Navigating Large Language Models for Recommendation: From Architecture to Learning Paradigms and Deployment

Lecture Tutorial For SIGIR 2025

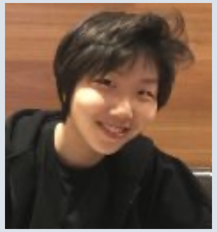
Organizers: Xinyu Lin, Keqin Bao, Jizhi Zhang, Sunhao Dai, Yang Zhang, Wenjie Wang, Fuli Feng, Xiangnan He

Outline

Our Team

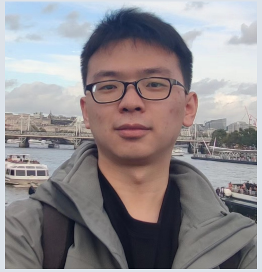
Keqin Bao
USTC, PhD Student

Jizhi Zhang
USTC, PhD Student

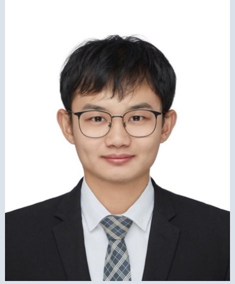


Xinyu Lin
NUS, PhD Student

Sunhao Dai
RUC, PhD Student



Yang Zhang
NUS, Research Fellow



Wenjie Wang
USTC, Prof

Fuli Feng
USTC, Prof

Xiangnan He
USTC, Prof

Tat-Seng Chua
NUS, Prof

Representative Work:

2023
TALLRec, RecSys
FaiRLLM, RecSys
…

2024
LETTER, CIKM
TransRec, KDD
DEALRec, SIGIR
…

2025
BIGRec, TORS
SETRec, SIGIR
CoLLM, TKDE
Rec4Agentverse, CACM
PersonalWAB, WWW

Previous Tutorials:

2023.11
1st Tutorial
@SIGIR-AP

2024.4
2nd Tutorial
@WWW

2024.7
3rd Tutorial
@SIGIR

2025.7
4th Tutorial
@SIGIR

Highlights for this tutorial:

- New structure: a systematic technical structure from pre-training, post-training, decoding, to deployment for LLM4Rec
- New directions (e.g., Reasoning4Rec)
- New future (e.g., Rec4Agent, E2E generative Rec)

Outline

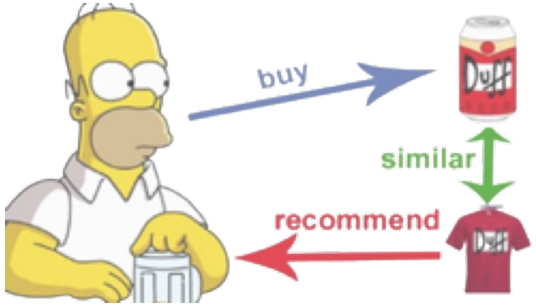
- 14:00-14:05 Introduction (Wenjie Wang)
- 14:05-14:20 Development of LLMs (Wenjie Wang)
- 14:20-17:15 Technical Stacks of LLM4Rec
  - 14:20-14:50: Model Architecture and Pre-training (Wenjie Wang)
  - Model Post-training
    - 14:50-15:30: Recommendation Accuracy (Yang Zhang)
    - 15:30-16:00: QA & Coffee Break
    - 16:00-16:15: Recommendation Efficiency (Wenjie Wang)
    - 16:15-16:45: Recommendation Trustworthiness (Sunhao Dai)
  - 16:45-17:15: Model Decoding and Deployment (Sunhao Dai)
- 17:15-17:30 Open Problems (Yang Zhang)
- 17:30-17:35 Future Direction & Conclusions (Yang Zhang)

Background of RecSys

❑ Information explosion era

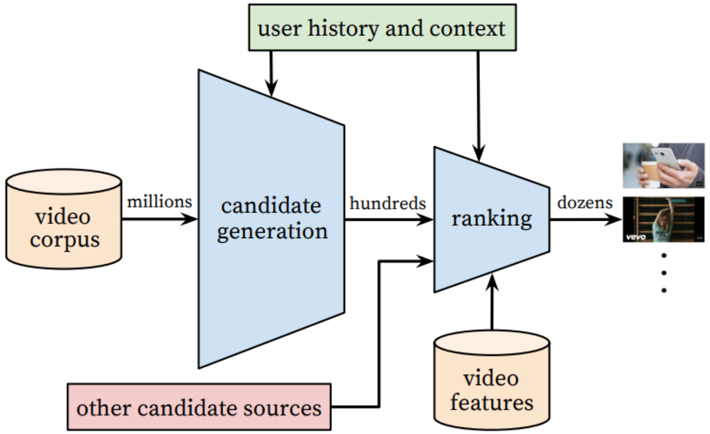
- E-commerce: 12 million items in Amazon.
- Social networks: 2.8 billion users in Facebook.
- Content sharing platforms: 720,000 hours videos uploaded to Youtube per day； 35 million videos posted on TikTok daily

❑ Recommender system



Information seeking via **user history feedback**

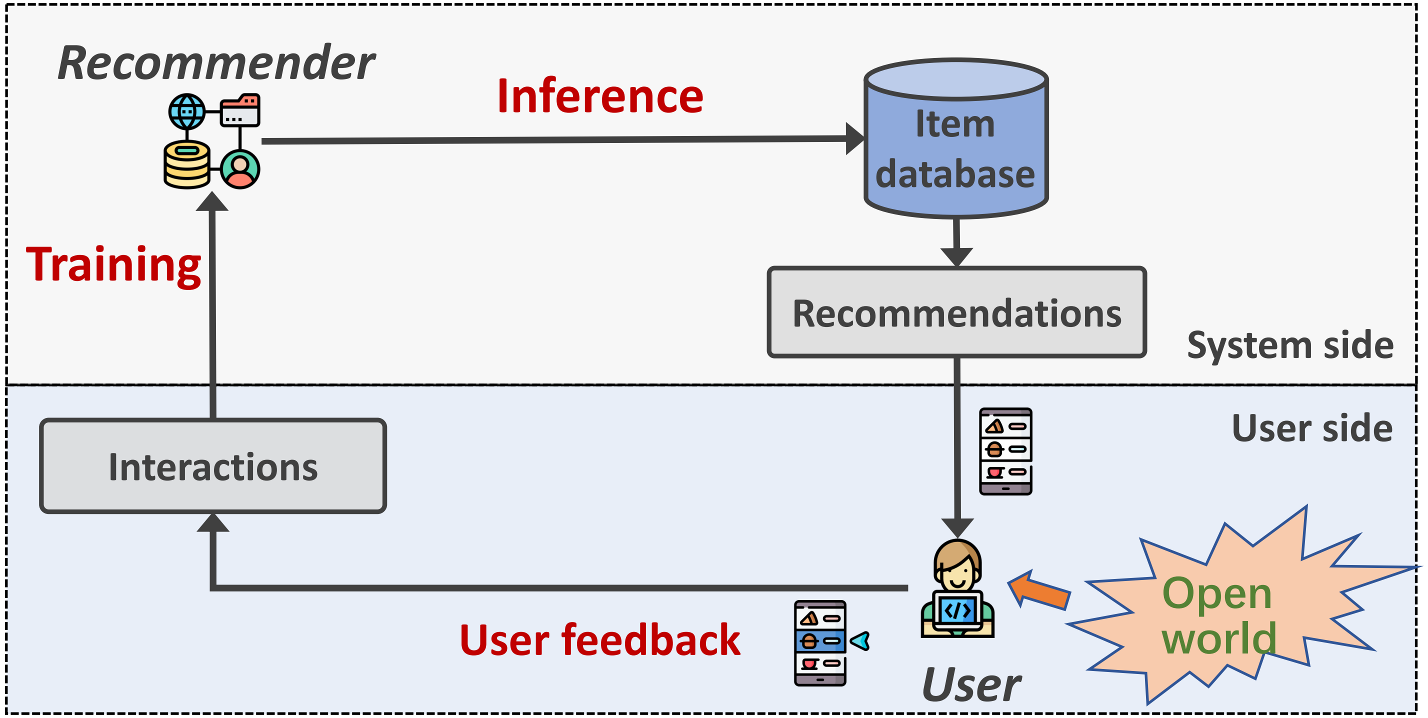
Recommendation



Images from: Deep neural networks for youtube recommendations

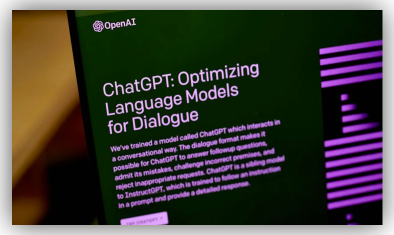
Background of RecSys

❑ Workflow of Recommender System



LLM4Rec: Model Architecture

- LLMs such as ChatGPT, GPT4, GPT-o1, DeepSeek-R1 have influenced many fields
  - LLMs change the paradigm of information seeking
  - Also affect research in NLP, IR, and MM domains.
  - How about recommendation?



ChatGPT

New Bing

Recommender System + LLMs?

LLMs for Recommendation

❑ How recommender systems benefit from LLMs

• **Representation:** Textual feature, item representation, knowledge representation	• **Interaction:** Acquire user information needs via dialog (**chat**)	• **Generalization:** cross-domain, knowledge compositional-generalization	• **Generation:** Personalized content generation, explanation generation

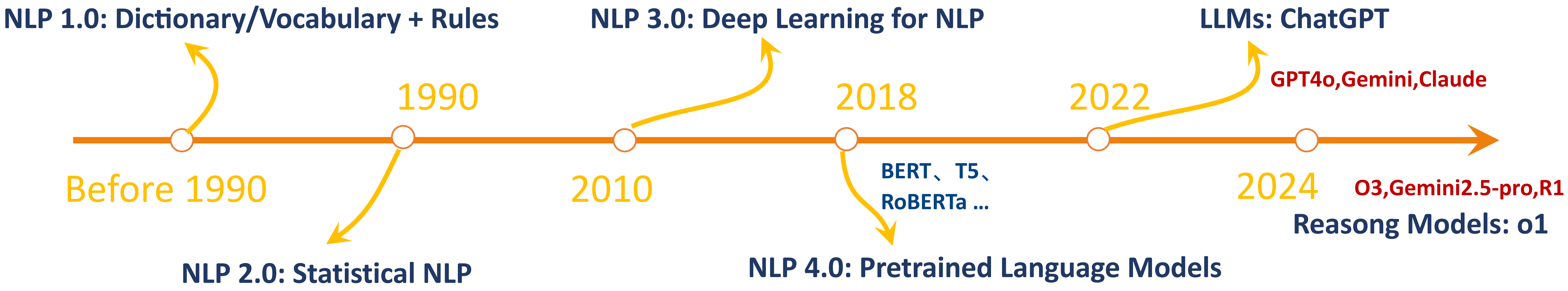
- **Learning paradigm:** Pretrain-finetune, Instruction-tuning, Preference-alignment

- **Model architecture:** Transformer、Self-attention,

Outline

- Introduction
- **Development of LLMs**
- Technical Stacks of LLM4Rec
- Open Problems
- Future Direction & Conclusions

The development of LLMs

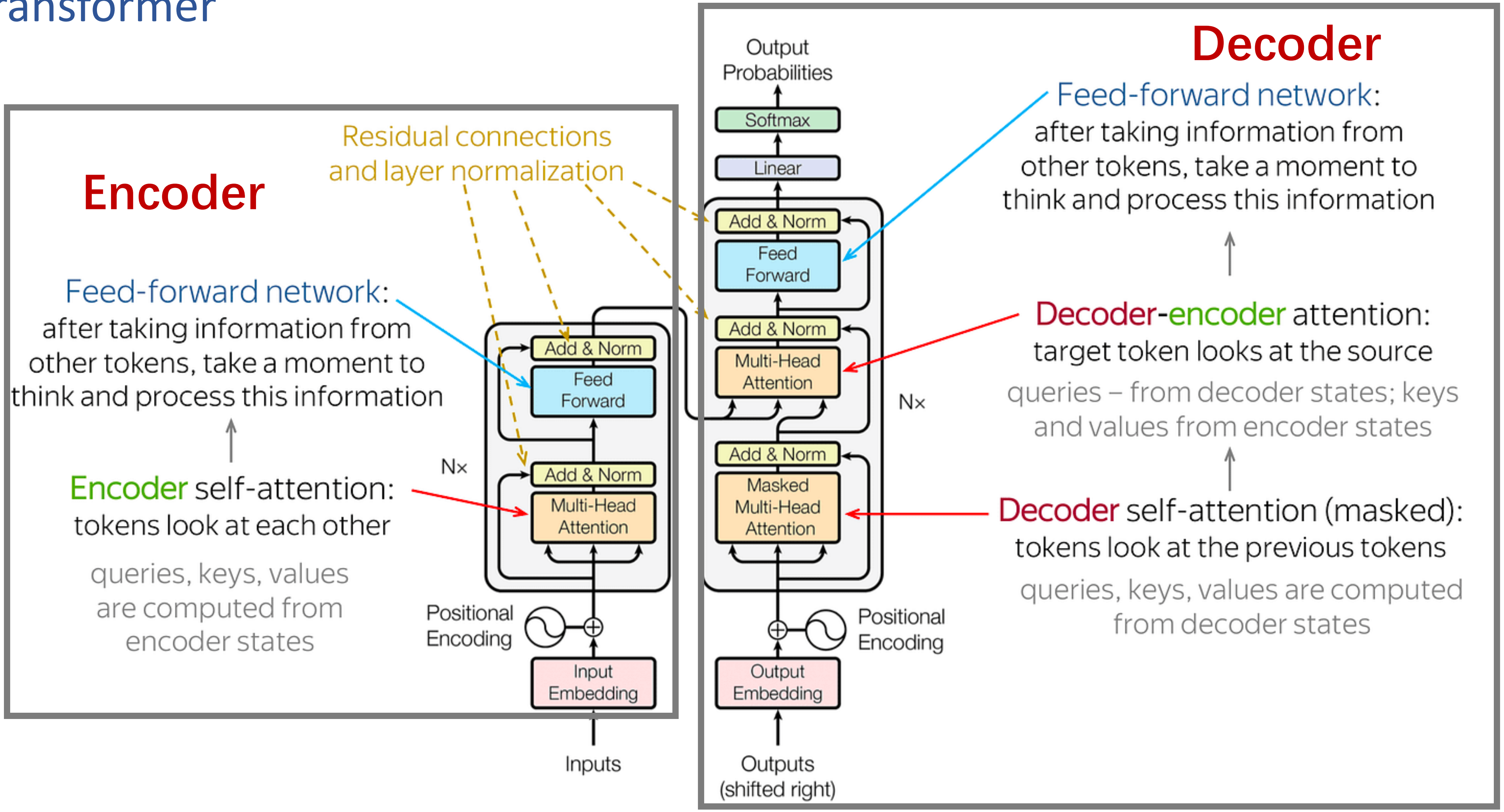


Large Language Model：billions of parameters, emergent capabilities

- Rich knowledge & Language Capabilities
- Instruction following
- In-context learning
- Chain-of-thought
- Reasoning
- …

Development of LLMs - architecture

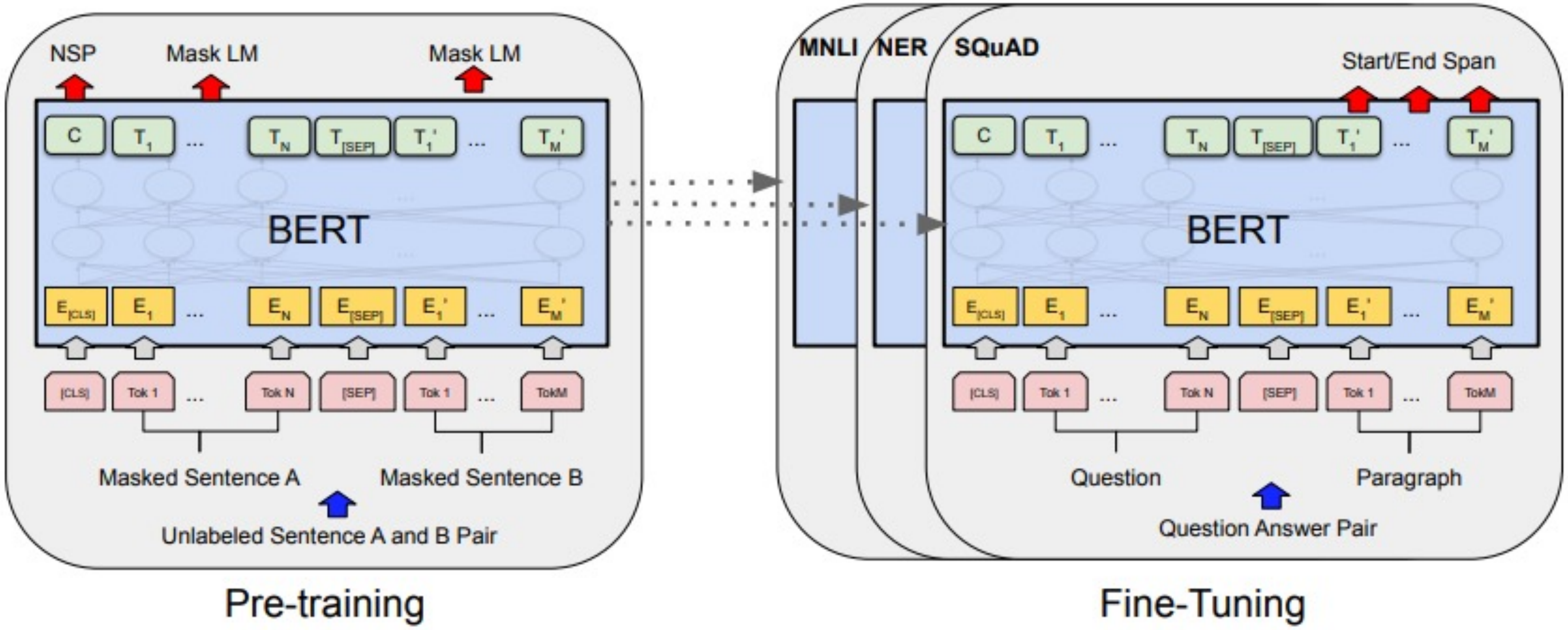
❑ Transformer



Ashish Vaswani et al. " Attention is All You Need". NIPS 2017.

Development of LLMs – architecture

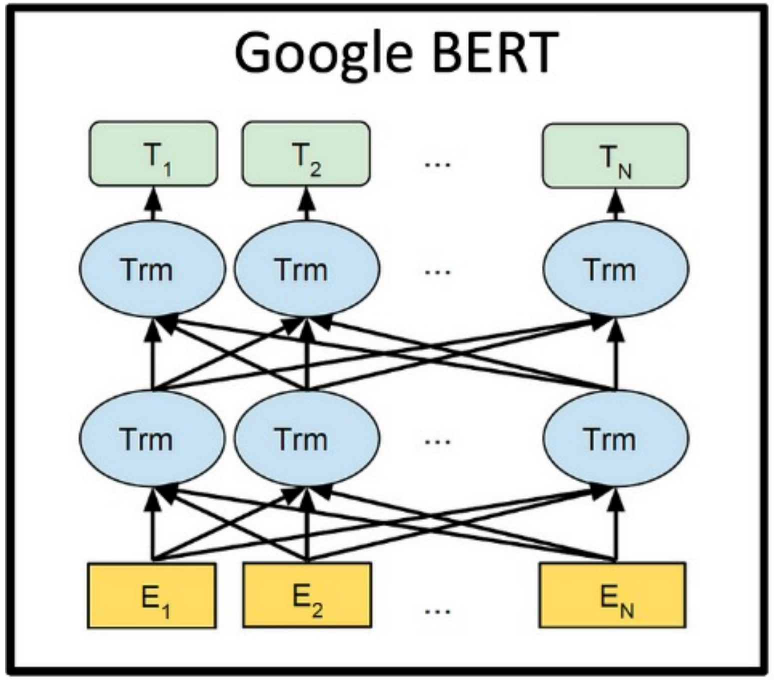
- ❑ BERT: pre-training of deep bidirectional transformers
  - ❑ Mask Language Modeling，bi-direction
  - ❑ Encoder (advantage) --> understanding

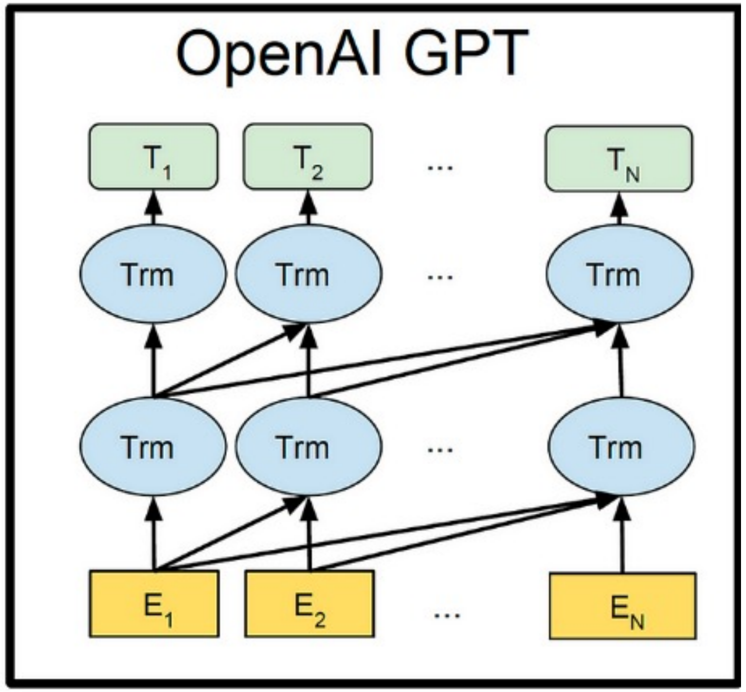


Development of LLMs – architecture

- GPT2: generative pre-trained transformer
  - Causal language modeling
  - Decoder (advantage) --> Generation
  - unsupervised multi-task learner

$$p(x) = \prod_{i=1}^{n} p(s_n | s_1, ..., s_{n-1})$$



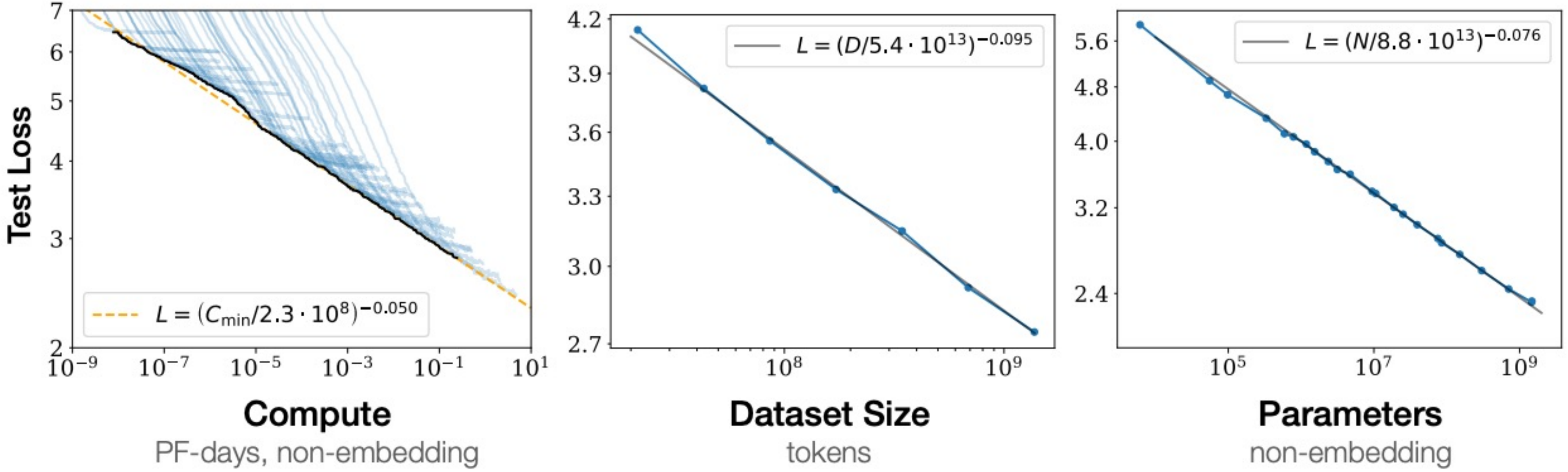


Alec Radford et al. Language Models are Unsupervised Multitask Learners. 2018.

