

Personalizing How Recommendation Agents Reason: Self-Evolving Per-User Skills for Recommendation Agents

Zhen Tao[♣], Riwei Lai[♣], Chenyun Yu^{◇*}, Weixin Chen[♣], Li Chen[♣],
Beibei Kong[♡], Lei Cheng[♡], Chengxiang Zhuo[♡], Zang Li[♡], Qingqiang Sun^{♣*}

[♣]Great Bay University [♣]Hong Kong Baptist University

[◇]Sun Yat-Sen University [♡]Platform and Content Group, Tencent

zhentao.tz@gmail.com, {csrwlai, cswxchen, lichen}@comp.hkbu.edu.hk

{echokong, raycheng, felixzhuo, gavinzli}@tencent.com

yuchy35@mail.sysu.edu.cn, qqsun@gbu.edu.cn

Abstract

LLM-based recommendation agents personalize *what* they know through evolving per-user semantic memory, yet *how* they perform reasoning relies on a universal, static system prompt shared across all users. When a recommendation fails, the agent only updates its memory, but never examines or revises the reasoning that leads to the failure. To close this gap, we propose **SAGER** (Self-Evolving **AG**ent for **PE**rsonalized **R**ecommendation), the first framework to equip each user with a dedicated, continuously evolving *policy skill*. SAGER incorporates three tailored mechanisms for recommendation scenarios: (i) a two-representation skill architecture constrained by a *Cognitive Injection Boundary*, decoupling a rich evolution substrate from a minimal inference-time injection; (ii) a *contrastive differential* chain-of-thought engine that diagnoses ranking errors against unchosen near-misses under a *refine-not-replace* principle; and (iii) skill-augmented listwise reasoning that mitigates the *Score-Tie Bottleneck*, enabling the slim skill to serve as an active tie-breaker. Evaluated on four public benchmarks, SAGER achieves state-of-the-art accuracy. Furthermore, plugging its policy skill layer into three structurally distinct host agents delivers consistent 9%–12% gains on the H@1 metric. These results indicate that policy skill evolution provides architecture-agnostic improvements that are largely complementary to memory accumulation.

1 Introduction

Large language model (LLM)-based recommendation agents represent the most capable instantiation of personalized recommendation to date (Wu et al., 2024; Zhang et al., 2025). By treating recommendation as an agentic reasoning process, these systems maintain a per-user *semantic memory* (a

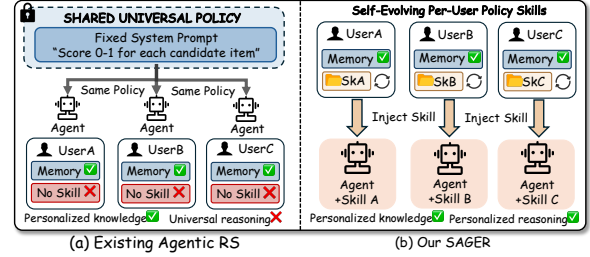


Figure 1: SAGER vs Existing Agentic RS: Evolving How They Think with Skill. (a) **Existing recommendation agents** personalize *what they know* (memory) but reason about every user with a fixed, shared policy. (b) **SAGER** attaches a self-evolving *policy skill* to any LLM-based recommender, personalizing *how* the agent reasons in addition to *what* it remembers.

structured natural-language record of user interaction histories and inferred preferences), retrieve relevant context at inference, and leverage the world knowledge of LLMs to generate grounded, interpretable rankings (Xu et al., 2025; Zhang et al., 2024b; Tang et al., 2025). Recent work further pushes two complementary forms of *self-evolution*: *memory-evolving* agents enrich each user’s memory with cross-user collaborative signals from dynamic semantic graphs (Chen et al., 2026), while *model-evolving* systems use LLM-driven optimizers to autonomously refine the recommender at the system scale (Wang et al., 2026a; Kim et al., 2026). Together, these efforts converge on a compelling vision: a recommendation agent that continuously improves through experience.

The one-policy-fits-all bottleneck. Despite this promising vision, a critical optimization dimension remains largely underexplored. As depicted in Figure 1(a), existing agents, including the self-evolving ones, focus on enriching what the agent *knows* about each user: building richer memories and retrieving more relevant context. Yet the way the agent *thinks*, specifically the reasoning strategy that translates memory into a final ranking, typ-

*Corresponding author.

ically remains fixed and shared across all users. We call this strategy the agent’s *ranking policy*: in nearly all current systems it is a single hand-written system prompt (e.g., “rank candidates by relevance to user history”), and is therefore unable to express the very different decision styles of different users (a price-sensitive shopper and an exploratory cinephile receive the same ranking heuristic). Worse still, when a recommendation fails, the agent updates its memory of the user but never asks: *what was wrong with the reasoning that produced this recommendation?* The agent knows *what* happened but never learns *why it was wrong*. We refer to this regime as the *one-policy-fits-all bottleneck*, and argue that closing it calls for a third, orthogonal form of self-evolution: *policy evolution*, which updates *how* the agent reasons for each individual user.

Skills as a candidate primitive. The agentic LLM literature has independently encountered an analogous problem. Agents relying on a fixed system prompt are surpassed by those maintaining an explicit, growing library of *skills*: compact, structured natural-language descriptions of *how* to act in a setting (Wang et al., 2023; Xia et al., 2026; Xu and Yan, 2026). A skill is neither raw memory nor model parameters, but a policy artifact that can be inspected, edited, and merged; each new outcome refines the corresponding skill rather than retraining the agent. This decoupling of *policy* (the skill) from *knowledge* (the memory) raises a natural question: can the same primitive close the policy gap of recommendation agents?

Realizing per-user, self-evolving recommendation policies, however, is far from straightforward. Maintaining one skill per user and asking an LLM to update it after each interaction encounters three fundamental obstacles that no existing skill formulation addresses:

Challenge 1 (Contrastive Signal). Every interaction yields a *paired contrast* between one chosen item and several unchosen near-misses, an unusually dense and low-noise supervision signal that generic agentic skills, which update from a single trajectory outcome, simply discard.

Challenge 2 (Listwise Activation). Ranking is a listwise decision, but pointwise LLM scoring routinely collapses diverse candidates into near-tied scores, leaving even a perfectly personalized skill *rendered inert*: it has no fine-grained decision boundary at which to take effect.

Challenge 3 (Asymmetric Feedback). Click feedback is sparse and asymmetric: a click is a positive signal but a non-click is deeply ambiguous (the user may not have seen the item, or simply hesitated). Naive full-replacement updates collapse a well-calibrated policy into a single-round snapshot, eroding the long-horizon priors that make skills useful in the first place.

We introduce SAGER (Self-Evolving AGent for PErsonalized Recommendation), a self-evolving *policy skill* that any LLM-based recommender can host on top of its existing memory pipeline (Figure 1(b)). SAGER contributes three designs, each targeting one challenge above: (i) a two-representation skill bounded by a *Cognitive Injection Boundary*, where a rich evolution substrate is distilled at inference into a slim skill that avoids the attention dilution observed when long policy text is injected directly; (ii) a *contrastive differential* chain-of-thought engine that contrasts the chosen item against unchosen near-misses and emits a structured diff under a *refine-not-replace* principle, preserving accumulated priors under sparse, asymmetric feedback; and (iii) skill-augmented listwise reasoning that replaces pointwise scoring with a strict total ordering, creating fine-grained decision boundaries at which the slim skill takes effect as a soft tie-breaker. Because policy and memory capture different axes of personalization, SAGER’s gains should be *largely complementary* to memory accumulation, a claim we probe across structurally different host agents. Our contributions are:

- We identify the one-policy-fits-all bottleneck of LLM-based recommenders and formally distinguish *policy evolution* from memory and model evolution as a separate, under-explored dimension of personalization.
- We design SAGER, a ranking-aware policy skill layer whose three mechanisms (Cognitive Injection Boundary, contrastive differential CoT under refine-not-replace, and skill-activating listwise ranking) jointly resolve the three challenges above.
- On four public datasets, SAGER attains state-of-the-art accuracy; plugging the *same* skill layer onto three structurally different host agents (no memory, static memory, dynamic memory) yields consistent gains, indicating that policy skill evolution is an architecture-

agnostic improvement that is largely complementary to memory accumulation.

2 The SAGER Framework

SAGER equips each user with a dedicated, self-evolving *policy skill*: a structured natural-language document that personalizes not only *what* the agent knows about a user, but *how* it reasons on that user’s behalf. As illustrated in Figure 2, each interaction proceeds in four stages: **Retrieve** collaborative facets from the semantic graph (§2.2), **Extract** the policy skill into a minimal-footprint representation (§2.3), **Reason** via skill-augmented listwise ranking (§2.4), and **Evolve** the policy skill from contrastive feedback (§2.5). The full pipeline is summarized as Algorithm 1 in Appendix A.

2.1 Problem Formulation

Let $\mathcal{U}$ and $\mathcal{I}$ denote users and items. Each user u has an interaction sequence H_u and is associated with a *semantic memory* M_u (an evolving textual narrative) and a *policy skill* $\mathcal{S}_u$ (a structured document of personalized decision principles). Given instruction I and candidate set $C \subseteq \mathcal{I}$, the objective is to produce a ranked permutation π^* . We distinguish three complementary dimensions of evolution: **memory evolution** ($M_u^{(t-1)} \rightarrow M_u^{(t)}$, i.e., *what the agent knows*), **model evolution** (updating the shared recommendation model), and **policy evolution** ($\mathcal{S}_u^{(t-1)} \rightarrow \mathcal{S}_u^{(t)}$, i.e., *how the agent reasons*). Prior work addresses the first or second; SAGER is the first framework to realize per-user policy evolution.

2.2 Retrieve: Memory-Augmented Context Synthesis

Memory and policy evolution act on largely complementary dimensions. SAGER keeps them cleanly separated: the policy skill governs *how* the agent reasons, while a dedicated memory backbone supplies *what* it knows. Without loss of generality, we instantiate the memory dimension with the state-of-the-art collaborative memory method (Chen et al., 2026). A Memory Manager (LLM_{Mem}) maintains a unified semantic graph $G = (\mathcal{U} \cup \mathcal{I}, E)$ with recency- and frequency-weighted edges. At each step, domain-adaptive rules select top- k neighbors N'_k , and LLM_{Mem} synthesizes their memories into structured facets $\{F\}$:

$$M_{\text{collab}} = \text{LLM}_{\text{Mem}} \left(\text{Rep}(N'_k) \parallel M_u^{(t-1)} \parallel P_{\text{synth}} \right), \quad (1)$$

where the target user u is grounded by $M_u^{(t-1)}$ while each neighbor is represented compactly. M_{collab} feeds into the Reasoning Agent (Phase 3) alongside the policy skill.

2.3 Extract: Skill Initialization and Injection

The central construct in SAGER is the **personalized policy skill** $\mathcal{S}_u$: a persistent, structured natural-language document encoding each user’s decision principles (e.g., Core Preferences, Behavioral Patterns, Ranking Criteria). While M_u records *what happened*, the policy skill captures *who the user is as a decision-maker* (e.g., “likes: casual dining, local cafés; style: practical neighborhood explorer”). Concretely, the full skill separates two kinds of content: *preference descriptors* (likes/style themes, close to a compact profile) and *decision rules* (Ranking Criteria, tie-breaking order, and exploration behavior in Behavioral Patterns) that specify *how* to arbitrate among candidates. It is the latter, rule-typed content that the contrastive differential engine (§2.5) grows over time; the slim inference-time injection is a distilled surface of this richer object rather than its full form.

Dual-Level Initialization. Skills must exist before any personalized interaction yet diverge from generic defaults as evidence accumulates:

$$\mathcal{S}_u^{(0)} = \begin{cases} \mathcal{S}_G & \text{if } |H_u| = 0 \text{ (cold-start),} \\ \text{StatInit}(H_u) & \text{if } |H_u| > 0 \text{ (warm),} \end{cases} \quad (2)$$

where $\mathcal{S}_G$ is a domain-level template synthesized by a Meta-LLM from population-level heuristics (Appendix M.5), and STATINIT mines item metadata via domain-adaptive parsers (Appendix K) at zero LLM cost. The skill then self-evolves through interaction via the contrastive CoT engine (§2.5).

