ECB: Evidence-Conditioned Continuous Bandit for Recommendation Agents

Anonymous authors
Paper under double-blind review

Abstract

Recommendation agents must decide where to retrieve evidence, how much evidence to acquire, and when further exploration is no longer worthwhile. This creates two fundamental challenges. First, reranking an initial candidate list cannot recover items that were never retrieved, making candidate coverage dependent on the retrieval workflow itself. Second, retrieval and candidate scoring consume limited resources, so exploration must balance the discovery of new items against its additional cost. We propose Evidence-Conditioned Continuous Bandit (ECB), a budgeted controller for adaptive workflow control. ECB introduces a gate-before-scoring mechanism that first examines the base recommender's output and determines whether further evidence should be acquired. When exploration is triggered, an Evidence-aware Ranker integrates user history, contextual information, item attributes, and construction evidence to produce candidate-level retrieval signals. These signals condition a joint continuous policy that controls interest focus, evidence mixing, and retrieval-source allocation. The policy therefore coordinates what to retrieve, how broadly to explore, and when to stop, while the base recommender ranks the expanded candidate pool. ECB learns this policy offline from sampled workflow trajectories using a target-conditioned proxy reward that jointly accounts for recommendation outcomes and resource costs. A likelihood-ratio update enables policy learning through the discrete retrieval executor. Experiments on multiple recommendation benchmarks show that ECB improves recommendation quality and target coverage, demonstrating the value of evidence-conditioned exploration and budget-aware workflow control.

1 Introduction

Recommendation agents combine a base recommender with retrieval tools that use user histories and contextual evidence. Their choice of retrieval sources determines which items reach the ranking stage and how much the workflow costs. Figure 1 illustrates why this choice matters. The upper path commits a request to one tool, leaving other useful signals unexplored. The lower path adjusts the contributions of multiple sources to the user's history and context. An item missed by one source can then enter through another. Effective recommendation therefore requires decisions about where to look, which evidence to use, and how much retrieval to allocate to each request.

These decisions face two constraints. First, reordering an initial shortlist cannot recover an item outside it. The system must acquire evidence that helps it search beyond that list. Second, candidate scoring and retrieval consume resources, so repeated expansion is useful only while its expected benefit justifies the additional cost. We learn this trade-off offline by sampling workflow configurations and evaluating their outcomes with a frozen proxy reward model. Scoring and retrieval budgets limit how many alternatives can be explored for each training context.

We introduce ECB (Evidence-Conditioned Continuous Bandit), a budgeted controller for this workflow. As shown in Figure 2, a gate first examines the base recommender's output. STOP returns the current list; UPGRADE invokes an Evidence-aware Ranker that scores candidates using organized history, context, item information, and construction evidence. The resulting score distribution guides a joint continuous action over interest focus (λ), evidence mix (ρ), and retrieval-source budget (ω). Retrieval expands the candidate pool, and the base recommender ranks the expanded pool before the

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.

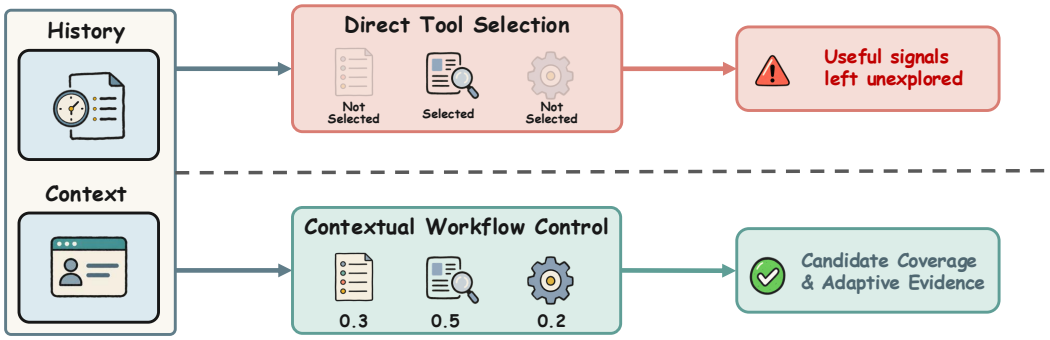

Figure 1: From direct tool selection to contextual workflow control. The upper path leaves unselected signals unexplored. The lower path adapts source contributions to history and context, broadening candidate coverage. The weights illustrate one possible allocation.

next gate decision. The original history persists throughout; the latest retrieval signal replaces the previous signal. Our code is open source at `https://anonymous.4open.science/r/ECB/`.

ECB learns these configurations from sampled workflow outcomes. A target-conditioned reward model, fitted on training records, evaluates each sampled training trajectory. Its score minus the incurred scoring and retrieval costs supplies the policy reward. A likelihood-ratio update with an action-independent baseline trains the continuous action policy through the discrete retrieval executor. At inference, candidate scores provide retrieval evidence, and the gate controls whether that evidence is acquired.

Experiments on Beauty, Yelp, Movies, and Toys show that ECB improves recommendation quality and target coverage. Candidate-expansion controls separate the value of discovering new items from reordering existing ones. Gate, input, action, and retrieval-source ablations explain how the workflow uses evidence and budget, while ranker replacement and history-length analyses examine its behavior across scoring choices and user histories.

Our contributions are:

- We formulate recommendation-agent improvement as budgeted candidate exploration, connecting limited candidate coverage with the cost of acquiring additional evidence.
- We introduce a gate-before-scoring workflow in which candidate-level evidence conditions a joint continuous policy over interest focus, evidence mixing, and retrieval allocation.
- We derive a finite-horizon stopping criterion and the likelihood-ratio update, and evaluate the design through recommendation, coverage, component, and resource analyses on four datasets.

2 Related Work

Recommendation models and agents. Sequential models learn from interaction histories (Kang & McAuley, 2018; Shin et al., 2024; Zhou et al., 2022; Liu et al., 2024), while HSTU, TIGER, and OneRec develop generative recommendation (Zhai et al., 2024; Rajput et al., 2023; Deng et al., 2025). TALLRec and HLLM adapt language models to recommendation; ItemRAG augments recommendation with retrieved items (Bao et al., 2023; Chen et al., 2024; Kim et al., 2026). Recommendation agents further introduce planning, deliberative reasoning, and tool use (Huang et al., 2025; Wu et al., 2025; Ma et al., 2025; Zhao et al., 2024). Generative user agents also support simulation of recommendation environments (Zhang et al., 2024). ECB controls the acquisition of evidence and candidates around a base recommender, adapting retrieval to each request and its remaining budget.

