Generative *vs.* Agentic Recommendation

Two LLM Paradigms for Recommender Systems, and the MemRec Framework

Weixin Chen (陈伟新)

Applied Scientist, Kuaishou Technology 快手科技
LLM-based recommendation, via the Kuaishou K-Star Talent Program (快 Star-X)

Invited Talk, Shenzhen University 深圳大学
September 15, 2026

深圳大学
SHENZHEN UNIVERSITY

生成式推荐 *vs.* 智能体推荐:
大模型时代推荐系统的两条路线与 MemRec 实践

KUAISHOU

About the speaker

- **Now**: Applied Scientist, Kuaishou Technology — LLM-based recommendation, via the Kuaishou K-Star Talent Program (快 Star-X).
- **Ph.D. 2026**: Hong Kong Baptist University (Prof. Li Chen).
- **2025–26**: visiting, Rutgers WISE Lab (Prof. Yongfeng Zhang).
- **B.Eng. 2020**: **Shenzhen University** (Prof. Weike Pan).

Agentic and generative recommendation; fairness and diversity; user modelling.

ACL, EMNLP, WWW, KDD, RecSys, TOIS, TORS.

Why this talk

First-hand experience on **both** sides:

- **academia** — agentic recommenders and their memory;
- **industry** — generative recommenders in production.

Outline

Part I — the map (∼22 min)

LLMs entered recommendation through **two different doors**. Which one wins where?

Part II — one deep dive (∼18 min)

MemRec (ACL 2026): agent memory is missing the most valuable signal in recommendation — **other people**.

Part 1

Background: Two Paradigms

Three eras of recommender systems

Era	A preference is...	Representative
Heuristic CF	a sparse row	User/Item-KNN
Representation learning	a dense vector	MF, FM, DNN, SASRec
Language / LLM	**tokens and text**	**Generative:** P5, TIGER, HSTU, OneRec **Agentic:** RecMind, MACRec, AgentCF, **MemRec**

Each era changed **what a preference is made of**.
The LLM era is the first in which that representation is also something we can **reason about in words**.

Why bring LLMs in? Four cracks in the classical cascade

The cascade

Retrieval $10^9 \to 10^4$

↓

Pre-ranking $10^4 \to 10^3$

↓

Ranking $10^3 \to 10^2$

↓

Re-ranking $10^2 \to 10$

Four models, four objectives.

1. **Objective fragmentation** — the best local decision is not the best global one.
2. **Cold start** — a new item's ID points to nothing.
3. **No world knowledge** — IDs carry no semantics.
4. **Scaling stalls** — more parameters stopped buying accuracy.

LLMs attack all four at once: one model, semantic item representations, pretrained knowledge, an unfinished scaling curve.

The fork in the road

"What should this person see next?"

Generative Rec

The LLM ***becomes*** the recommender.

Items become tokens; the model *generates* the next item the way a language model generates the next word.

P5 · TIGER · HSTU · OneRec

Training and serving.

Agentic Rec

The LLM ***harnesses*** the recommenders you already have.

It plans, calls retrieval/ranking tools, keeps a memory of the user, reflects on mistakes — and decides.

RecMind · MACRec · AgentCF · **MemRec**

Reasoning and interaction.

The LLM4Rec landscape

Paradigm A — Generative Rec

Recommendation as **autoregressive token generation**, trained end-to-end.

- Unified language paradigm — **P5**, M6-Rec
- Semantic ID + generative retrieval — **TIGER**, LC-Rec
- Native sequences, at scale — **HSTU**
- Industrial end-to-end + RL — **OneRec**

Paradigm B — Agentic Rec

LLM as a **decision brain**: planning, tools, memory, reflection.

- Tool-augmented — **InteRecAgent**, RecMind
- Multi-agent roles — **MACRec**
- User simulation — **Agent4Rec**, RecAgent
- Agentic CF & memory — **AgentCF**, **MemRec**

Part 2 walks down the left column, Part 3 the right, Part 4 puts them side by side.

Part 2

Generative Recommendation

1. Background: Two Paradigms
2. Generative Recommendation
3. Agentic Recommendation
4. Comparing the Two Paradigms
5. MemRec: Collaborative Memory
6. Outlook

From scoring to generation

The old formulation — scoring.

$$\hat{y}_{ui} = f(\mathbf{e}_u, \mathbf{e}_i) \qquad \text{for every } i \in \mathcal{I}$$

Cost grows with $|\mathcal{I}|$. Hence the cascade.