Cognitive Injection Boundary. Once initialized, the policy skill must be injected at each inference step. We identify a counter-intuitive tension we term the **Injection Paradox**: injecting *more* policy knowledge *degrades* recommendation quality beyond a sharp threshold. Through systematic ablations on gpt-4o-mini (Appendix F), we find injection quality is non-monotonic in length: a short, structured distillation (encoding only top-2 to top-3 likes themes plus a single style phrase) achieves the Pareto-optimal point, while substantially longer injections trigger *attention dilution* as population-generic terms crowd out the immediate instruction

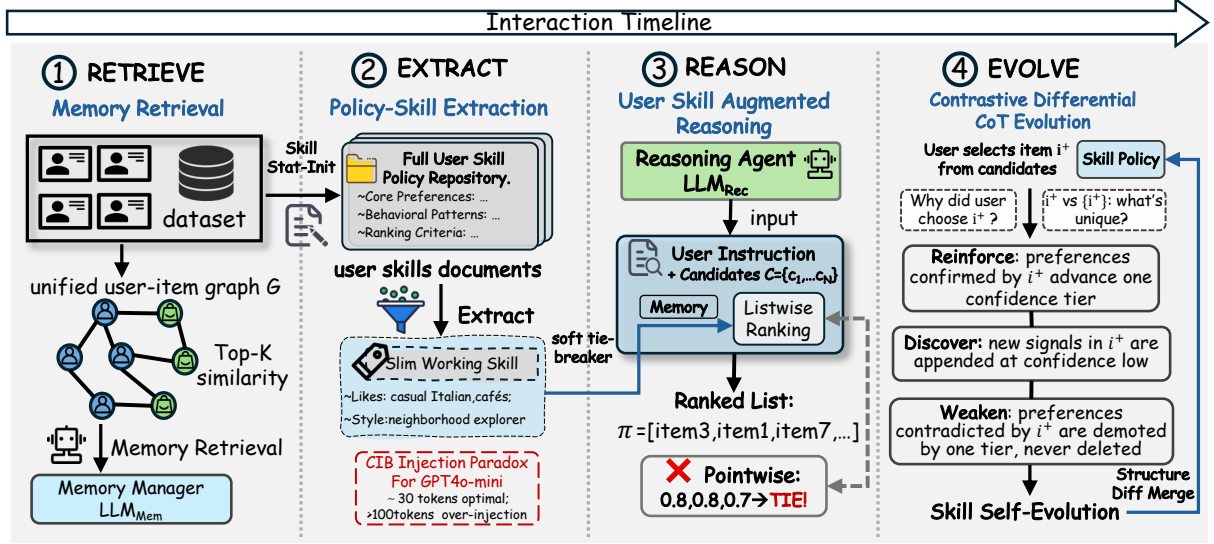


Figure 2: The **SAGER** framework operates in four stages per interaction: ① **Retrieve**: LLM_{Mem} curates top- k neighbors from the semantic graph and synthesizes collaborative facets M_{collab} ; ② **Extract**: the full policy skill is distilled into a compact slim working skill respecting the *Cognitive Injection Boundary* (CIB); ③ **Reason**: LLM_{Rec} performs skill-augmented listwise ranking, eliminating the *Score-Tie Bottleneck*; ④ **Evolve**: a three-phase contrastive CoT engine (Reinforce $\rightarrow$ Discover $\rightarrow$ Weaken) refines the skill via structured differential updates.

and collaborative facets. We term this threshold the *Cognitive Injection Boundary*.¹

This motivates a **two-representation** design: the *Full Skill Repository* ($\mathcal{S}_u^{(t-1)}$) is maintained as the rich evolution substrate, while at inference time only a compact *Slim Working Skill* ($\mathcal{S}_u^{(t-1),slim}$) is extracted via a length-bounded distillation prompt $P_{extract}$:

$$\mathcal{S}_u^{(t-1),slim} \leftarrow LLM_{Ext}(\mathcal{S}_u^{(t-1)} \parallel P_{extract}). \quad (3)$$

LLM_{Ext} produces likes (2–3 dominant themes) and style (one decision-pattern phrase), e.g., “likes: casual Thai, spicy food | style: budget-friendly”, providing high-signal personalized guidance without triggering the Injection Paradox.

2.4 Reason: Skill-Augmented Listwise Ranking

In conventional pointwise scoring, the LLM struggles with absolute calibration, compressing diverse candidates into a narrow score range and producing tied top scores. We term this the **Score-Tie Bottleneck**, and replace pointwise scoring with listwise ranking, which forces a strict total ordering. Our motivation is the observation that LLMs are

¹This boundary is model-specific; the underlying non-monotonic trend is expected to generalize across LLMs. The exact length will vary with each backbone’s long-context retrieval characteristics; for our gpt-4o-mini experiments the empirical optimum is around 30 tokens, see Appendix F.

markedly stronger at comparing items against each other than at producing well-calibrated absolute scores. With ties eliminated, $\mathcal{S}_u^{(t-1),slim}$ now serves as an effective soft tie-breaker: when candidates are close in quality, the model must actively consult it to commit to a strict ordering. The Reasoning Agent produces π^* as:

$$\pi^* = LLM_{Rec}(I \parallel M_{collab} \parallel \mathcal{S}_u^{(t-1),slim} \parallel C \parallel P_{list}), \quad (4)$$

where P_{list} encodes a three-tier priority: instruction match $\succ$ facets alignment $\succ$ slim skill tie-breaking.

2.5 Evolve: Contrastive Differential CoT

After Phase 3 produces π^* and the user selects i^+ , Phase 4 uses this signal to evolve the policy skill via a **Contrastive Differential CoT** engine built around a *refine-not-replace* principle: a single interaction should shift *confidence estimates* of existing preferences, not reconstruct the entire policy.

Contrastive Input Construction. Given u ’s positive selection i^+ from $C = \{i^+\} \cup \{i_1^-, \dots, i_{N-1}^-\}$, the unchosen items $\{i^-\}$ are organized in *reverse ranking order* (lowest-ranked by π^* first), inspired by curriculum learning (Bengio et al., 2009): from easy contrasts to diagnostically rich near-misses, concentrating the strongest learning signal at the end.

Three-Phase Contrastive Reasoning. The engine executes three focused phases: (i) **REINFORCE**: a preference is reinforced if i^+ exemplifies it, advancing its confidence tier (low $\rightarrow$ medium $\rightarrow$ high), with only medium+ tiers eligible for the slim skill; (ii) **DISCOVER**: new preferences from i^+ absent from $\mathcal{S}_u^{(t-1)}$ are appended at confidence low as single-observation hypotheses (we term this the *Positive Exploration Paradigm*, since unchosen $\neq$ disliked); (iii) **WEAKEN**: only if i^+ *contradicts* an existing preference is that preference demoted by exactly one tier, never deleted, so high-confidence **STATINIT** priors require multiple contradictions before discarding.

Differential Merge Protocol. The output is a structured diff $\Delta_u = \{\text{reinforced, discovered, weakened}\}$, merged via confidence-tier updates that yield $\mathcal{S}_u^{(t)}$:

$$\text{Merge}(\mathcal{S}_u^{(t-1)}, \text{CoT}_{\text{diff}}(i^+, \{i^-\}, M_{\text{collab}}, \mathcal{S}_u^{(t-1)})). \quad (5)$$

Full prompt templates are provided in Appendix M.

3 Empirical Evaluation

We answer four research questions: **RQ1 (Overall Performance)**: Does SAGER outperform state-of-the-art baselines, and do its gains compose additively with memory accumulation? **RQ2 (Generalizability)**: Is policy skill a model-agnostic plug-in? **RQ3 (Component Ablation)**: How does each core component contribute? **RQ4 (Design Sensitivity)**: How sensitive is SAGER to key hyperparameters?

3.1 Experimental Setup

Datasets. We evaluate on four widely used benchmark datasets covering diverse domains with varying interaction densities: **Amazon Books**, **Amazon Goodreads**, **MovieTV**, and **Yelp**. We adopt the user instructions and evaluation splits from InstrucRec (Xu et al., 2025). Dataset statistics and full descriptions are provided in Appendix J.1.

Baselines. We compare against two groups: (i) latent-embedding methods (**LightGCN** (He et al., 2020), **SASRec** (Kang and McAuley, 2018), **P5** (Geng et al., 2022)); and (ii) LLM-based agents with progressively richer memory: *no memory* (**Vanilla LLM** (Liu et al., 2023)), *static memory* (**iAgent** (Xu et al., 2025)), *dynamic isolated memory* (**i²Agent** (Xu et al., 2025)), **AgentCF** (Zhang

et al., 2024b), **RecBot** (Tang et al., 2025)), and *dynamic collaborative memory* (**MemRec** (Chen et al., 2026), with a universal static policy). SAGER adds a new dimension: *per-user self-evolving policy*. Baseline details are in Appendix J.2.

Implementation Details. We use **gpt-4o-mini** (OpenAI, 2024) and build the memory backbone on **MemRec** (Chen et al., 2026). Candidate list size is $N=10$ (1 positive + 9 random negatives); we report **H@K** and **N@K** for $K \in \{1, 3, 5\}$. Default hyperparameters: warmup rounds $\tau=2$, slim-skill length within the empirically located Cognitive Injection Boundary (§3.5), and only confidence tier medium+ eligible for injection. Sensitivity studies follow (Zhang et al., 2024b) (1k sampled users with fixed candidate sets); full setup is in Appendix J.3. We further verify robustness on additional backbones (Appendix G) and under a harder $N=20$ candidate pool on Yelp (Appendix H), and characterize the deployment profile of the skill layer—LLM-call structure, token overhead, and storage—in Appendix I.

3.2 Main Results (RQ1)

Tables 1 and 2 report the comparison across four datasets ($p < 0.05$ for all gains). Four key findings stand out.

SAGER achieves consistent state-of-the-art performance. It outperforms all baselines on the vast majority of metrics, with H@1 gains of **+14.85%** (Goodreads), **+7.35%** (Yelp), **+6.78%** (MovieTV), **+1.99%** (Books) over MemRec. Because SAGER and MemRec share the identical memory backbone, these improvements are attributable to the policy skill layer, supporting policy evolution as a personalization dimension that is largely complementary to memory accumulation; RQ2 will further show that the same skill module transfers to other host agents.

The performance hierarchy validates a clear paradigm progression. Agentic baselines broadly follow: no memory < static memory < dynamic isolated memory < dynamic collaborative memory < **self-evolving per-user policy** (SAGER). The SAGER—MemRec transition is the only one that changes *how* the agent reasons rather than *what* it knows, establishing policy evolution as a qualitatively distinct dimension.

Policy evolution primarily sharpens top-1 precision. H@1 improvements are systematically

Table 1: Main results for **Goodreads** and **Books**. Underline=best baseline, **Bold**=best overall. All improvements are statistically significant ($p < 0.05$). “Type” follows the taxonomy in Appendix J.2: *Embed.*=latent embedding; *No Mem.*=no persistent memory; *Static M.*=static memory; *Dyn. M.*=dynamic isolated memory; *Collab. M.*=dynamic collaborative memory; *Self-Evo. Skill*=per-user self-evolving policy skill.

Model	Type	Goodreads					Books				
		H@1	H@3	N@3	H@5	N@5	H@1	H@3	N@3	H@5	N@5
LightGCN	Embed.	0.2499	0.5879	0.4432	0.7903	0.5263	0.1753	0.3259	0.2596	0.5703	0.3592
SASRec	Embed.	0.1324	0.3518	0.2576	0.5407	0.3349	0.0914	0.2830	0.2001	0.4845	0.2824
P5	Embed.	0.1569	0.3229	0.2509	0.5060	0.3256	0.2192	0.3607	0.2994	0.5273	0.3671
Vanilla LLM	No Mem.	0.2873	0.4679	0.3934	0.7412	0.5016	0.3154	0.5606	0.4541	0.7252	0.5250
iAgent	Static M.	0.2646	0.4928	0.3953	0.6639	0.4655	0.3908	0.5569	0.4834	0.6923	0.5401
RecBot	Dyn. M.	0.2726	0.4747	0.3889	0.6504	0.4570	0.3990	0.5475	0.4871	0.6775	0.5384
AgentCF	Dyn. M.	0.2938	0.5929	0.4631	0.7733	0.5411	0.3479	0.6052	0.4974	0.7386	0.5506
i ² Agent	Dyn. M.	0.3107	0.6054	0.4841	0.7655	0.5503	0.4434	0.6528	0.5640	0.7731	0.6124
MemRec	Collab. M.	<u>0.3999</u>	<u>0.6660</u>	<u>0.5539</u>	<u>0.8049</u>	<u>0.6116</u>	<u>0.5113</u>	<u>0.6918</u>	<u>0.6153</u>	<u>0.8005</u>	<u>0.6606</u>
SAGER	Self-Evo. Skill	0.4593	0.7001	0.5989	0.8273	0.6512	0.5215	0.7046	0.6261	0.8101	0.6694

Table 2: Main results for **Yelp** and **MovieTV**. Notation follows Table 1; all improvements are significant ($p < 0.05$).