Learning from limited feedback. Contextual bandits learn action values from observed rewards (Li et al., 2010). Counterfactual risk minimization and doubly robust methods learn and evaluate policies from logged feedback (Swaminathan & Joachims, 2015; Dudík et al., 2011); continuous-

treatment methods extend policy evaluation and optimization to continuous actions (Kallus & Zhou, 2018). ECB learns continuous retrieval configurations from sampled workflow outcomes. Candidate scores supply the context for these actions, and a cost-adjusted reward trains the policy. A finite-horizon stopping rule connects the value of further exploration with the cost of acquiring evidence.

Continuous control of discrete retrieval. Learned action representations make large discrete action spaces tractable (Dulac-Arnold et al., 2016; Chandak et al., 2019), and approximate nearest-neighbor indices support efficient catalog search (Malkov & Yashunin, 2020). ECB uses continuous variables to set interest focus, evidence mixture, and retrieval-source allocation. A discrete executor converts their joint setting into candidate additions. The likelihood-ratio estimator (Williams, 1992) trains the policy from the resulting outcomes while leaving retrieval and ranking as discrete operations.

3 Workflow-Control Formulation

Let H_u be user u's observed history, c the request context, and r_t the latest retrieval signal. The workflow input is $x_t = (H_u, c, r_t)$, with $r_0 = \varnothing$. History and context persist across rounds; each upgrade replaces the retrieval signal. The retained candidate pool is a separate object, initialized as P_0 and growing as new items are retrieved.

At round t, the base recommender ranks the available pool and emits a shortlist B_t with base-score distribution p_t^{base}. Writing P_t for the retained candidate pool, the pre-scoring state is $s_t = (x_t, P_t, B_t, p_t^{\text{base}}, b_t, t)$, where b_t is the remaining budget. A gate acts on this pre-scoring state:

$$z_t = g_\psi(s_t) \in \{0, 1\}. \tag{1}$$

STOP ($z_t = 0$) returns the current base-ranked list. UPGRADE ($z_t = 1$) acquires Evidence-aware Ranker scores $e_{t,i}$ for the exposed candidates $i \in B_t$. Their relative distribution is

$$d_{t,i}^E = \frac{\exp(e_{t,i})}{\sum_{j \in B_t} \exp(e_{t,j})}. \tag{2}$$

The continuation state $h_t = (s_t, d_t^E)$ conditions a joint action $a_t = (\lambda_t, \rho_t, \omega_t)$ over interest focus, evidence mixing, and retrieval-source allocation. We factor its density in workflow order:

$$\pi_\theta(a_t \mid h_t) = \pi_\lambda(\lambda_t \mid h_t)\pi_\rho(\rho_t \mid h_t, \lambda_t)\pi_\omega(\omega_t \mid h_t, \lambda_t, \rho_t). \tag{3}$$

The executor builds $r_{t+1} = G(s_t, d_t^E, a_t)$, retrieves candidates, and merges them into the pool. The base recommender ranks the expanded pool before the next gate decision. Stopping or reaching the resource or round limit returns the latest base-ranked list.

A frozen reward model q_ϕ, fitted on training data, evaluates the sampled training trajectory τ using its history, context, and training target y. The objective deducts the scoring and retrieval cost penalties, expressed in the same utility units as the reward:

$$R^{\text{train}}(\tau) = q_\phi(H_u, c, y \mid \tau) - C^E(\tau) - C^R(\tau), \qquad J(\theta) = \mathbb{E}_{\tau \sim p_\theta}[R^{\text{train}}(\tau)]. \tag{4}$$

The policy learns from sampled training outcomes. During held-out evaluation, the controller receives observed history, request context, candidate information, and acquired evidence; a separate evaluator accesses the next item after the workflow returns its list.

Proposition 1 (Shortlist coverage ceiling)**.** *For any ranking rule whose output is contained in a shortlist B,* HR@$k \le \Pr[y \in B]$.

Proof. If $L_k \subseteq B$ denotes the returned top-k set, then $\mathbf{1}[y \in L_k] \le \mathbf{1}[y \in B]$. Taking expectations proves the bound. □

The bound makes candidate acquisition consequential: expanding the pool can raise the ceiling available to the final ranking stage.

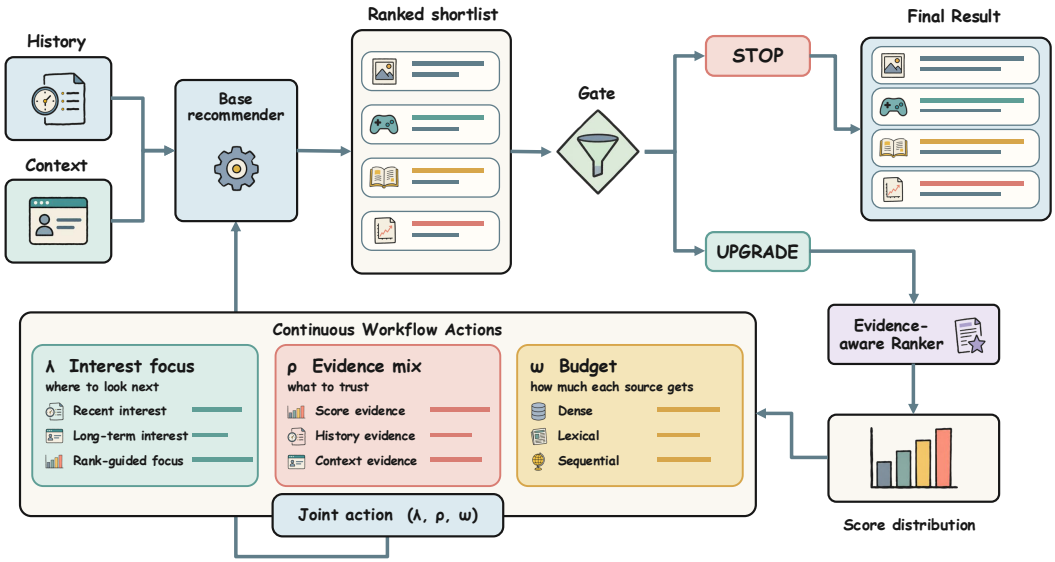

Figure 2: ECB workflow. The gate examines the base-ranked list before paying for candidate scoring. On UPGRADE, the Evidence-aware Ranker supplies scores that condition the joint action (λ, ρ, ω). Retrieval expands the pool, and the base recommender ranks it before the next gate decision. STOP returns the current list.

