology.org/2024.eacl-long.172/.

Yaoning Yu, Ye Yu, Peiyan Zhang, Kai Wei, Haojing Luo, and Haohan Wang. Sipdo: Closed-loop prompt optimization via synthetic data feedback. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 142701–142730, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/e6da278cdd692077b7e4a99d55573d9c-Paper-Conference.pdf.

Klim Zaporojets, Lucie-Aimée Kaffee, Johannes Deleu, Thomas Demeester, Chris Develder, and Isabelle Augenstein. Tempel: linking dynamically evolving and newly emerging entities. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Pengyu Zhang, Congfeng Cao, Klim Zaporojets, and Paul Groth. Cycle: Cross-year contrastive learning in entity-linking. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 3197–3206, 2024.

Fangwei Zhu, Jifan Yu, Hailong Jin, Lei Hou, Juanzi Li, and Zhifang Sui. Learn to not link: Exploring NIL prediction in entity linking. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 10846–10860, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.690. URL https://aclanthology.org/2023.findings-acl.690.

A SUPPLEMENTARY RESULTS

Table 6 summarizes the matched TEST accuracy of every Solver on every benchmark (Section 3.2).

Table 6: Cross-model TEST accuracy (%), SEEKA-BASE→SEEKA. The TempEL column pools all 192,860 Continual and New Entities mentions.

Model	NEL OAC	ShadowLink	TempEL	DynamicER
Llama 4 Maverick	82.07→82.38	83.98→85.05	17.42→17.54	85.31→86.15
Grok 4.3	87.51→88.52	89.07→90.11	21.50→22.45	89.30→89.64
GPT-5.6 Terra	89.93→90.94	89.74→90.73	21.35→22.00	89.66→89.90
GPT-5.5	90.13→91.04	90.92→91.66	23.65→24.65	89.94→90.27

A.1 OPEN-WEIGHT LLAMA 4 MAVERICK RESULTS

We additionally evaluate SEEKA-BASE and SEEKA with the open-weight Llama 4 Maverick Solver on the same frozen candidates, with Llama 4 Maverick in every role. Because Llama 4 Maverick exposes no reasoning-effort setting, the Solver decodes at temperature 0 with top-p of 0.9, and the Proposer samples at temperature 0.7. The NEL Llama skill was selected on a 160-row tuning set and confirmed on 808 further VALID rows that had also been used to reject an earlier candidate; the ShadowLink Llama skill was accepted on synthetic feedback without a held selection fold. Table 7 reports the complete TEST results. SEEKA improves overall accuracy on NEL, ShadowLink, TempEL, and DynamicER by 0.30, 1.07, 0.12, and 0.84 points, respectively.

NEL

Method	Non-NIL	NIL	Overall (OAC)	Improvement
SEEKA-BASE	**86.97**	72.37	82.07	–
SEEKA	84.55	**78.08**	**82.38**	1.7

ShadowLink

Method	Tail	Shadow	Top	Overall	Improvement
SEEKA-BASE	97.11	75.77	79.09	83.98	–
SEEKA	**97.23**	**77.88**	**80.09**	**85.05**	8.6

TempEL

Method	Continual Entities				New Entities				Overall	Improvement
	2019	2020	2021	2022	2019	2020	2021	2022		
SEEKA-BASE	17.08	17.20	17.04	16.70	**33.31**	30.21	26.90	**32.41**	17.42	–
SEEKA	**17.23**	**17.37**	**17.17**	**16.80**	32.48	**30.48**	**27.03**	31.72	**17.54**	1.1

DynamicER

Method	Baseball			Soccer			Overall	Improvement
	1112	0102	0304	1112	0102	0304		
SEEKA-BASE	84.39	90.49	86.72	85.08	83.55	86.71	85.31	–
SEEKA	**84.72**	**90.96**	**87.06**	**85.92**	**84.69**	**87.47**	**86.15**	13.6

Table 7: Llama 4 Maverick TEST results on all four datasets (top-1 accuracy, %). Bold marks the better value within a matched SEEKA-BASE/SEEKA pair. Improvement as in Table 1.