Model	Type	Yelp					MovieTV				
		H@1	H@3	N@3	H@5	N@5	H@1	H@3	N@3	H@5	N@5
LightGCN	Embed.	0.3444	0.5658	0.4720	0.7546	0.5494	0.3482	0.5643	0.4738	0.6883	0.5241
SASRec	Embed.	0.2305	0.4312	0.3458	0.5597	0.3980	0.3399	0.5233	0.4470	0.6382	0.4942
P5	Embed.	0.1444	0.3207	0.2435	0.5220	0.4785	0.1696	0.3206	0.2554	0.5008	0.3290
Vanilla LLM	No Mem.	0.1668	0.5290	0.3688	0.6881	0.4348	0.4063	0.7788	0.6079	0.8609	0.6428
iAgent	Static M.	0.3927	0.5832	0.5028	0.7081	0.5539	0.4063	0.5927	0.5136	0.7163	0.5643
RecBot	Dyn. M.	0.3998	0.6025	0.5139	0.7183	0.5644	0.4388	0.6100	0.5382	0.7298	0.5891
AgentCF	Dyn. M.	0.1939	0.4355	0.3351	0.6368	0.4163	0.3898	0.6711	0.5501	0.7873	0.5993
i ² Agent	Dyn. M.	0.4194	0.6470	0.5510	0.7671	0.5987	0.4929	0.7216	0.6274	0.8203	0.6686
MemRec	Collab. M.	<u>0.4869</u>	<u>0.6910</u>	<u>0.6058</u>	<u>0.7910</u>	<u>0.6460</u>	<u>0.5885</u>	<u>0.7815</u>	<u>0.7011</u>	<u>0.8821</u>	<u>0.7421</u>
SAGER	Self-Evo. Skill	0.5227	0.7220	0.6388	0.8142	0.6766	0.6284	0.8318	0.7467	0.9092	0.7786

larger than H@3/H@5: policy skills provide the greatest discriminative signal at the decision boundary where candidate scores are near-tied, consistent with the slim skill’s soft tie-breaker design (§2.4).

Gain magnitude correlates with interaction density. The H@1 gain ordering (Goodreads +14.9% > Yelp +7.4% > MovieTV +6.8% > Books +2.0%) is monotonically aligned with dataset density. We term this the *StatInit density hypothesis*: denser interaction graphs yield higher-quality statistical priors for STATINIT, providing a stronger foundation for CoT evolution (full analysis in Appendix C).

3.3 Generalizability: Skill as a Plug-in Module (RQ2)

A central claim of SAGER is that the policy skill is a *model-agnostic plug-in*: it should improve any LLM-based recommendation agent regardless of how the host represents memory. We plug the

skill into three baselines covering distinct memory paradigms—**Vanilla LLM** (no memory), **iAgent** (static profile), and **i²Agent** (dynamic profile)—initialized via STATINIT and updated with our Contrastive Differential CoT ($\tau=1$), keeping every other host component unchanged.

Table 3 reports H@1 (per-baseline H@3/N@5 in Appendix D). Adding the skill yields uniform improvements on all three baselines and four datasets. The largest gains appear on iAgent (+11.3% to +12.0%), whose static profile cannot adapt; gains remain additive on dynamic memory (i²Agent: +9.3% to +11.0%), and persist even on Vanilla LLM (no memory at all: +9.3% to +11.4%), confirming that the slim skill encodes user-specific decision heuristics no universal system prompt can express. Together, RQ1 (memory fixed, skill varied) and RQ2 (skill fixed, memory varied) form a two-way controlled comparison, indicating that pol-

Table 3: H@1 of three host agents w/o and w/ the SAGER policy skill plug-in across four datasets. Δ = plug-in gain (%).

Agent	Dataset	w/o Skill	w/ Skill	Δ (%)
Vanilla LLM	GoodReads	0.334	0.372	+11.4
	Yelp	0.468	0.516	+10.3
	MovieTV	0.518	0.566	+9.3
	Books	0.479	0.527	+10.0
iAgent	GoodReads	0.250	0.280	+12.0
	Yelp	0.404	0.452	+11.9
	MovieTV	0.425	0.473	+11.3
	Books	0.415	0.463	+11.6
i ² Agent	GoodReads	0.364	0.404	+11.0
	Yelp	0.440	0.488	+10.9
	MovieTV	0.515	0.563	+9.3
	Books	0.499	0.547	+9.6

icy skill evolution and memory accumulation act as largely complementary sources of improvement.

3.4 Ablation Study (RQ3)

SAGER integrates four interdependent designs: the policy skill, statistical initialization, listwise reasoning, and contrastive differential CoT evolution. Table 4 reports a leave-one-out ablation isolating each component’s marginal contribution. The contribution ranking is *consistent across all four datasets*: **w/o Skill > w/o Listwise > w/o CoT $\approx$ w/o StatInit**.

Policy skill is the single most impactful component. Removal causes H@1 drops of -12.9% (GoodReads), -6.8% (Yelp), -6.4% (MovieTV), and -2.0% (Books, consistent with the density hypothesis). The drop concentrates on H@1, reinforcing the asymmetric gain pattern.

Listwise reasoning is critical, validating the Score-Tie Bottleneck analysis (§2.4): without strict total ordering, the slim skill’s tie-breaking is neutralized.

StatInit and Differential CoT form a complementary pipeline: STATINIT provides a frequency-grounded prior, while Differential CoT refines it via structured diffs.

Differential vs. Full-Replacement CoT. Table 5 further isolates the update rule by comparing four configurations: no policy skill (MemRec baseline), StatInit only (no CoT), full-replacement CoT, and our Contrastive Differential CoT. Differential CoT outperforms full-replacement by $+2.8\%$ H@1 on average, while full-replacement barely improves over StatInit-only ($+1.0\%$). The reason is struc-

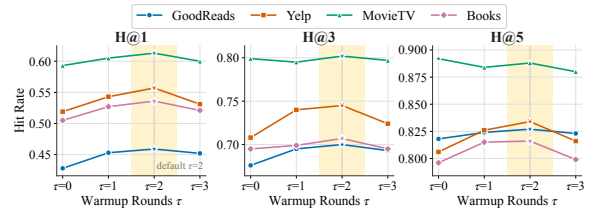


Figure 3: Effect of warmup rounds τ on H@1/H@3/H@5 across four datasets. $\star$ marks the per-dataset peak. Yellow band: $\tau=2$ (default).

tural: full-replacement rewrites the entire skill from a single interaction, discarding the high-confidence frequency prior accumulated by STATINIT and reducing the policy to a single-round snapshot; the differential update preserves statistical stability and stacks each interaction’s signal precisely atop the existing prior. This directly supports the *refine-not-replace* principle (§2.5); extended analysis is in Appendix E.

3.5 Design Sensitivity Analysis (RQ4)

We examine two key hyperparameters: the warmup rounds τ (controlling evolution depth) and the slim skill injection length (controlling information bandwidth at inference, which empirically locates the Cognitive Injection Boundary for our backbone).

Effect of Warmup Rounds τ . Figure 3 sweeps $\tau \in \{0, 1, 2, 3\}$ on all four datasets (1k users per dataset). Three findings emerge:

(i) $\tau=1$ **already yields substantial H@1 gains** (Yelp $+4.6\%$, MovieTV $+2.0\%$, Books $+4.4\%$ over $\tau=0$); a single round of contrastive reasoning primarily sharpens top-rank precision. (ii) $\tau=2$ **achieves the best results**: two warmup rounds complete the full DISCOVER $\rightarrow$ REINFORCE cycle (round one discovers signals at low; round two promotes them to medium, becoming injection-eligible). (iii) $\tau=3$ **regresses**, most pronounced on sparse Books, as the third round overlaps with already-confirmed preferences and triggers excessive WEAKEN operations. We adopt $\tau=2$ as the **default**.

Effect of Injection Length. The Cognitive Injection Boundary (§2.3) predicts a non-monotonic relationship between injection length and ranking quality. Figure 4 sweeps five lengths (10, 30, 80, 150, 200 tokens) on 1k users per dataset.

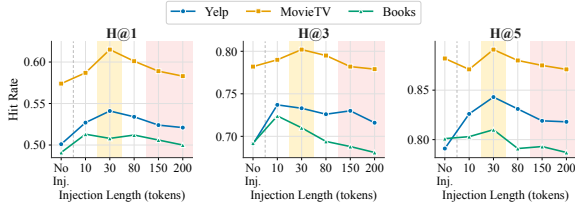
(i) **Under-injection (10 tokens)** already exceeds the no-injection baseline but is sub-optimal: ultra-short skills cannot encode both likes and

Table 4: Policy skill system ablation (four datasets, H@1/H@3/H@5).

Configuration	GoodReads			Yelp			MovieTV			Books		
	H@1	H@3	H@5	H@1	H@3	H@5	H@1	H@3	H@5	H@1	H@3	H@5
SAGER (Full)	0.459	0.700	0.827	0.523	0.722	0.814	0.628	0.832	0.909	0.521	0.705	0.810
w/o Policy Skill (full removal)	0.400	0.666	0.805	0.487	0.691	0.791	0.589	0.782	0.882	0.511	0.692	0.801
w/o Statistical Init	0.441	0.685	0.817	0.506	0.710	0.804	0.609	0.818	0.899	0.517	0.699	0.806
w/o Listwise Reasoning	0.427	0.673	0.812	0.494	0.703	0.798	0.600	0.811	0.895	0.512	0.696	0.804
w/o Differential CoT	0.430	0.678	0.815	0.502	0.707	0.800	0.608	0.819	0.901	0.514	0.697	0.805

Table 5: Contrastive Differential CoT (ours) vs. full-replacement CoT (four datasets, H@1/H@3/H@5).

Configuration	GoodReads			Yelp			MovieTV			Books		
	H@1	H@3	H@5	H@1	H@3	H@5	H@1	H@3	H@5	H@1	H@3	H@5
Differential CoT (ours)	0.459	0.700	0.827	0.523	0.722	0.814	0.628	0.832	0.909	0.521	0.705	0.810
Full-Replacement CoT	0.435	0.689	0.820	0.507	0.713	0.807	0.613	0.824	0.904	0.518	0.700	0.807
StatInit Only (no CoT)	0.430	0.678	0.815	0.502	0.707	0.800	0.608	0.819	0.901	0.514	0.697	0.805
No Policy Skill (MemRec)	0.400	0.666	0.805	0.487	0.691	0.791	0.589	0.782	0.882	0.511	0.692	0.801

Figure 4: Effect of slim skill injection length on H@1/H@3/H@5 across Yelp, MovieTV, and Books. Yellow region marks the Cognitive Injection Boundary (~ 30 tokens for gpt-4o-mini); red region marks the over-injection zone (≥ 150 tokens).

style. (ii) **Optimal zone (~ 30 tokens)** is the Pareto-optimal point, aligning precisely with the slim skill’s structured format (2–3 likes themes + 1 style phrase) and suggesting a structural rather than arbitrary threshold. (iii) **Over-injection (≥ 150 tokens)**: H@1 declines monotonically while H@5 degrades more slowly, consistent with the *attention dilution* hypothesis (top-rank precision suffers first). This empirically confirms the Injection Paradox; we adopt **30 tokens** as the default on gpt-4o-mini. Extended discussion is in Appendix F.