Developments of LLMs – pre-training

❑ Scaling Laws

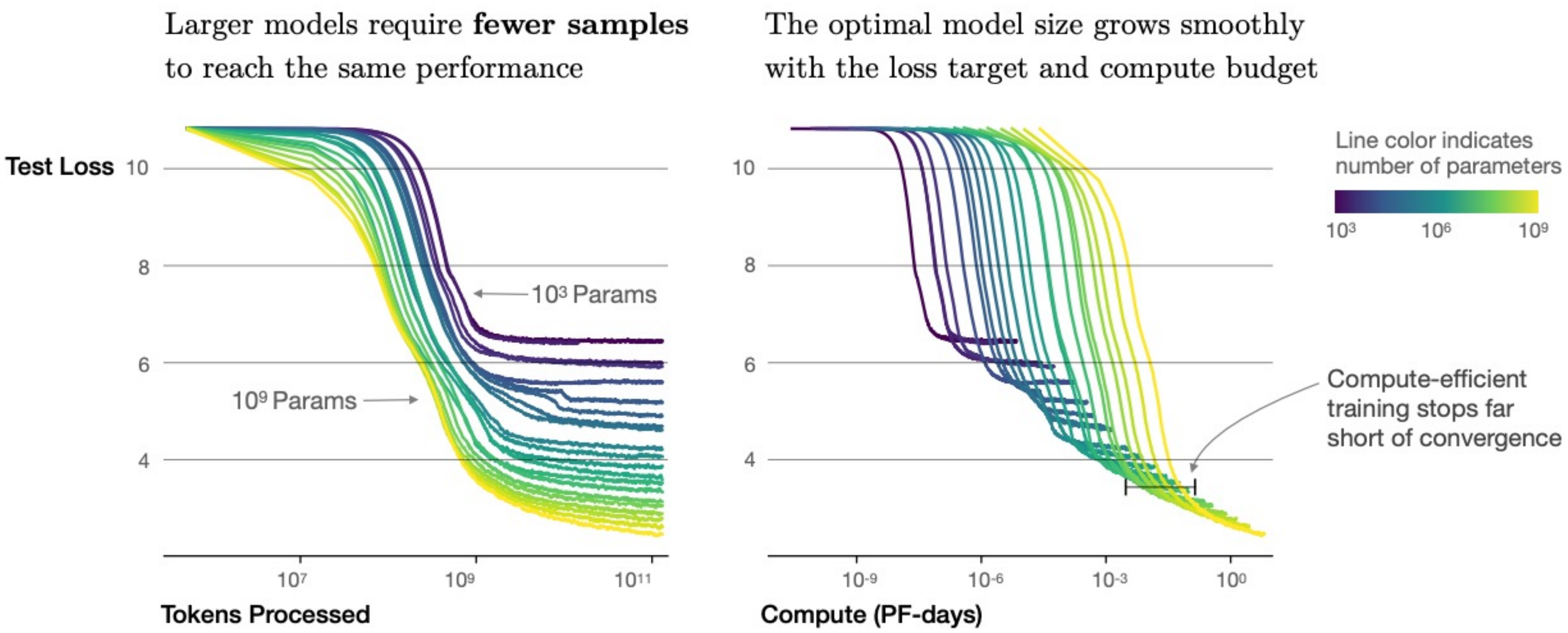
- ❑ The greater the amount of the data and the model parameters, the better the performance of the model
- ❑ Performance can be predicted



Jared Kaplan et al. Scaling Laws for Neural Language Models arxiv

Developments of LLMs - pre-training

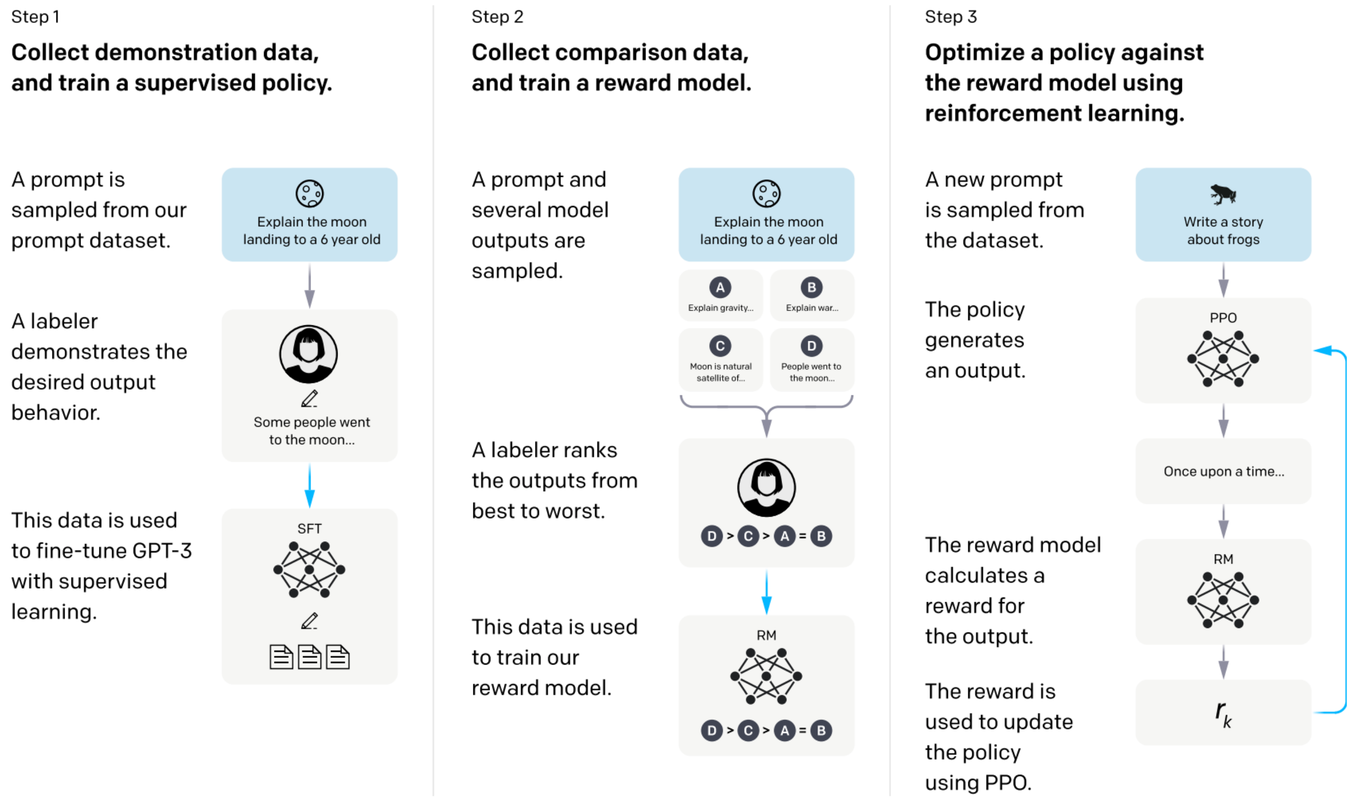
❑ Scaling Laws

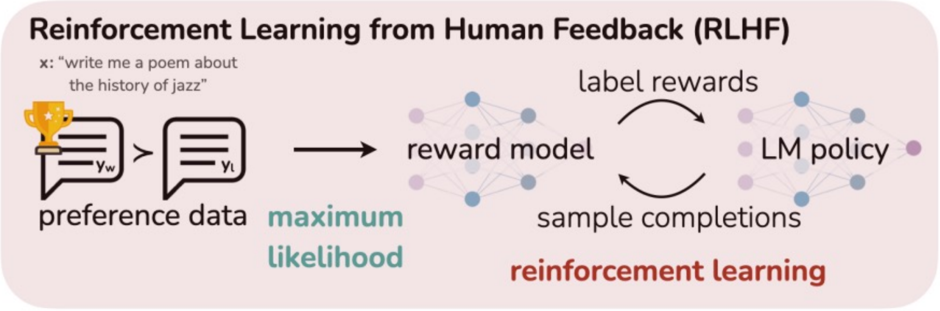


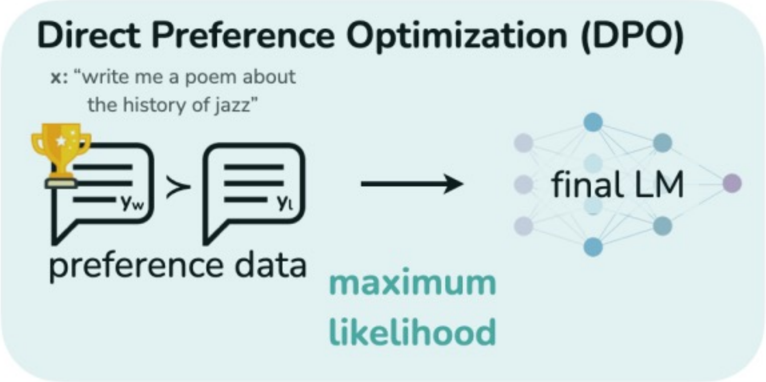
Jared Kaplan et al. Scaling Laws for Neural Language Models arxiv

Developments of LLMs – post-training

❑ Align with human

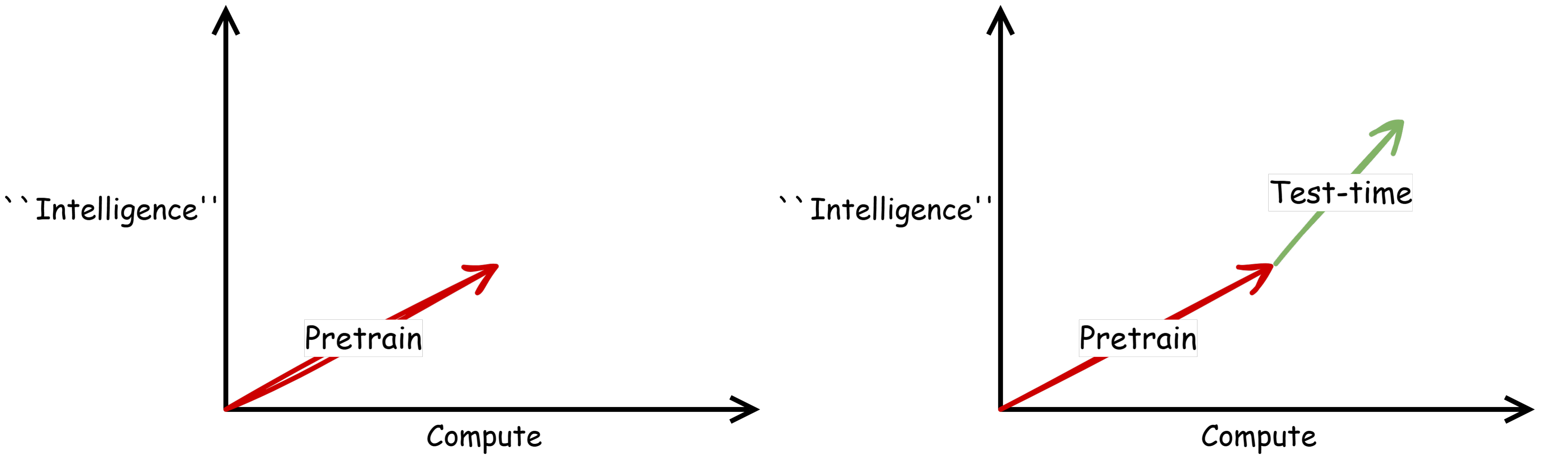






Long Ouyang et al., 2022 Training language models to follow instructions with human feedback NeurIPS 2022
Rafael Rafailov et al., 2023 Direct Preference Optimization: Your Language Model is Secretly a Reward Model NeurIPS 2023

Developments of LLMs – post-training



Increasing pretraining-time compute yields consistent performance improvements

Increasing test-time compute yields consistent performance improvements

Figure From https://github.com/zhangqiyuan-rico

Developments of LLMs – post-training

- ❑ **Training Paradiam**
  - ❑ RL : GRPO/PPO & RLVR
  - ❑ Transfers far beyond SFT, unlocking genuine generalization
- ❑ **Test Time Scaling**
  - ❑ Parallel Decoding
  - ❑ More compute at inference = consistent accuracy gains
- ❑ **Performance**
  - ❑ Reasoning drives breakthroughs in Math, STEM & Code
- ❑ **But how far can it push Recommendation?**

Augmented capabilities of LLMs

- Emergent abilities of LLM
  - Sufficient world knowledge
  - Chatting
  - In-context Learning & Instruction Following
  - Reasoning & Planning
  - Tool using
  - LLM as an Agent
  - ...

LLMs for Recommendation

Large Language Model Stack	
Deployment	Conversation, Math, Chat…
Decoding	Beam Search, greedy decoding ……
Post-training	RLHF, DPO, SFT for (safety) alignment RL for reasoning enhancement
Pre-training	Next-token prediction for Content understanding
Architecture	Self-attention Transformer

❑ Benefits built upon LLMs stack for recommendation

- **Representation:** Textual feature, item representation, knowledge representation
- **Interaction:** Acquire user information needs via dialog (**chat**)
- **Generalization:** cross-domain, knowledge compositional-generalization
- **Generation:** End2end Recommendation; Personalized content generation

LLMs for Recommendation

- **Key Challenge**
  - Mismatch between LLM objective and recommendation: emerging new items, dynamic user interests, etc.
  - LLMs tend to rely on semantics, and another important aspect of recommendation tasks is collaborative information.

Pathways for LLM4Rec

- **Incorporate recommendation knowledge to LLMs**

Large Foundation Model Stack		Personalized LLM/LFM Stack
Deployment Conversation, Math, Chat…	LLM utilization for recommendation tasks →	**Deployment** Various LLM roles: LLM as recommender, LLM as user simulator, LLM as rec agent
Decoding Beam Search, greedy decoding ……	Satisfy the decoding needs for recommendation →	**Decoding** Speculative decoding, Parallel decoding,
Post-training RLHF, DPO, SFT for (safety) alignment RL for reasoning enhancement	Teach the model for recommendation →	**Post-training** Behavior understanding, preference alignment and reasoning to pursue accuracy, efficiency, and trustworthiness.
Pre-training Next-token prediction for Content understanding	Pre-trained recommendation knowledge →	**Pre-training** Recommendation pre-training (open problem)
Architecture Self-attention Transformer	Model expressiveness for recommendation →	**Architecture** Item tokenizer, Memory for LLMs

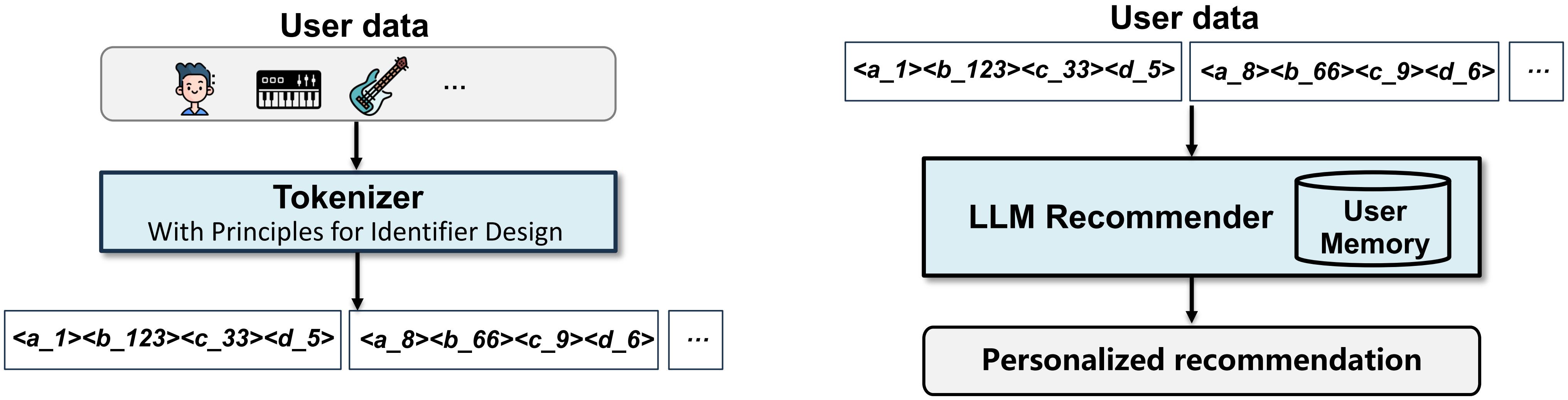
Outline

- Introduction
- Development of LLMs
- **Technical Stacks of LLM4Rec**
  - Model Architecture and Pre-training
  - Model Post-training
  - QA & Coffee Break
  - Model Post-training
  - Decoding and Deployment
- Open Problems
- Future Direction & Conclusions

Model Architecture

- **Enhance model expressiveness for recommendation**

Architecture	Model expressiveness for recommendation →	Architecture
Self-attention Transformer		Item tokenizer, Memory for LLMs



(a) Item Tokenization   **(b) Recommendation Generation**

Overview of LLM4Rec Architecture

- **Item Tokenizer**
  - ID-based: BERT4Rec, SASRec, ...
  - Text-based: Recformer, BIGRec, TransRec ...
  - Codebook-based: TIGER, LETTER …
  - Multi-facet: TransRec …
  - Set-based: SETRec …
  - …
- **LLM Recommender (with memory)**
  - Encoder-only: BERT4Rec …
  - Encoder-decoder: P5 …
  - Decoder-only: TALLRec, BIGRec …
  - Memory: ReLLa, LIBER …

Model Architecture: Item Tokenizer

- **Evolution of item tokenizer:**

Timeline

Vanilla ID

- Random ID
- Collaborative ID
- e.g., “1234”

Rich CF info.
Poor semantics

Model Architecture: Item Tokenizer

❑ ID-based: BERT4Rec

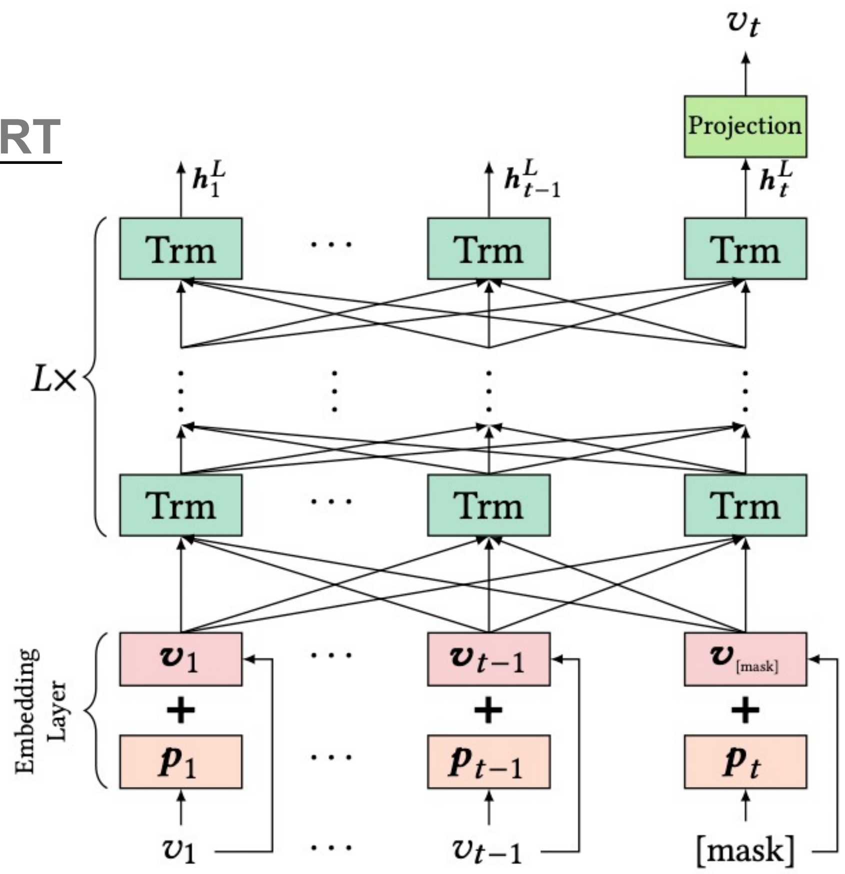
BERT

Natural Language:

- Token sequence
- Inter-token correlations

RecSys:

- ID sequence
- Inter-item correlations



(b) BERT4Rec model architecture.

Next-item prediction

Self-attention

Item ID embedding

Position embedding

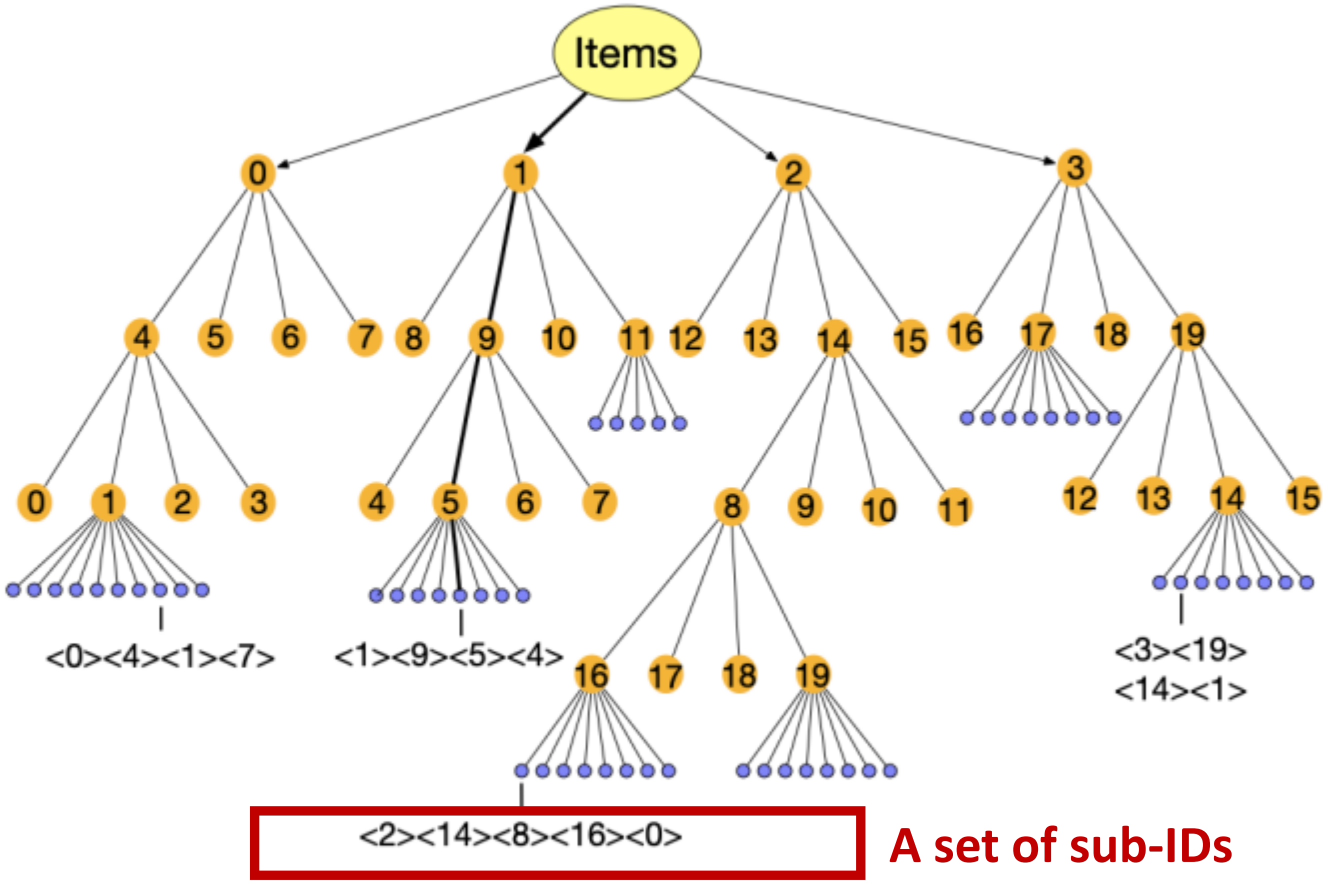
Training recommender by masked item prediction as BERT.