The new formulation — generation.

$$p(i_{t+1} \mid i_{1:t}) = \prod_{k=1}^{L} p\big(c_k \mid c_{<k},\, i_{1:t}\big)$$

Each item is a string of L tokens. Beam search gives top-K without scoring the catalogue.

Everything that follows comes from one move: **stop treating an item as an atomic ID, start treating it as a sequence of tokens**.

The enabler: Semantic IDs

item content $\longrightarrow$ content encoder (Sentence-T5) $\longrightarrow$ residual quantiser $\longrightarrow$ $\langle c_1, c_2, c_3 \rangle$

RQ-VAE (TIGER) or RQ-Kmeans (OneRec): quantise, then quantise the residual, ...

	Atomic ID	**Raw text**	**Semantic ID**
Vocabulary size	$\lvert\mathcal{I}\rvert$ (billions)	sub-words (long)	$K \times L$ (thousands)
Carries semantics	no	yes	yes
Cold-start capable	no	yes	yes
Decoding length	1 step	dozens of tokens	L steps (3–4)
Collaborative signal	yes (learned)	no	**needs alignment**

The last row is the open wound of this line of work — we return to it at the end.

P5 — “everything is language”

Geng et al., RecSys 2022 ▪ Rutgers

- Rewrite *every* rec task as text-to-text; train one T5 with personalised prompts.
- Five task families collapse into **one model, one loss**.
- Tasks that were separate pipelines become **different prompts**.
- **Zero-shot transfer** to unseen prompts.

Prompt-as-task

```
% sequential recommendation
I find the purchase history list of user_15466:
item_7391, item_1028, … What is the next item to
recommend?

% explanation generation
Help user_15466 generate a 3-star explanation about
item_7391

% rating prediction
What star rating will user_15466 give item_7391?
```

What it could not do

Items were **atomic IDs spelled as text** (“item_7391”), so the model had to memorise every one. No generalisation to new items.

TIGER — generative retrieval with Semantic IDs

Rajput et al., NeurIPS 2023 ▪ Google

- Quantize each item's content embedding into a **Semantic ID** with RQ-VAE.
- Train seq2seq to *generate* the next item's Semantic ID.
- **Generative retrieval**: beam search returns top-K — no ANN index, no dot products.
- **Cold start works**: a new item gets an ID from content alone.

Decoding

Input $\langle 12, 240, 7\rangle\ \langle 3, 88, 19\rangle\ \langle 12, 17, 5\rangle$

Decode $\hat{c}_1 = 12,\ \hat{c}_2 = 205,\ \hat{c}_3 = 41$

Output $\langle 12, 205, 41\rangle \Rightarrow$ a concrete item

Semantic IDs are the most transferable idea in this paradigm — almost every system after TIGER, including the one in production at Kuaishou, uses a version of them.

HSTU — the scaling law arrives

Zhai et al., ICML 2024 ▪ Meta

- Reformulate **ranking and retrieval** as sequential transduction over one unified event stream.
- A native attention unit for long, **non-stationary** vocabularies — what rec has and language does not.
- $5.3\times$–$15.2\times$ faster than FlashAttention-2 at 8192 length; $+65.8\%$ NDCG over baselines.
- Deployed at Meta at **1.5T parameters**; $+12.4\%$ in online A/B tests.

Why it mattered

For a decade, rec models stopped improving when made bigger. HSTU showed a clean **power law** up to GPT-3 scale.

The question shifted from *"what feature do we add?"* to *"how much compute can we afford?"*

The catch

Scaling laws are useful only if you have Meta's data and GPUs.

OneRec — the cascade actually gets replaced

Kuaishou, 2025 ▪ two reports, deployed in the main scenario

Three ingredients

- **Session-wise generation** — generate a whole list, so ordering and diversity are modelled, not patched.
- **Balanced Semantic IDs** (RQ-Kmeans) — avoids the codebook collapse RQ-VAE hits at industrial scale.
- **Preference alignment** — a reward model over watch time and interaction drives iterative DPO.

Reported results

OneRec (arXiv 2502.18965) — 1% of main-page traffic: $+1.68\%$ total watch time, $+6.56\%$ average view duration.