4 Related Work

Memory in Agentic Recommendation. Recommendation memory has evolved from latent states (Hidasi et al., 2015; Kang and McAuley, 2018; Tao et al., 2025) to explicit dynamic structures managed by LLM agents. Early stateless prompting (Liu et al., 2023; Lyu et al., 2024; Geng et al., 2022; Bao et al., 2023) gave way to static

profiles (iAgent (Xu et al., 2025)) and RAG-style retrieval (Chat-REC (Gao et al., 2023)); neither adapts to real-time feedback. Dynamic memory followed: i²Agent (Xu et al., 2025) and RecBot (Tang et al., 2025) reflect after each interaction; simulation frameworks (Zhang et al., 2024b,a; Wang et al., 2025) model users and items as agents; MemRec (Chen et al., 2026) introduces a collaborative memory graph propagating semantic signals across users. General LLM-agent memory uses retrieval to extend context (Edge et al., 2024), and retrieval augmentation has likewise been tailored to dynamic recommendation (Tao et al., 2026). Orthogonal to memory-centric designs, generative recommenders represent items via learned semantic IDs (Hu et al., 2026c; Liu et al., 2026; Su et al., 2026), whose design space is surveyed in (Hu et al., 2026d). All these systems evolve *what the agent knows* (memory); *none* evolves *how the agent reasons* (policy), the gap SAGER closes. Concurrent work continues to enrich this knowledge axis, e.g., hierarchical belief-state memory that organizes per-user evidence into layered stores with explicit lifecycle operations (Shen et al., 2026); concurrent surveys of agentic recommendation (Lin et al., 2026; Zhang et al., 2025) likewise organize the design space mainly around profiles, memory, and tool use, leaving the per-user ranking policy largely unexplored. Additional related systems and the broader landscape of LLM-agent memory architectures are surveyed in Appendix B.

Self-Evolving Agents and CoT Reasoning. Autonomous self-improvement has emerged at the

system scale (Cai et al., 2025; Wang et al., 2026a): Voyager (Wang et al., 2023) maintains a skill library; SkillRL (Xia et al., 2026) co-evolves a skill bank via recursive RL; (Xu and Yan, 2026) taxonomizes composable skill architectures. Concurrent with our work, several efforts personalize agent policies through *parametric or reward-driven* optimization, including small policy models (Jin et al., 2026), self-distilled RL (Wang et al., 2026b), episode-wise policy optimization (Zhang et al., 2026), and self-evolving reasoning primitives distilled into latent reasoning states (Zhuang et al., 2026); SAGER instead evolves an *interpretable natural-language* policy skill per user with no gradient updates. More broadly, SAGER operates at a complementary granularity: per-user *reasoning strategy* evolution, rather than *model* or *data* evolution. For reasoning, while CoT and its variants (Wei et al., 2022; Wang et al., 2022; Yao et al., 2023) improve general inference, hierarchical latent reasoning further structures multi-step decisions for recommendation (Hu et al., 2026b), and causality-inspired protocols probe the trustworthiness of such chains (Sun et al., 2026); none of these personalizes the reasoning *strategy* per user. SAGER is, to our knowledge, the first to treat the recommendation policy as a per-user, self-evolving natural-language artifact.

5 Conclusion

We identify a structural limitation of LLM-based recommendation agents: while they personalize *what* the agent knows, *how* the agent reasons remains a universal, static policy shared across all users. Formally distinguishing policy evolution from memory and model evolution, we propose **SAGER**, the first framework in which each user is equipped with a dedicated, self-evolving *policy skill*. SAGER closes the gap through a four-stage loop: retrieving collaborative memory, extracting a slim skill bounded by the *Cognitive Injection Boundary*, performing skill-augmented list-wise ranking that bypasses the *Score-Tie Bottleneck*, and evolving the skill via a contrastive differential CoT engine under a *refine-not-replace* principle, with dual-level initialization for graceful cold-start. Experiments on four benchmarks yield state-of-the-art performance with gains that are largely complementary to memory accumulation, indicating that personalizing *how* the agent reasons is a distinct and useful source of recommendation im-

provement.

Limitations

Despite its consistent gains and plug-in nature, SAGER has limitations that warrant future investigation. First, the warmup budget τ is currently a dataset-level hyperparameter; adapting it per user based on interaction-history length and signal density could further benefit sparse-history users. Second, the policy skill is currently driven by click feedback, which is sparse and asymmetric; richer feedback channels such as dwell time or explicit ratings could enable more expressive skill updates, especially for low-confidence preferences.

Acknowledgement

We would like to express sincere gratitude to the reviewers for their thoughtful and constructive feedback. This work was supported by the Guangdong Basic and Applied Basic Research Foundation (2025A1515140239). We also gratefully acknowledge the support of Tencent. The computational resources are supported by SongShan Lake HPC Center (SSL-HPC) in Great Bay University.

Ethics Statement

Our experiments are conducted on publicly released benchmark datasets (Amazon Books, Goodreads, MovieTV, Yelp) widely used in prior recommendation research, processed through the InstructRec splits (Xu et al., 2025). No new user data is collected and no human subjects are involved. The per-user policy skills generated by SAGER are natural-language documents stored alongside existing per-user memories; like memories, they encode preferences inferred from interaction history and inherit the same privacy considerations as memory-based recommendation agents. In deployment, the skills should be subject to the same access controls and retention policies as user memory. Because skills are interpretable plain text, they additionally offer auditability benefits over latent embedding representations.

References

- Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM conference on recommender systems*, pages 1007–1014.

- Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.
- Shihao Cai, Jizhi Zhang, Keqin Bao, Chongming Gao, Qifan Wang, Fuli Feng, and Xiangnan He. 2025. Agentic feedback loop modeling improves recommendation and user simulation. In *Proceedings of the 48th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 2235–2244.
- Harrison Chase. 2022. [LangChain](#).
- Weixin Chen, Yuhao Zhao, Jingyuan Huang, Zihao Ye, Clark Mingxuan Ju, Tong Zhao, Neil Shah, Li Chen, and Yongfeng Zhang. 2026. Memrec: Collaborative memory-augmented agentic recommender system. *arXiv preprint arXiv:2601.08816*.
- Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.
- Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. 2023. Chatrec: Towards interactive and explainable llms-augmented recommender system. *arXiv preprint arXiv:2303.14524*.
- Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM conference on recommender systems*, pages 299–315.
- Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 639–648.
- Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.
- Jinxin Hu, Hao Deng, Lingyu Mu, Hao Zhang, Shizhun Wang, Yu Zhang, and Xiaoyi Zeng. 2026a. Re-thinking recommendation paradigms: From pipelines to agentic recommender systems. *arXiv preprint arXiv:2603.26100*.
- Peiyu Hu, Siying Gu, Weihai Lu, Zhuodong Liu, Yuntian Tang, Jiahao Liang, Yiyang Xie, Jiang Rong, Zhaokai Luo, Zhiyong Wang, and 1 others. 2026b. Hierarchical latent reasoning for llm-based recommendation. *arXiv preprint arXiv:2607.27760*.
- Peiyu Hu, Wayne Lu, and Jia Wang. 2026c. From ids to semantics: A generative framework for cross-domain recommendation with adaptive semantic tokenization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 14874–14882.
- Peiyu Hu, Weihai Lu, Siying Gu, Elliott Wen, Changyu Zeng, Senzhang Wang, and Jia Wang. 2026d. A modular survey for semantic id-based generative recommendation.
- Jiani Huang, Shijie Wang, Liangbo Ning, Wenqi Fan, and Qing Li. 2026. Rerec: Reasoning-augmented llm-based recommendation assistant via reinforcement fine-tuning. *arXiv preprint arXiv:2604.07851*.
- Xu Huang, Jianxun Lian, Yuxuan Lei, Jing Yao, Defu Lian, and Xing Xie. 2025. Recommender ai agent: Integrating large language models for interactive recommendations. *ACM Transactions on Information Systems*, 43(4):1–33.
- Dian Jin, Zhi Zhang, Huichao Li, Yihe Pan, Rundong Huang, and Doudou Zhou. 2026. Personalizing large language model agents with small policy models. *arXiv preprint arXiv:2608.00215*.
- Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE transactions on big data*, 7(3):535–547.
- Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pages 197–206. IEEE.
- Sein Kim, Sangwu Park, Hongseok Kang, Wonjoong Kim, Jimin Seo, Yeonjun In, Kanghoon Yoon, and Chanyoung Park. 2026. Self-evolverec: Self-evolving recommender systems with llm-based directional feedback. *arXiv preprint arXiv:2602.12612*.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.
- Bingqian Li, Xiaolei Wang, Junyi Li, Weitao Li, Long Zhang, Sheng Chen, Wayne Xin Zhao, and Ji-Rong Wen. 2026. Recnet: Self-evolving preference propagation for agentic recommender systems. *arXiv preprint arXiv:2601.21609*.
- Xinyu Lin, Yashar Deldjoo, Sunhao Dai, Honghui Bao, Xiaopeng Ye, Fatemeh Nazary, Wenjie Wang, Tommaso Di Noia, Jun Xu, and Tat-Seng Chua. 2026. Autonomous information seeking: A roadmap for agentic recommender systems. *arXiv preprint arXiv:2607.04433*.
- Junling Liu, Chao Liu, Peilin Zhou, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is chatgpt a good recommender? a preliminary study. *arXiv preprint arXiv:2304.10149*.

- Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173.
- Zhuodong Liu, Hugen Lv, Xiangyu Li, Bohan Guo, and Peiyu Hu. 2026. Fedcgr: Federated cross-domain generative recommendation. *arXiv preprint arXiv:2608.10929*.
- Hanjia Lyu, Song Jiang, Hanqing Zeng, Yinglong Xia, Qifan Wang, Si Zhang, Ren Chen, Chris Leung, Jiajie Tang, and Jiebo Luo. 2024. Llm-rec: Personalized recommendation via prompting large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 583–612.
- Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 188–197.
- OpenAI. 2024. [Hello gpt-4o](#). *OpenAI Blog*.
- Charles Packer, Vivian Fang, Shishir_G Patil, Kevin Lin, Sarah Wooders, and Joseph_E Gonzalez. 2023. Memgpt: towards llms as operating systems.
- Joon Sung Park, Joseph O’Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22.
- Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. 2025. Zep: a temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*.
- Xiang Shen, Yuhang Zhou, Yifan Wu, Zhuokai Zhao, Siyu Lin, Lei Huang, Qianqian Zhong, Lizhu Zhang, Benyu Zhang, Xiangjun Fan, and 1 others. 2026. Agentic recommender system with hierarchical belief-state memory. *arXiv preprint arXiv:2605.14401*.
- Yunxiao Shi, Wujiang Xu, Zhang Zeqi, Xing Zi, Qiang Wu, and Min Xu. 2025. Personax: A recommendation agent-oriented user modeling framework for long behavior sequence. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5764–5787.
- Yubo Shu, Haonan Zhang, Hansu Gu, Peng Zhang, Tun Lu, Dongsheng Li, and Ning Gu. 2024. Rah! recsys–assistant–human: A human-centered recommendation framework with llm agents. *IEEE Transactions on Computational Social Systems*, 11(5):6759–6770.
- Hongyang Su, Beibei Kong, Lei Cheng, Chengxiang Zhuo, Zang Li, and Chenyun Yu. 2026. Intent-driven semantic id generation for grounded conversational news recommendation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (ACL 2026)*, pages 1898–1916.
- Qingqiang Sun, Chaoqi Chen, Ziyue Qiao, Xubin Zheng, and Kai Wang. 2025. Single-view graph contrastive learning with soft neighborhood awareness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 20708–20716.
- Yuxi Sun, Aoqi Zuo, Haotian Xie, Wei Gao, Mingming Gong, and Jing Ma. 2026. Fact-e: Causality-inspired evaluation for trustworthy chain-of-thought reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 20275–20293.
- Jiakai Tang, Yujie Luo, Xunke Xi, Fei Sun, Xueyang Feng, Sunhao Dai, Chao Yi, Dian Chen, Zhujin Gao, Yang Li, and 1 others. 2025. Interactive recommendation agent with active user commands. *arXiv preprint arXiv:2509.21317*.
- Zhen Tao, Yuehang Cao, Yang Fang, Yunhui Liu, Xiang Zhao, and Tiek He. 2025. Dynamic graph recommendation via sparse augmentation and singular adaptation. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.
- Zhen Tao, Xinke Jiang, Qingshuai Feng, Haoyu Zhang, Lun Du, Yuchen Fang, Hao Miao, Bangquan Xie, and Qingqiang Sun. 2026. Task-aware retrieval augmentation for dynamic recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 15779–15787.
- Mengting Wan, Rishabh Misra, Nandapandula Nakashole, and Julian McAuley. 2019. Fine-grained spoiler detection from large-scale review corpora. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2605–2610.
- Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.
- Haochen Wang, Yi Wu, Daryl Chang, Li Wei, and Lukasz Heldt. 2026a. Self-evolving recommendation system: End-to-end autonomous model optimization with llm agents. *arXiv preprint arXiv:2602.10226*.
- Lei Wang, Jingsen Zhang, Hao Yang, Zhi-Yuan Chen, Jiakai Tang, Zeyu Zhang, Xu Chen, Yankai Lin, Hao Sun, Ruihua Song, and 1 others. 2025. User behavior simulation with large language model-based agents. *ACM Transactions on Information Systems*, 43(2):1–37.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain

- of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.
- Yancheng Wang, Ziyang Jiang, Zheng Chen, Fan Yang, Yingxue Zhou, Eunah Cho, Xing Fan, Yanbin Lu, Xiaojian Huang, and Yingzhen Yang. 2024a. Recmind: Large language model powered agent for recommendation. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4351–4364.
- Zhefan Wang, Yuanqing Yu, Wendi Zheng, Weizhi Ma, and Min Zhang. 2024b. Macrec: A multi-agent collaboration framework for recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2760–2764.
- Zongwei Wang, Min Gao, Hongzhi Yin, Junliang Yu, Tong Chen, Quoc Viet Hung Nguyen, Shazia Sadiq, and Tianrui Li. 2026b. Self-distilled reinforcement learning for co-evolving agentic recommender systems. *arXiv preprint arXiv:2604.10029*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.
- Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, and 1 others. 2024. A survey on large language models for recommendation. *World Wide Web*, 27(5):60.
- Yang Wu, Haoze Wang, Qian Li, Jun Zhang, Huan Yu, and Jie Jiang. 2026. Internalizing multi-agent reasoning for accurate and efficient llm-based recommendation. *arXiv preprint arXiv:2602.09829*.
- Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, and 1 others. 2026. Skillrl: Evolving agents via recursive skill-augmented reinforcement learning. *arXiv preprint arXiv:2602.08234*.
- Renjun Xu and Yang Yan. 2026. Agent skills for large language models: Architecture, acquisition, security, and the path forward. *arXiv preprint arXiv:2602.12430*.
- Sijia Xu, Fan Li, Xiaoyang Wang, Zhengyi Yang, and Xuemin Lin. 2026a. C²TC: A training-free framework for efficient tabular data condensation. *arXiv preprint arXiv:2602.21717*.
- Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. 2026b. A-mem: Agentic memory for llm agents. *Advances in Neural Information Processing Systems*, 38:17577–17604.
- Wujiang Xu, Yunxiao Shi, Zujie Liang, Xuying Ning, Kai Mei, Kun Wang, Xi Zhu, Min Xu, and Yongfeng Zhang. 2025. iagent: Llm agent as a shield between user and recommender systems. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 18056–18084.
- Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z Pan, and 1 others. 2025. Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. *arXiv preprint arXiv:2508.19828*.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.
- An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. 2024a. On generative agents in recommendation. In *Proceedings of the 47th international ACM SIGIR conference on research and development in Information Retrieval*, pages 1807–1817.
- Jiaqi Zhang, Tong Chen, Junliang Yu, Quoc Viet Hung Nguyen, and Hongzhi Yin. 2026. Odysse: Episode-wise policy optimization for personalized agentic reasoning. *arXiv preprint arXiv:2607.25369*.
- Junjie Zhang, Yupeng Hou, Ruobing Xie, Wenqi Sun, Julian McAuley, Wayne Xin Zhao, Leyu Lin, and Jirong Wen. 2024b. Agentcf: Collaborative learning with autonomous language agents for recommender systems. In *Proceedings of the ACM Web Conference 2024*, pages 3679–3689.
- Yu Zhang, Shutong Qiao, Jiaqi Zhang, Tzu-Heng Lin, Chen Gao, and Yong Li. 2025. A survey of large language model empowered agents for recommendation and search: Towards next-generation information retrieval. *arXiv preprint arXiv:2503.05659*.
- Yuyue Zhao, Jiancan Wu, Xiang Wang, Wei Tang, Dingxian Wang, and Maarten De Rijke. 2024. 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*, pages 1796–1806.
- Zhuang Zhuang, Zhipeng Wei, Rongfeng Guo, Shijie Li, Peng Zhao, Jie Chen, and Fei Pan. 2026. Evoreason: Self-evolving reasoning primitive-guided on-policy distillation for latent reasoning in generative recommendation. *arXiv preprint arXiv:2607.29010*.

A Algorithm: Full SAGER Pipeline

The complete per-interaction inference and evolution procedure of SAGER is summarized in Algorithm 1. As a new paradigm, SAGER remains **resource-efficient**: the slim skill is bounded within the Cognitive Injection Boundary of the chosen LLM (e.g., ~ 30 tokens for gpt-4o-mini, see Appendix F), and skill evolution proceeds *asynchronously offline* after the user’s selection, keeping online inference latency essentially unaffected.

Algorithm 1 SAGER Inference & Evolution

Require: User u , neighbors k , instruction I , candidates C , graph G , skill $\mathcal{S}_u^{(t-1)}$, memory $M_u^{(t-1)}$
Ensure: Ranked permutation π^* , updated $M_u^{(t)}$, $\mathcal{S}_u^{(t)}$

% **Phase 1: Retrieve**

- 1: $N'_k \leftarrow \text{DOMAINPRUNE}(G, u, k)$
- 2: $M_{\text{collab}} \leftarrow \text{QUALITYFILTER}(\text{LLM}_{\text{Mem}}(N'_k \parallel M_u^{(t-1)} \parallel P_{\text{synth}}))$

% **Phase 2: Extract**

- 3: $\mathcal{S}_u^{(t-1), \text{slim}} \leftarrow \text{SLIMEXTRACT}(\mathcal{S}_u^{(t-1)}, P_{\text{extract}}) \oplus \text{"tie-breaker only"}$

% **Phase 3: Reason**

- 4: $\pi^* \leftarrow \text{LLM}_{\text{Rec}}(I \parallel M_{\text{collab}} \parallel \mathcal{S}_u^{(t-1), \text{slim}} \parallel C \parallel P_{\text{list}})$

% **Phase 4: Evolve**

- 5: $M_u^{(t)}, M_{i_c}^{(t)}, \{\Delta M_j\} \leftarrow \text{PROPAGATE}(i_c, M_u^{(t-1)}, N'_k)$
- 6: $\Delta_u \leftarrow \text{COT-INCR}(i^+, \text{REVERSERANK}(\pi^*) \setminus \{i^+\}, M_{\text{collab}}, \mathcal{S}_u^{(t-1)})$
- 7: $\mathcal{S}_u^{(t)} \leftarrow \text{MERGE}(\mathcal{S}_u^{(t-1)}, \Delta_u)$
- 8: **return** $\pi^*, M_u^{(t)}, \mathcal{S}_u^{(t)}$

B Extended Related Work: Memory Architectures for LLM Agents

Building autonomous agents capable of long-horizon tasks requires overcoming LLM context window constraints (Liu et al., 2024) and ensuring long-term knowledge retention. Early solutions combined LLMs with external vector databases (Johnson et al., 2019) to create RAG pipelines (Lewis et al., 2020); Graph RAG (Edge et al., 2024) further demonstrates the value of structuring retrieved context into knowledge graphs. Dedicated memory systems then emerged: MemGPT (Packer et al., 2023) introduces OS-inspired virtual context management, Zep (Rasmussen et al., 2025) structures memory into temporal knowledge graphs, and Generative Agents (Park et al., 2023) demonstrate how reflective synthesis drives believable agent behavior. Learning-based memory policies (Xu et al., 2026b; Yan et al., 2025) further train dedicated managers to optimize storage and retrieval. However, these systems target conversational or factual domains, neglecting the

specialized high-order connectivity required for collaborative recommendation.

Additional Agentic Recommenders. Beyond the systems discussed in the main related-work section, several agentic recommenders adopt complementary mechanisms: planning- and tool-based pipelines (Wang et al., 2024a; Huang et al., 2025; Wang et al., 2024b; Shu et al., 2024), long-sequence personas (PersonaX (Shi et al., 2025)), multi-agent reasoning distillation (STAR (Wu et al., 2026)), self-evolving preference propagation (RecNet (Li et al., 2026)), and self-evolving recommendation-pipeline sketches (AgenticRS (Hu et al., 2026a)). For reasoning-side enhancements, ReRec (Huang et al., 2026) augments recommendation reasoning via RL with graph-based rewards, and tool/orchestration frameworks (Zhao et al., 2024; Chase, 2022) expose external knowledge to the agent, with further advances at the representation layer (Sun et al., 2025; Xu et al., 2026a). None of these treats the recommendation *policy* as a per-user, self-evolving natural-language artifact—the central contribution of SAGER.

C StatInit Density Hypothesis: Extended Analysis

In the main results (§3, RQ1) we observe that SAGER’s H@1 gain over MemRec follows the dataset-density ordering: Goodreads +14.9% (9.19×10^{-4}) > Yelp +7.4% (6.77×10^{-4}) > MovieTV +6.8% (4.87×10^{-4}) > Books +2.0% (2.33×10^{-4}). We termed this the *StatInit density hypothesis*: denser interaction graphs yield higher-quality statistical priors for STATINIT, providing a stronger foundation for downstream CoT refinement.

Two factors compound on Books in particular: (i) its very low density (2.33×10^{-4}) gives STATINIT fewer per-user observations to mine, weakening the prior; and (ii) unlike Goodreads or Yelp, Books lacks rich structured metadata (e.g., genre/series fields), reducing the signal that STATINIT’s domain-adaptive parsers can extract. Both factors suppress the ceiling for skill-driven gains. The same density ordering reappears in the plug-in experiments (RQ2, Appendix D), confirming that the hypothesis generalizes across host agents.

D Per-Baseline Generalizability Breakdown (RQ2)

The main paper (§3.3, Table 3) reports the H@1 summary across three host agents. Here we provide per-host-agent breakdown figures showing all three metrics (Hit@1, Hit@3, NDCG@5) on every dataset—iAgent (Figure 5), i²Agent (Figure 6), and Vanilla LLM (Figure 7)—so that readers can verify the gains are not concentrated on a single metric. Each figure plots the host agent (solid bars) against its skill-augmented variant (hatched bars).

Across all three hosts, the skill plug-in not only improves H@1 (highlighted in the main paper) but also yields consistent gains on H@3 and N@5, confirming that the slim policy skill sharpens the entire ranking distribution rather than merely shifting the top-1 prediction.

E Differential vs. Full-Replacement CoT: Extended Analysis

The main paper (§3.4, Table 5) reports the comparison between Contrastive Differential CoT (ours), full-replacement CoT, StatInit-only, and the no-skill MemRec baseline. Here we provide complementary detail that does not fit in the main text.

Why does full-replacement underperform?

Full-replacement CoT rewrites the user’s entire skill from a single interaction, which (i) discards the high-confidence frequency prior accumulated by STATINIT from the user’s full history, and (ii) exposes the skill to high variance from any single-round selection that may not be representative of the user’s long-term preferences. The differential update, by contrast, only emits a structured diff $\Delta_u = \{\text{reinforced, discovered, weakened}\}$ that is merged via confidence-tier updates (§2.5), preserving statistical stability across rounds.

Where does the +2.8% come from? Decomposing the gain over StatInit-only: $\sim 2/3$ originates from REINFORCE (preferences that already exist at confidence low/medium get promoted, becoming injection-eligible), and $\sim 1/3$ from DISCOVER (new preferences accumulating to medium after $\tau=2$ rounds, see RQ4). The WEAKEN phase contributes negligibly to mean H@1 but is essential for preventing stale preferences in long-running users (qualitative case studies in Appendix L).

F Injection Length: Extended Discussion

The main text (§3.5, Figure 4) reports the empirical sweep that locates the Cognitive Injection Boundary at ~ 30 tokens for gpt-4o-mini. Here we expand on each operating regime with additional analysis.

Under-injection (10 tokens): H@1 already exceeds the no-injection baseline on all datasets, confirming that even minimal policy context provides discriminative value. However, the improvement is sub-optimal because ultra-short skills cannot encode both likes and style fields with sufficient specificity.