Sun, Fei, et al. "BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer." CIKM. 2019.

Model Architecture: Item Tokenizer

❑ ID-based: inject CF information into identifier

- Collaborative indexing: Clustering collaborative information to create IDs



- Construct item co-occurrence matrix
- Hierarchically cluster the factorized Laplacian matrix
- generate IDs based on cluster indices.

Advantages:

1) Add constraints on item IDs

2) Reduce the token spaces

Increase the learning efficacy.

[1] Hua et.al. How to Index Item IDs for Recommendation Foundation Models. SIGIR-AP 2023.

Model Architecture: Item Tokenizer

- **Evolution of item tokenizer:**

Timeline

Vanilla ID
- Random ID
- Collaborative ID
- e.g., “1234”

Rich CF info.
Poor semantics

Textual Identifiers
- Recformer [7], BIGRec [1],
- IDGenRec [2]
- e.g., titles

Rich semantics
Limited CF info.

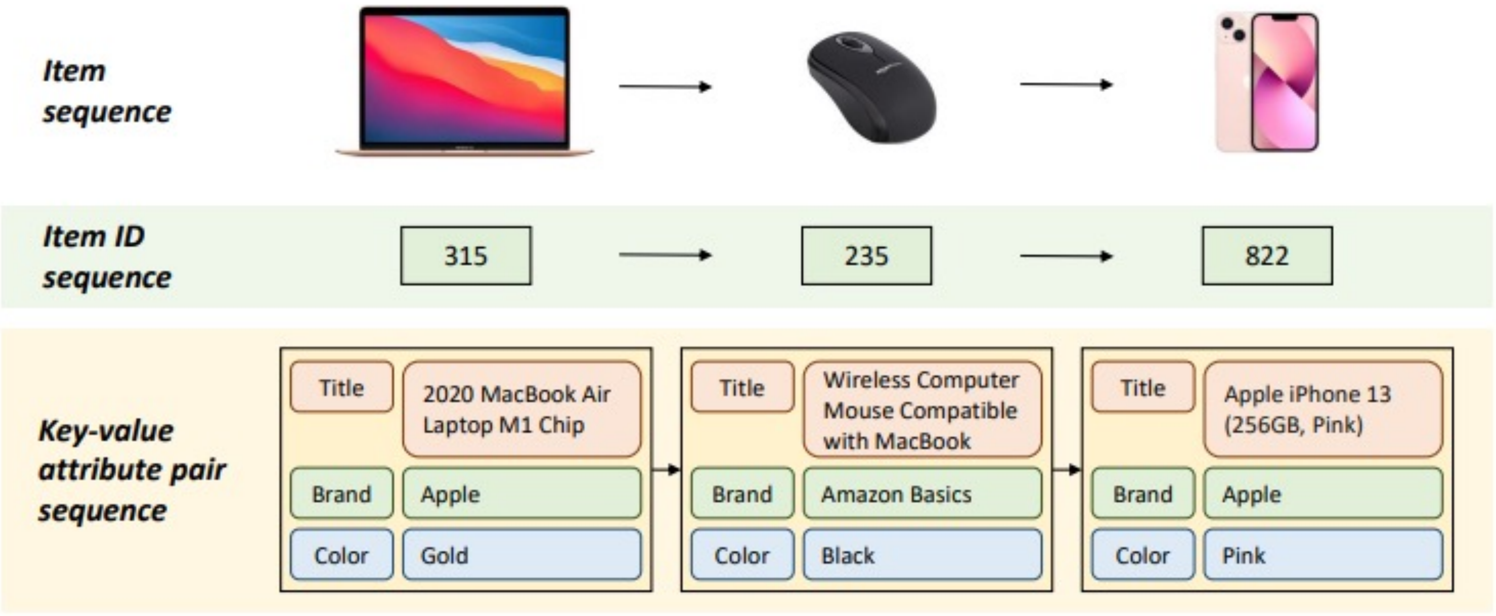
[1] Bao et al. A bi-step grounding paradigm for large language models in recommendation systems. TORS’24.
[2] Tan et al. IDGenRec: LLM-RecSys Alignment with Textual ID Learning. SIGIR24.
[3] Wang et al. Learnable Tokenizer for LLM-based Generative Recommendation. CIKM’25.
[4] Lin et al. Bridging items and language: A transition paradigm for large language model-based recommendation. KDD’24.
[5] Lin et al. Order-agnostic Identifier for Large Language Model-based Generative Recommendation. Arxiv 2025.
[6] Hou et al. Generating Long Semantic IDs in Parallel for Recommendation. KDD 2025.
[7] Li et al. Text Is All You Need: Learning Language Representations for Sequential Recommendation. KDD 2023.

Model Architecture: Item Tokenizer

❑ Text-based: Recformer

❑ Text is all you need (NO item ID)

- Only use texts to represent items.
- Low resource, better cold-start recommendation.

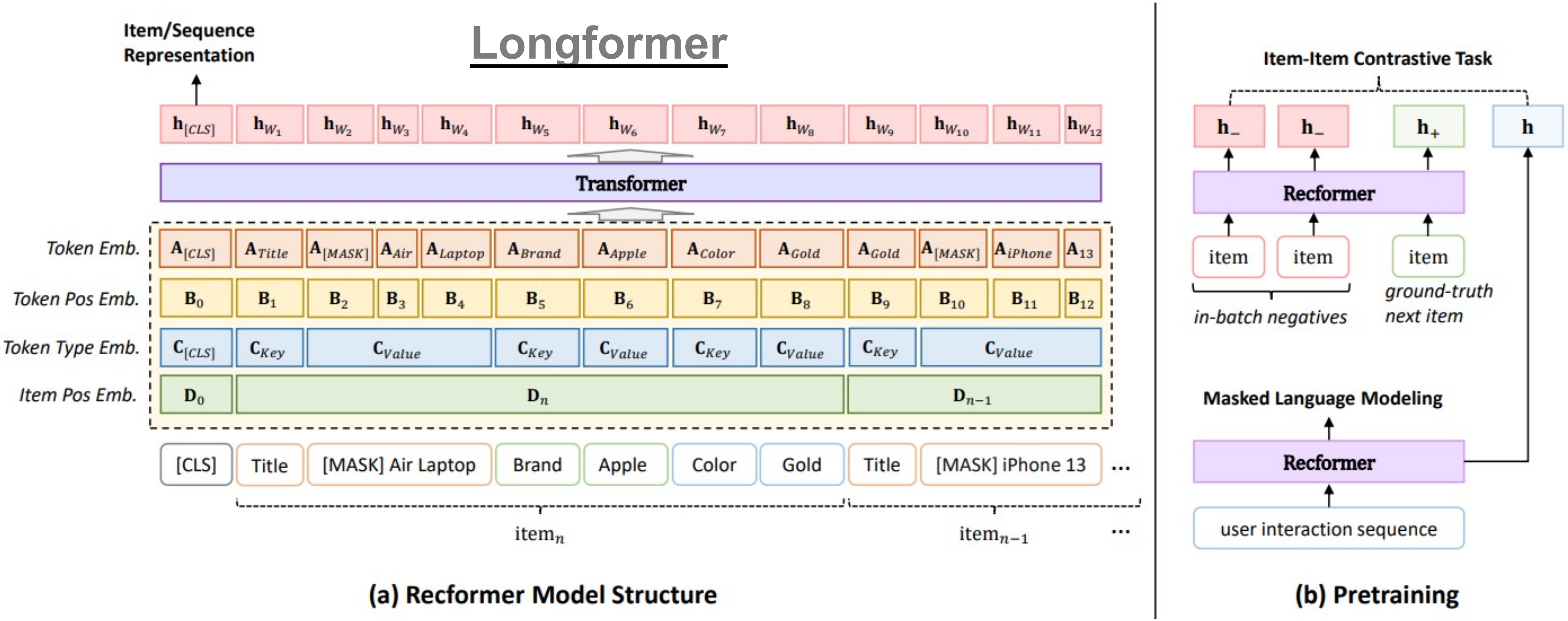


Li Jiacheng et al. "Text Is All You Need: Learning Language Representations for Sequential Recommendation“ KDD 2023.

Model Architecture: Item Tokenizer

❑ Text-based: Recformer

- ❑ Text is all you need (NO item ID)



Li Jiacheng et al. "Text Is All You Need: Learning Language Representations for Sequential Recommendation" KDD 2023.

Model Architecture: Item Tokenizer

❑ Text-based: BIGRec

Instruction Input	
Instruction:	Given ten movies that the user watched recently, please recommend a new movie that the user likes to the user.
Input:	The user has watched the following movies before: "Traffic (2000)", "Ocean's Eleven (2001)", ... "Fargo (1996)"
Instruction Output	
Output:	"Crouching Tiger, Hidden Dragon (Wu hu zang long) (2000)"

Iron Man (2008)

Titanic (2008)

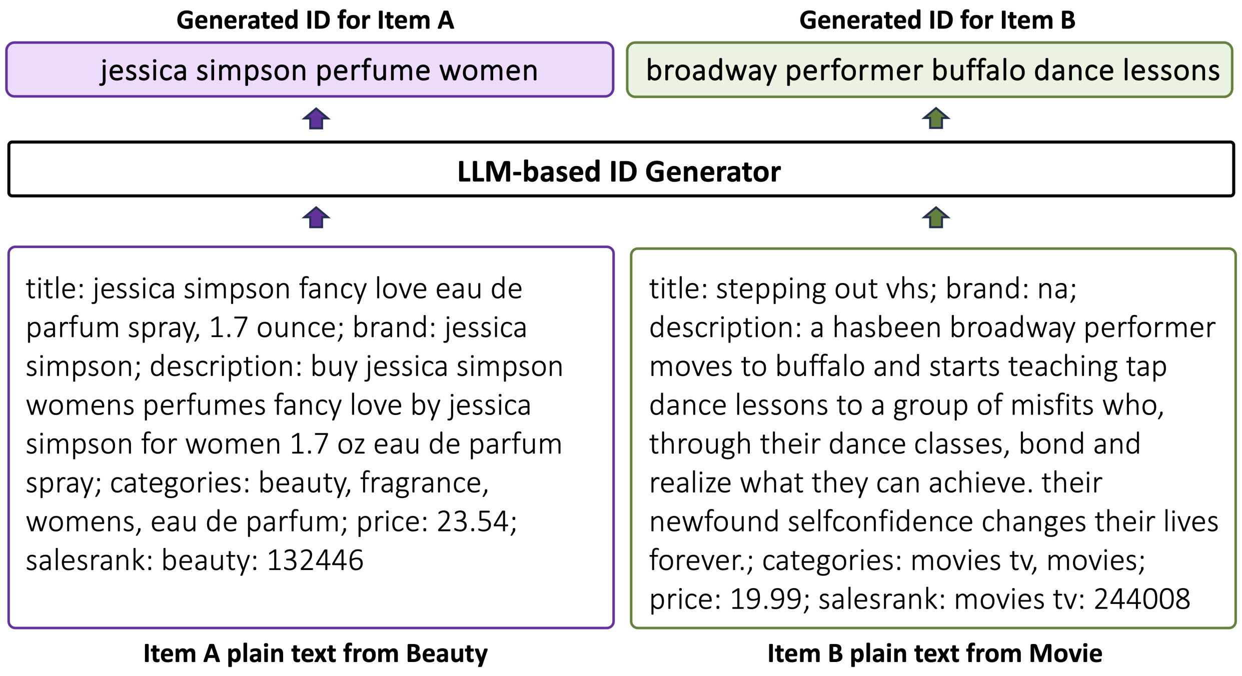
...

Item title as item identifier. The user history is transformed into a sequence of item title. LLMs will generate the next item title as recommendation.

Model Architecture: Item Tokenizer

❑ Text-based: inject CF information into identifier

- **IDGenRec: generate textual ID aligned with user behavior**



Textual ID generator: text-to-text

- **ID Generator**
  - Input plain text of item information into fine-tuned tag generator
  - Generate several short, informative, and unique tags in natural language

Advantages:

1) Use text tokens, thus harnessing LLMs' semantic knowledge
2) Generalize to new items
3) Align with user bevhavior

[1] Tan et.al. Towards LLM-RecSys Alignment with Textual ID Learning. SIGIR 2024.

Model Architecture: Item Tokenizer

- **Evolution of item tokenizer:**

Timeline

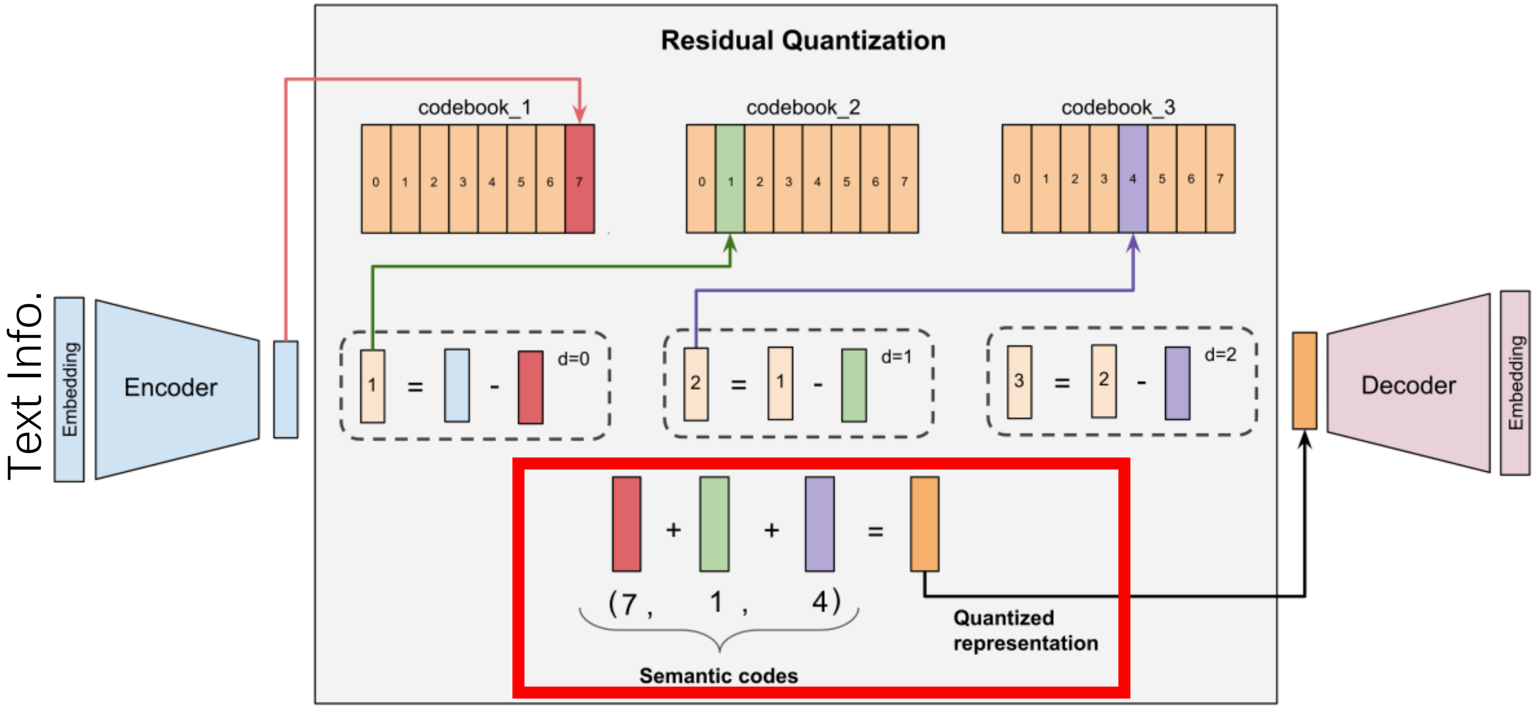
Vanilla ID	Textual Identifiers	Codebook-based Identifiers
• Random ID • Collaborative ID • e.g., “1234”	• Recformer [7],BIGRec [1], IDGenRec [2] • e.g., titles	• TIGER • LETTER [3] • i.e., hierarchical info.
Rich CF info. Poor semantics	Rich semantics Limited CF info.	Rich semantics Limited CF info.

[1] Bao et al. A bi-step grounding paradigm for large language models in recommendation systems. TORS’24.
[2] Tan et al. IDGenRec: LLM-RecSys Alignment with Textual ID Learning. SIGIR24.
[3] Wang et al. Learnable Tokenizer for LLM-based Generative Recommendation. CIKM’25.
[4] Lin et al. Bridging items and language: A transition paradigm for large language model-based recommendation. KDD’24.
[5] Lin et al. Order-agnostic Identifier for Large Language Model-based Generative Recommendation. Arxiv 2025.
[6] Hou et al. Generating Long Semantic IDs in Parallel for Recommendation. KDD 2025.
[7] Li et al. Text Is All You Need: Learning Language Representations for Sequential Recommendation. KDD 2023.

Model Architecture: Item Tokenizer

❑ Codebook-based

- Semantic-aware ID (Tiger/LC-Rec): quantizing text embedding to generate IDs



Quantization: RQ-VAE

- Convert text content information into embeddings
- Quantization: represent the text embedding with several sub-embeddings, generating semantic ID
- Several sub-IDs form a semantic ID

Advantages:

1) Reduce the token spaces, $N \rightarrow K \cdot N^{1/K}$
2) 2) Could deal with new items

[1] Zheng et.al. Adapting Large Language Models by Integrating Collaborative Semantics for Recommendation. ICDE 2024.

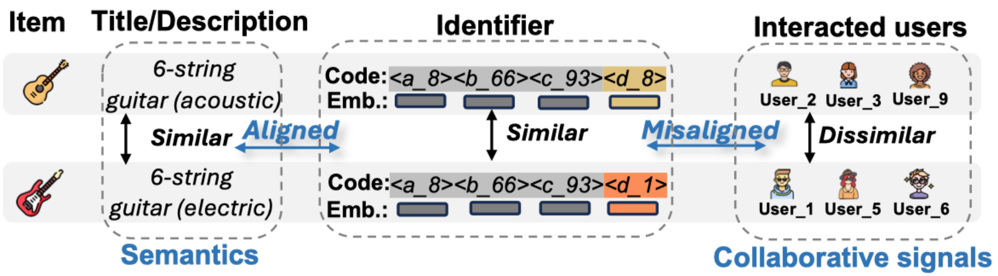
[2] Rajput el.al. Recommender Systems with Generative Retrieval. NeurIPS 2023.

Model Architecture: Item Tokenizer

❑ Codebook-based: inject CF into identifier

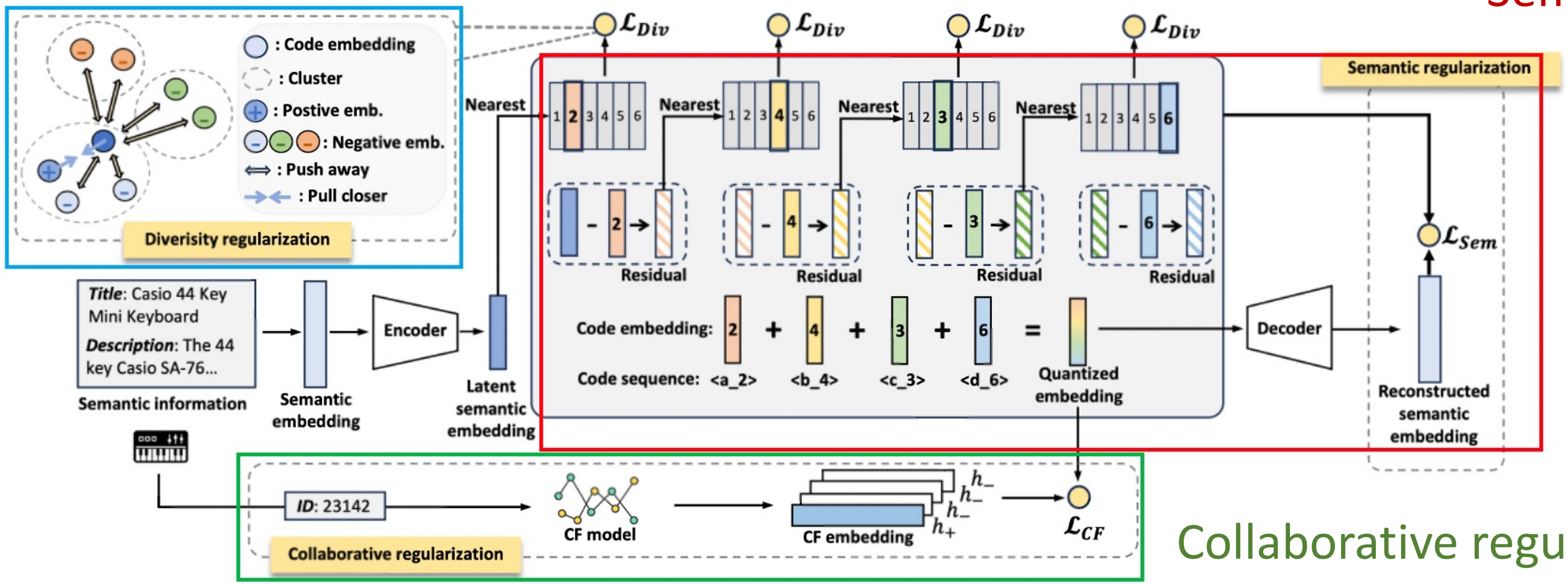
- Semantic and CF-aware ID (LETTER): align generated identifier with CF information

Semantic ID might misalign with user behaviors.



Diversity regularization

Semantic regularization



Collaborative regularization

Model Architecture: Item Tokenizer

- **Evolution of item tokenizer:**

Timeline

Vanilla ID
- Random ID
- Collaborative ID
- e.g., “1234”

Rich CF info.
Poor semantics

Textual Identifiers
- Recformer[7], BIGRec [1], IDGenRec [2]
- e.g., titles

Rich semantics
Poor CF info.

Codebook-based Identifiers
- TIGER
- LETTER [3]
- i.e., hierarchical info.

Rich semantics
Limited CF info.

Multi-facet Identifiers
- TransRec [4]
- i.e., hybrid info

Rich semantics
Rich CF info.