Technical Report (arXiv 2506.13695) — scaled to $\sim 25\%$ of QPS:

- App Stay Time $+0.54\%$ / $+1.24\%$ (Kuaishou / Lite)
- MFU $4.6\% \rightarrow 23.7\%$ train, $11.2\% \rightarrow 28.8\%$ infer
- OPEX $= \mathbf{10.6}\%$ of the cascade

The efficiency number is the real story: one dense model turns recommendation into a workload GPUs are good at.

Generative Rec: four papers, one trajectory

	Item repr.	Core idea	Why it mattered
P5 '22	atomic ID as text	all tasks as text-to-text with prompts	the "everything is language" framing
TIGER '23	**Semantic ID**	generate the next item's token sequence	generalisation and cold start
HSTU '24	unified event sequence	rec as sequential transduction	the first credible **scaling law**
OneRec '25	semantic tokens	one model replaces the cascade, aligned by RL	**deployed**, order-of-magnitude efficiency

P5 made it a language problem → TIGER made it *generalise* → HSTU made it *scale* → OneRec made it *ship*.

What generative rec cannot do

The price of the main pipeline

- **Explainability near zero.** The output is a token sequence; there is no rationale to show a user or a regulator.
- **No conversation.** It cannot take "shorter than *Dune*, for a weekend trip" as an instruction.
- **Decoding overhead.** Autoregressive beam search over a large vocabulary is 10–100$\times$ slower than ANN lookup; production SLA demands dedicated accelerators.
- **Tokenizer lock-in.** Change the Semantic ID scheme and everything downstream retrains.

Generative Rec wins the **main pipeline**. It has nothing to say about the user who wants to *talk* to the system.

Part 3

Agentic Recommendation

The LLM as a decision brain

The agent toolkit

- **Planning** — decompose a vague intent
- **Tools** — retrieval, ranking, search, SQL
- **Memory** — who this user is
- **Reflection** — check, correct, iterate

Memory is what Part 5 is about.

Traditional RecSys — stateless

You click a sci-fi book → your embedding shifts → another sci-fi book silently appears.

Agentic RecSys — stateful

You say *"something like Dune, but shorter, for a weekend trip."* The agent reads your textual memory, plans, calls a retrieval tool, checks the constraint, and answers **with a reason**.

Nothing is trained end-to-end. The LLM **harnesses** existing models as its tools.

Four families of agentic recommenders

1. Tool-augmented orchestration

InteRecAgent [TOIS'25], **RecMind** [NAACL'24]

The LLM never ranks; it drives your existing models as tools. A **candidate bus** passes items between calls.

2. Multi-agent roles

MACRec [SIGIR'24]

Manager, user analyst, item analyst, reflector, searcher — and let them argue.

3. User simulation

Agent4Rec [SIGIR'24], **RecAgent** [RUC]

1,000 profiles become 1,000 LLM agents with taste, memory and an exit rule. Offline evaluation without an A/B test.

4. Agentic CF & memory

AgentCF [WWW'24], **MemRec** [ACL'26]

Users *and* items are agents carrying natural-language memory. A wrong prediction makes both sides reflect and rewrite.

What agents are good at — and what they are not

Strengths

- **Explainability is native**
- **Complex, long-tail intent**
- **Cold start** via world knowledge
- **Controllable** — new tool, no retraining
- **Simulation** for evaluation

Limitations

- **Latency and cost** — seconds and cents
- **Not viable** at billion-scale real time
- **Brittle** — prompt and model dependent
- **Weak on pure accuracy** offline
- **Hallucination**

Agentic Rec is not competing for the feed. It competes for the **conversation**, the **hard query** and the **simulator**.

Part 4

Comparing the Two Paradigms

Head to head

	Generative Rec	Agentic Rec
Essence	rec *is* sequence generation	the LLM plans, calls tools, remembers
Training	large-scale pretrain / finetune	mostly prompting and orchestration
Item repr.	Semantic IDs	text; delegates to existing retrieval
Intelligence lives in	the **weights**	the **control flow**
Strengths	scalable, low-latency, strong metrics	explainable, interactive, cold-start
Limits	opaque, costly to build, mute	slow, fragile, not for the feed
Home turf	the industrial main pipeline	conversation, hard queries, simulation

They are not two answers to the same question. They are answers to **two different questions** that happen to share a name.

Not replacement — layering

Interaction layer — Agentic — seconds · thousands of requests

conversational search · multi-constraint intent · explanation · "why am I seeing this?"