On NEL, the three-hit net gain combines 19 additional correct NIL decisions with 16 lost linked decisions ($p = .78$) and leaves the final Llama result below CLINK, the task-trained reference system of the NEL benchmark (Zhu et al., 2023). ShadowLink improves by 29 hits on all three subsets. TempEL gains 240 hits overall (2,882 fixes, 2,642 regressions), but accuracy decreases on the 2019 and 2022 New Entities slices. DynamicER gains 271 hits across all six slices (498 fixes, 227 regressions) but remains 1.45 points below TempCCA, the task-trained model released with DynamicER, which fine-tunes a BERT-base bi-encoder with temporal segmented clustering

and continual adaptation (Kim et al., 2024). A likely reason is that a task-trained model can encode domain- and time-specific evidence in its parameters, whereas SEEKA keeps Llama frozen and modifies only its external skill; skill evolution improves recurring decision rules without closing the underlying Solver-capability gap.

B CONFIGURATION OF REPORTED SYSTEMS

Table 8 records, for every SEEKA result in Section 3.2 and Appendix A.1, the models besides the Solver, the deployed skill components, the number of ranking views, and the development data used to revise, fit, and select the skill. Within each matched pair, SEEKA-BASE uses the same Solver, reasoning effort, and candidates with a single zero-shot ranking view.

The Development data column counts every labeled mention outside the test split that a system consumed, by role: *feedback* mentions, whose Solver errors the Optimizer inspects; *selection* mentions, which accept or reject each candidate revision; and *certification* mentions, scored once to confirm the final choice. Feedback and selection mentions are always disjoint. The source is fixed per benchmark by what the benchmark releases outside its test split, and the fold sizes were fixed before each run rather than tuned on results. Because every candidate skill is scored on the full selection fold in every cycle, the size of that fold sets the per-cycle Solver cost, so we kept it to a few hundred mentions and reserved the larger folds for the single certification pass.

Table 8: Configuration of every reported SEEKA system. Each SEEKA system uses the same Solver, reasoning effort (MEDIUM; Llama 4 Maverick exposes none), and candidates as its matched SEEKA-BASE: frozen Cohere Embed v4 top-100 lists, and on NEL the released lists plus NIL. Proposer / Optimizer gives the models that developed the skill; Views counts the ranking views combined per query (SEEKA-BASE: one).

Solver	Proposer / Optimizer	Deployed skill	Views	Development data
NEL				
Grok 4.3	Grok 4.3	RRF of the parent ranking and 7 candidate-order permutation views, NIL bias	8	VALID
GPT-5.6 Terra	GPT-5.6 Terra	8 ranking and verifier views, logistic gate (C=0.1)	8	VALID
GPT-5.5	GPT-5.5	two-stage workflow with confidence and evidence gates	2	992 VALID
Llama 4 Maverick	Llama 4 Maverick	evolved skill	n/r	160 selection and 808 certification (968 VALID)
TempEL				
Grok 4.3	Grok 4.3	RRF (k=10) of Grok 4.3, and short-description Grok 4.3 (weight 0.75) views	3	2,000 from 2018
GPT-5.6 Terra	GPT-5.6 Terra	evolved instruction, 2017 temporal mention prior	1	400 feedback (2017), 400 selection and 2,000 certification (2018); 2,800 total
GPT-5.5	GPT-5.5	validation-ranked cascade over 5 component runs with tree gates	5	VALID (2017–2018)
Llama 4 Maverick	Llama 4 Maverick	evolved instruction, 2017 temporal mention prior	1	as GPT-5.6 Terra
ShadowLink				
Grok 4.3	Grok 4.3	Grok 4.3 skill fused with a challenger ranking	n/r	synthesized cases only
GPT-5.6 Terra	GPT-5.6 Terra	Terra views with e5-small-v2 features	n/r	Neutral subset
GPT-5.5	GPT-5.5	GPT-5.5 adjudicator over 22 frozen GPT-5.5 ranking views	22	Neutral subset: 200 per cycle, 400 for selection
Llama 4 Maverick	Llama 4 Maverick	evolved skill	n/r	synthesized cases only
DynamicER				
Grok 4.3	Grok 4.3	4,109-entry temporal alias memory	1	240 feedback, 480 selection (pre-evaluation); 720 total
GPT-5.6 Terra	GPT-5.6 Terra	evolved instruction, 741-entry alias memory, rank-10 gate	1	240 selection (pre-evaluation)
GPT-5.5	GPT-5.5	evolved instruction, 741-entry alias memory, rank-10 gate	1	240 selection (pre-evaluation)
Llama 4 Maverick	Llama 4 Maverick	evolved instruction	1	pre-evaluation mentions