4 ECB: Evidence-Conditioned Continuous Bandit

A recommender can rank only the items that retrieval makes available. Improving the order of a fixed pool therefore leaves an important decision unresolved: where should the next useful candidate come from? Applying the same expansion to every request also ignores whether the current list already meets the user's needs. ECB connects these decisions in one loop (Figure 2). It first decides whether to acquire evidence, uses that evidence to configure retrieval, and learns the configuration from the resulting recommendation outcome and cost.

4.1 Acquire evidence only when it can guide useful expansion

Scoring candidates before deciding whether to continue incurs a cost even for requests that can already stop. ECB places its gate before the Evidence-aware Ranker. The gate reads the current base output, workflow state, and remaining budget, all available without another scoring call. STOP returns the current list; UPGRADE acquires scores and starts one retrieval round. This order makes the cost of evidence acquisition part of the decision itself.

The gate learns to compare the value of the current list with the expected value of continuing. Training rollouts supply the continuation advantage $A_t = (U_t^{\text{continue}} - C_t) - U_t^{\text{stop}}$, where U_t^{continue} estimates terminal utility under a fixed continuation policy, U_t^{stop} estimates utility on stopping now, and C_t covers the same continuation suffix. The frozen training reward model supplies both utility estimates. They supervise a gate using only pre-scoring inputs, separately from the continuous action policy. Consequently, the gate can learn when the expected improvement warrants another acquisition step. Appendix A.1 derives the corresponding finite-horizon stopping criterion.

Once the gate chooses UPGRADE, the controller needs evidence about which candidates fit the request and why they entered the pool. The Evidence-aware Ranker evaluates one record per exposed candidate, combining observed history, request context, item information, and candidate-construction evidence. Each record receives a scalar score; Equation 2 normalizes the scores within the exposed set. The continuation policy encodes candidate–score pairs and pools valid records with padding masked. Pairing identities with scores preserves which evidence supports each item, allowing the policy to guide retrieval from candidate-level differences. The same interface supports alternative rankers.

Table 1: Dataset statistics. Mean length is the number of interactions per user.

Dataset	Users	Items	Interactions	Mean length
Beauty	9,159	19,450	39,427	4.30
Yelp	530,330	149,659	5,211,090	9.83
Movies	1,534,020	646,522	11,195,641	7.30
Toys	1,446,010	703,307	8,136,525	5.63

4.2 Coordinate retrieval through a joint continuous action

A fixed source mixture assigns the same retrieval effort to requests with different histories and candidate evidence. Selecting a single source can likewise leave complementary signals unused. ECB instead conditions its retrieval configuration on the evidence acquired for the current request. Three action components connect the interpretation of that evidence to the allocation of retrieval capacity.

Interest focus λ controls how recent and long-term interests, guided by candidate scores, shape the next query. Evidence mixture ρ weights the available score, history, and context evidence. Source allocation ω divides retrieval capacity among semantic, lexical, and sequential sources. Both ρ and ω are nonnegative weight vectors whose entries sum to one. These components address linked decisions: the interest being pursued affects which evidence is useful, and that evidence affects where retrieval effort should be spent. Accordingly, Equation 3 samples the components in a conditional sequence. The mixture head observes the sampled interest focus, and the allocation head observes both earlier components. This factorization coordinates the three choices while retaining a tractable joint density for learning.

The executor converts the sampled action into a retrieval signal, requests candidates, removes duplicates, and merges additions into the retained pool. The base recommender then ranks the expanded pool. Original history remains available throughout the loop, while the latest retrieval signal replaces the previous one. Thus the pool accumulates useful discoveries without requiring an ever-growing sequence of retrieval signals in the model input. The next gate examines the updated base output before acquiring more evidence. Together, these steps let candidate scores influence which new items become rankable, directly addressing the coverage ceiling of the initial pool.

4.3 Learn retrieval choices from their outcomes and costs

The executor contains discrete retrieval and ranking operations, so a useful training signal must connect a sampled continuous action to the list it ultimately produces. ECB samples workflow trajectories and evaluates them with the frozen target-conditioned reward model fitted on training records. Equation 4 subtracts the realized scoring and retrieval cost penalties from its terminal score. An expansion is therefore rewarded for the improvement it achieves after accounting for the resources it consumes.

For an executed upgrade, let G_t denote terminal proxy reward minus the cost penalties from round t onward, and let $b_\eta(h_t)$ be an action-independent baseline. The likelihood-ratio update is

$$\widehat{\nabla_\theta J} = \sum_{t:z_t=1} \big(G_t - b_\eta(h_t)\big)\nabla_\theta \log \pi_\theta(a_t \mid h_t). \tag{5}$$

The return weights the log-probability gradient of the sampled action: configurations with a higher return relative to the baseline receive a positive update. The joint log-density is the sum of the three conditional head log-densities, allowing all components to learn from the same workflow outcome. This update uses outcomes supplied by the executor without differentiating through its discrete operations. The baseline controls variance; entropy regularization encourages exploration, and feasible action supports enforce the remaining resource limits. Appendix A.2 states the assumptions and derives the estimator.

The three scoring roles remain distinct. The frozen reward model evaluates training outcomes using the training target. At execution time, the Evidence-aware Ranker supplies candidate evidence for retrieval. A separate evaluator accesses held-out targets only after the workflow returns its list. Validation selects the operating configuration. ECB thus executes the learned policy using only information available at request time.

Table 2: Recommendation results. ECB improves over the base model across all four datasets. H1, H5, and N5 denote HR@1, HR@5, and NDCG@5; leading zeros are omitted. Bold values indicate statistically significant improvements over the base model, OneRec ($p < 0.05$).