↕

Main pipeline — Generative — milliseconds · billions of requests

end-to-end retrieval + ranking · Semantic IDs · session generation · RL alignment

↕

Shared substrate — offline

item tokeniser · user/item memory store · reward model · agent-driven simulation

Generative owns throughput. **Agentic** owns the conversation. They meet at the **tokeniser** and the **reward model**.

A personal thesis: different trajectories, different ceilings

Generative Rec — evolutionary

- The natural continuation of seq rec.
  text-based seq rec → Semantic ID → generation
- **TIGER** plays the role **SASRec** played: a clean baseline that everyone can build on.
- Plugs into the existing pipeline — add a module or replace the retriever.
- Lower risk; ceiling is readable from the scaling law.

Agentic Rec — potentially revolutionary

- Breaks the assumption that the technology base is *fixed*. Mature product forms (Kuaishou feed, Amazon homepage) were designed around fixed tech.
- **No unified paradigm yet.** E-commerce agents and short-video agents will likely look very different.
- The right question: for which features does the **extra agent cost pay off**?
- Higher risk, but a higher — and still uncharted — ceiling.

The missing TIGER moment for agentic rec is simultaneously the field's biggest open problem **and** its biggest opportunity.

One critical piece of that missing foundation — **memory** — is the rest of this talk.

Part 5

MemRec: Collaborative Memory

What should an agent remember?

An agent without memory is a **stateless function**.

Everything that makes agentic recommendation interesting — personalisation, adaptation, the sense that the system *knows you* — lives in the memory module.

The field's answer has changed three times.
I will argue it needs to change a fourth.

MemRec

Collaborative Memory-Augmented Agentic Recommender System

Weixin Chen, Yuhan Zhao, Jingyuan Huang, Zihe Ye, Clark Mingxuan Ju, Tong Zhao, Neil Shah, Li Chen, Yongfeng Zhang

ACL 2026, Main Conference



The evolution of memory

	How it works	**Example**
1. No explicit memory	raw history in the prompt; nothing persists	Vanilla LLM, LLMRank
2. Static memory	a profile written once, retrieved as context; frozen	iAgent
3. Dynamic memory	the agent reflects and rewrites its own profile	AgentCF, i^2Agent, RecBot

Each step made the memory **more current**. None of them made it **less lonely**: every one updates a single user's profile in a silo, reading only that user's own history.

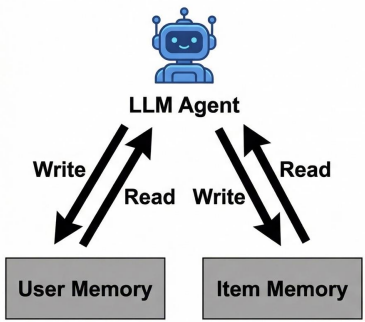
The gap: dynamic, but isolated

Recommendation's founding insight is that **you are informative about me**. Collaborative filtering has run on that for thirty years.

Yet a state-of-the-art agentic recommender builds its memory from **one user's history alone**.

What gets lost

- **Neighbour evidence** — similar users already explored what this user has not.
- **Community drift** — a rising genre stays invisible.
- **Sparse users never recover** — five interactions give self-reflection nothing to reflect on.



Today's paradigm: every user is an island.

"Then just put the neighbours in the prompt"

1. Cognitive overload

Raw neighbourhoods **degrade the reasoning they were meant to help**.

- hundreds of neighbours $\times$ their histories $=$ tens of thousands of noisy tokens;
- classic *lost-in-the-middle*;
- past a point, more context lowers accuracy.

2. Prohibitive cost

Keeping the memory current means updating **everyone the interaction touches**.

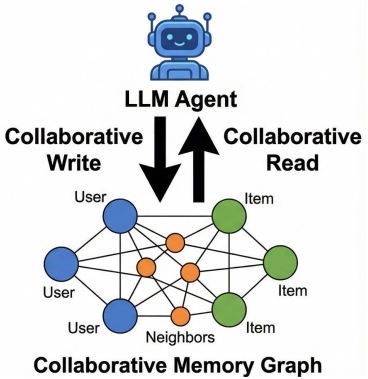
- one click should update user, item *and* neighbourhood;
- synchronously that is $O(|\mathcal{N}|)$ LLM calls **on the critical path**;
- at industrial traffic, impossible.

These two bottlenecks are why nobody had done it.
One is solved by **curation**, the other by **asynchrony**.