[1] Bao et al. A bi-step grounding paradigm for large language models in recommendation systems. TORS’24.
[2] Tan et al. IDGenRec: LLM-RecSys Alignment with Textual ID Learning. SIGIR24.
[3] Wang et al. Learnable Tokenizer for LLM-based Generative Recommendation. CIKM’25.
[4] Lin et al. Bridging items and language: A transition paradigm for large language model-based recommendation. KDD’24.
[5] Lin et al. Order-agnostic Identifier for Large Language Model-based Generative Recommendation. Arxiv 2025.
[6] Hou et al. Generating Long Semantic IDs in Parallel for Recommendation. KDD 2025.
[7] Li et al. Text Is All You Need: Learning Language Representations for Sequential Recommendation. KDD 2023.

Model Architecture: Item Tokenizer

❑ Multi-facet identifier: incorporate both CF and semantics in parallel

ID 15826 **(Distinctiveness)**

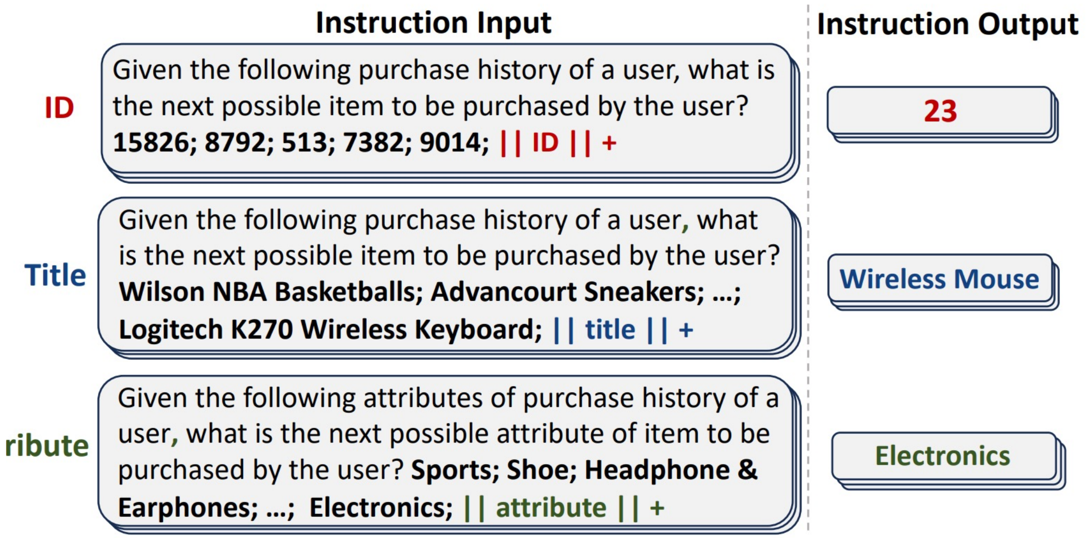
Title Wilson Indoor Basketballs

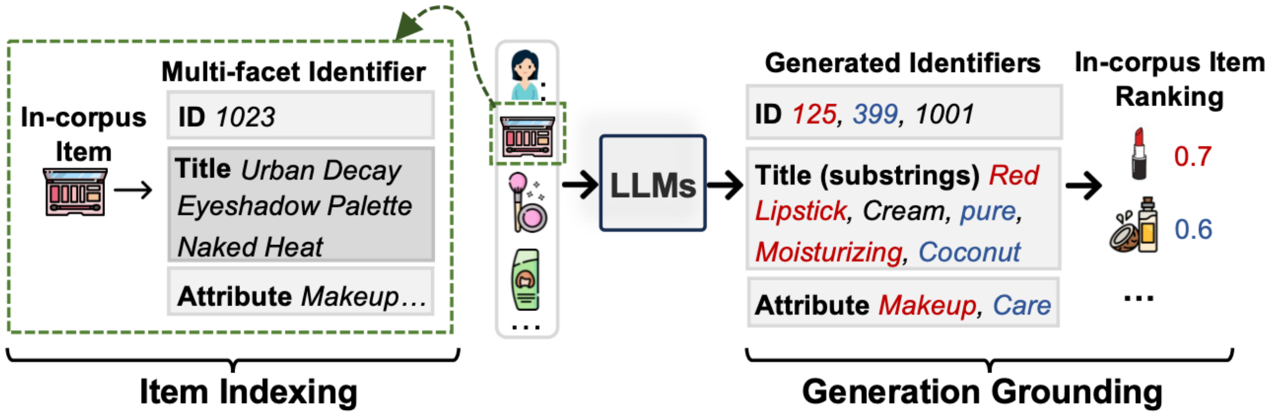
Attribute Sports

(Semantics)

Each item is represented by three different facets

- **Instruction data**





Each item is generated in three different facets, in parallel

[4] Lin et al. Bridging items and language: A transition paradigm for large language model-based recommendation. KDD'24.

Model Architecture: Item Tokenizer

- **Evolution of item tokenizer:**

Timeline

Vanilla ID
- Random ID
- Collaborative ID
- e.g., “1234”

Rich CF info.
Poor semantics

Textual Identifiers
- Recformer[7], BIGRec [1], IDGenRec [2]
- e.g., titles

Rich semantics
Poor CF info.

Codebook-based Identifiers
- TIGER
- LETTER [3]
- i.e., hierarchical info.

Rich semantics
Limited CF info.

Multi-facet Identifiers
- TransRec [4]
- i.e., hybrid info

Rich semantics
Rich CF info.

Critical issue: autoregressive generation → low efficiency, local optima issue

Set Identifiers
- SETRec [5]
- order-agnostic tokens

Rich semantics & CF info.
Order-agnostic tokens

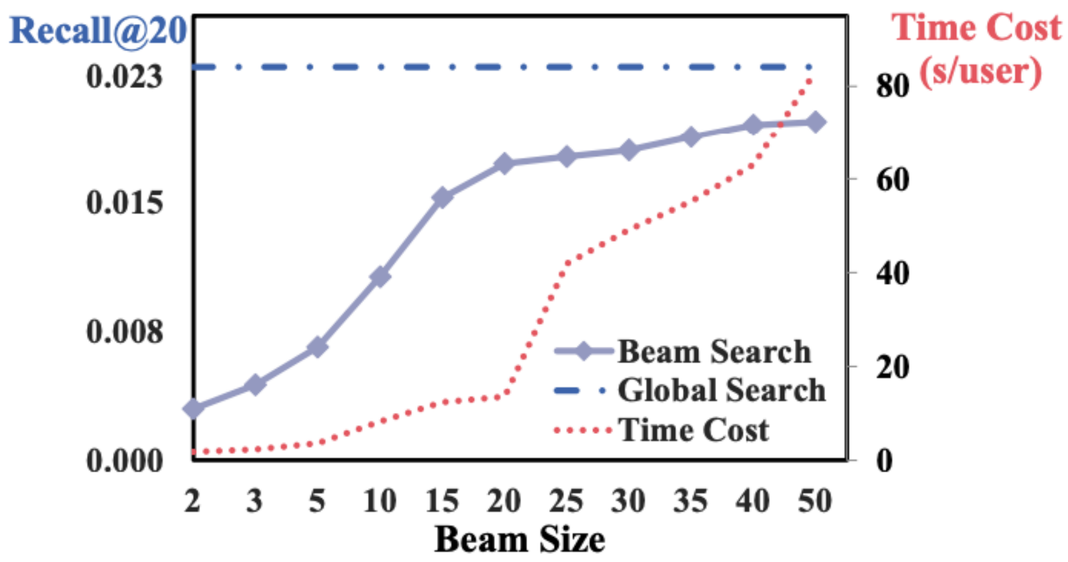
High efficiency
Globally optimal results

[1] *Bao et al. A bi-step grounding paradigm for large language models in recommendation systems. TORS’24.*
[2] *Tan et al. IDGenRec: LLM-RecSys Alignment with Textual ID Learning. SIGIR24.*
[3] *Wang et al. Learnable Tokenizer for LLM-based Generative Recommendation. CIKM’25.*
[4] *Lin et al. Bridging items and language: A transition paradigm for large language model-based recommendation. KDD’24.*
[5] *Lin et al. Order-agnostic Identifier for Large Language Model-based Generative Recommendation. Arxiv 2025.*
[6] Hou et al. Generating Long Semantic IDs in Parallel for Recommendation. KDD 2025.
[7] *Li et al. Text Is All You Need: Learning Language Representations for Sequential Recommendation. KDD 2023.*

Model Architecture: Item Tokenizer

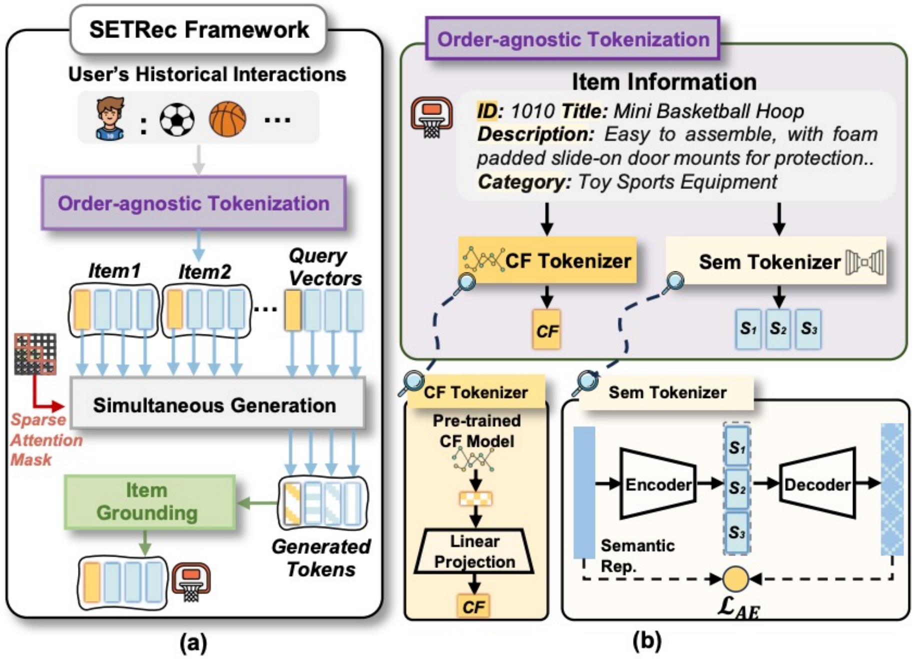
❑ Set identifier

- SETRec: a set of order-agnostic tokens for simultaneous encoding and decoding



- Existing token sequence identifier suffers from local optima via beam search

- **Set identifier tokenizer**
  - **CF tokenizer**: assign each item an ID embedding
  - **Semantic tokenizer**: assign each item a set of semantic embeddings



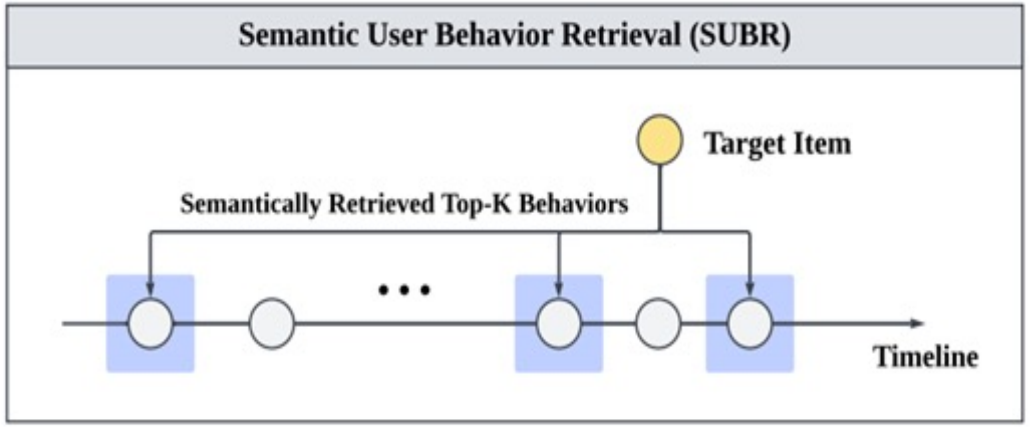
Model Architecture: Memory

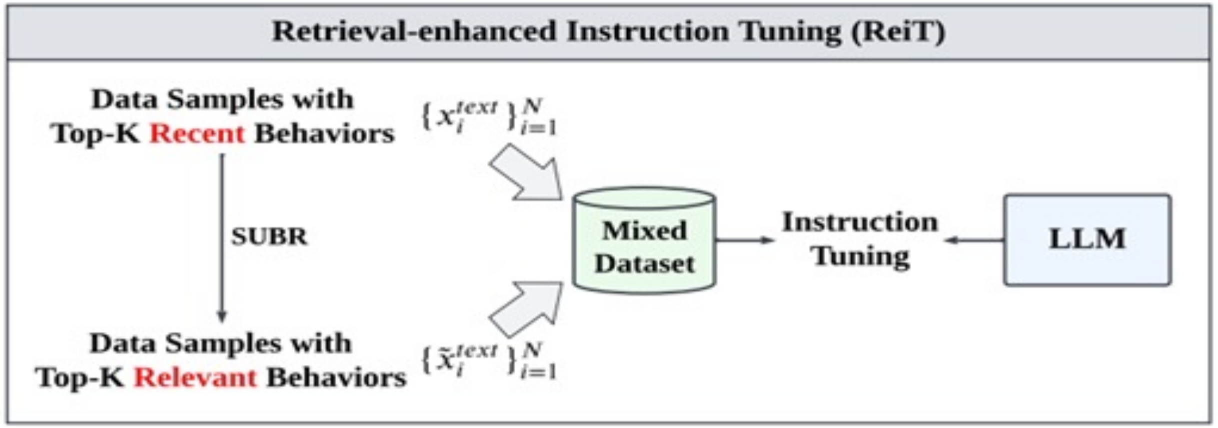
Static external memory:

- **Rella --- just retrieve most (semantically) similar items from the history**

Step1: For a target item, retrieve the top-K semantically similar items from the history, forming a new sample

Step2: Leverage the original sample and new sample to fine tune LLM for recommendation



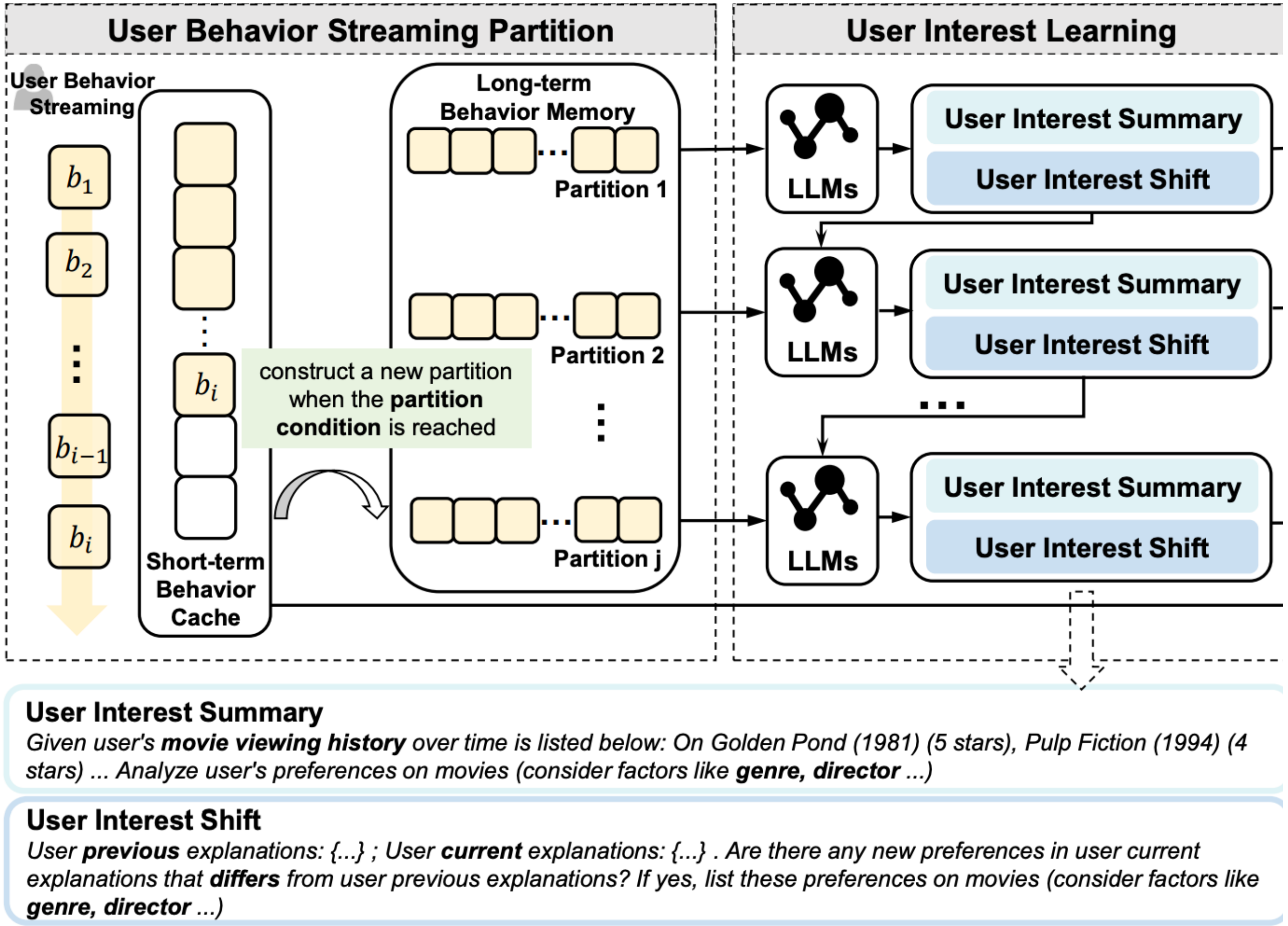


- Limitations: heavily depends on “target attention, not applicable when the input lacks target items.
- Future: may need to explore other solutions like memory.

Li et al. ReLLa: Retrieval-enhanced Large Language Models for Lifelong Sequential Behavior Comprehension in Recommendation. WWW, 2024.

Model Architecture: Memory

Dynamically updated memory：



User Behavior Streaming Partition

- Short-term cache and constructs a new partition once a predefined condition is met (e.g., time window, behavior count).
- Captures evolving user preferences across different interest stages.

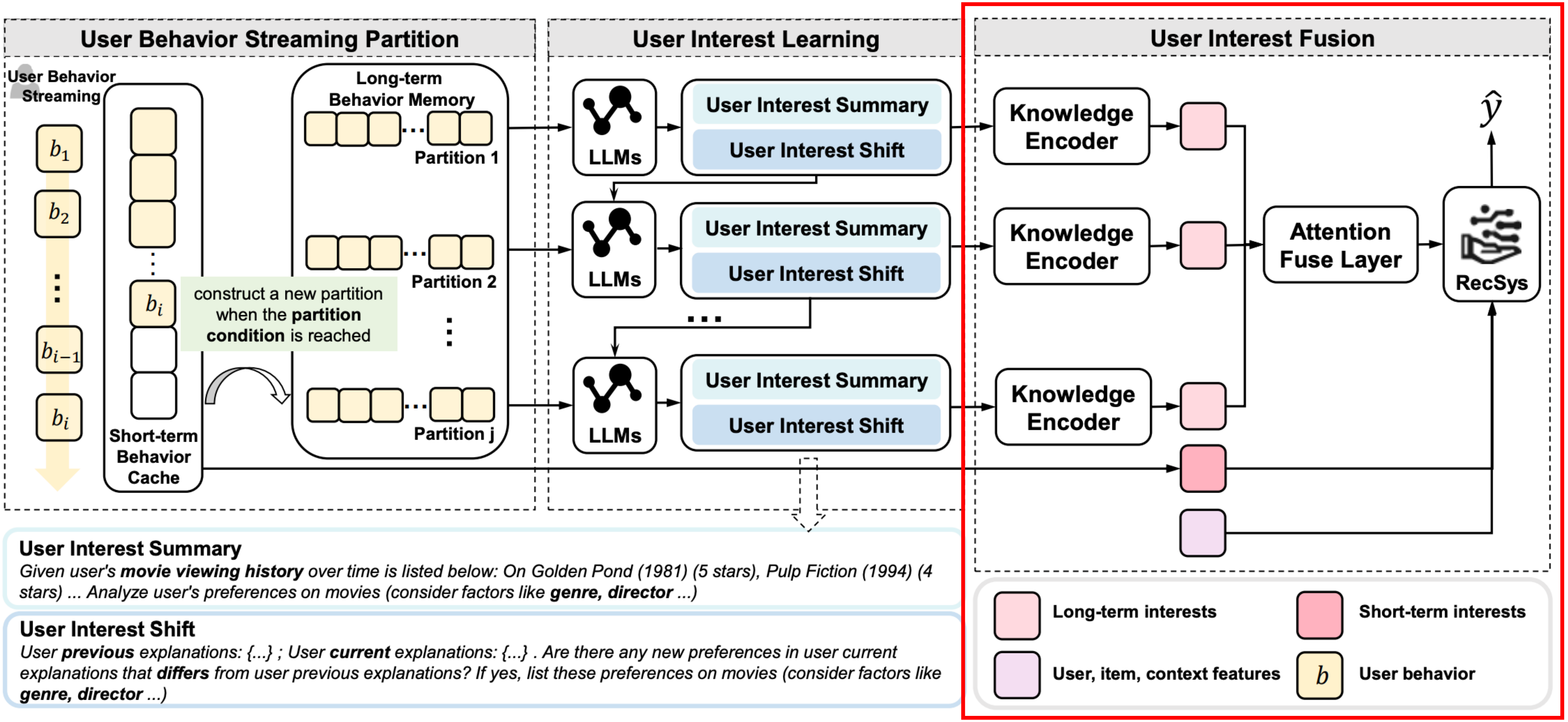
User interest Learning

- Iteratively update the user interests temporally, to capture evolving preference.

Zhu et al. LIBER: Lifelong User Behavior Modeling Based on Large Language Models. arxiv 2024

Model Architecture: Memory

Dynamically updated memory：



Zhu et al. LIBER: Lifelong User Behavior Modeling Based on Large Language Models. arxiv 2024

Model Architecture: Memory

Difference-aware memory:

Problem How to build memory better?