C Audit Implementation Details

This appendix details the audit loop of Figure 2 (Section 2.2). The current implementation uses the configured Proposer model for the separate zero-temperature audit call with a strict JSON contract; malformed judge outputs are retried until they satisfy the typed schema. Refinement is selective: accepted anchor slots are frozen and only rejected slots are regenerated. On a retry, the Proposer is instructed to use substantially different wording, recheck every fact against the gold description, make the gold denotation direct and uniquely supported, and retain a plausible wrong-candidate confounder. The implementation preserves the original slot order and curriculum size; after 20 unsuccessful strict audits, an optional reserve anchor may replace an unsynthesizable anchor and a gold-only grounding instruction is applied. If any slot still fails after the configured attempt budget, strict runs abort rather than silently admit it; non-strict diagnostic runs may retain the final rejected draft with an explicit fallback flag. Judge requests are content-addressed in the cache, and each row stores the model, reasoning effort, pass decision, attempt count, and full audit history. Structural validators reject duplicate or malformed identifiers, unsupported labels, missing descriptions when evidence is required, copied evaluation examples, and incomplete rankings. Generation traces retain the candidate set, context, gold identifier, Solver output, and gate result. The gate measures semantic validity, not Solver difficulty: the frozen Solver is evaluated only after a row passes.

D Complete Algorithm and Consolidation

Algorithm 1 One SEEKA run. $\rho_m(e)$ is the set of m nearest KB neighbors of e under the frozen retriever, $\mathrm{Acc}(s; D)$ is the top-1 accuracy of the frozen Solver equipped with skill s on a labeled set D, L_τ is the learnability reward of Eq. (3), OPTIMIZE is the Optimizer's revision step (Section 2.3), and UPDATEPROPOSER is the Proposer instruction update (Section 2.4).

Require: catalog $\mathcal{K}$; frozen retriever ρ; initial skill s_0; initial Proposer instruction q_0; feedback fold D_{fb} and selection fold D_{sel}; target accuracy τ; number of confusers m; Proposer acceptance margin ϵ; number of cycles T, cases per cycle N, skill revisions per cycle S, and audit attempts per case A