MemRec: from isolated memory to a collaborative memory graph

1. **Collaborative memory graph** — users *and* items are nodes carrying semantic memory; interactions are edges.
2. **Architectural decoupling** — a cheap model *manages memory*, a strong model *reasons*.
3. **Asynchronous propagation** — graph maintenance leaves the critical path; the online request stays $O(1)$.

The agent now consults a community library instead of re-reading its own diary.



Memory that reaches across users and items.

Architectural decoupling: the Librarian and the Brain

The Librarian ($\mathrm{LM}_{\mathrm{Mem}}$)

Role: memory manager.

Model: small and cheap — gpt-4o-mini, or a local 7B.

Does: zero-shot curation, noise filtering, background graph updates.

Runs constantly, off the critical path.

The Brain ($\mathrm{LLM}_{\mathrm{Rec}}$)

Role: reasoning core.

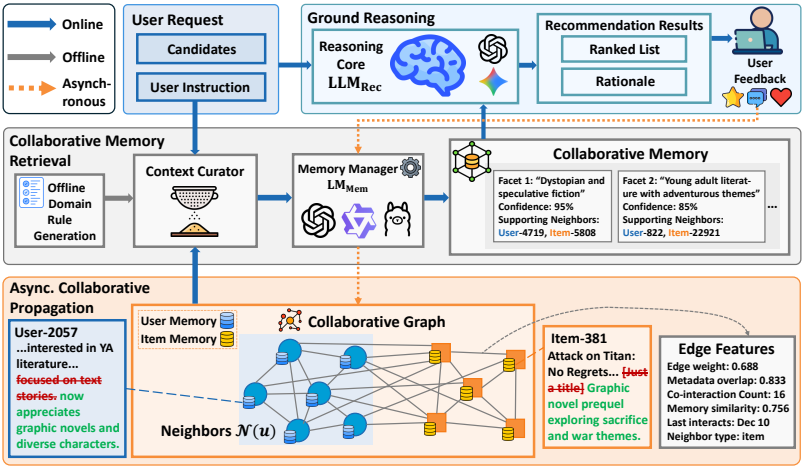
Model: high-capacity — gpt-4o class.

Does: consumes *clean, synthesised* collaborative context and produces the ranking plus a rationale.

Runs once per request, on clean input.

Cognitive overload is not a context-length problem.
It is a **division-of-labour** problem.

The three-stage pipeline



Stage-R: retrieve & synthesise → Stage-ReRank: grounded reasoning → Stage-W: async propagation

Full pipeline walkthrough: `memrec.weixinchen.com`

Experimental setup

Datasets — chosen to disagree

- **Books** — sparse, content-driven
- **Goodreads** — dense, content-driven
- **MovieTV** — volatile, recency-sensitive
- **Yelp** — spatial and categorical constraints

Baselines — one per rung

- Classical / deep: LightGCN, SASRec, P5
- No memory: Vanilla LLM
- Static memory: iAgent
- Dynamic memory: AgentCF, i^2Agent, RecBot

gpt-4o-mini for *both* roles, $k = 16$ neighbours, $N_f = 7$ facets, candidate list $N = 10$.

Two evaluation protocols — keep them apart

Main results run on the **full test sets**. Every **analysis** afterwards (efficiency, overload, ablation, sensitivity) uses a **random 1,000-user subset**, following AgentCF. Absolute numbers therefore differ between the table and the figures.

Main results

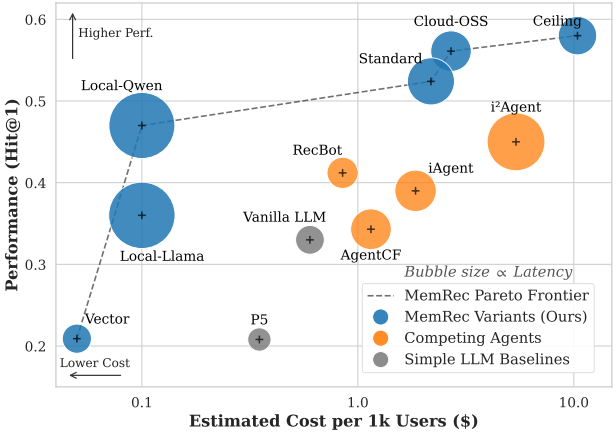
Full test sets ▪ abridged; SASRec, P5, RecBot and the H@3 / N@5 columns are in the paper