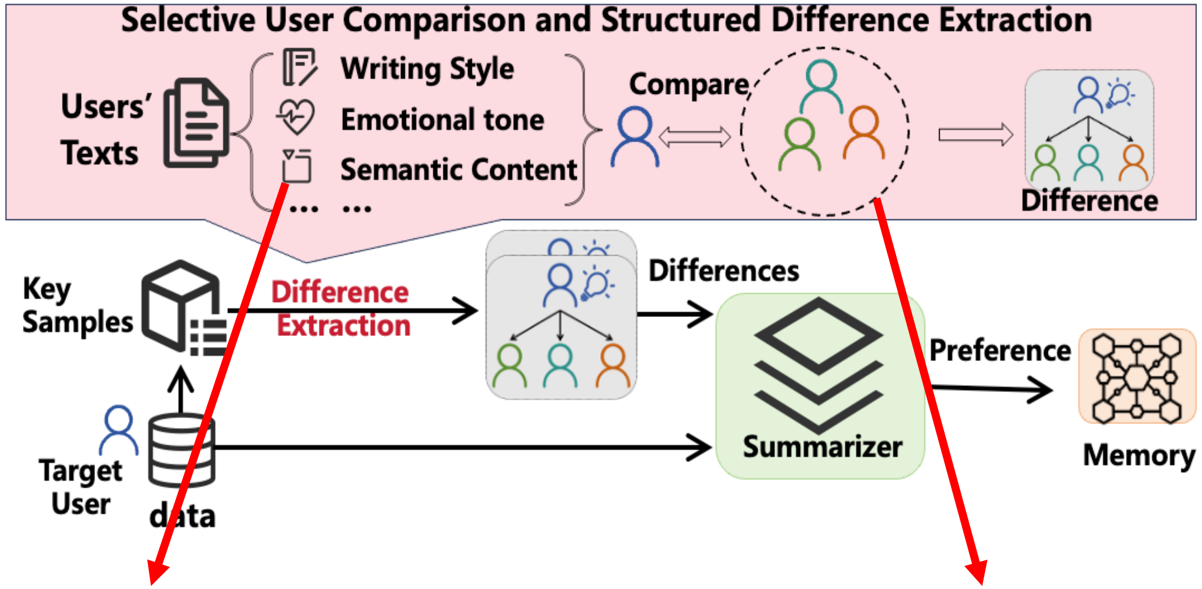
Challenge User data could be redundant, not all information is important for personalization

Our difference makes us unique; uniqueness shapes the preference

Existing methods usually ***capture Personal data patterns*** ignore the ***inter-user comparative analysis***, which is essential for personalization

Solution

- A Difference-aware method for Memory Construction (DPL)
- Emphasize extracting inter-user differences in memory construction for personalization tasks.



1 Predefine dimensions to structure the difference extraction

2 Select representative users for comparison (via clustering)

Yilun Qiu, et al. Measuring What Makes You Unique: Difference-Aware User Modeling for Enhancing LLM Personalization. ACL 2025 Submission.

Pre-training

Pre-training

Next-token prediction for Content understanding

Architecture

Self-attention Transformer

Pre-trained recommendation knowledge →

Model expressiveness for recommendation →

Pre-training

Recommendation Pretraining

Architecture

Item tokenizer, Memory for LLMs

- **Open problems: pre-train LLMs specifically for recommendation**
  - Objective: incorporate rich recommendation knowledge in pre-training stage
  - **Three key aspects:**

Item knowledge in different dimensions

Semantics

Temporal

User knowledge in different dimensions

User profile

User social relations

User-item interaction knowledge in different dimensions

Collaborative

Temporal

Align LLM with Recommendation Task

Pre-training

Next-token prediction for Content understanding

Architecture

Self-attention Transformer

Pre-trained recommendation knowledge →

Model expressiveness for recommendation →

Pre-training

Recommendation Pretraining

Architecture

Item tokenizer, Memory for LLMs

- **With pre-trained LLMs on general knowledge such as LLaMA, can we do recommendations?**

In-context learning is possible

ICL

❑ In-context learning

- LLMs has rich world knowledge, wonderful abilities like reasoning, instruction following, in-context learning.
- The LLMs itself could be leveraged for recommendation by in context learning.
- Existing works on in-context learning:
  - Ask LLM for recommendation
  - Serving as knowledge augmentation for traditional recsys
  - Optimize the prompt used for recommendation
  - Directly used for conversational recommender system

In-context learning: directly ask LLMs for recommendation

- Prompt construction

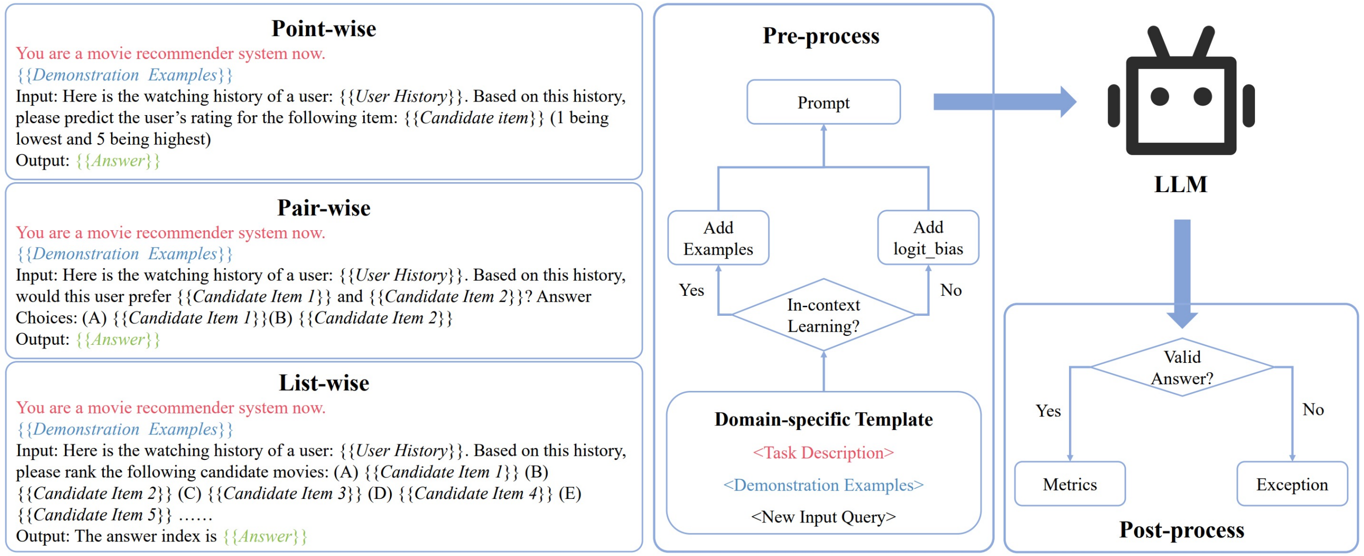


Figure 1: The overall evaluation framework of LLMs for recommendation. The left part demonstrates examples of how prompts are constructed to elicit each of the three ranking capabilities. The right part outlines the process of employing LLMs to perform different ranking tasks and conduct evaluations.

Three different ways of measuring ranking abilities:

$$\hat{y}'_i = LLM_{\text{point}}\left(I, \mathcal{D}, f(\mathbf{h}', \mathbf{c}' \mid u)\right)$$

$$\hat{y}'_{i_m > i_n} = LLM_{\text{pair}}\left(I, \mathcal{D}, f(\mathbf{h}', \mathbf{c}' \mid u)\right)$$

$$\hat{y}'_{i_1}, \hat{y}'_{i_2}, \cdots, \hat{y}'_{i_k} = LLM_{\text{list}}\left(I, \mathcal{D}, f(\mathbf{h}', \mathbf{c}' \mid u)\right)$$

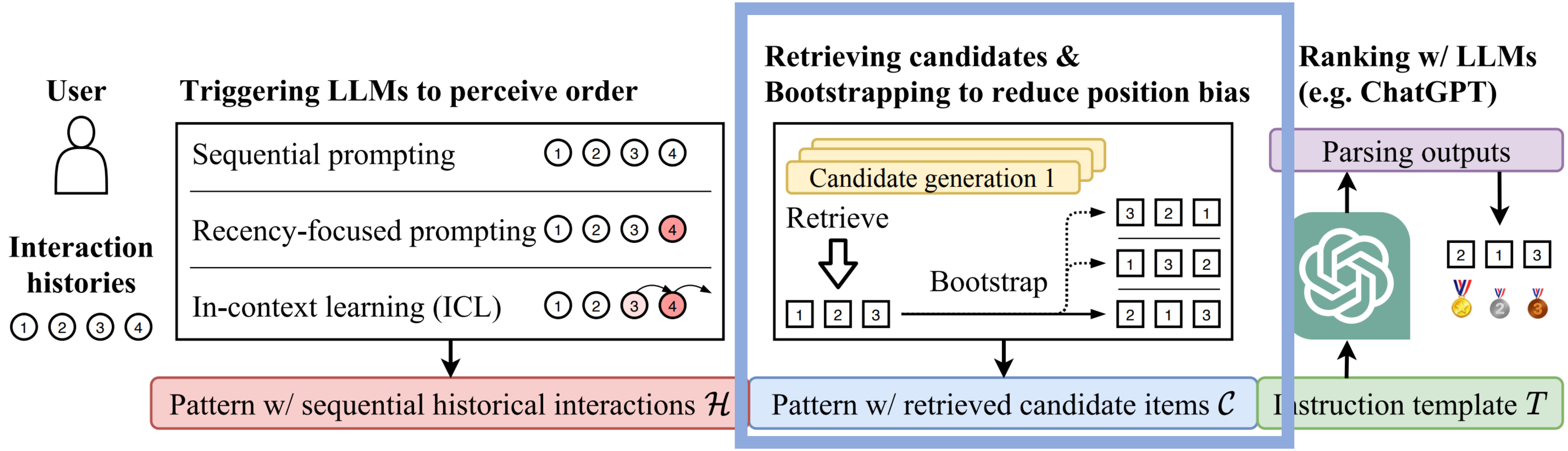
Uncovering ChatGPT's Capabilities in Recommender Systems, RecSys, 2023

ICL: LLMRank

❑ In-context learning: re-ranking given candidated items

❑ Task formulation:

- ***Using historical interaction to rank items retrieved by exsiting recsys.***
- ***Input:** language instructions created with historical interactions and candidate items*
- ***Output:** ranking of the candidate items*



Yupeng Hou et al. Large Language Models are Zero-Shot Rankers for Recommender Systems ECIR 2024

ICL: LLMRank

- **In-context learning: ranking given candidated items**
  - **Tree types of prompts:**

- **Sequential prompting: describing History using language**

  "I've watched the following movies in the past in order: '0. Multiplicity', '1. Jurassic Park', ..."

- **Recency-focused prompting**： **emphasize most recent interactions**

  "I've watched the following movies in the past in order: '0. Multiplicity', '1. Jurassic Park', ... Note that my most recently watched movie is Dead Presidents. . . ."

- **In-context learning (ICL)**： **providing recommendation example**

  " If I've watched the following movies in the past in order: '0. Multiplicity', '1. Jurassic Park', . . ., then you should recommend Dead Presidents to me and now that I've watched Dead Presidents, then . . ."

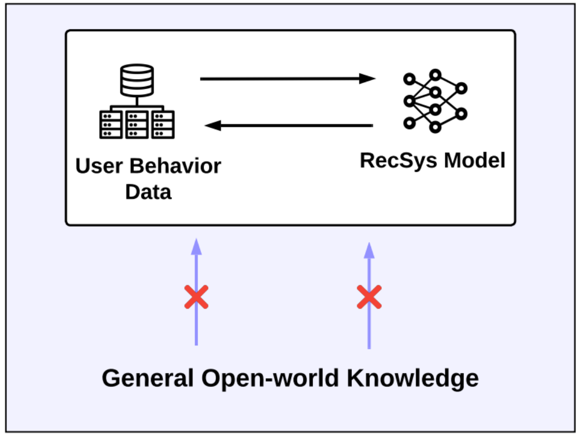
Yupeng Hou et al. Large Language Models are Zero-Shot Rankers for Recommender Systems ECIR 2024

ICL: KAR

In-context learning: knowledge enhancement

Traditional RecSys vs ICL-based RecSys

Traditional RecSys



Inference fast but being colsed system, generating recommendations relying on local dataset

Directly ask LLMs for recommendaiton

Given the user's historical interactions, please determine whether the user will enjoy the target new movie by answering "Yes" or "No".

Could **leverage open-world knowledge**, but:

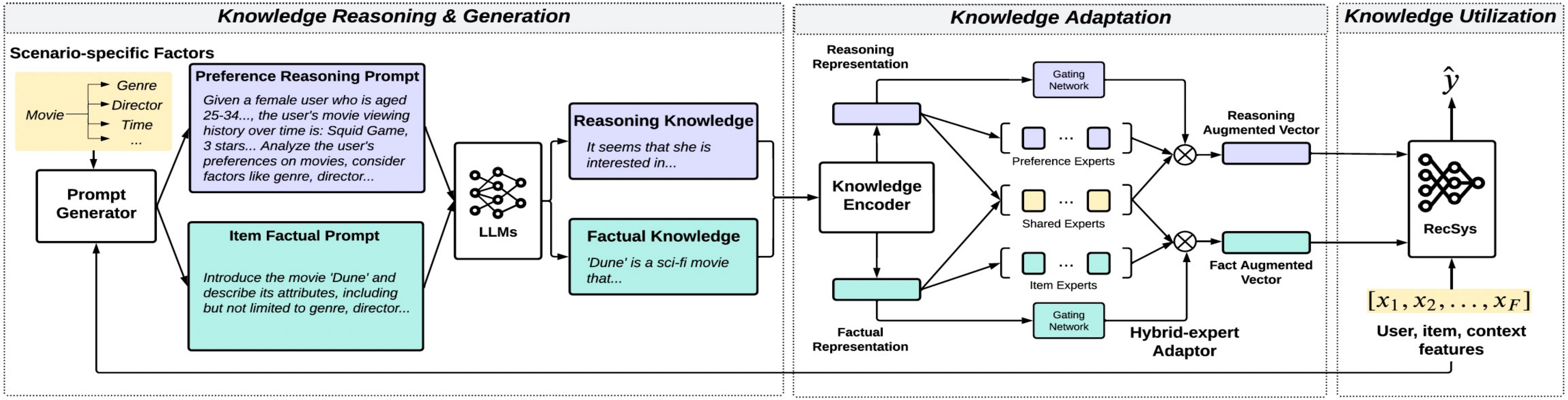
1) not trained on specific recommendation task
2) Inference slowly
3) hard to correctly answer compoitional questions

Extract and inject LLM's world knowledge into traditional recommender system

Xi Yunjia et al. "Towards Open-World Recommendation with Knowledge Augmentation from Large Language Models" arXiv 2023.

ICL: KAR

❑ In-context learning: knowledge enhancement



Obtain knowledge beyond local rec dataset：

1） Generate reasoning knowledge on user preference (factors affect preference)

2） Generate factual knowledge about items

Knowledge Adaptation Stage

encode the textual knowledge and maping it into recommendation space

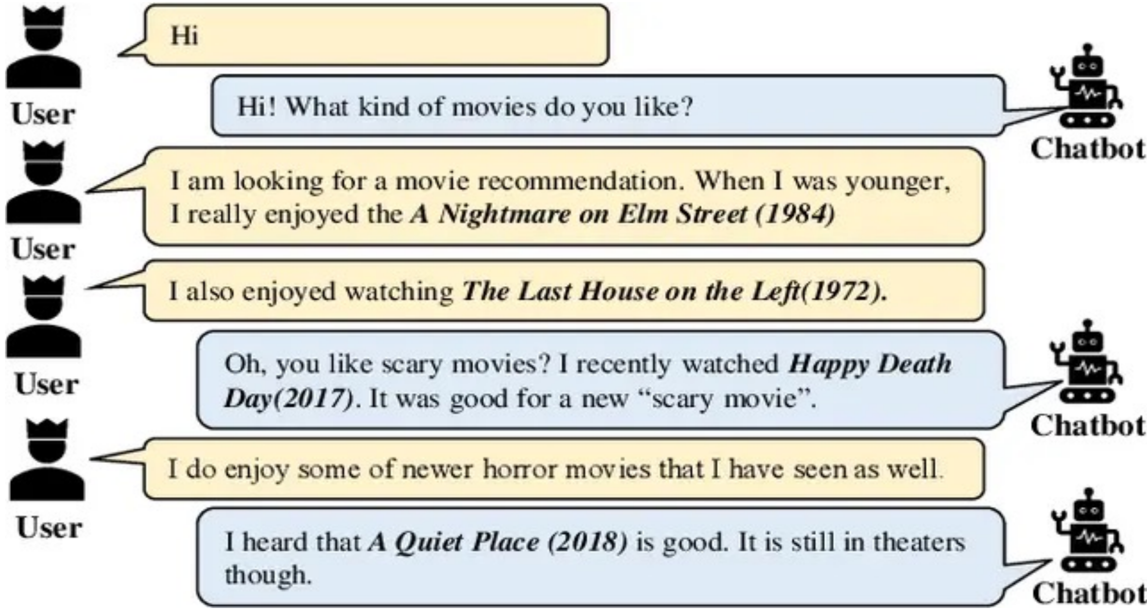
Knowledge Utilization

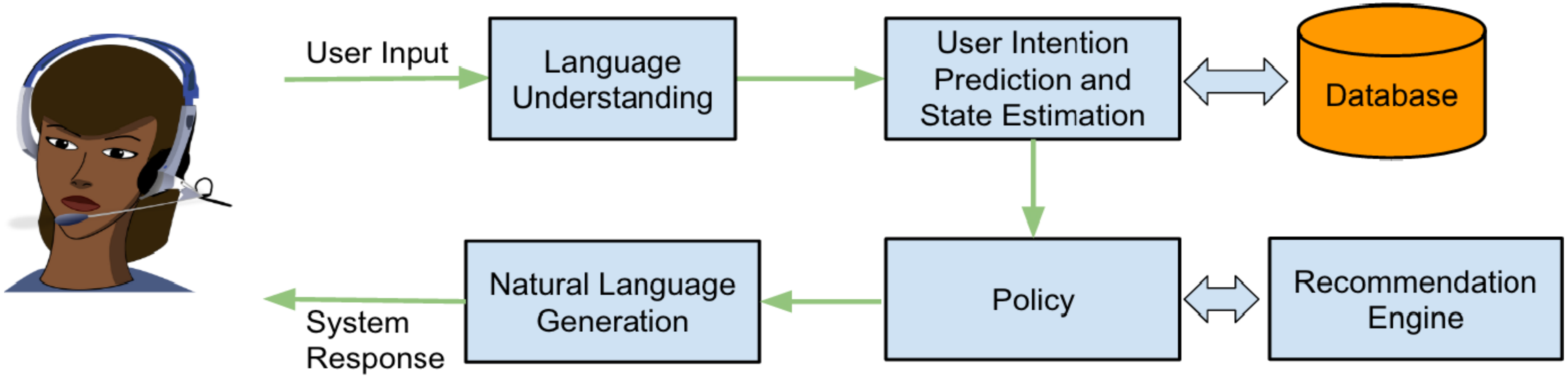
Use the knowledge obtained from LLMs as additional features

Xi Yunjia et al. "Towards Open-World Recommendation with Knowledge Augmentation from Large Language Models“ arXiv 2023.

ICL: KECRS

- ICL for conversational recommender system
  - Users chat with chatbot with natural language
  - Chatbot analyses user interest
  - Chatbot provide recommendation

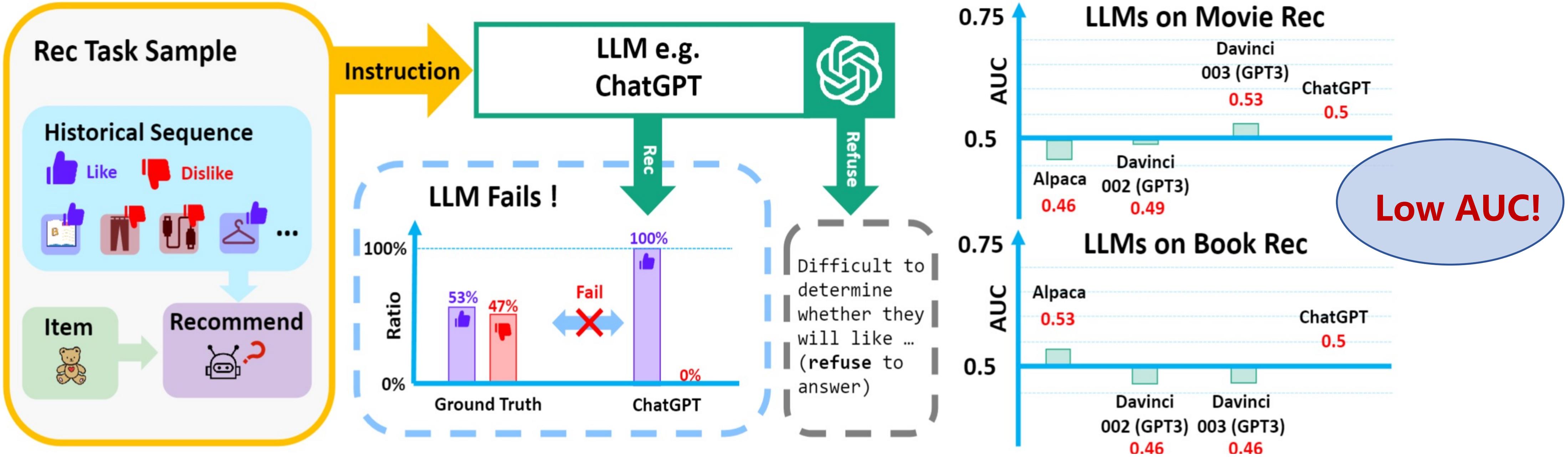




KECRS: Towards Knowledge-Enriched Conversational Recommendation System

ICL: might not be enough

- **In-context learning is not enough.**
- In complex scenarios, ChatGPT usually gives positive ratings or refuse to answer.



Need to **align** LLM with recommendation task!

Keqin Bao et al. Recsys, TALLRec: An Effective and Efficient Tuning Framework to Align Large Language Model with Recommendation. 2023

Outline

- Introduction
- Development of LLMs
- **Technical Stacks of LLM4Rec**
  - Model Architecture and Pre-training
  - **Model Post-training – accuracy (Yang Zhang, NUS postdoc)**
  - QA & Coffee Break
  - Model Post-training – efficiency and trustworthiness
  - Decoding and Deployment
- Open Problems
- Future Direction & Conclusions

Model Post-training

Align LLMs with recommendation tasks via post-training

Post-training

RLHF, DPO, SFT for (safety) alignment
RL for reasoning enhancement

Pre-training

Next-token prediction for Content understanding

Architecture

Self-attention Transformer

Large Foundation Model Stack

Align the model with recommendation task better

NA

Model expressiveness for recommendation

Post-training Behavior understanding, preference alignment and reasoning to pursue accuracy, efficiency, and trustworthiness.

Pre-training

N/A

Architecture

Item tokenizer, Memory for LLMs

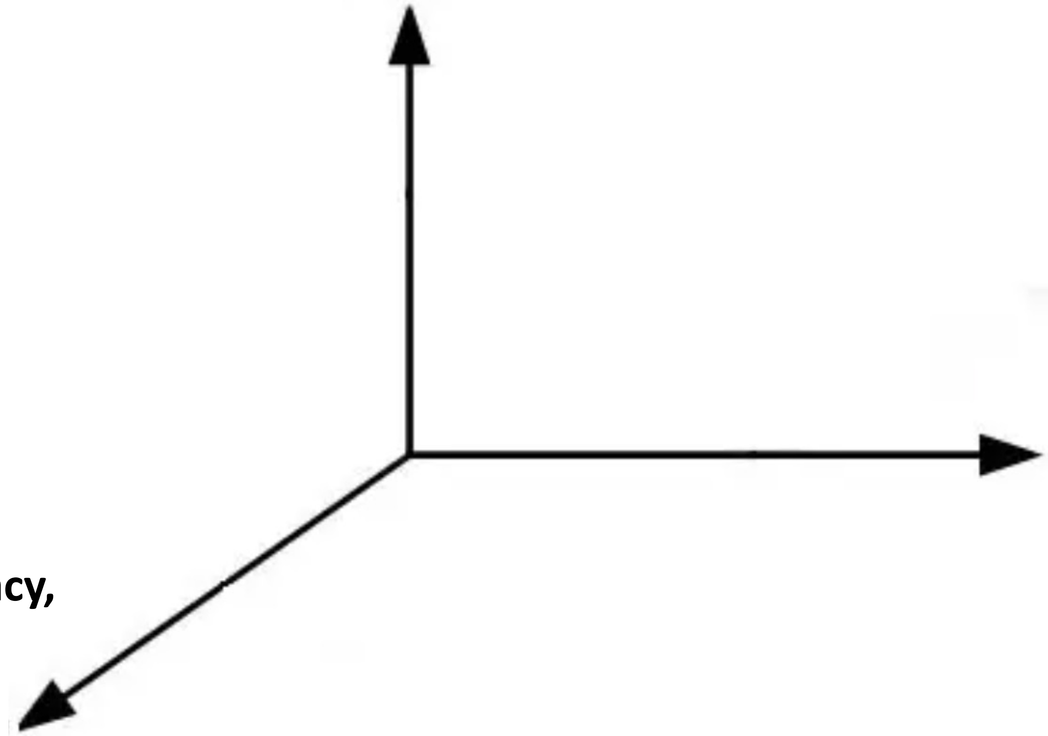
Personalized LLM/LFM Stack

Model Post-training

Three dimensions:

Accuracy

Learn to capture user preference and generate items for accurate recommendation



Efficiency

Data-efficient, parameter-effici post-training, etc

Trustworthiness

Beyond accuracy such as privacy, fairness, etc.