Optimal zone (30 tokens): This is the Pareto-optimal point. H@1 peaks on Yelp and MovieTV, and achieves robust performance across all datasets without systematic degradation on any metric. The 30-token budget aligns precisely with the slim skill’s structured format (2–3 likes themes + 1 style phrase), suggesting that the Cognitive Injection Boundary reflects a structural constraint rather than an arbitrary threshold.

Over-injection (≥ 150 tokens): H@1 declines monotonically while H@5 degrades more gradually, consistent with the *attention dilution* hypothesis: excessively long injections introduce population-generic terms that crowd out the immediate instruction and collaborative signals, with top-rank precision suffering first. This confirms the Injection Paradox: richer does not mean better.

Based on these results, **30 tokens** is adopted as the default injection length on the gpt-4o-mini backbone.

G Robustness across LLM Backbones

Our main experiments use gpt-4o-mini. To probe whether the policy skill layer remains effective on other backbones, we additionally run SAGER on two representative alternatives—gemini-3-flash (closed-source) and qwen3-30b (open-source)—across all four datasets. These runs are conducted on **1,000 randomly sampled users per dataset with a single seed**; the absolute numbers are therefore subject to LLM-API fluctuation and should be read as an indicative robustness check rather than a precise re-ranking of backbones.

As shown in Table 6, the policy skill layer improves the same-backbone MemRec baseline on all four datasets and all five metrics under both backbones, consistent with the model-agnostic trend observed in the main plug-in study (RQ2). We treat

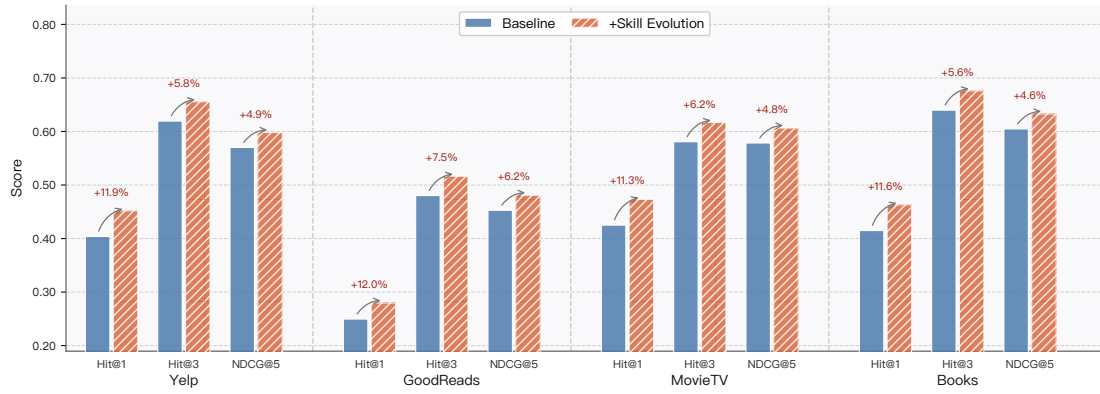


Figure 5: **iAgent** (static profile) with and without SAGER skill evolution across four datasets. Consistent improvements across all metrics demonstrate that skill evolution is effective even for static-profile agents.

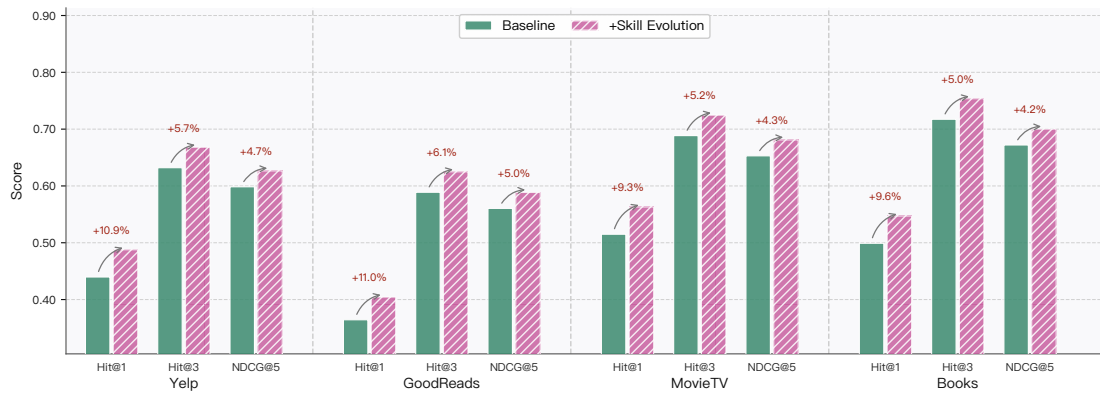


Figure 6: **i²Agent** (dynamic profile) with and without SAGER skill evolution. Skill evolution provides additive gains on top of dynamic profile updates, indicating that profile-level memory and policy-level skill are largely complementary.

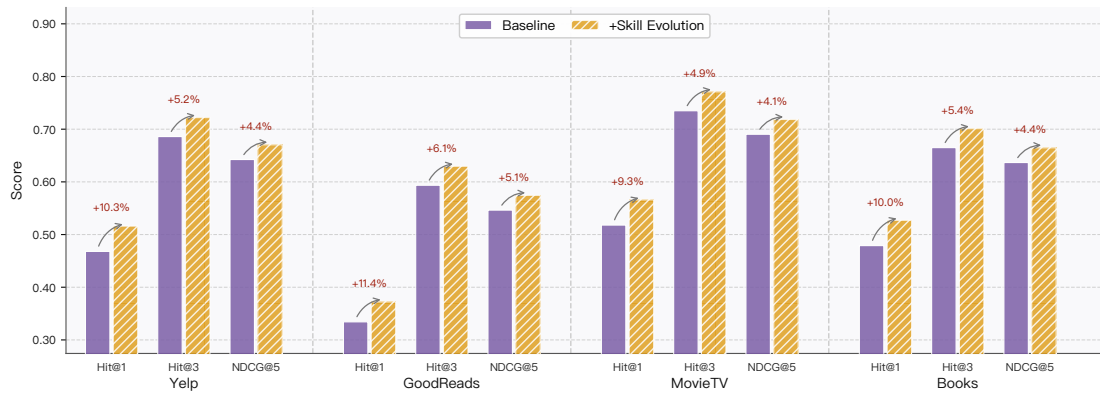


Figure 7: **Vanilla LLM** (no memory) with and without SAGER skill evolution. Skill evolution yields consistent improvements across all datasets even without any memory module, confirming that the skill itself encodes user-specific decision heuristics.

these as preliminary evidence: a full multi-seed, full-benchmark backbone sweep is left to future work.

H Larger Candidate Pool: N=20 Results on Yelp

The main-paper protocol follows the standard agentic-recommendation setup of $N=10$ candidates (1 positive + 9 random negatives), matching

Table 6: SAGER vs. MemRec across two additional backbones (1k randomly sampled users, single run, $N=10$). Numbers are indicative and subject to API-side variance; SAGER improves over the same-backbone MemRec baseline on every dataset and metric in our runs.

Dataset	Method	H@1	H@3	N@3	H@5	N@5
<i>Backbone: gemini-3-flash</i>						
Yelp	MemRec	0.429	0.623	0.541	0.739	0.588
	SAGER	0.461	0.651	0.571	0.761	0.615
Books	MemRec	0.572	0.665	0.586	0.838	0.709
	SAGER	0.583	0.678	0.597	0.848	0.719
Good.	MemRec	0.293	0.572	0.453	0.764	0.531
	SAGER	0.337	0.601	0.489	0.785	0.565
Mov.	MemRec	0.466	0.667	0.584	0.803	0.640
	SAGER	0.498	0.709	0.622	0.828	0.671
<i>Backbone: qwen3-30b</i>						
Yelp	MemRec	0.488	0.728	0.629	0.828	0.670
	SAGER	0.524	0.760	0.664	0.852	0.702
Books	MemRec	0.503	0.777	0.690	0.814	0.712
	SAGER	0.578	0.792	0.702	0.879	0.738
Good.	MemRec	0.313	0.606	0.481	0.760	0.544
	SAGER	0.360	0.637	0.520	0.781	0.579
Mov.	MemRec	0.473	0.659	0.583	0.787	0.634
	SAGER	0.505	0.701	0.620	0.811	0.665

AgentCF (Zhang et al., 2024b) and the reported protocol of MemRec (Chen et al., 2026). To verify that SAGER’s ranking advantage is not an artifact of a small candidate pool, we additionally evaluate under a harder $N=20$ setup (1 positive + 19 random negatives) on the full Yelp benchmark, comparing against all seven baselines on H@1 / H@5 / N@5 / H@10 / N@10 (Table 7). The candidate-set size describes the shortlist handed to the reranking stage; it does not change the underlying two-stage retrieval-then-rank pipeline.

Table 7: Full Yelp benchmark under the harder $N=20$ candidate pool (1 positive + 19 negatives). **Bold**=best overall. SAGER continues to lead every LLM-based baseline on H@1, confirming that the skill-augmented listwise ranking advantage persists beyond the 1+9 setup.

Model	H@1	H@5	N@5	H@10	N@10
LightGCN	0.2549	0.5437	0.4046	0.7481	0.4692
SASRec	0.1678	0.3993	0.2879	0.5590	0.3389
P5	0.0634	0.2492	0.1537	0.5051	0.2354
Vanilla LLM	0.0330	0.1352	0.0874	0.5006	0.2107
i ² Agent	0.3181	0.5746	0.4368	0.6631	0.4682
MemRec	0.3723	0.6260	0.5083	0.7616	0.5555
SAGER	0.4183	0.6563	0.5450	0.7854	0.5866

Under $N=20$, SAGER beats MemRec on all five metrics on Yelp, with an H@1 gain of +0.046

(0.3723 $\rightarrow$ 0.4183) of the same order as the $N=10$ gain in the main paper. Enlarging the candidate pool therefore does not dilute the policy-skill advantage: the slim skill still supplies the discriminative tie-breaking signal that promotes the true positive to the top rank even against a larger set of distractors.

I Efficiency and Deployment Considerations

We report the cost profile of the policy skill layer, combining *architectural properties* (which hold by construction) with *measured statistics* from our Yelp full-benchmark run (2,950 users, gpt-4o-mini).

Online inference adds no extra LLM call.

SAGER reuses the host agent’s existing retrieval and listwise-ranking calls; the only online-side change is that the ranking prompt is augmented with the slim skill. On the online serving path both SAGER and its MemRec backbone issue the same **two** LLM calls per interaction (memory-augmented synthesis + listwise ranking). Skill evolution (the contrastive differential CoT) is triggered *after* the user’s selection and runs asynchronously offline, so it is off the serving path. The skill layer therefore adds exactly **one offline LLM call per user per warmup round** and *zero* online LLM calls.

Token overhead is small and mostly offline.

Because SAGER reuses the backbone’s memory-synthesis and ranking prompts, its two online calls carry essentially the same token distribution as the backbone; on our gpt-4o-mini proxy the backbone’s per-user pipeline uses on the order of $\sim 6.7K$ tokens. The skill layer’s added tokens come from two sources, both small. *Online*: the ranking prompt gains only the ≤ 35 -token slim skill (typically 15–25), i.e., $\leq 2.6\%$ of the ranking call’s average input—a negligible online increment. *Offline*: the single extra contrastive differential CoT call consumes roughly ~ 700 tokens per user per warmup round (≈ 400 input / ≈ 300 output on the same proxy). The net marginal cost is thus about ~ 700 tokens per user per warmup round, raising the per-user total to on the order of $\sim 7.4K$ tokens, and it is paid once during offline warmup rather than at every online interaction. We report these as proxy-measured approximations rather than exact per-call constants, since token counts vary slightly with prompt content across users.

The slim skill is a bounded, cached injection.

By the Cognitive Injection Boundary, the injected slim skill is capped at a short structured form (≤ 35 tokens, typically 15–25, for gpt-4o-mini; Appendix F)—a negligible fraction ($\leq 2.6\%$) of a typical ranking prompt. It is extracted from the full skill by deterministic parsing (no LLM call, < 1 ms), cached per user, and re-extracted only when the full skill is updated rather than at every inference step.