	Books (sparse)			**Goodreads** (dense)		
	H@1	N@3	H@5	H@1	N@3	H@5
LightGCN	0.1753	0.2596	0.5703	0.2499	0.4432	0.7903
Vanilla LLM no mem.	0.3138	0.4533	0.7270	0.2864	0.3948	0.7390
iAgent static	0.3925	0.4858	0.6905	0.2617	0.3954	0.6591
AgentCF dynamic	0.3457	0.4960	0.7403	0.2951	0.4654	0.7726
i^2Agent dynamic	0.4453	0.5649	0.7708	0.3099	0.4825	0.7675
MemRec	**0.5117**	**0.6152**	**0.8007**	**0.3997**	**0.5540**	**0.8052**
Improv.	+14.91%	+8.90%	+3.88%	+28.98%	+14.82%	+1.89%

- Consistent gains on **all four** datasets over the strongest isolated-memory agent (MovieTV $+19.75\%$, Yelp $+15.77\%$ H@1; all significant at $p < 0.05$).
- Gains concentrate in **Hit@1**: collaborative memory is not finding *more* plausible items, it is **picking the right one**.

Does the decoupling pay for itself?

Analysis studies use a random 1,000-user subset, so absolute numbers run slightly above the full-set table

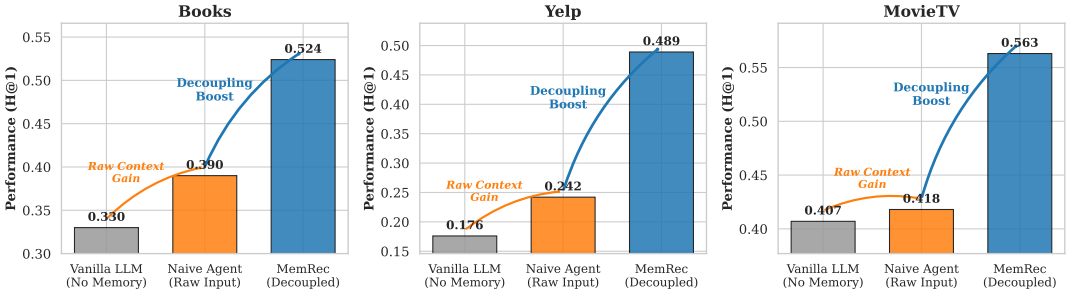


A collaborative agent that called a frontier model for every memory update would be strictly worse than a cheap isolated one: better numbers, unusable economics.

A **small** model for memory and **asynchronous** graph maintenance put MemRec on a new **cost–performance Pareto frontier** — more accurate *per dollar*.

Cognitive overload, measured

Random 1,000-user subset ▪ $+34\%$ relative H@1 over the naive collaborative agent on Books



Naive collaborative agent

Fed raw neighbourhoods, it **plateaus early**: extra context stops helping and starts hurting.

MemRec

Curation and decoupling **break the plateau**: more evidence keeps turning into accuracy.

A case study, end to end

User 2057 — a sparse reader with an unusual request

History: a handful of YA fantasy titles. **Instruction:** *"visually immersive... like a **graphic novel**."*

Stage-R — the Librarian consults the community

Curates **16 similar readers** and synthesises their memories: *"this community is strongly engaged with **dystopian fiction** and **emotionally layered storytelling**."*

None of this is in User 2057's own history.

Stage-ReRank — the Brain decides

Recommendation: *Attack on Titan: No Regrets*

Rationale: *"This is a **graphic novel**, matching your format request. Based on similar readers, it also fits the community's interest in **dystopian settings** and **complex storytelling**."*

The constraint comes from the user; the taste prior comes from the community.

What MemRec says about agentic recommendation

1. **Collaborative signal does not become obsolete when the agent gets smart.** World knowledge is not a substitute for what *this* population did last week.
2. **Architecture is destiny.** Separating memory management from reasoning is what makes collaborative context usable at all.
3. **Scalability has to be designed in.** Asynchronous background maintenance is what gives the idea a deployable cost profile.

External validation

In Meta's **MARS** paper, MemRec is singled out as *"the strongest competitor"* among LLM-based methods.