Post-training: Accuracy Perspective

Motivation: lack of recommendation task tuning in LLM pre-training

→ tune LLMs with the recommendation data to align with the recommendation task better

Research directions:

2023.10 | 2024.06 | now

Task alignment

Collaborative Info-focused Alignment

Reasoning-focused alignment

Reformulate rec task as language task:

TALLRec@RecSys'23, BIGRec@ArXiv'23 (TORS), InstructRec@TOIS, TRSR@WWW'24, TransRec@KDD'24 IGD@arXiv'25

Inject collaborative filtering signals into LLMs:

CoLLM@ArXiv'23(TKDE), LC-Rec@ICDE'24, LLaRa@SIGIR24, BinLLM@ACL'24, A-LLMRec@KDD'24, Cora@AAAI'25...

Enhance recommendation-specific reasoning

RecSAVER@ACL'24, ReasoningRec@ArXiv'24, LatentR^3@ArXiv'25, Reason4Rec@AriXiv25, R^{2}Rec@ArXiv25, ReaRec@ArXiv'25

Post-training: Accuracy Perspective

Direction 1: Task alignment – perform rec task as language task better

Early: Recommendation paradigm (2022.12-2024)

Discriminative manner

Following **traditional rec task**,
provide candidates:
pointwise, pairwise, listwise

TALLRec@RecSys'23

Generative manner

Following **the pretraining task**,
do not provide candidates:
directly generate items

BIGRec@TORS, InstructRec@TOIS,
TransRec@KDD'24

Later: Rec-specific focus (2024-now)

Aligning with Rec-specific Focus

Enhance **long sequence** modeling;
Enhance **rec-specific token learning**
Adapt to **dynamic preference**

TRSR@WWW'24, MSL@ArXiv'25
IGD@ArXiv'25, RecICL@ACL'25

Task Alignment: Discriminative Formulation

❑ TALLRec: Instruction-tuning to predict preference for target items

❑ Task formulation:

Like or not

↑

LLM with LoRA

↑

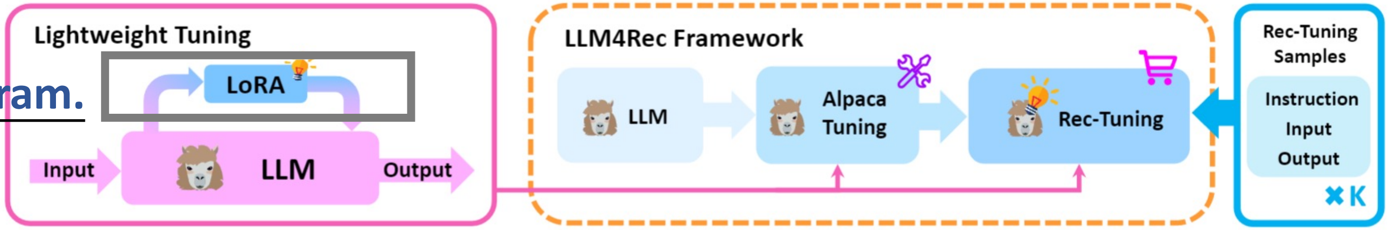
User his + item features (in language)

- Use target item (y) as the input

$$\max_{\Theta} \sum_{(x,y)\in\mathcal{Z}} \sum_{t=1}^{|y|} \log\left(P_{\Phi+\Theta}(y_t|x, y_{<t})\right),$$

❑ Tuning implementation:

4M param.



7B LLAMA

Task Alignment: Discriminative Formulation

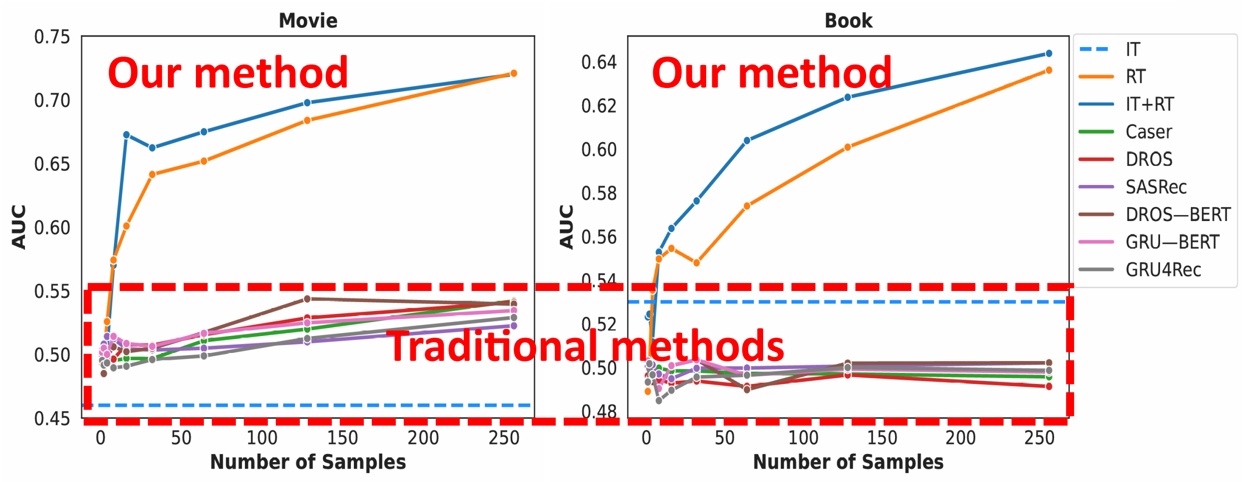
❑ TALLRec: Instruction-tuning to predict preference for target items

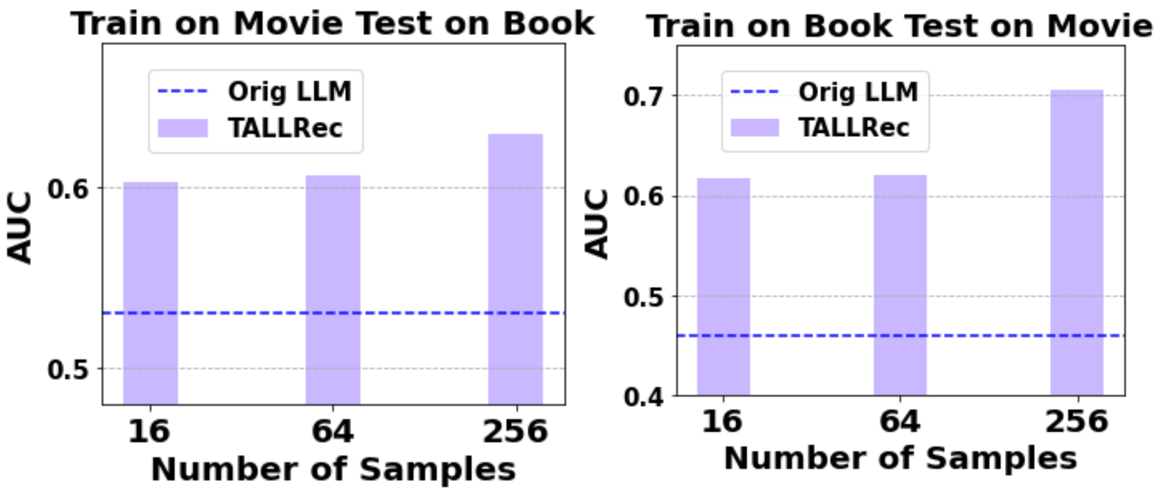
- **Good few-shot learning ability**

  Performance significantly improves by fine-tuning few-shot samples

- **Good cross-domain performance**

  Learning from movie scenario can directly recommend on books, and vice versa





Keqin Bao et al. Recsys, TALLRec: An Effective and Efficient Tuning Framework to Align Large Language Model with Recommendation. 2023

Task Alignment: Generative Formulation

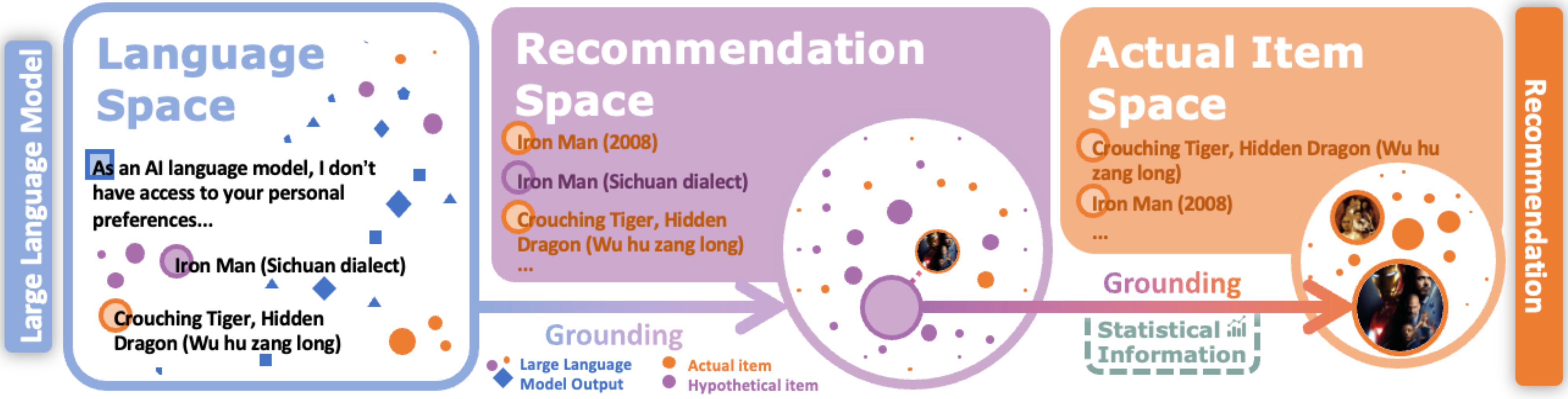
- **Task Formulation:** User his (in language) → LLM $\xrightarrow{\text{generate}}$ Next item
- **BIGRec: Bi-step grounding solution**

Challenges:

- LLMs do not know how to represent an item via token sequence in the recommendation scenario.
- Besides, the item generated by the LLM may not exist in **the actual world**.

- **Two-step Grounding Solution**

Language Space $\xrightarrow{\textit{Step1: instruction tuning}}$ Recommendation Space $\xrightarrow{\textit{Step2: L2 distance grounding}}$ Actual Item Space



Bao Keqin et al. " A Bi-Step Grounding Paradigm for Large Language Models in Recommendation Systems". ACM TORS.

Task Alignment: Generative Formulation

❑ BIGRec

- Few-shot tuning

Dataset	Model	NG@1	NG@3	NG@5	NG@10	NG@20	HR@1	HR@3	HR@5	HR@10	HR@20
Movie	GRU4Rec	0.0015	0.0034	0.0047	0.0070	0.0104	0.0015	0.0047	0.0079	0.0147	0.0281
	Caser	0.0020	0.0035	0.0052	0.0078	0.0109	0.0020	0.0046	0.0088	0.0171	0.0293
	SASRec	0.0023	0.0051	0.0062	0.0082	0.0117	0.0023	0.0070	0.0097	0.0161	0.0301
	P5	0.0014	0.0026	0.0036	0.0051	0.0069	0.0014	0.0035	0.0059	0.0107	0.0176
	DROS	0.0022	0.0040	0.0052	0.0081	0.0112	0.0022	0.0051	0.0081	0.0173	0.0297
	GPT4Rec-LLaMA	0.0016	0.0022	0.0024	0.0028	0.0035	0.0016	0.0026	0.0030	0.0044	0.0074
	BIGRec (1024)	**0.0176**	**0.0214**	**0.0230**	**0.0257**	**0.0283**	**0.0176**	**0.0241**	**0.0281**	**0.0366**	**0.0471**
	Improve	**654.29%**	**323.31%**	**273.70%**	**213.71%**	**142.55%**	**654.29%**	**244.71%**	**188.39%**	**111.97%**	**56.55%**
Game	GRU4Rec	0.0013	0.0016	0.0018	0.0024	0.0030	0.0013	0.0018	0.0024	0.0041	0.0069
	Caser	0.0007	0.0012	0.0019	0.0024	0.0035	0.0007	0.0016	0.0032	0.0048	0.0092
	SASRec	0.0009	0.0012	0.0015	0.0020	0.0025	0.0009	0.0015	0.0021	0.0037	0.0057
	P5	0.0002	0.0005	0.0007	0.0010	0.0017	0.0002	0.0007	0.0012	0.0023	0.0049
	DROS	0.0006	0.0011	0.0013	0.0016	0.0022	0.0006	0.0015	0.0019	0.0027	0.0052
	GPT4Rec-LLaMA	0.0000	0.0000	0.0000	0.0001	0.0001	0.0000	0.0000	0.0000	0.0002	0.0002
	BIGRec (1024)	**0.0133**	**0.0169**	**0.0189**	**0.0216**	**0.0248**	**0.0133**	**0.0195**	**0.0243**	**0.0329**	**0.0457**
	Improve	**952.63%**	**976.26%**	**888.19%**	**799.64%**	**613.76%**	**952.63%**	**985.19%**	**660.42%**	**586.11%**	**397.10%**

- BIGRec significantly surpasses baselines by few-shot tuning.

- Improvement of BIGRec is significantly higher on Game compared to on Movie.

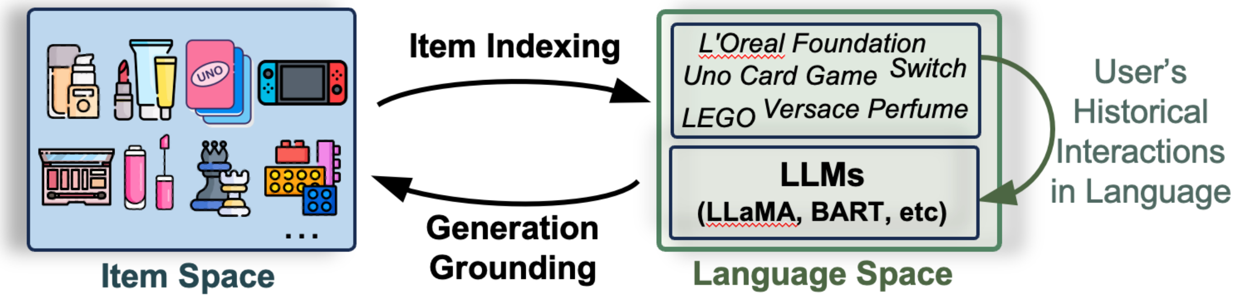
  - possibly due to the varying properties of popularity bias between the two datasets.

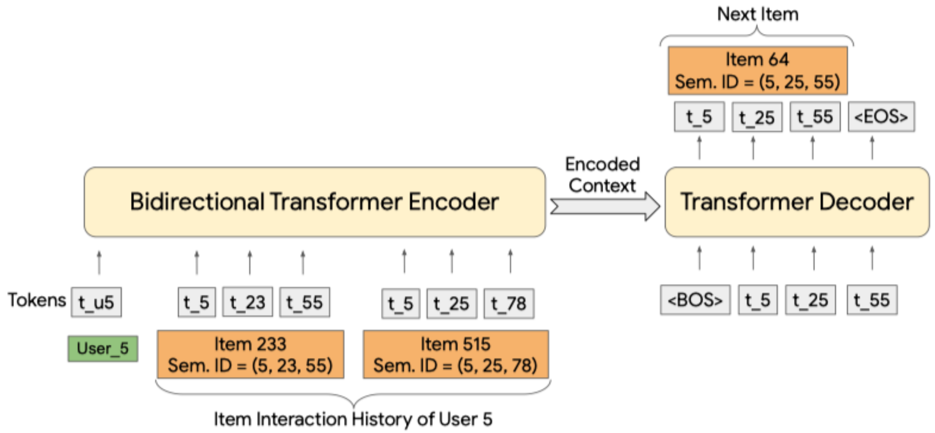
Bao Keqin et al. " A Bi-Step Grounding Paradigm for Large Language Models in Recommendation Systems" ACM TORS.

Task Alignment: Generative Formulation

❑ TransRec

LLM for generative recommendation





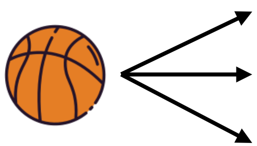
- Two key problems of LLM4Rec
  - Item tokenization: index items into language space
  - Item generation: generate items as recommendations

Lin Xinyu et al. " A Multi-facet Paradigm to Bridge Large Language Model and Recommendation " KDD 2024.

Task Alignment: Generative Formulation

❑ TransRec

- **Item indexing: multi-facet identifier**



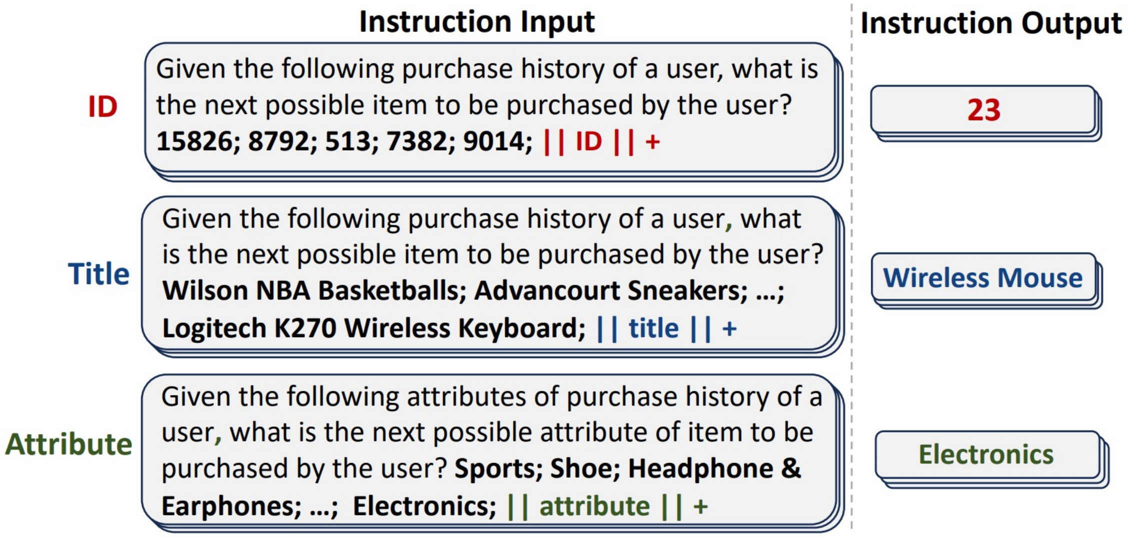
ID 15826 **(Distinctiveness)**

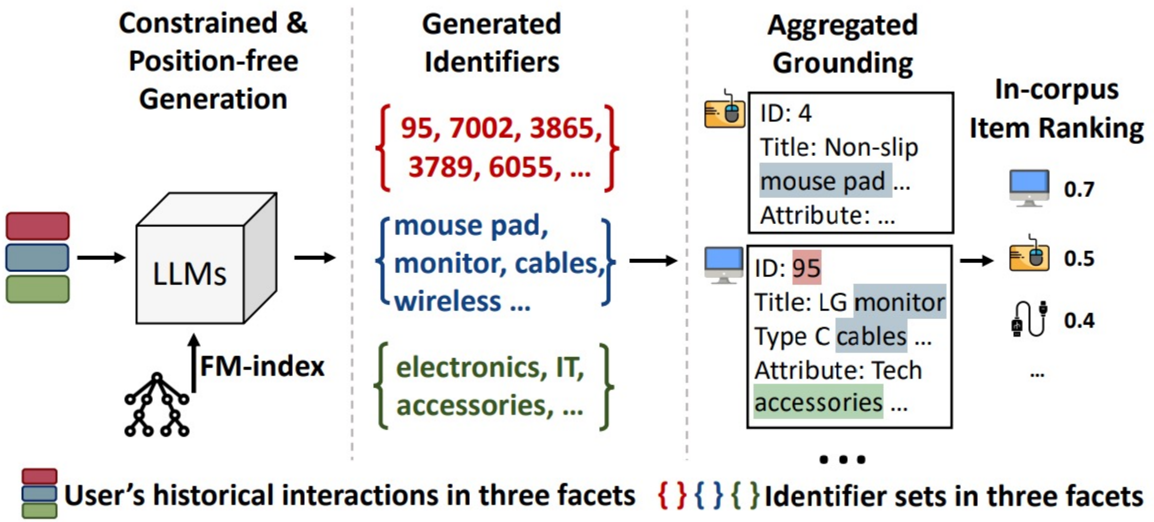
Title Wilson Indoor Basketballs
Attribute Sports
(Semantics)

- **Generation grounding:**
  - Position-free constrained generation

FM-index: special prefix tree that supports search from any position of the identifier corpus.

- **Instruction data reconstruction**



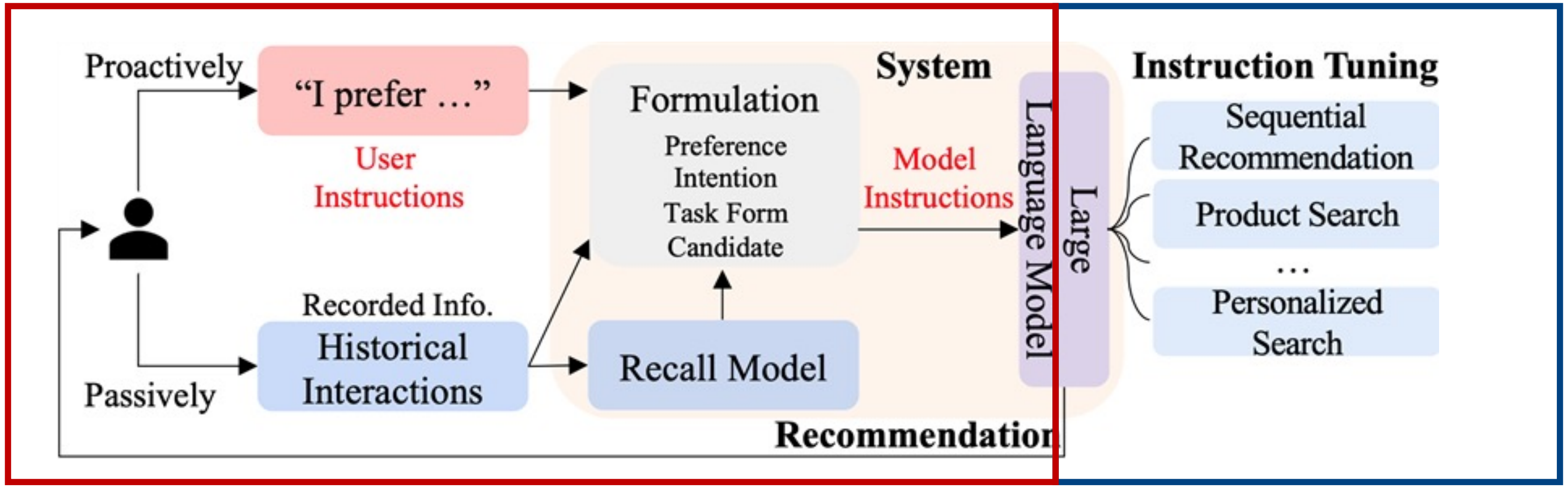


Lin Xinyu et al. " A Multi-facet Paradigm to Bridge Large Language Model and Recommendation " KDD 2024.