Ensure: frozen Proposer instruction q and skill s

$s \leftarrow s_0$; $q \leftarrow q_0$
for $t = 1, \ldots, T$ **do**
  $\mathcal{D}_t \leftarrow \emptyset$ ▷ audited curriculum of cycle t
  for $i = 1, \ldots, N$ **do**
    sample an intended entity $e^\star \in \mathcal{K}$ and a decision type $z \in \{\mathrm{link}, \mathrm{NIL}\}$, with $z = \mathrm{link}$ whenever the benchmark does not allow abstention
    if $z = \mathrm{link}$ **then** $C \leftarrow \{e^\star\} \cup \rho_m(e^\star)$, $y^\star \leftarrow e^\star$ **else** $C \leftarrow \rho_m(e^\star)$, $y^\star \leftarrow \mathrm{NIL}$
    the Proposer, following q, writes a mention-in-context x for $(e^\star, C)$; the audit checks $(x, C, y^\star)$ and returns failed criteria for rewriting, for at most A attempts
    if the audit passes **then** $\mathcal{D}_t \leftarrow \mathcal{D}_t \cup \{(x, C, y^\star)\}$ **else** replace $e^\star$ with a reserve anchor and repeat this iteration (a strict run aborts instead), so that $|\mathcal{D}_t| = N$
  end for
  $F_{\mathrm{syn}} \leftarrow$ cases in $\mathcal{D}_t$ that the Solver with skill s ranks incorrectly
  $F_{\mathrm{opt}} \leftarrow F_{\mathrm{syn}}$, representative correctly ranked cases in $\mathcal{D}_t$, and Solver errors on D_{fb}
  for $j = 1, \ldots, S$ **do**
    $s' \leftarrow \mathrm{OPTIMIZE}(s, F_{\mathrm{opt}})$ ▷ revises only the declared components of s
    if $\mathrm{Acc}(s'; D_{\mathrm{sel}}) > \mathrm{Acc}(s; D_{\mathrm{sel}})$ **then** $s \leftarrow s'$
  end for
  $a_t \leftarrow \mathrm{Acc}(s; \mathcal{D}_t)$ ▷ scored with the updated s, so both curricula below share one skill
  $q' \leftarrow \mathrm{UPDATEPROPOSER}(q, a_t, \tau, L_\tau(a_t), \mathrm{Summary}(F_{\mathrm{syn}}))$ ▷ Summary: aggregate error summary
  if $L_\tau(\mathrm{Acc}(s; \mathcal{D}')) \geq L_\tau(a_t) + \epsilon$ for a trial curriculum $\mathcal{D}'$ generated with q' **then** $q \leftarrow q'$
end for
optionally consolidate views of s on synthetic and development data
return q and s, frozen before evaluation on the held-out test set

Algorithm 1 states one complete SEEKA run (Section 2.5). After the loop, an optional consolidation stage may combine complementary Solver behaviors. A *view* is a complete candidate ranking produced by one fixed Solver configuration, such as an accepted instruction, a candidate-order permutation, a memory lookup, or a verifier call. Using only synthetic and designated development data, consolidation selects and combines views through reciprocal-rank fusion or a small deterministic gate. Because multiple views increase inference computation, we report the number of views and retained skill components for each system (Table 8). All instructions, memories, gates, view definitions, and fusion weights are frozen before evaluation on the official held-out test set.

E ROLE PROMPTS AND LEARNED SKILLS

This section gives the core prompt templates used by the three roles. The runner appends the instance fields and strict JSON output schema; bracketed fields below are populated with the current run state.

E.1 ROLE PROMPT TEMPLATES

Solver.

> Act as an entity-linking listwise reranker. Given a marked mention in context and a numbered list of candidate entities, return a complete best-to-worst ranking of all supplied decisions. Rank candidates by whether the marked mention denotes that exact entity in this context. Use aliases, entity type, sense, time, location, relationships, and factual compatibility. Candidate or retrieval order is not evidence, and surface-form similarity alone is insufficient. For NEL, rank NIL first when none of the listed entities is the referent. Never add a decision outside the supplied list.

Proposer.

> Generate one genuinely new, natural entity-linking example from the supplied knowledge-base entity pool. Create the marked mention and its context. Ground all factual clues in the target description, keep confusable entities plausible, and make exactly one entity-or-NIL label justified. Prefer an ambiguous surface shared by the target and a distractor. Include at most one subtle target-exclusive clue; keep the remaining clues compatible with at least two decisions. Produce natural, self-contained prose rather than a quiz, definition, or candidate comparison, and never expose candidates, labels, pools, benchmarks, or entity-linking mechanics.

Optimizer.