	Beauty			Yelp			Movies			Toys		
Method	H1	H5	N5	H1	H5	N5	H1	H5	N5	H1	H5	N5
Sequential recommendation												
SASRec	.0922	.1227	.1091	.0035	.0121	.0078	.0068	.0267	.0167	.0041	.0085	.0064
BSARec	.1182	.1329	.1266	.0027	.0091	.0059	.0106	.0161	.0135	.0086	.0114	.0101
FMLP-Rec	.1192	.1376	.1292	.0042	.0115	.0079	.0114	.0236	.0179	.0086	.0149	.0120
Mamba4Rec	.1094	.1282	.1197	.0041	.0094	.0068	.0115	.0228	.0176	.0081	.0140	.0113
HSTU	.1199	.1372	.1294	.0030	.0114	.0072	.0122	.0195	.0161	.0079	.0136	.0103
Generative recommendation												
TIGER	.1231	.1385	.1291	.0086	.0132	.0108	.0119	.0201	.0171	.0085	.0150	.0122
OneRec	.1302	.1698	.1547	.0103	.0181	.0149	.0201	.0278	.0241	.0187	.0249	.0211
LLM-based recommendation												
TALLRec	.1234	.1533	.1400	.0064	.0082	.0073	.0123	.0286	.0207	.0121	.0204	.0164
HLLM	.1462	.1708	.1566	.0117	.0195	.0152	.0256	.0448	.0349	.0179	.0317	.0265
ItemRAG	.1306	.1656	.1467	.0127	.0176	.0155	.0209	.0454	.0311	.0148	.0314	.0235
Recommendation agents												
InteRec Agent	.1067	.1345	.1221	.0076	.0201	.0155	.0193	.0357	.0257	.0135	.0234	.0178
STARec	.1237	.1577	.1409	.0114	.0182	.0142	.0211	.0374	.0281	.0183	.0213	.0196
AgentRec	.1137	.1342	.1248	.0091	.0182	.0141	.0199	.0378	.0271	.0175	.0201	.0192
ToolRec	.1051	.1137	.1101	.0080	.0154	.0127	.0195	.0372	.0263	.0148	.0177	.0166
ECB	**.1610**	**.1842**	**.1754**	**.0140**	**.0291**	**.0221**	**.0283**	**.0483**	**.0360**	**.0203**	**.0349**	**.0288**

5 Experiments

5.1 Experimental setup

We evaluate ECB on the Amazon Beauty, Movies, and Toys categories[1] and Yelp[2] (Table 1). Each context contains observed history and one held-out next item. OneRec supplies the initial candidates and ranks every expanded pool. The gate precedes scoring, and the workflow permits at most three expansion rounds.

For a target at rank r, HR@k is $\mathbf{1}[r \leq k]$ and NDCG@5 is $\mathbf{1}[r \leq 5]/\log_2(r+1)$. Metrics average over all evaluation contexts, including targets absent from the pool. Models use training records; validation selects the configuration fixed for held-out evaluation. Only the evaluator accesses the target after the list is returned.

We compare sequential recommenders (Kang & McAuley, 2018; Shin et al., 2024; Zhou et al., 2022; Liu et al., 2024; Zhai et al., 2024), generative recommenders (Rajput et al., 2023; Deng et al., 2025), LLM-based methods (Bao et al., 2023; Chen et al., 2024; Kim et al., 2026), and recommendation agents (Huang et al., 2025; Wu et al., 2025; Ma et al., 2025; Zhao et al., 2024). Component studies then isolate candidate acquisition, stopping, retrieval actions, and ranker evidence.

5.2 Recommendation quality

ECB leads HR@1, HR@5, and NDCG@5 across the four datasets (Table 2). Its NDCG@5 gains over OneRec are 13.4%, 48.3%, 49.4%, and 36.5% on Beauty, Yelp, Movies, and Toys. Improvements in both hit rate and discounted gain indicate that the workflow helps relevant items reach the returned list and rank near its top.

[1] http://snap.stanford.edu/data/amazon/productGraph/categoryFiles/

[2] https://www.yelp.com/dataset

Table 3: Candidate expansion. N5 denotes NDCG@5; Cov. is target coverage (%).

	Beauty		Yelp		Movies		Toys	
Setting	N5	Cov.	N5	Cov.	N5	Cov.	N5	Cov.
Original OneRec	0.1547	20.41%	0.0149	2.90%	0.0241	5.07%	0.0211	3.86 %
Rerank initial pool	0.1582	20.41	0.0162	2.90	0.0266	5.07	0.0234	3.86
Fixed retrieval	0.1606	22.77	0.0175	3.87	0.0301	6.78	0.0248	5.27
Full ECB	**0.1754**	**24.35%**	**0.0221**	**4.64%**	**0.0360**	**7.80%**	**0.0288**	**5.88 %**

Table 4: Gating. N5 denotes NDCG@5; Rounds is the mean expansion count. Random gating matches ECB in mean rounds.

	Beauty		Yelp		Movies		Toys	
Setting	N5	Rounds	N5	Rounds	N5	Rounds	N5	Rounds
Always stop	0.1547	0	0.0149	0	0.0241	0	0.0211	0
Always continue	0.1750	3	0.0221	3	0.0367	3	0.0290	3
Random gate	0.1655	1.24	0.0188	1.76	0.0315	1.55	0.0253	1.41
Learned gate (ECB)	**0.1754**	**1.24**	**0.0221**	**1.76**	**0.0360**	**1.55**	**0.0288**	**1.41**

Table 5: Expansion-policy ablations, measured by NDCG@5.

Setting	Beauty	Yelp	Movies	Toys
Full ECB	**0.1754**	**0.0221**	**0.0360**	**0.0288**
Policy without candidate scores	0.1668	0.0185	0.0312	0.0250
Fixed interest focus λ	0.1720	0.0213	0.0342	0.0277
Fixed evidence mixture ρ	0.1718	0.0201	0.0347	0.0274
Fixed source quotas ω	0.1705	0.0213	0.0344	0.0276
All expansion parameters fixed	0.1608	0.0175	0.0302	0.0240

5.3 Candidate expansion and stopping

Finding candidates beyond the initial pool. Table 3 separates reranking from acquisition. Reranking the initial pool improves NDCG@5 but leaves coverage unchanged: its opportunity to find a target remains limited to the same items. Fixed retrieval raises coverage, and ECB further improves both coverage and NDCG@5 in every dataset. On Beauty, coverage progresses from 20.41% for the initial pool to 22.77% with fixed retrieval and 24.35% with ECB. The improvement therefore involves discovering targets that the initial recommender could not rank. Coverage measures target availability; NDCG@5 measures its returned rank. Their joint increase shows that ranking can use the opportunities created by retrieval.

Choosing when to expand. Table 4 tests whether the controller spends its rounds effectively. Random gating matches ECB's mean expansion count, yet ECB achieves higher NDCG@5 on all four datasets. The allocation of rounds across requests thus matters even at the same average count. The gate allocates limited acquisition steps across contexts, continuing requests that can benefit from more evidence. Relative to always expanding for three rounds, ECB uses 41.3–58.7% fewer rounds. Its NDCG@5 is higher on Beauty, equal on Yelp, and lower by 0.0007 and 0.0002 on Movies and Toys. This comparison quantifies the quality–cost tradeoff of stopping. Each stopped round also avoids an Evidence-aware Ranker call because the gate precedes scoring.