Task Alignment: Unified Formulation

❑ InstructRec

- User could express their needs diversely: vague or specific; implicit or explicit
- LLM should understand and follow different instructions for recommendation



Recommendation instruction construction

Instruction tuning: tuning LLMs with the instruction data

Junjie Zhang et al. Recommendation as Instruction Following: A Large Language Model Empowered Recommendation Approach. TOIS.

Task Alignment: Unified Formulation

❑ InstructRec: Instruction construction:

- **Format:** Preference: none/implicit/explicit   Intention: none/vague/specific   task: pointwise/pairwise/listwise

Instantiation	Model Instructions
$\langle P_1, I_0, T_0 \rangle$	The user has purchased these items: <historical interactions> . Based on this information, is it likely that the user will interact with <target item> next?
$\langle P_2, I_0, T_3 \rangle$	You are a search engine and you meet a user's query: <explicit preference> . Please respond to this user by selecting items from the candidates: <candidate items>.
$\langle P_0, I_1, T_2 \rangle$	As a recommender system, your task is to recommend an item that is related to the user's <vague intention> . Please provide your recommendation.
$\langle P_0, I_2, T_2 \rangle$	Suppose you are a search engine, now the user search that <specific Intention> , can you generate the item to respond to user's query?
$\langle P_1, P_2, T_2 \rangle$	Here is the historical interactions of a user: <historical interactions> . His preferences are as follows: <explicit preference> . Please provide recommendations .
$\langle P_1, I_1, T_2 \rangle$	The user has interacted with the following <historical interactions> . Now the user search for <vague intention> , please generate products that match his intent.
$\langle P_1, I_2, T_2 \rangle$	The user has recently purchased the following <historical items>. The user has expressed a desire for <specific intention>. Please provide recommendations.

- **Instruction generation: #1** using ChatGPT to generate user preferences and intentions based on interactions

Interaction
↓
Explicit preference

[Raw Behavior Sequence]:
"1. Resident Evil: Revelations 2 - PS 4
→ 2. Resident Evil 4 - PS 4."
[Generated Explicit Preference]:
"He prefers horror-based games with a strong narrative."

[Raw Target Review]:
"My son loves ... of the game. I'm happy I bought this for him."
[Generated Vague Intention]:
"I enjoy buying games for my son that he enjoys."

review
↓
vague intention

#2 Increasing the instruction diversity via multiple strategies such as CoT

Junjie Zhang et al. Recommendation as Instruction Following: A Large Language Model Empowered Recommendation Approach. TOIS.

Task Alignment: Unified Formulation

InstructRec

- **Instruction construction**
  - **Quality: human evaluation**

Statistic	
# of fine-grained instructions	252,730
- # of user-described preferences	151,638
- # of user intention in decision making	101,092
ave. instruction length (in words)	23.5
# of coarse-grained instructions	39
- # of preferences related instructions	17
- # of intentions related instructions	9
- # of combined instructions	13
ave. instruction length (in words)	41.4

Quality Review Question	Preference	Intention
Is the instruction generated from the user's related information?	93%	90%
Does the teacher-LLM provide related world knowledge?	87%	22%
Does the instruction reflect the user's preference/ intention?	88%	69%
Is the instruction related to target item?	48%	69%

- **Instruction tuning:**
  - **Supervised fine-tuning, tuning all model parameters (3B Flan-T5-XL)**

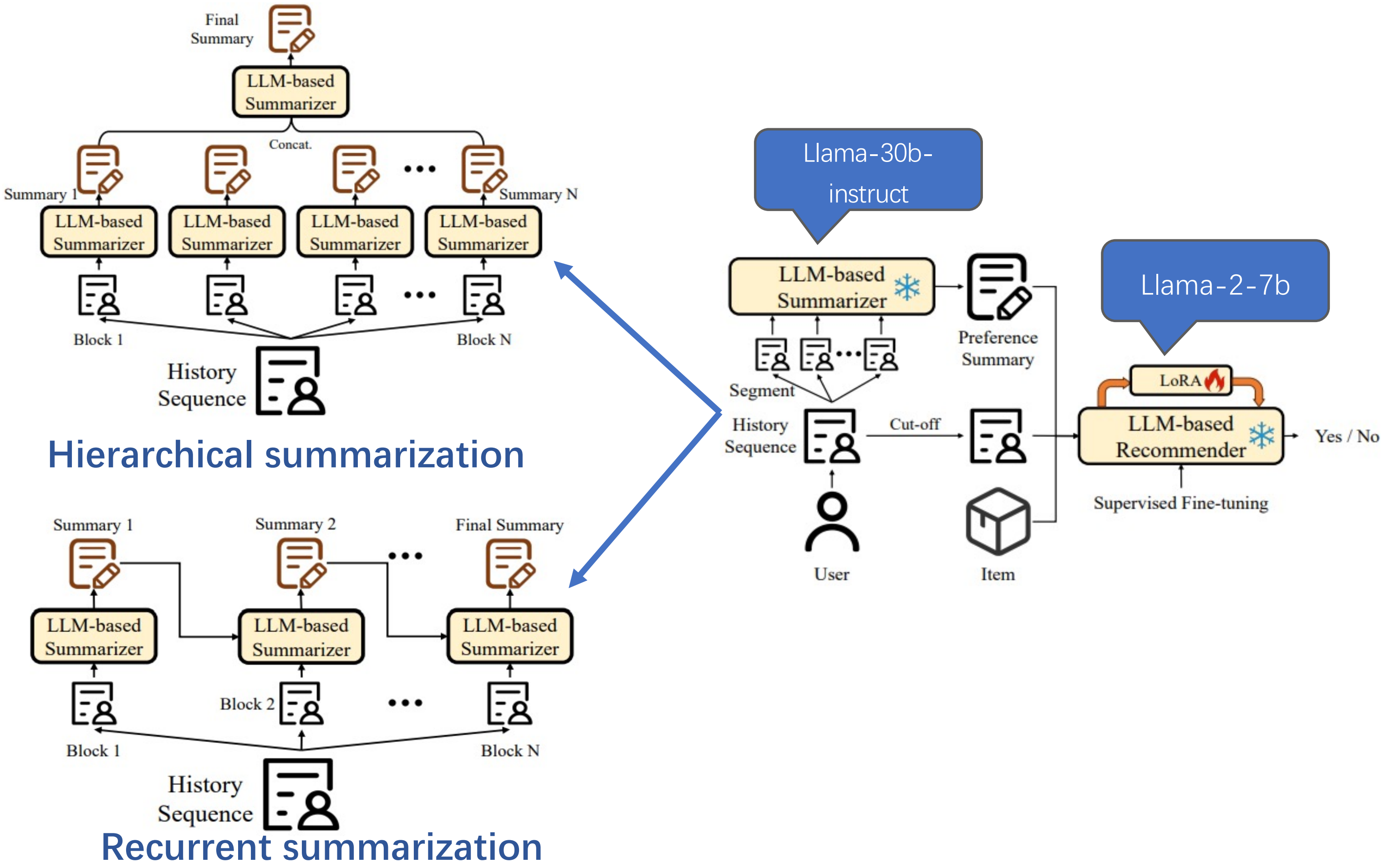
$$\mathcal{L} = \sum_{k=1}^{B} \sum_{j=1}^{|Y_k|} \log P\left(Y_{k,j} \mid Y_{k,<j}, I_k\right), \qquad (1)$$

where Y_k is the desired system responses for the k-th instance, I_k is the instruction of the k-th instance, and B is the batch size.

Task Alignment: Long History

❑ TRSR: Text-Rich Sequential Recommendation

- Use summarization to deal long history
- LLM for preference summary
  - Hierarchical summarization
  - Recurrent summarization
- Supervised fine-tuning
  - Given user preference summary, recently interacted items, and candidate items, LLMs are tuned for recommendation



Zhi Zheng et al. Harnessing Large Language Models for Text-Rich Sequential Recommendation. WWW 2024

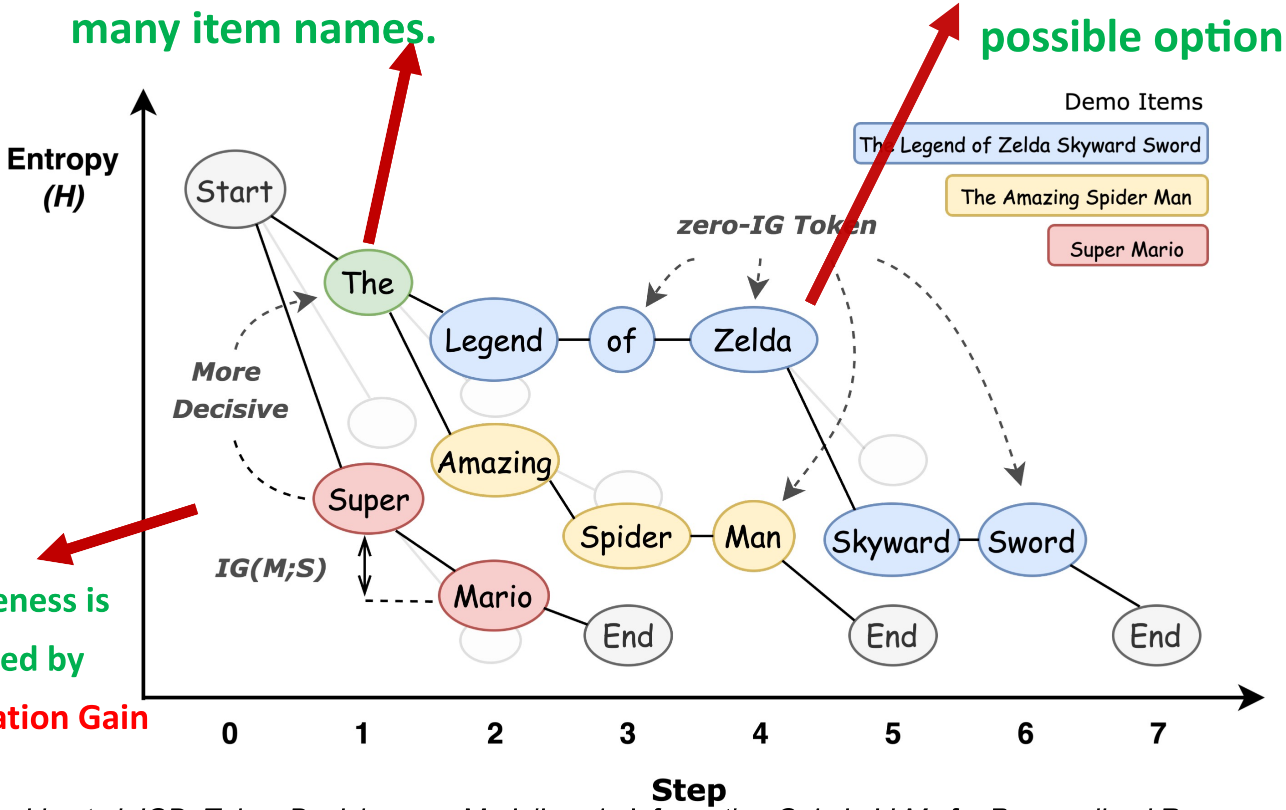
Task Alignment: Item Token Learning

IGD: Address Over-optimization of Less-decisive Tokens in LLM4Rec

- LLM4Rec optimizing token likelihood without considering token importance.

Token “The” is less decisive than “Super” for identifying an item, as it appears in many item names.

Tokens “of,” “Zelda,” and “Man” do not help to decide an item in the item name space, as each only has one possible option.



Tokens decisiveness is measured by Information Gain

- Token generation can be modeled as a decision process, with token decisiveness quantified by **information gain (IG).**
- Low-IG tokens, especially those with zero-IG (i.e., less-decisive tokens), are prone to over-optimization.

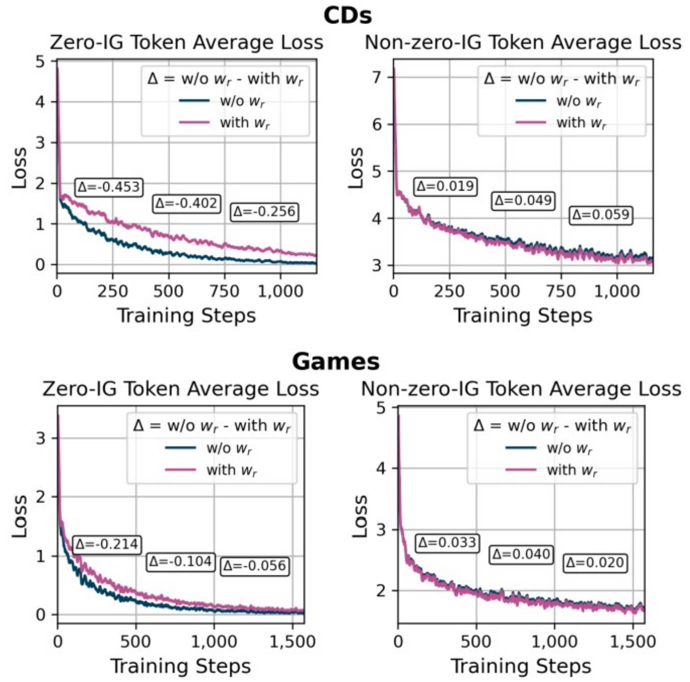
Lin et al. IGD: Token Decisiveness Modeling via Information Gain in LLMs for Personalized Recommendation. ArXiv 2025

Task Alignment: Item Token Learning

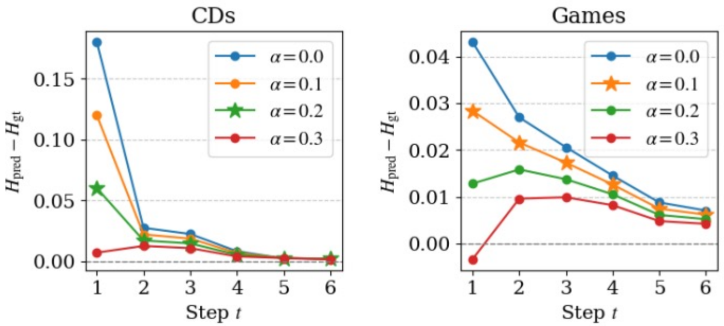
IGD: Address Over-optimization of Less-decisive Tokens in LLM4Rec

- IGD reweights tokens during tuning and inference according to token IG.
- Consistently improves recommendation performance.

Methods	CDs				Games			
	N@5	N@10	H@5	H@10	N@5	N@10	H@5	H@10
GRU4Rec	0.0248	0.0288	0.0342	0.0467	0.0169	0.0221	0.0261	0.0423
SASRec	0.0477	0.0535	0.0647	0.0824	0.0237	0.0290	0.0338	0.0502
BIGRec	0.0502	0.0553	0.0623	0.0782	0.0317	0.0381	0.0430	0.0631
+Pos	0.0511	0.0566	0.0632	0.0802	0.0319	0.0396	0.0423	0.0665
+CFT	0.0509	0.0566	0.0631	0.0810	0.0349	0.0414	0.0482	0.0686
+IGD	**0.0540**	**0.0593**	**0.0669**	**0.0833**	**0.0423**	**0.0507**	**0.0576**	**0.0833**
Improvement	*+7.78%*	*+7.82%*	*+9.33%*	*+9.04%*	*+33.4%*	*+33.1%*	*+34.0%*	*+32.0%*
D3	0.0716	0.0767	0.0882	0.1040	0.0415	0.0477	0.0581	0.0773
+Pos	0.0729	0.0779	0.0902	0.1053	0.0429	0.0489	0.0581	0.0767
+CFT	0.0736	0.0786	0.0917	0.1069	0.0437	0.0499	0.0613	0.0806
+IGD	**0.0748**	**0.0801**	**0.0929**	**0.1092**	**0.0518**	**0.0598**	**0.0705**	**0.0946**
Improvement	*+4.47%*	*+4.43%*	*+5.33%*	*+5.00%*	*+25.6%*	*+29.2%*	*+26.7%*	*+22.7%*



Encourage LLMs to focus on high-IG (decisive) tokens during tuning.



Align IG of predicted tokens with ground truth at each decoding step.

Lin et al. IGD: Token Decisiveness Modeling via Information Gain in LLMs for Personalized Recommendation. ArXiv 2025

Task Alignment: Dynamic Preference

Background

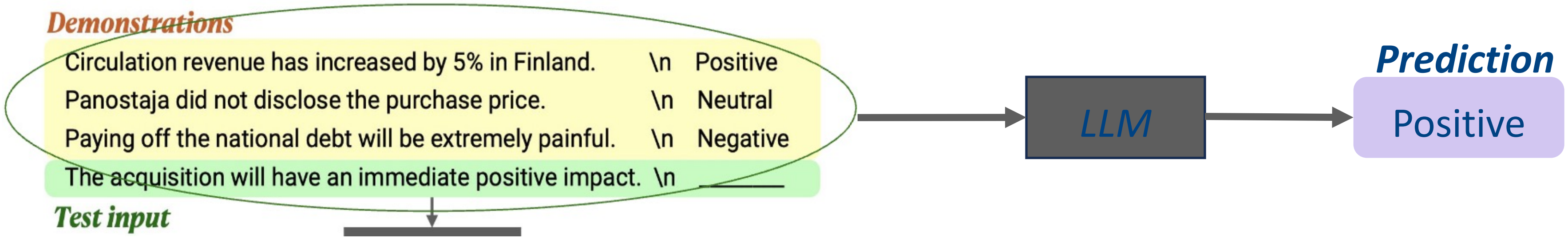
- User preference would drift with time going

Objective

- **How to effectively align to user dynamic prefernece**

Motivation

- ICL enables **learning new tasks without retraining**—can it also capture evolving preferences to eliminate the retraining costs?



Challenge

- Original LLMs have the ICL abilities but not the personalization abilities
- LLMs aligned to user preference with existing methods usually lose the ICL abilities

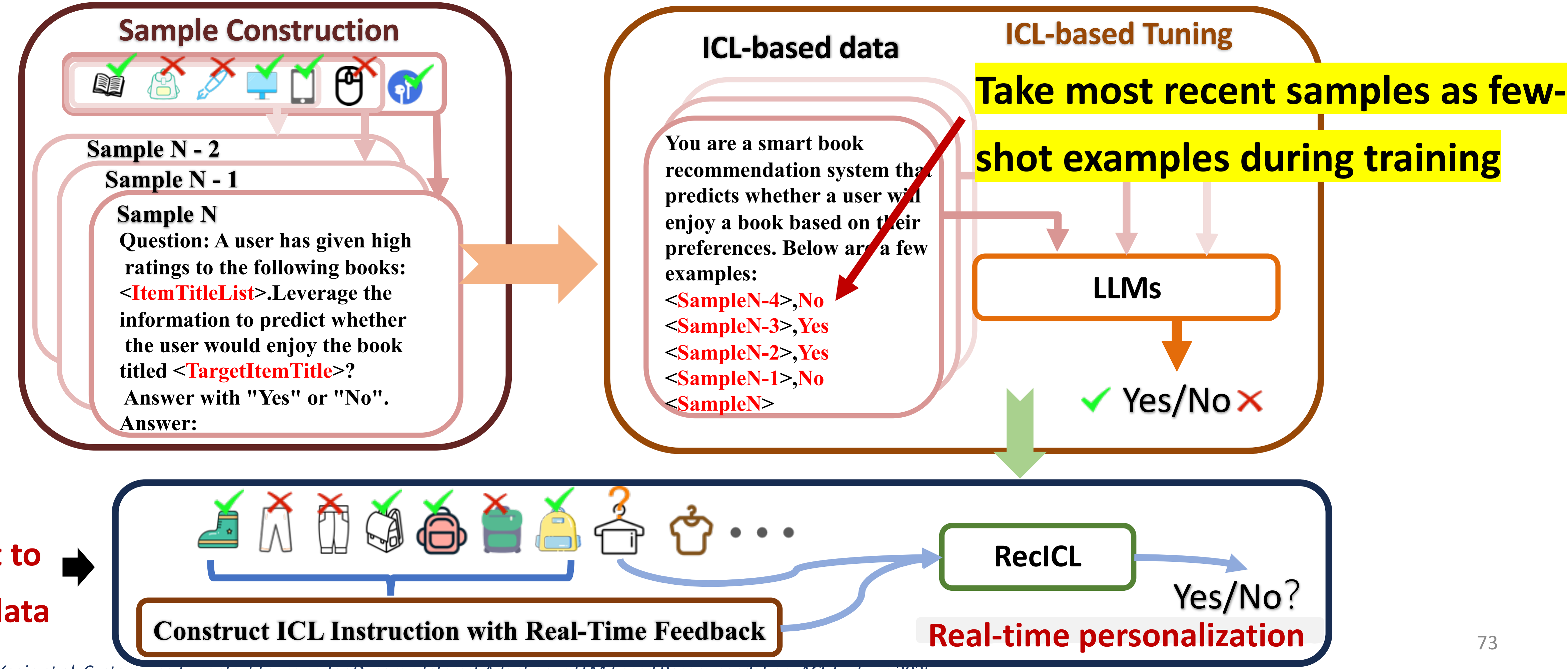
Bao Keqin et al. Customizing In-context Learning for Dynamic Interest Adaption in LLM-based Recommendation. ACL findings 2025

Task Alignment: Dynamic Preference

Solution

RecICL: Perform alignment in an ICL-tuning manner

- Align to personalized tasks while preserving the ICL capability



Adapt to new data

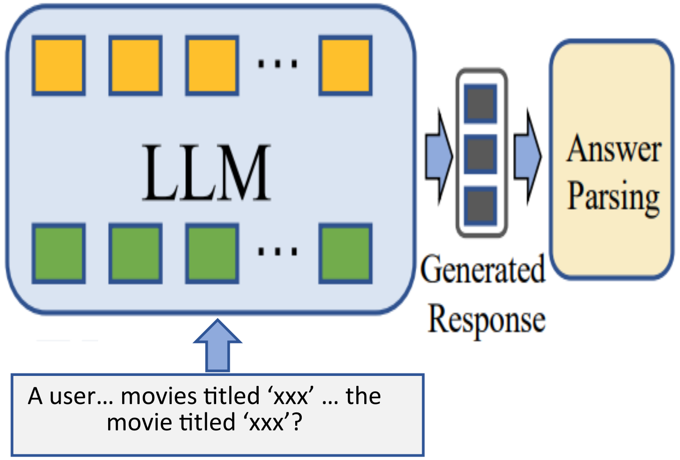
Bao Keqin et al. Customizing In-context Learning for Dynamic Interest Adaption in LLM-based Recommendation. ACL findings 2025

Post-training: Accuracy Perspective

❑ Direction 2: Collaborative Filtering Info-focused Alignment （CF-focused Alignment）

LLM4Rec methods

User/Item: Text

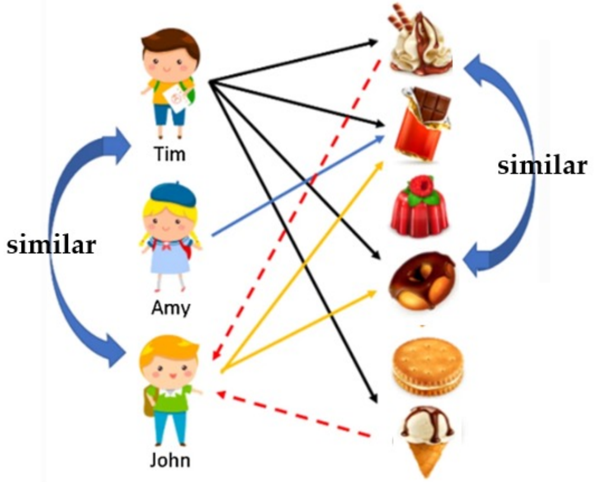


May lack of some information

Textually similar item may have distinct collab. info.