Per-user storage is small and interpretable.

Measured over all 2,950 Yelp users after warmup, each user’s full policy skill occupies on average **2.07 KB** of plain text (range 1.74–2.31 KB; 6.25 MB in total for the whole benchmark), and its slim form is ≤ 140 B. Extrapolated to 100M users this is ≈ 200 GB in total—on the same order as the per-user memory that memory-based agents already maintain, and comfortably held on a single commodity KV-store shard. Because the skill is human-readable text rather than a latent vector, it adds interpretability and auditability at comparable storage cost.

Serving-path impact. Since the slim-skill addition to the ranking prompt is $\leq 2.6\%$ of the input and the skill update runs off the serving path, online serving cost is essentially identical to the MemRec backbone; the only added compute is the offline CoT call, which can be batched or scheduled off-peak.

J Experimental Setup and Implementation Details

J.1 Dataset Details

We utilize four datasets widely used in recommendation research, encompassing diverse domains such as e-commerce, social reading, entertainment, and local services. As mentioned in Section 3, we adopt the versions of these datasets augmented with natural language user instructions from **InstructRec** (Xu et al., 2025). Table 8 reports basic statistics; the original data sources and detailed domain descriptions follow.

Books. Derived from the **Amazon review dataset**² (Ni et al., 2019), this subset focuses on book recommendations. It is characterized by incredibly sparse interactions and a vast item space.

²https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/

Table 8: Statistics of the datasets used in experiments. $|U|$: users; $|I|$: items; $|E|$: interactions; $\bar{L}_u$: average sequence length per user; **Density** = $|E|/(|U| \cdot |I|)$.

Dataset	$ U $	$ I $	$ E $	$\bar{L}_u$	Density
Books	7.4K	120.9K	207.8K	28.2	2.33e-4
GoodReads	11.7K	57.4K	618.3K	52.7	9.19e-4
MovieTV	5.6K	29.0K	79.7K	14.1	4.87e-4
Yelp	3.0K	31.6K	63.1K	21.4	6.77e-4

User preferences in this domain are typically stable and highly content-driven, focusing on specific genres, authors, or themes.

Goodreads. Collected from the **Goodreads** social book cataloging website³ (Wan et al., 2019), this dataset is notably dense compared to others. It features strong community interactions and rich metadata about books, including series information. Users on Goodreads often exhibit series-aware reading behaviors and are influenced by social signals.

MovieTV. Also originating from the **Amazon review dataset** (Ni et al., 2019), this dataset covers movies and TV shows. The domain is marked by volatile user preferences often influenced by immediate context or trending content. While metadata like genre and cast are important, item recency frequently plays a critical role in user decision-making.

Yelp. Sourced from the **Yelp Dataset**⁴, this dataset consists of reviews for local businesses like restaurants and services. It is characterized by strong categorical constraints (e.g., cuisine type) and the critical importance of attributes like price range and location. User preferences here are often highly context-dependent.

J.2 Baseline Model Details

This appendix provides detailed descriptions of the baseline models used in our comparative evaluation. Following the categorization in the main text, we group these baselines based on their underlying memory paradigms into two major categories: traditional pre-LLM methods using latent embeddings, and memory-based approaches developed in the post-AgentRS era using semantic memory.

³<https://cseweb.ucsd.edu/~jmcauley/datasets/goodreads.html>

⁴<https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset>

J.2.1 Traditional Pre-LLM Methods (Latent Embeddings)

These models represent the conventional paradigm where historical information is encoded and preserved using dense latent vectors, without explicit semantic memory structures for reasoning agents.

- **LightGCN** (He et al., 2020): A state-of-the-art graph collaborative filtering model that simplifies the Graph Convolutional Network (GCN) design by removing feature transformation and nonlinear activation. It learns user and item embeddings by linearly propagating them on the user-item interaction graph, capturing high-order collaborative signals through structural connections.
- **SASRec** (Kang and McAuley, 2018): A leading sequential recommendation model based on the self-attention mechanism. It models the entire user sequence to capture long-term semantics and dynamic dependencies, using an attention mechanism to selectively focus on relevant items in the history for making predictions.
- **P5** (Geng et al., 2022): A unified framework that formulates various recommendation tasks as sequence-to-sequence language modeling problems. It utilizes a pre-trained T5 backbone and represents users and items as sequence tokens (IDs) within personalized prompts. While LLM-based, the original P5 relies on pre-trained knowledge related to these IDs and does not incorporate an evolving, descriptive memory component.

J.2.2 Memory-based Approaches (Post-AgentRS Era)

This category encompasses approaches developed in the era following AgentRS, utilizing LLMs with varying degrees of semantic memory capabilities.

(1) Models with No Explicit Memory. These models operate by directly processing raw interaction histories without maintaining a persistent, structured semantic memory store.

- **Vanilla LLM (Zero-Shot Prompting)** (Liu et al., 2023): This baseline represents the direct application of a powerful instruction-tuned LLM (e.g., GPT-4o-mini) via API calls. For each prediction, the user's entire sequence of historical interactions is converted into a natural language string and fed into the LLM as a static context prompt. The model performs zero-shot selection

from candidate items based solely on this provided raw history, serving as a baseline to measure the LLM's inherent capabilities independent of designed memory architectures.

(2) Static Memory Agents. These agents utilize descriptive semantic information about users and items, but this "memory" remains fixed as a static context during inference and does not evolve.

- **iAgent** (Xu et al., 2025): An LLM-based autonomous agent designed for recommendation. It employs a static profile for each user, constructed from their historical interactions and available descriptive data. This fixed profile is fed into the LLM as context to generate recommendations. The key characteristic is that its understanding of the user does not adapt over time after initial construction.

(3) Dynamic Memory Agents (Isolated Updates). These agents possess a dynamic memory mechanism, allowing them to reflect on interactions and update their understanding. However, these updates are isolated to the individual agent and do not propagate collaboratively.

- **i²Agent** (Xu et al., 2025): An extension of iAgent that introduces a "reflection" mechanism. After recommendations, the agent can reflect on user feedback to refine its internal state or strategy for future interactions. While dynamic, these reflections are confined to the individual agent's experience with a specific user.

- **AgentCF** (Zhang et al., 2024b): An agent-based collaborative filtering framework that simulates user-item interactions. Agents representing users and items can autonomously interact, learn from these interactions, and update their own preferences or characteristics. The memory update is dynamic but remains localized to the individual agents involved in the direct interaction.

- **RecBot** (Tang et al., 2025): A conversational recommender system that uses an LLM to engage with users. It maintains a dynamic dialogue history and can update its understanding of user preferences based on the ongoing conversation. This dynamic memory allows for multi-turn interactions but is limited to the context of the current user session.

(4) Dynamic Collaborative Memory.

- **MemRec** (Chen et al., 2026): A memory-augmented LLM agent for personalized recommendation that serves as our primary baseline. MemRec retrieves user context via semantic similarity and collaborative signals from a user-item interaction graph, and synthesizes structured memory facets for re-ranking. It operates with a universal, static reasoning policy. SAGER extends this backbone by introducing a per-user self-evolving policy skill that personalizes not only what the agent knows, but how it reasons.

J.3 Implementation Details

Model Deployment. All experiments are conducted using gpt-4o-mini accessed via a unified OpenAI-compatible API endpoint, with temperature set to 0.0 for reproducibility. We use the same API endpoint and model for all pipeline stages (retrieval, re-ranking, skill evolution).

Hyperparameters. The core memory and retrieval hyperparameters follow the original MemRec (Chen et al., 2026) configuration to ensure fair comparison. All retrieval, pruning, and re-ranking parameters are kept identical to MemRec defaults. For skill evolution, the warmup rounds τ are searched over $\{0, 1, 2, 3\}$. The slim skill injection length is searched over $\{10, 30, 80, 150, 200\}$ tokens; based on the averaged trend across multiple independent runs, 30 tokens achieves the best overall performance across datasets (individual runs may show a smaller gap between 30 tok and 80 tok, but 30 tokens is the only setting with no systematic degradation across any dataset or truncation point), and is therefore used as the default; confidence tiers have three levels (low/medium/high), with only medium and above eligible for injection. Our code is available at <https://anonymous.4open.science/r/EvoSkill-Sager-00AE/>.

K Domain-Aware Statistical Initialization: Extended Analysis

K.1 Cross-Domain Metadata Structure and Parser Design

The STATINIT component must handle fundamentally different metadata structures across domains. Yelp provides richly structured metadata (categories, price tier, ambience tags, city, meal type, star ratings), while Books and MovieTV datasets contain only free-text title and description

fields formatted as Python lists. Without domain-specific parsing, all regex patterns designed for Yelp return empty matches on Books/MovieTV, producing identical “empty template” skills for every user—completely defeating the purpose of statistical initialization.

We diagnose this failure empirically: on Books, the initial domain-agnostic implementation produced 100% identical skill files (1.04KB Yelp-formatted templates), with Core Preferences sections populated with Yelp-specific terms (“Venues”, “Ambience”, “Location proximity”) irrelevant to book recommendation.

K.2 Domain-Adaptive Parser Architecture

We introduce a **domain-adaptive multi-parser architecture** that selects the appropriate metadata extraction pathway based on dataset type:

Table 9: Domain-aware StatInit signal extraction. Each domain uses a specialized parser with dataset-appropriate dimensions.

Domain	Dimension 1	Dimension 2	Dimension 3
Yelp (baseline)	categories (regex)	price tier + city	ambience + meal type
Books/GoodReads	genres (15-class keyword)	favorite authors	reading moods (7-class)
MovieTV	genres (17-class + qualifier)	directors	viewing moods (8-class)

K.3 Concrete Skill Examples: Books and MovieTV

L Qualitative Case Study: Skill Evolution in Action

L.1 Case A: Yelp — Emerging Preference Discovery (User 0)

User 0 has extensive interactions in Las Vegas, primarily in the Restaurants category. **StatInit** ($\tau = 0$, zero LLM cost) constructs the initial skill from interaction frequency statistics. The initial skill and subsequent evolution traces are shown in Figure 10.

L.2 Case B: Books — Domain-Adaptive StatInit (User 0)

The Books dataset has only free-text title and description fields. The domain-adaptive parser extracts genre/mood signals as shown in Figure 11.

M Prompt Templates and Global Skill Example

This section gathers all the verbatim prompt templates and the cold-start global skill example. Because several of these contents are long and formatted, we render them as full-width boxes spanning

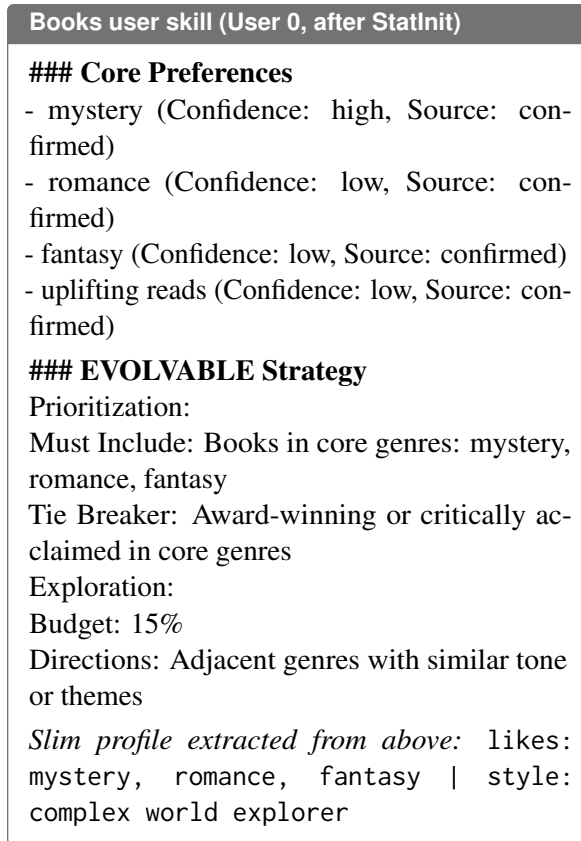


Figure 8: Books user skill example (User 0, after STATINIT).

both columns at the end of the appendix to avoid layout overflow.

M.1 Slim Profile Extraction Prompt

The slim profile extractor compresses the full skill repository to an injection representation bounded by the Cognitive Injection Boundary (Appendix F; e.g., a ~30-token budget on gpt-4o-mini), using only signals with confidence medium or above. The prompt is shown in Figure 12 (page-spanning, end of appendix).