5.4 What makes expansion effective?

Table 5 examines the configuration of each expansion. Removing candidate scores lowers NDCG@5 by 0.0086, 0.0036, 0.0048, and 0.0038 across the four datasets. Fixing interest focus, evidence mixture, or source allocation also reduces quality. Fixing all three yields the lowest score in each dataset, although retrieval expansion remains available.

Table 6: Sensitivity to the per-round retrieval budget, measured by NDCG@5. $1\times$ is the default per-round budget.

Budget	Method	Beauty	Yelp	Movies	Toys
$0.5\times$	Fixed retrieval	0.1562	0.0164	0.0270	0.0231
$0.5\times$	ECB	0.1656	0.0187	0.0317	0.0253
$1\times$	Fixed retrieval	0.1606	0.0175	0.0301	0.0248
$1\times$	ECB	**0.1754**	**0.0221**	**0.0360**	**0.0288**
$2\times$	Fixed retrieval	0.1655	0.0191	0.0330	0.0262
$2\times$	ECB	0.1755	0.0224	0.0360	0.0291

Table 7: Evidence-aware Ranker replacement. N5 denotes NDCG@5; Time is mean scoring time in seconds. Shading marks the default ECB configuration.

	Beauty		Yelp		Movies		Toys	
Scorer	N5	Time (s)	N5	Time (s)	N5	Time (s)	N5	Time (s)
Fixed source scores	0.1662	< 0.001	0.0193	< 0.001	0.0320	< 0.001	0.0255	< 0.001
Feature-based scorer	0.1703	0.0047	0.0202	0.0053	0.0331	0.0050	0.0273	0.0053
Default (ECB)	**0.1754**	**0.1210**	**0.0221**	**0.1462**	**0.0360**	**0.1083**	**0.0288**	**0.1208**
JEv	0.1760	1.1001	0.0222	1.1505	0.0366	1.1203	0.0280	1.1107

5.5 Retrieval budget and scoring choices

Directing the per-round budget. Table 6 varies the number of candidates requested in each retrieval round. At half the default budget, ECB exceeds fixed retrieval at the default budget on every dataset. At the default budget, it also exceeds fixed retrieval with twice that budget. Thus a larger candidate request alone does not reproduce the gains from adapting retrieval to the request. The policy's contribution is visible at multiple operating points, connecting the action ablation to an explicit resource comparison.

Doubling ECB's per-round budget beyond the default increases NDCG@5 by at most 0.0003. In contrast, replacing fixed retrieval with ECB at the default budget produces a larger gain in every dataset. Within the tested range, these comparisons show that directing the available retrieval capacity is more consequential than simply increasing the request size. They complement the gate study: the gate controls how many rounds are executed, while the action controls what each executed round retrieves. The two controls govern different parts of resource use. Increasing a per-round budget broadens each retrieval request; executing another round lets the controller respond to the newly expanded pool. A budgeted workflow can therefore adjust retrieval breadth and the number of decisions separately. The matched-round experiment tests the allocation of continuation decisions, while Table 6 tests retrieval breadth.

Choosing a ranker for the evidence interface. Table 7 changes the scoring implementation while retaining the controller workflow. Fixed source scores and a simple feature-based scorer both reduce NDCG@5 on every dataset. Candidate evidence therefore benefits from a ranker that combines the available record fields. The result also motivates examining the inputs themselves: a scoring model can only exploit the information and correspondence supplied to it. The simple feature-based scorer is faster than the default, but its lower NDCG@5 reveals a quality–cost tradeoff at the evidence interface. JEv provides the complementary comparison at a higher scoring cost. These alternatives help distinguish the benefit of useful candidate evidence from the computational expense of producing it.

JEv changes NDCG@5 by -0.0008 to $+0.0006$ across datasets, while its mean scoring time is 7.9–10.3 times that of the default ranker. The default thus provides recommendation quality close to JEv with shorter scoring time. These candidate scores affect which items the workflow acquires. OneRec retains the final ranking role, so the replacement study evaluates the evidence interface through its downstream effect on retrieval and recommendation. These times refer to the scoring component; the gate separately determines whether that component is invoked. Together, the replacement and stopping studies describe two ways to control evidence-acquisition effort: select an economical scorer and call it when continuation is useful.

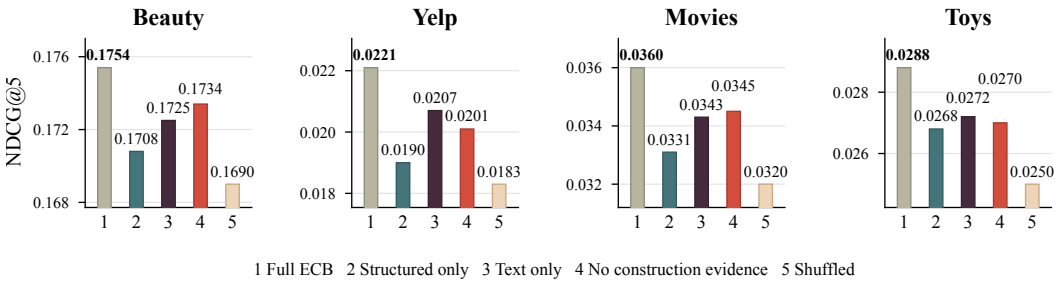

Figure 3: Ranker-input ablations (NDCG@5). Construction evidence describes how a candidate entered the pool; shuffling breaks its correspondence to candidates. Each panel uses a local, nonzero vertical range.

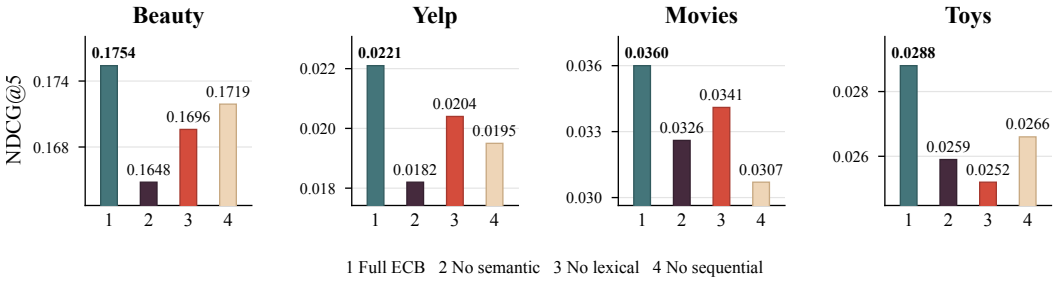

Figure 4: Retrieval-source ablations (NDCG@5). Removing a source redistributes its budget among the remaining sources. Each panel uses a local, nonzero vertical range.