Part 6

Outlook

Summary

1. **Generative Rec** turns recommendation into token generation and wins on throughput, latency and metrics. **Agentic Rec** turns it into planning, tools and memory, and wins on explanation, interaction and cold start.
2. Their trade-off profiles are nearly **inverted**, so the field is settling into **layering**, not replacement.
3. Inside agentic rec, **memory** is where personalisation lives — but the field kept making it *fresher*, when the real gap was making it *collaborative*.
4. **MemRec** adds the missing axis — a collaborative memory graph, a Librarian/Brain split, asynchronous propagation. **Up to** $+28.9\%$ **Hit@1**.
5. Semantic IDs, scaling and preference alignment are the **shared substrate** of both paradigms.

Thank you

Q & A

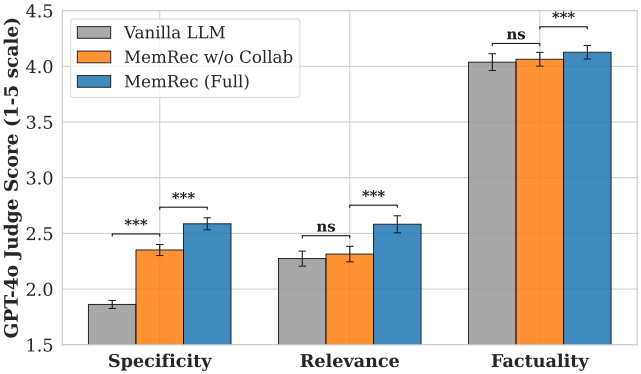
Paper	aclanthology.org/2026.acl-long.2061.pdf
Code	github.com/rutgerswiselab/memrec
Project	memrec.weixinchen.com
Homepage	weixinchen.com

Backup slides

for Q & A

Backup — is the *explanation* better, or only the ranking?

GPT-4o as judge, Books, random 1,000-user subset

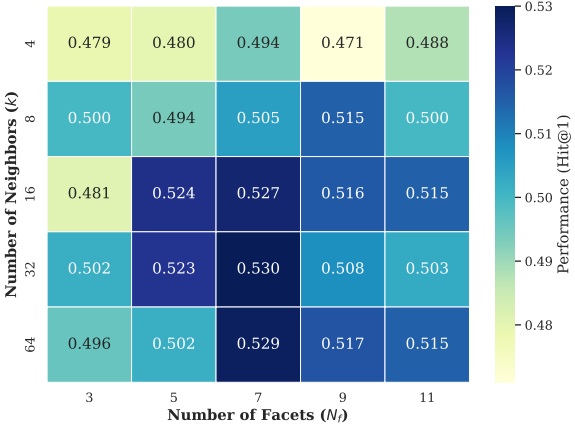


A better Hit@1 could in principle come with the same vague boilerplate rationale.

Grounding generation in M_{collab} improves the **specificity and relevance** of the rationale, not only the ranking: the explanation now cites evidence that exists.

Backup — sensitivity to neighbourhood settings

Books, random 1,000-user subset ▪ neighbours k vs. facets N_f; the deck uses $k = 16$, $N_f = 7$



Hit@1 across neighbourhood size and propagation settings.

Performance sits on a broad plateau rather than a single tuned point — and the drop at very large neighbourhoods is the cognitive-overload effect reappearing as a hyperparameter.

Backup — references

Generative Recommendation

- Geng et al. **P5**: *Recommendation as Language Processing*. RecSys 2022.
- Rajput et al. **TIGER**: *Recommender Systems with Generative Retrieval*. NeurIPS 2023.
- Zheng et al. **LC-Rec**: *Adapting LLMs by Integrating Collaborative Semantics*. ICDE 2024.
- Zhai et al. **HSTU**: *Actions Speak Louder than Words*. ICML 2024.
- Kuaishou. **OneRec**: *Unifying Retrieve and Rank with Generative Recommender and Iterative Preference Alignment*. 2025.

Agentic Recommendation

- Huang et al. **InteRecAgent**: *Recommender AI Agent*. TOIS 2025.
- Wang et al. **RecMind**: *Large Language Model Powered Agent for Recommendation*. Findings of NAACL 2024.
- Wang et al. **MACRec**: *Multi-Agent Collaboration Framework for Recommendation*. SIGIR 2024.
- Zhang et al. **Agent4Rec**: *On Generative Agents in Recommendation*. SIGIR 2024.
- Zhang et al. **AgentCF**: *Collaborative Learning with Autonomous Language Agents*. WWW 2024.
- **Chen et al. MemRec**: *Collaborative Memory-Augmented Agentic Recommender System*. ACL 2026.