> Optimize the Solver instruction block from the current skill, generated-data metrics, bounded Solver rollouts, representative successes and failures, and recent validation decisions. Diagnose errors in denotation, entity type, scope, work/version, temporal evidence, and NIL calibration. Produce one complete replacement instruction block containing general rules that transfer to unseen examples. Do not memorize identifiers, exact examples, entity titles, candidate positions, validation demonstrations, or dataset proportions. Return one JSON object with the proposed skill name, instruction block, and rationale.

E.2 EXAMPLES OF OPTIMIZED SOLVER SKILLS

The following lightly shortened excerpts show the learned decision procedures in four frozen, post-optimization Solver skills. They are distinct from the generic role templates above. These excerpts come from the GPT-5.5 artifacts, in which GPT-5.5 serves as the development-time Proposer and Optimizer. Full prompts, memories, gates, and hashes are retained in the released run artifacts.

NEL.

> Infer the contextual referent and its type and scope before testing candidates. Then explicitly compare the best-supported entity with NIL: link only when contextual identity evidence is stronger than an ordinary-language or absent- referent explanation and no hard type, scope, or work-version contradiction remains. One learned closed rule handles digital products:

select an exact-title source work only when the context directly attributes the digital product to the same creator and no candidate represents the product itself.

TempEL.

Resolve the marked span in its stated temporal snapshot. A candidate description may omit later facts, use an older name, or emphasize another period; missing later information alone is not a contradiction. Prefer the candidate whose entity type, alias, time period, geography, role, and relationships explain the local clues, rather than the most famous entity or the closest surface-title match.

ShadowLink.

First classify the surface as a full name, shortened name, surname, title fragment, acronym, alias, or ambiguous common phrase. Give title and known alias matches strong weight unless the context contradicts them, but do not select a semantically related neighbor from one overlapping word. Broad words such as "group," "line," or "track" may be incidental framing rather than hard entity-type constraints.

DynamicER.

Parse the marked sports phrase and decide whether it names a person, club, place, venue, competition, nickname, category, or fan expression. Do not turn a league phrase into a member club, a stadium issue into its resident team, a bare place into a team without club-language evidence, or a parent club into a more specific subpage. Use surrounding rosters, competitions, and dates only as tie-breakers among candidates the phrase could plausibly denote.

E.3 GPT-5.5 SYNTHESIZED CURRICULUM EXAMPLE

The following accepted NEL item illustrates how the Proposer creates a hard NIL case. For readability, we show the strongest confuser and several representative alternatives rather than the complete stored inventory.

Mention and context. Collectors' blurbs can make [the New Morning record] seem like a routine leader-led album, the sort of entry filed by title and musician rather than by a band name. Its basic facts fit standard album shorthand—format, artist-catalog placement, and label-credit notes—although, in one track note, its second life is noted through both a shelved six-string pass and a border-crossing vocal remake. The rest of the discussion can stay broad, treating it as an artist-catalog installment with an optimistic title. For casual browsing, the sunrise wording and leader-album framing are more noticeable than that small trail through versions.

Gold decision. NIL. The intended referent is Bob Dylan's *New Morning*, which is deliberately absent from the candidate list.

Strongest confuser. *New Morning (Johnny Coles album)*. Representative alternatives include *New Morning (Misia album)*, *New Morning (Alpha Rev album)*, and *New Morning (club)*.

Why it is learnable but difficult. The surface and album framing make the Johnny Coles record plausible, while the low-salience clue about later guitar and vocal versions identifies the absent Bob Dylan album, making NIL the unique correct decision.

F LIMITATIONS

Scope. Reranking cannot exceed the frozen retriever, whose top-100 recall of 28.0–28.8% on TempEL Continual Entities caps every reranker below the task-trained CYCLE system. NEL ranks NIL as an in-list option, earlier headline runs used curriculum depths between 10 and 100, benchmarks and curricula are English, and synthesized cases and learned skills can reproduce model and catalog biases.

Supervision. SEEKA is not label-free: besides the KB-labeled synthesized cases, skill revision, acceptance, and fitted components also use a small labeled development set (Section 3.1).