5.6 Evidence inputs and complementary retrieval sources

Figure 3 compares five ranker-input configurations. Text-only input exceeds structured-only input, and the full combination leads on every dataset. Removing construction evidence reduces quality; shuffling its correspondence to candidates causes a larger decrease. Both evidence content and its association with items guide retrieval.

Figure 4 removes individual retrieval sources while preserving the total budget. Every removal reduces NDCG@5 across all datasets. Semantic retrieval contributes most on Beauty and Yelp, sequential retrieval on Movies, and lexical retrieval on Toys. Together with the fixed-ω ablation, these differences support retaining complementary sources and adapting their allocation to the request.

History length and a concrete example. ECB improves over OneRec in every history-length group (Appendix B.4, Table 8). Gains are largest for histories with 0–3 interactions: 20.7%, 68.6%, 79.9%, and 50.8% across the four datasets. In the Beauty example (Appendix C), one upgrade retrieves an initially absent target, which OneRec then ranks second. Evidence-guided acquisition makes this target rankable.

6 Discussion and Conclusion

ECB learns when to expand a recommendation workflow and how to direct that expansion. Candidate scores guide retrieval, while the gate controls their acquisition cost. Across four datasets, the candidate-expansion controls show gains in target coverage and ranking quality. Matched-round gating and budget comparisons connect these gains to the allocation of retrieval effort.

The experiments measure offline next-item quality, expansion rounds, and ranker scoring time. Interactive evaluation can assess live user response and end-to-end latency. ECB extends an existing recommender by learning where additional evidence and candidates are worth acquiring.

AI Use Statement

AI tools were used to assist with manuscript writing and polishing, literature retrieval and discovery, and research ideation and execution. The authors reviewed the manuscript, checked cited sources, and checked the reported results and figures against the experiment records. The authors remain responsible for the final content of the paper.

Ethics Statement

This work uses public recommendation benchmarks and offline next-item evaluation. Deployment should account for biases in historical interactions and for how retrieval decisions affect item exposure and user experience.

Reproducibility Statement

The code is open source at `https://anonymous.4open.science/r/ECB/`. The paper reports the workflow, evaluation metrics, dataset statistics, and experimental comparisons; the appendix provides the training objective, cost accounting, metric definitions, and the structured case study.

References

Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. TALLRec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pp. 1007–1014. ACM, 2023. doi: 10.1145/3604915.3608857. URL https://doi.org/10.1145/3604915.3608857.

Yash Chandak, Georgios Theocharous, James Kostas, Scott Jordan, and Philip Thomas. Learning action representations for reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 941–950. PMLR, 2019. URL https://proceedings.mlr.press/v97/chandak19a.html.

Junyi Chen, Lu Chi, Bingyue Peng, and Zehuan Yuan. HLLM: Enhancing sequential recommendations via hierarchical large language models for item and user modeling. arXiv preprint arXiv:2409.12740, 2024. URL https://arxiv.org/abs/2409.12740.

Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. OneRec: Unifying retrieve and rank with generative recommender and iterative preference alignment. arXiv preprint arXiv:2502.18965, 2025. URL https://arxiv.org/abs/2502.18965.

Miroslav Dudík, John Langford, and Lihong Li. Doubly robust policy evaluation and learning. In *Proceedings of the 28th International Conference on Machine Learning*, pp. 1097–1104, 2011. URL https://arxiv.org/abs/1103.4601.

Gabriel Dulac-Arnold, Richard Evans, Hado van Hasselt, Peter Sunehag, Timothy Lillicrap, Jonathan Hunt, Timothy Mann, Theophane Weber, Thomas Degris, and Ben Coppin. Deep reinforcement learning in large discrete action spaces. arXiv preprint arXiv:1512.07679v2, 2016. URL https://arxiv.org/abs/1512.07679v2. Version 2, 4 April 2016; first posted 24 December 2015.

Xu Huang, Jianxun Lian, Yuxuan Lei, Jing Yao, Defu Lian, and Xing Xie. Recommender AI agent: Integrating large language models for interactive recommendations. *ACM Transactions on Information Systems*, 43(4):1–33, 2025. doi: 10.1145/3731446. URL https://doi.org/10.1145/3731446.

Nathan Kallus and Angela Zhou. Policy evaluation and optimization with continuous treatments. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84, pp. 1243–1251. PMLR, 2018. URL https://proceedings.mlr.press/v84/kallus18a.html.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206. IEEE, 2018. doi: 10.1109/icdm.2018.00035. URL https://doi.org/10.1109/ICDM.2018.00035.

Sunwoo Kim, Geon Lee, Kyungho Kim, Jaemin Yoo, and Kijung Shin. ItemRAG: Item-based retrieval-augmented generation for LLM-based recommendation. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 3884–3888. ACM, 2026. doi: 10.1145/3805712.3809942. URL https://doi.org/10.1145/3805712.3809942.

Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pp. 661–670. ACM, 2010. doi: 10.1145/1772690.1772758. URL https://doi.org/10.1145/1772690.1772758.

Chengkai Liu, Jianghao Lin, Jianling Wang, Hanzhou Liu, and James Caverlee. Mamba4Rec: Towards efficient sequential recommendation with selective state space models. arXiv preprint arXiv:2403.03900, 2024. URL https://arxiv.org/abs/2403.03900.

Bo Ma, Hang Li, ZeHua Hu, XiaoFan Gui, LuYao Liu, and Simon Lau. AgentRec: Next-generation LLM-powered multi-agent collaborative recommendation with adaptive intelligence. arXiv preprint arXiv:2510.01609, 2025. URL https://arxiv.org/abs/2510.01609.

Yu A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 2020. doi: 10.1109/tpami.2018.2889473. URL https://doi.org/10.1109/TPAMI.2018.2889473.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, Maciej Kula, Ed Chi, and Maheswaran Sathiamoorthy. Recommender systems with generative retrieval. In *Advances in Neural Information Processing Systems*, volume 36, pp. 10299–10315. Curran Associates, Inc., 2023. doi: 10.52202/075280-0452. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/20dcab0f14046a5c6b02b61da9f13229-Paper-Conference.pdf.