M.2 Listwise Ranking Prompt

The listwise ranking prompt instructs the LLM to produce a complete ordering rather than independent scores; the core structure appears in Figure 13. This formulation eliminates the score-tie problem: every item receives a unique rank position, and the LLM leverages its reasoning strength rather than struggling with absolute calibration.

M.3 Contrastive Differential CoT Evolution Prompt

The Contrastive Differential CoT evolution follows a three-phase *refine-not-replace* protocol. Rather

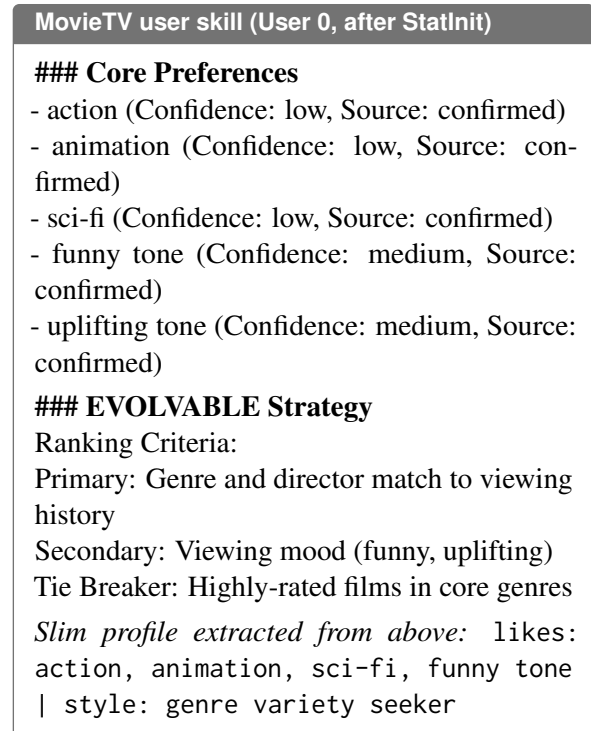


Figure 9: MovieTV user skill example (User 0, after STATINIT).

than regenerating the full skill from a single interaction (which destroys the statistical prior), it outputs only a structured differential update: `new_preferences`, `reinforced`, and `weakened`. The full template appears in Figure 14.

M.4 Memory Retrieval Prompt

The memory retrieval and update components of SAGER are implemented on top of an existing memory-based recommendation agent (MemRec (Chen et al., 2026) in our experiments). SAGER is designed to be agent-agnostic: the policy skill can be plugged into any memory recommendation agent that exposes per-user memory and collaborative neighbor context. The prompt below is inherited from the underlying agent and is shown for completeness in Figure 15; it synthesizes per-user memory and collaborative neighbor signals into structured preference facets that are subsequently used in the listwise ranking stage.

M.5 Global Skill Example: Yelp Dataset

The global skill $\mathcal{S}_G$ provides a population-level prior for users with no interaction history (cold start, Level 1 of the dual-level initialization). It is synthesized once per dataset by a meta-LLM from domain statistics, and is inherited as the starting

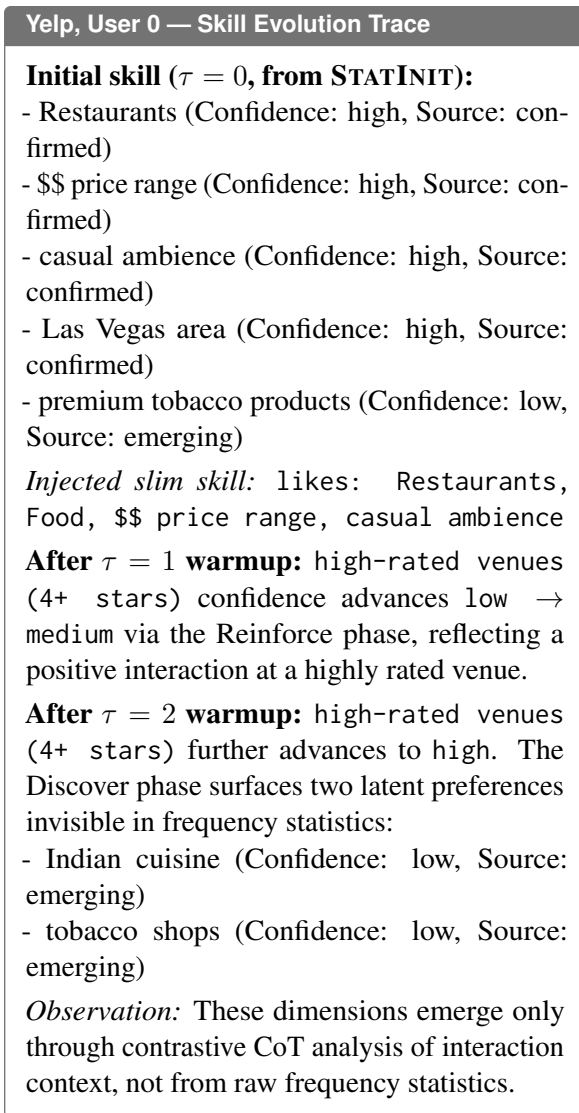


Figure 10: Yelp, User 0: skill evolution trace across warmup rounds ($\tau = 0, 1, 2$).

point before any per-user CoT evolution. The full global skill for Yelp is shown in Figure 16.

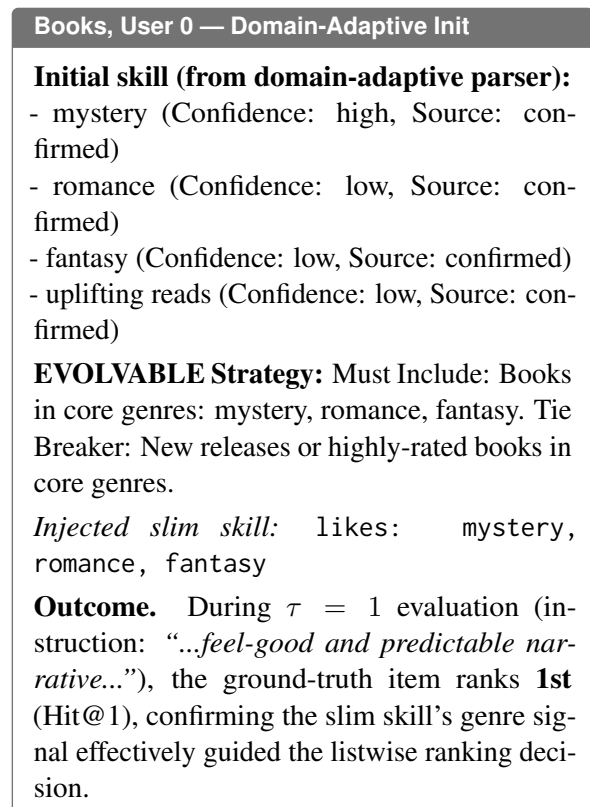


Figure 11: Books, User 0: domain-adaptive STATINIT skill and evaluation outcome.

Slim Profile Extraction Prompt (Core)

Purpose: Convert verbose user memory into a minimal structured profile (within the Cognitive Injection Boundary of the chosen backbone, e.g., ~30 tokens for gpt-4o-mini) for tie-breaking in ranking.

Extract: (1) **likes:** top 2–3 preference themes most strongly distinguishing this user; (2) **style:** one phrase describing their overall taste pattern.

Only include patterns with Confidence: medium or high. Output must respect the configured token budget.

Output format: likes: X, Y, Z | style: [decision style phrase]

Figure 12: Slim Profile Extraction Prompt used to compress the full skill into a ~30-token injection representation.

Listwise Ranking Prompt (Core Structure)

You are an expert personalized recommendation system.

User's Current Request: {instruction}

User Preferences (Memory): {formatted_facets}

Per-User Preference Skill: {slim_skill}

(Use as tie-breaker only when candidates are equally relevant.)

Candidate Items: {formatted_candidates}

Task: Rank ALL candidate items from most to least relevant. Output a JSON array of item IDs in ranked order, with a brief rationale for each.

Output format: {"ranking": [{"item_id": ..., "rationale": "..."}, ...]}

Figure 13: Listwise Ranking Prompt used for skill-augmented candidate re-ranking (Score-Tie Bottleneck bypass).

Contrastive Differential CoT Update Template (Abbreviated)

Purpose: Apply a differential update to an existing user skill based on a new interaction. Output ONLY the changes (additions, reinforcements, and weakening of existing preferences).

Key Principle: The existing skill was built from long-term history statistics. A single new interaction should REFINES it, not REPLACE it. Preserve high-confidence long-term preferences.

Phase 1 (Reinforce): Does this interaction REINFORCE any existing preferences? Which existing Core Preferences are supported by this choice?

Phase 2 (Discover): Does this interaction REVEAL new preferences not in the existing skill? What new signals emerge? Are they exploratory or consistent with existing patterns?

Phase 3 (Weaken): Does this interaction WEAKEN any existing preferences? Only if the positive choice clearly contradicts an existing preference. Be conservative: a single interaction rarely invalidates a long-term pattern.

Output: {"analysis": "2-3 sentences reasoning", "incremental_update": {"new_preferences": [{"attribute": ..., "reason": ...}], "reinforced": [{"attribute": ..., "evidence": ...}], "weakened": [{"attribute": ..., "reason": ...}]}}

Constraints: (1) Conservative—prefer reinforcing over weakening; (2) Specific—concrete attributes only; (3) No Must-Avoid rules from a single non-selection.

Figure 14: Contrastive Differential CoT Skill Evolution Prompt (abbreviated). Full template includes few-shot examples.

Memory Retrieval Prompt (Core Structure)

You are an intelligent memory retrieval system for personalized recommendation. Your task is to analyze the user's personal memory and collaborative memories from their neighbors to extract preference facets.

Target User: User {user_id}

User's Personal Memory: {user_mem_bullets}

Collaborative Neighbor Memories: The following neighboring users and items provide collaborative signals for understanding this user's preferences: {neighbor_table_json}

Task: Identify {n_facets} distinct preference facets. For each facet, provide: (1) a concise natural language description; (2) a confidence score in [0, 1]; (3) a list of supporting neighbor IDs.

Output format: {"facets": [{"facet": ..., "confidence": ..., "supporting_neighbors": [...]], ...}, {"support_edges": [{"from": ..., "to": ..., "w": ...}], ...}]}

Figure 15: Memory Retrieval Prompt.

Global Skill Example: Yelp Dataset (full)

Persona Clusters (interaction-based): (i) *Light Reviewers* (<15 interactions, ~30%): limited signal; rely on popularity/location. (ii) *Moderate Reviewers* (15–50 interactions, ~50%): emerging category preferences. (iii) *Heavy Reviewers* (>50 interactions, ~20%): stable taste profiles, neighborhood regulars.

Interest Clusters: Cuisine-Loyal | Price-Sensitive | Neighborhood-Bound | Experience-Seeker.

Strategy 1 (Guided Discovery, Light Reviewers): Prioritize high-rating local businesses; restrict to 2–3 dominant categories; apply geo-proximity filter; avoid niche or low-review establishments.

Strategy 2 (Preference Reinforcement, Moderate Reviewers): 70% from confirmed cuisine/category facets; 30% from adjacent categories; maintain price-tier consistency.

Strategy 3 (Exploratory, Heavy Reviewers, high category spread): 50% established facets; 50% novel/adjacent; 5–7 categories; cross-district exploration enabled.

Strategy 4 (Comfort Zone, Heavy Reviewers, low category spread): 85% from top-3 confirmed category/price facets; 15% closely related variants.

Dynamic Adjustment Rules: (a) Recency weighting: last 3 interactions 1.8×; interactions 4–15 1.0×; >15 0.6×. (b) If >60% of neighborhood-category businesses visited: expand geo-radius by one zone. (c) Category skipped ≥3 consecutive recommendations: suppress for 10 cycles.

Memory Integration: Prioritize item-based CF over user-based CF given the relatively sparse interaction graph. Neighbor agreement >65%: boost facet confidence by +20%. Leverage Yelp category taxonomy to generalize sparse interactions.

Default priority order: accuracy > locality > diversity > novelty.

Figure 16: Global skill S_G for the Yelp dataset, serving as the cold-start population-level prior.