Yehjin Shin, Jeongwhan Choi, Hyowon Wi, and Noseong Park. An attentive inductive bias for sequential recommendation beyond the self-attention. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(8):8984–8992, 2024. doi: 10.1609/aaai.v38i8.28747. URL https://doi.org/10.1609/aaai.v38i8.28747.

Adith Swaminathan and Thorsten Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pp. 814–823. PMLR, 2015. URL https://proceedings.mlr.press/v37/swaminathan15.html.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992. doi: 10.1007/bf00992696. URL https://doi.org/10.1007/BF00992696.

Chenghao Wu, Ruiyang Ren, Junjie Zhang, Ruirui Wang, Zhongrui Ma, Qi Ye, and Wayne Xin Zhao. STARec: An efficient agent framework for recommender systems via autonomous deliberate reasoning. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 3355–3365. ACM, 2025. doi: 10.1145/3746252.3760995. URL https://doi.org/10.1145/3746252.3760995.

Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Jiayuan He, Yinghai Lu, and Yu Shi. Actions speak louder than words: Trillion-parameter sequential transducers for generative recommendations. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 58484–58509. PMLR, 2024. URL https://proceedings.mlr.press/v235/zhai24a.html.

An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. On generative agents in recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1807–1817. ACM, 2024. doi: 10.1145/3626772.3657844. URL https://doi.org/10.1145/3626772.3657844.

Yuyue Zhao, Jiancan Wu, Xiang Wang, Wei Tang, Dingxian Wang, and Maarten de Rijke. Let me do it for you: Towards LLM empowered recommendation via tool learning. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1796–1806. ACM, 2024. doi: 10.1145/3626772.3657828. URL https://doi.org/10.1145/3626772.3657828.

Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. Filter-enhanced MLP is all you need for sequential recommendation. In *Proceedings of the ACM Web Conference 2022*, pp. 2388–2399. ACM, 2022. doi: 10.1145/3485447.3512111. URL https://doi.org/10.1145/3485447.3512111.

A Stopping and Policy-Gradient Analysis

A.1 Bellman criterion for the budgeted gate

The gate chooses between returning the current list and invoking the continuation policy. Costs in this section use the utility units in Equation 4. Let s_t be a sufficient pre-scoring state containing $(x_t, B_t, p_t^{\text{base}}, b_t)$, the retained candidate pool, and any trajectory summary needed to determine terminal training utility. Fix the lower-level joint policy π_θ and let $U_t(s_t)$ be the expected utility of returning the current base-ranked list from this information state; it is conditional on information available before scoring. With at most T upgrades starting at $t = 0$, the terminal value is $V_T^{*,\pi}(s_T) = U_T(s_T)$. For $t < T$, the optimal stopping recursion over gate rules for this fixed lower-level policy is

$$V_t^{*,\pi}(s_t) = \max\{V_{\text{stop},t}(s_t), Q^{\pi}_{\text{continue},t}(s_t)\}, \qquad V_{\text{stop},t}(s_t) = U_t(s_t), \tag{6}$$

where the upgrade value integrates over ranker outcomes and their actual costs, the joint action, and the discrete executor transition:

$$Q^{\pi}_{\text{continue},t}(s_t) = \mathbb{E}_{d_t^E, k_t^E, a_t, s_{t+1}, k_t^R}\left[-k_t^E - k_t^R + V_{t+1}^{*,\pi}(s_{t+1})\right], \tag{7}$$

Specifically, the Evidence-aware Ranker first returns d_t^E and actual cost k_t^E according to the ranker service kernel $P_E(d_t^E, k_t^E \mid s_t)$; this kernel covers any replaceable point-wise ranker implementation used in the replacement study. The controller then samples $a_t \sim \pi_\theta(\cdot \mid h_t)$; the executor samples its transition and actual retrieval cost (s_{t+1}, k_t^R) according to $P_{\text{exec}}(s_{t+1}, k_t^R \mid s_t, d_t^E, a_t)$, with remaining budget updated inside s_{t+1}. Thus service-cost variability, retries, and executor transitions are in the same conditional expectation. The recursion assumes a budget-feasible service and transition kernel. When no feasible upgrade remains, STOP is the only admissible decision. The normative rule is to continue when $Q^{\pi}_{\text{continue},t}(s_t) > V_{\text{stop},t}(s_t)$; it explains why the gate must compare expected incremental value with newly incurred cost. This recursion optimizes the stopping rule for a fixed continuation policy π_θ. The learned gate approximates this value comparison from training rollouts. The recursion follows by conditioning on the first decision: STOP yields $V_{\text{stop},t}$, whereas CONTINUE yields the displayed expected one-step costs plus the value of the sufficient next state. Taking the larger of these two admissible values gives the finite-horizon optimum over stopping rules. Starting from the specified terminal value, backward induction proves the recursion at every earlier round.

A.2 Likelihood-ratio optimization through the discrete executor

Let $J(\theta) = \mathbb{E}_{\tau \sim p_\theta}[R^{\text{train}}(\tau)]$, where the environment, Evidence-aware Ranker service, target-conditioned reward model, and discrete executor may be non-differentiable but, conditional on the sampled action and trajectory, do not directly depend on θ. The actor update covers the three action heads only; the binary gate is a separate routing function, and its parameters ψ plus target-conditioned reward-model parameters ϕ are held fixed in this derivative. Under an action support $\mathcal{A}(h_t)$ fixed with respect to θ for each state, a differentiable policy density on that support, finite expected reward, and conditions allowing differentiation under the integral, the likelihood-ratio identity is

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta}\left[R^{\text{train}}(\tau) \sum_{t: z_t = 1} \nabla_\theta \log \pi_\theta(a_t \mid h_t)\right]. \tag{8}$$

To see the identity, write $\nabla_\theta J = \nabla_\theta \int R^{\text{train}}(\tau) p_\theta(\tau)\, d\tau = \int R^{\text{train}}(\tau) \nabla_\theta p_\theta(\tau)\, d\tau$. Factor the trajectory density into policy terms and policy-independent transition/service terms, and use $\nabla_\theta p_\theta(\tau) = p_\theta(\tau) \nabla_\theta \log p_\theta(\tau)$; the gate, ranker-service, and target-conditioned reward-model terms are fixed in this derivative. The factorization in Eq. 3 makes the sum of the three head log-probabilities explicit. The action constraints define the feasible support $\mathcal{A}(h_t)$ and the budget is updated using realized costs in the transition kernel. For the unbiased Monte Carlo estimator, replace the trajectory return with the causal reward-to-go G_t and subtract an action-independent critic or running baseline:

$$\widehat{g}_\theta = \sum_{t: z_t = 1} \big(G_t - b_\eta(h_t)\big) \nabla_\theta \log \pi_\theta(a_t \mid h_t), \tag{9}$$

Entropy regularization can be added to the objective subject to the same action and budget constraints. The baseline has its gradient stopped during the actor update and does not change the expected gradient because, conditional on h_t,

$$\mathbb{E}_{a_t \sim \pi_\theta}\left[b_\eta(h_t)\nabla_\theta \log \pi_\theta(a_t \mid h_t) \mid h_t\right] = b_\eta(h_t)\nabla_\theta \int_{\mathcal{A}(h_t)} \pi_\theta(a \mid h_t)\, da = 0. \tag{10}$$

The unbiasedness result applies to Monte Carlo returns and an action-independent baseline. Replacing these returns with a bootstrapped critic introduces approximation error. The score-function form allows both choices to train a policy whose retrieval and ranking executor is discrete.

B Workflow and Evaluation Details

B.1 Candidate records and state

The runtime input consists of observed user history, request context, candidate identity and text, and available retrieval evidence. The gate reads the base recommender's current output. On continuation, the ranker evaluates each exposed candidate and returns the scores used in Equation 2. The continuation policy preserves candidate–score alignment, masks padding, and pools the valid records before sampling its action.

The retained candidate pool and the model's retrieval signal serve different purposes. The pool stores items available for ranking; the signal summarizes the latest retrieval direction. Each upgrade replaces the signal and merges retrieved items into the pool. If the gate stops or an execution limit is reached, the workflow returns the latest complete base-ranked list.

B.2 Costs and information flow

For a trajectory ending at round T, the number of logical candidate evaluations is

$$N_E(\tau) = \sum_{t=0}^{T} \mathbf{1}[z_t = 1]\, |B_t|. \tag{11}$$

Executed expansion rounds count the terms with $z_t = 1$. Physical service requests additionally depend on batching, caching, and retries. These quantities distinguish expansion frequency from the amount of scoring within an expansion.

Training feedback uses the frozen reward model on sampled training trajectories and deducts their scoring and retrieval costs. Validation selects the operating configuration. During held-out evaluation, a separate evaluator uses the target after the workflow completes. All runtime candidate records are constructed from information available at the request time.

B.3 Ranking metrics and candidate coverage

For one held-out target at rank r, HR@k is $\mathbf{1}[r \le k]$ and NDCG@5 is $\mathbf{1}[r \le 5]/\log_2(r+1)$. A missing target contributes zero. Dataset-level values average over the complete set of evaluation contexts. Candidate coverage is the fraction of these contexts whose candidate pool contains the target.

The single-target definition also links the metrics. Rank-one hits contribute one to both HR@1 and NDCG@5; hits at ranks two through five contribute discounts between $1/\log_2 6$ and $1/\log_2 3$. Hence

$$\mathrm{H1} + \frac{\mathrm{H5} - \mathrm{H1}}{\log_2 6} \le \mathrm{N5} \le \mathrm{H1} + \frac{\mathrm{H5} - \mathrm{H1}}{\log_2 3}, \tag{12}$$

where H1, H5, and N5 denote the corresponding dataset averages.

B.4 Results by history length

Table 8 reports the per-group scores underlying the history-length comparison in Section 5. Groups use observed history only; the held-out target is excluded.

Table 8: NDCG@5 by observed history length. The held-out target interaction is excluded from the grouping.

History length	Method	Beauty	Yelp	Movies	Toys
0–3	OneRec	0.1253	0.0086	0.0134	0.0126
0–3	ECB	**0.1512**	**0.0145**	**0.0241**	**0.0190**
4–7	OneRec	0.1622	0.0141	0.0230	0.0221
4–7	ECB	**0.1833**	**0.0228**	**0.0354**	**0.0307**
≥ 8	OneRec	0.1885	0.0201	0.0316	0.0284
≥ 8	ECB	**0.2014**	**0.0275**	**0.0430**	**0.0362**

C Case Study: Recovering an Initially Missing Item

The following four records describe one Beauty interaction for anonymous user U-18427. They present the workflow inputs and outputs in a compact format. The held-out target is shown for evaluation and is absent from runtime inputs.

Record 1: History and initial candidates

History, oldest to newest: Vanicream Gentle Facial Cleanser; CeraVe Hydrating Facial Cleanser; The Ordinary Hyaluronic Acid 2% + B5; Aquaphor Healing Ointment.
Initial OneRec top five: (1) CeraVe PM Facial Moisturizing Lotion; (2) Neutrogena Hydro Boost Water Gel; (3) Olay Regenerist Micro-Sculpting Cream; (4) Aveeno Daily Moisturizing Lotion; (5) COSRX Advanced Snail 92 All in One Cream.
Evaluator-only target: Vanicream Moisturizing Cream, absent from the entire initial candidate pool.

Record 2: Gate, scores, and retrieval action

Gate decision: UPGRADE, before acquiring ranker scores.
Evidence-aware Ranker scores: CeraVe PM 0.91; Aveeno 0.82; COSRX 0.61; Neutrogena 0.46; Olay 0.32.
Selected action components: Interest focus $\lambda = 0.72$; source allocation $\omega = (0.50, 0.35, 0.15)$ for semantic, lexical, and sequential retrieval, respectively. The score ordering provides additional evidence beyond the initial OneRec order, where Aveeno was fourth.

Record 3: Retrieved candidates

Lexical: Vanicream Moisturizing Cream; Eucerin Advanced Repair Cream.
Semantic: La Roche-Posay Lipikar Balm AP+M; the target also appears at rank 3 in the semantic results.
Sequential: CeraVe Moisturizing Cream.
Pool update: Merge retrieved items with existing candidates and remove duplicates. The target is now available to the base recommender.

Record 4: Final OneRec output and stopping

Updated top five: (1) CeraVe PM Facial Moisturizing Lotion; (2) **Vanicream Moisturizing Cream**; (3) CeraVe Moisturizing Cream; (4) La Roche-Posay Lipikar Balm AP+M; (5) Aveeno Daily Moisturizing Lotion.
Next gate decision: STOP; return this list without another ranker call or retrieval round.
Outcome: One expansion; the initially missing target finishes at rank 2.

The case illustrates the transition from candidate evidence to retrieval and then to the base recommender's updated ranking. Acquiring the target outside the initial pool is necessary for this outcome.