Traditional methods

User/Item: features + ID



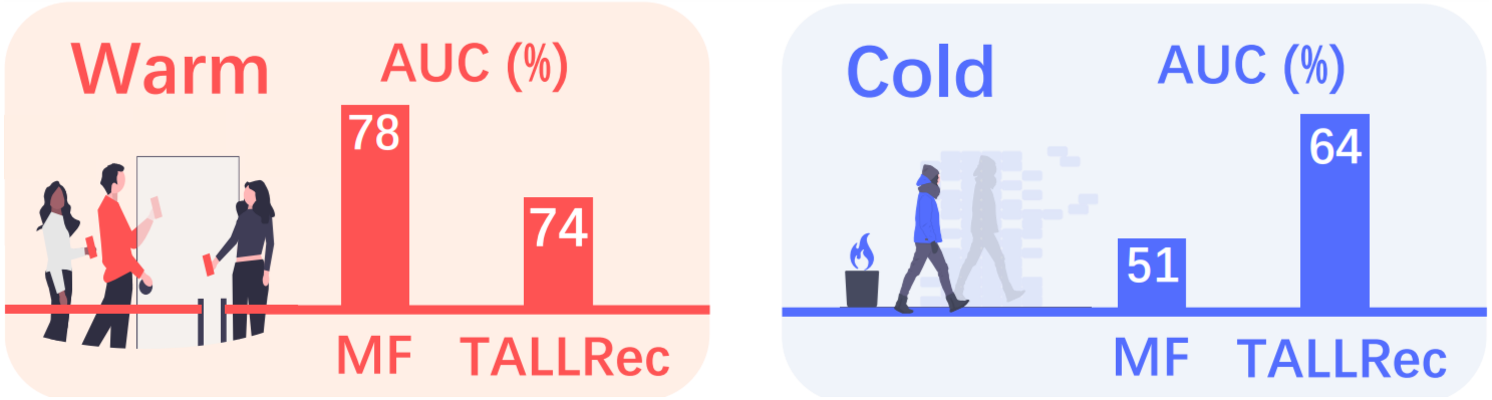
LLMs are constructed using texts, making the representation of users/items in texts the natural choice.

Features (content) alone are insufficient to depict users and items, mainly behavioral similarities (collaborative info). IDs are utilized.

CF-focused Alignment

Integrate collaborative information:

- **Why?**



LLM Rec vs Traditional CF Model：

#:Excellent at old-start scenarios #: Poor at warm-start scenarios

Zhang et al. CoLLM: Integrating Collaborative Embeddings into Large Language Models for Recommendation. ArXiv 2023.

CF-focused Alignment

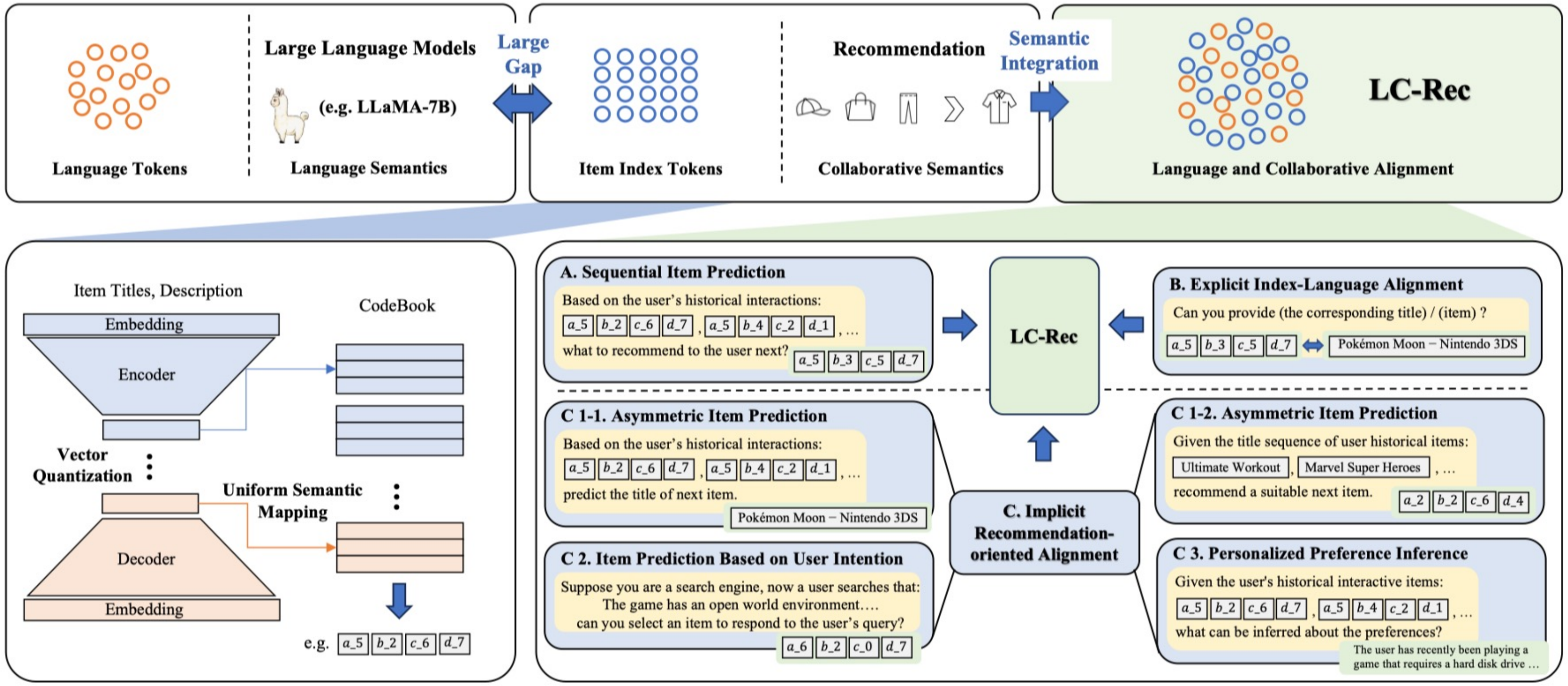
❑ Technical directions：

ID-based method	Feature-based method	Parameter-based method
Following **methods like MF, add ID to represent items, and learn ID embedding to encode CF info by fitting interactions**	**Leverage external model to encode CF info, and treat the encoded CF info as features that the LLM can leverage**	**Leverage external CF info to customize some parameters, and merge them with the original LLM parameters**
LC-Rec@ICDE'24	CoLLM@TKDE, BinLLM@ACL'24	Cora@AAAI'25

CF-focused Alignment: ID-based

ID-based method: **learning collaborative information via ID embedding update**

- **LC-Rec**
  - Item indexing: utilize Residual-Quantized Variational AutoEncoder (RQ-VAE) to encode item semantic information as identifiers.
  - Multiple alignment tasks to inject collaborative signals



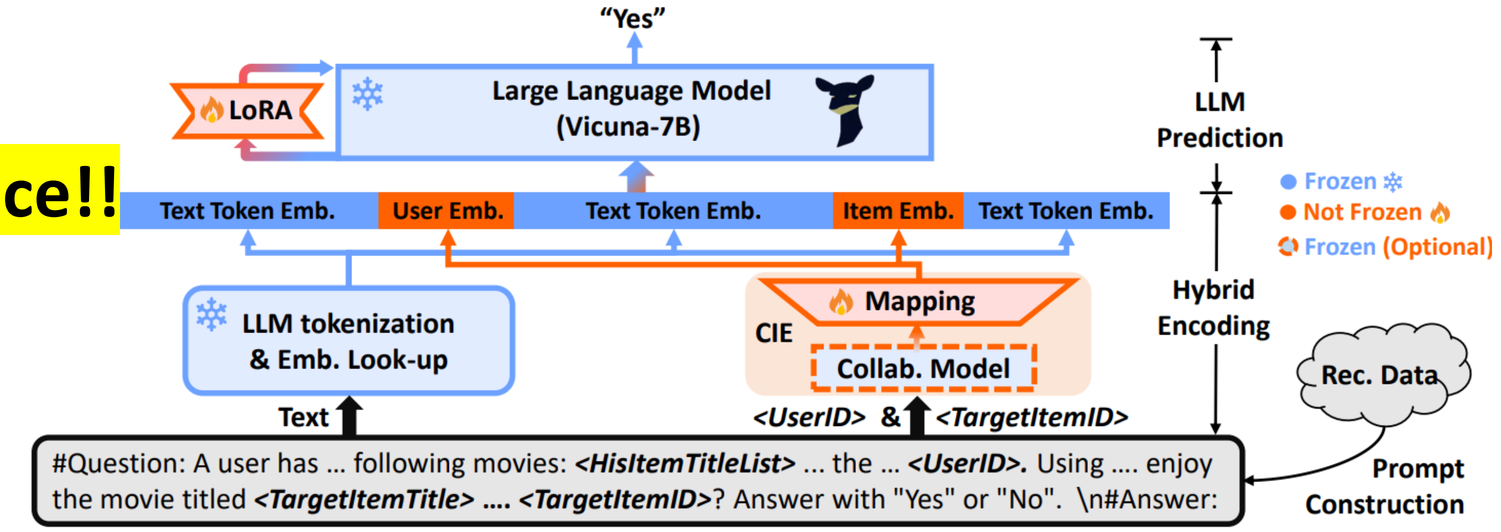
Bowen Zheng et al. "Adapting Large Language Models by Integrating Collaborative Semantics for Recommendation" ICDE 2024.

CF-focused Alignment: Feature-based

Feature-based method: feed external collaborative information into LLM

- **Work#1: CoLLM —— mapping collaborative embeddings into LLM's Latent space**

Latent space!!



- Prompt construction: add <UserID> and <TargetID> for placing the Collab. Info.
- Hybrid Encoding:
  - text: tokenization & LLM emb Lookup;
  - user/item ID: CIE --- extract info with collab. model (low rank), then map it to the token embedding space
- LLM prediction: add a LoRA module for recommendation task learning

Yang Zhang et al. CoLLM: Integrating Collaborative Embeddings into Large Language Models for Recommendation. ArXiv 2023.

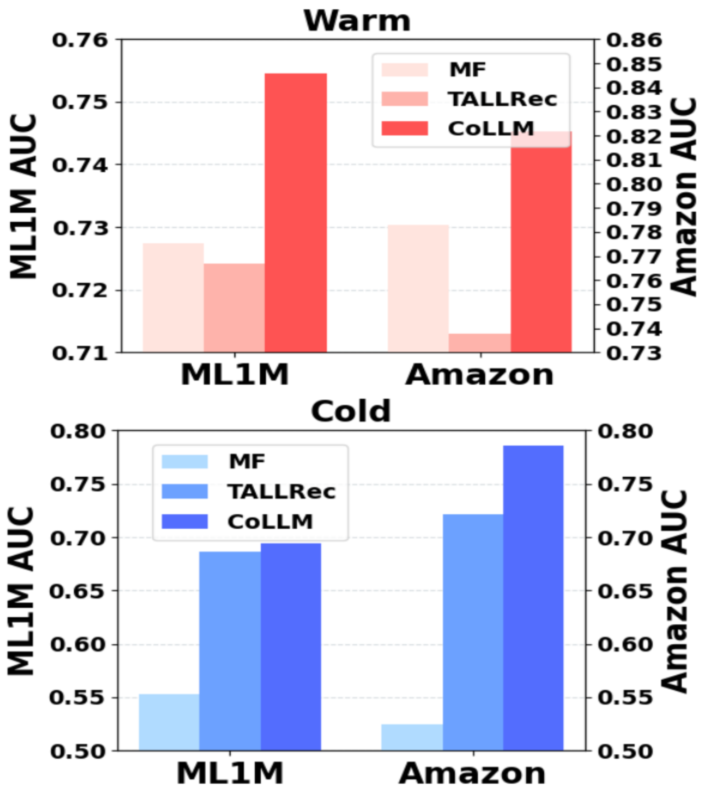
CF-focused Alignment: Feature-based

Feature-based method: feed external collaborative information into LLM

- **Work#1: CoLLM —— mapping collaborative embeddings into LLM's Latent space**

Overall Performance

Dataset		ML-1M			Amazon-Book		
Methods		AUC	UAUC	Rel. Imp.	AUC	UAUC	Rel. Imp.
Collab.	MF	0.6482	0.6361	10.3%	0.7134	0.5565	12.8%
	LightGCN	0.5959	0.6499	13.2%	0.7103	0.5639	10.7%
	SASRec	0.7078	0.6884	1.9%	0.6887	0.5714	8.4%
LLMRec	ICL	0.5320	0.5268	33.8%	0.4820	0.4856	48.2%
	Soft-Prompt	0.7071	0.6739	2.7%	0.7224	0.5881	10.4%
	TALLRec	0.7097	0.6818	1.8%	0.7375	0.5983	8.2%
Ours	CoLLM-MF	0.7295	0.6875	-	0.8109	0.6225	-
	CoLLM-LightGCN	0.7100	0.6967	-	0.7978	0.6149	-
	CoLLM-SASRec	0.7235	0.6990	-	0.7746	0.5962	-



- CoLLM brings performance improvements over traditional models and current LLM Rec in most cases

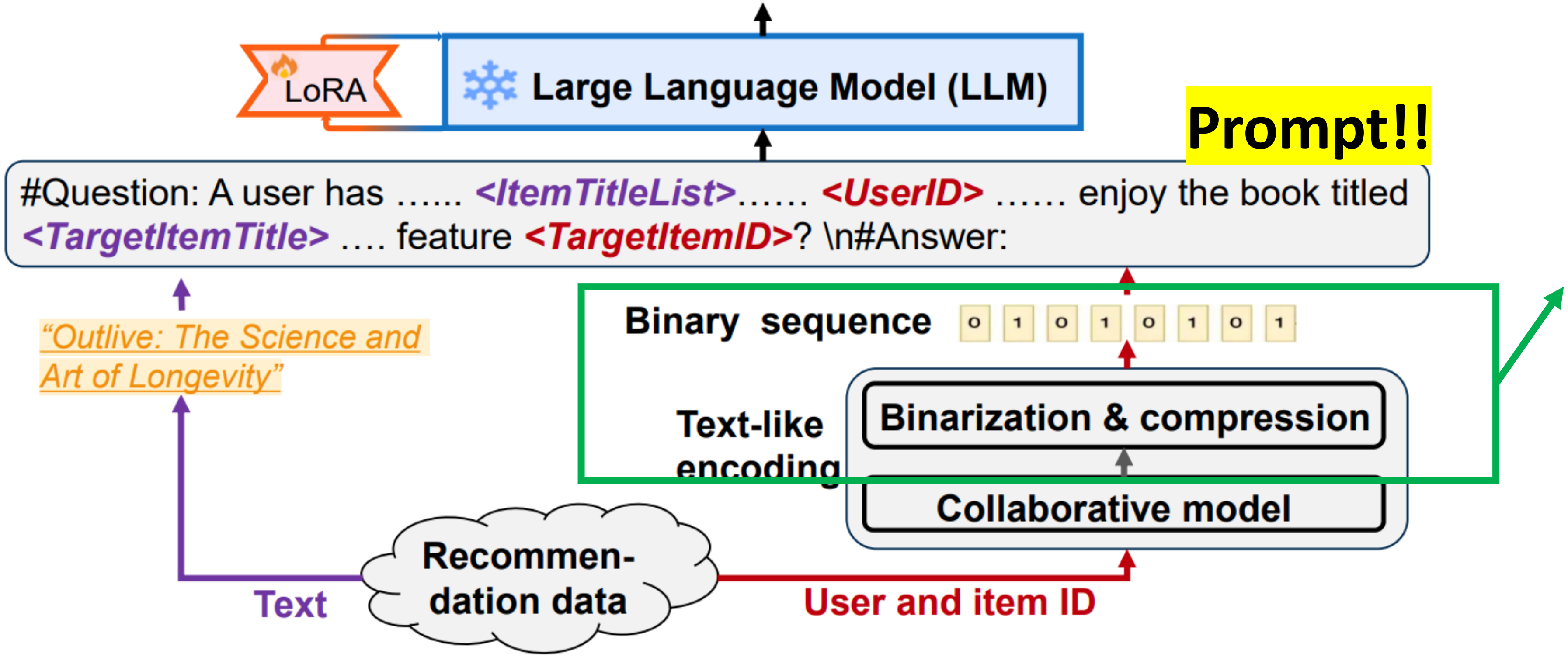
- CoLLM significantly improves the warm performance of LLM4Rec, while ensuring cold performance

Yang Zhang et al. CoLLM: Integrating Collaborative Embeddings into Large Language Models for Recommendation. ArXiv 2023.

CF-focused Alignment: Feature-based

Feature-based method: feed external collaborative information into LLM

- **Work#2: BinLLM —— Encoding collaborative embeddings in a text-like format for LLM**



transform the collaborative embeddings into binary sequence, treating them as textual features directly usable by LLMs

- LLMs could naturally perform bitwise operations
- Binarizing collaborative embeddings could keep performance.

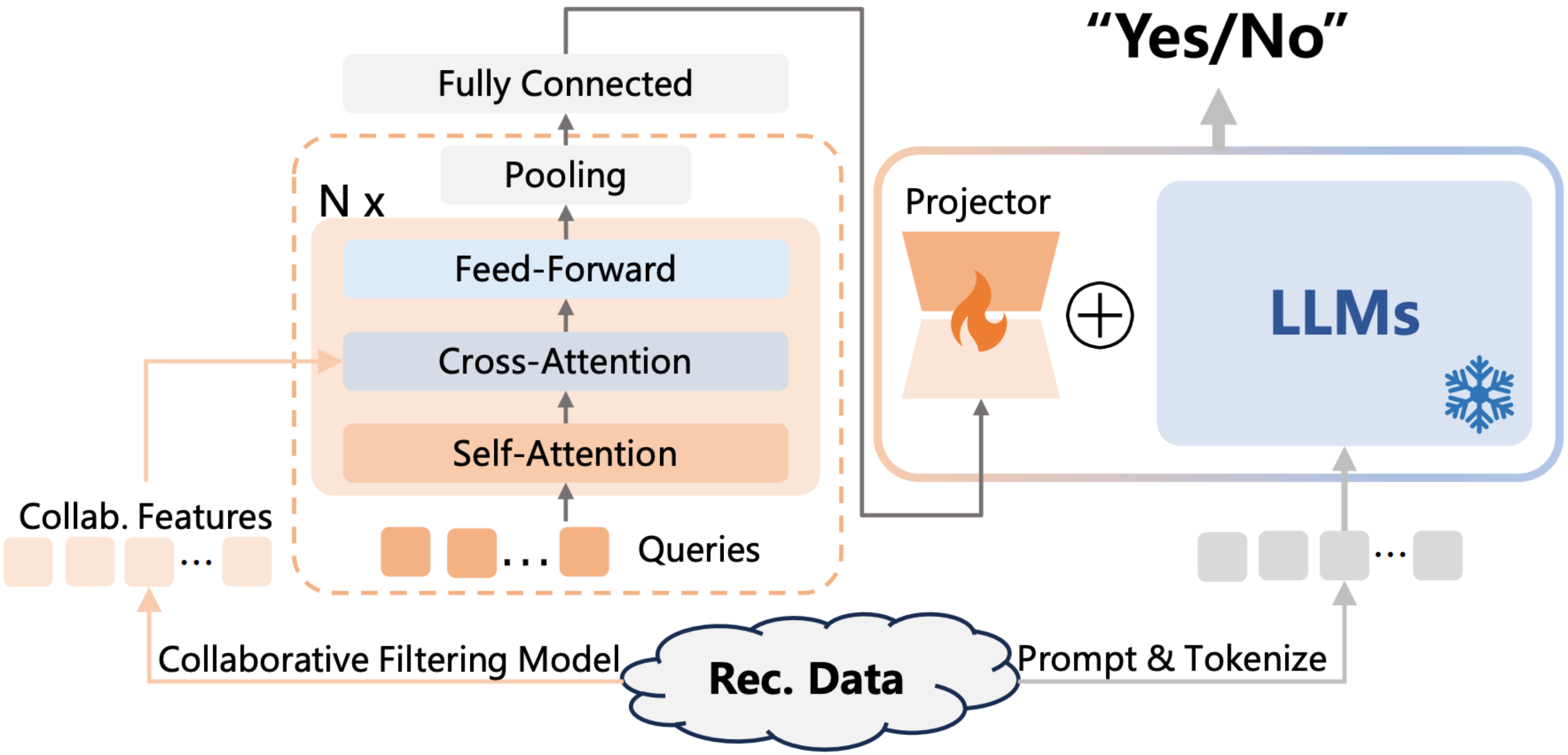
Feed collaborative information into prompts

Yang Zhang et al. Text-like Encoding of Collaborative Information in Large Language Models for Recommendation. ACL ARR preprint.

CF-focused Alignment: Parameter-based

Feature-based method: feed external collaborative information into LLM

- **Cora ——convert collaborative features into incremental weights addable to LLMs**



- Leverage self/corss-attention mechanism to convert model weights
- add the model weights to the original LLMs

Liu, Yuting, et al. "Cora: Collaborative information perception by large language model's weights for recommendation." AAAI 2025.

Tuning LLM4Rec: Inject CF info.

Integrate collaborative information: feed external collaborative information into LLM

- **More works**

[1] Liao et al. Large Language-Recommendation Assistant. ArXiv 2023.

[2] Yang et al. Large Language Model Can Interpret Latent Space of Sequential Recommender. ArXiv 2023.

[3] Yu et al. "RA-Rec: An Efficient ID Representation Alignment Framework for LLM-based Recommendation." arXiv 2024.

[4] Li et al. "E4SRec: An elegant effective efficient extensible solution of large language models for sequential recommendation." arXiv 2023.

[5] Kim et al. “Large Language Models meet Collaborative Filtering: An Efficient All-round LLM-based Recommender System”. KDD 2024.

[6] Zhu et al. “Collaborative Retrieval for Large Language Model-based Conversational Recommender Systems”. WWW 2025.

…

More papers can be found at https://github.com/Linxyhaha/LLM4Rec-Papers

Post-training: Accuracy Perspective

- **Direction 3: Reasoning-enhanced alignment**

  Core: Enhance recommendation performance by incorporating an explicit or implicit deliberative thinking process

Explicit CoT reasoning

Tune the model to incorporate implicit chain-of-thought (CoT) reasoning to enhance recommendation performance

RecSAVER@ACL’24

Reason4Rec@arXiv’25

R^2Rec@arXiv’25

Latent reasoning

Tune the model to incorporate latent reasoning to enhance recommendation performance

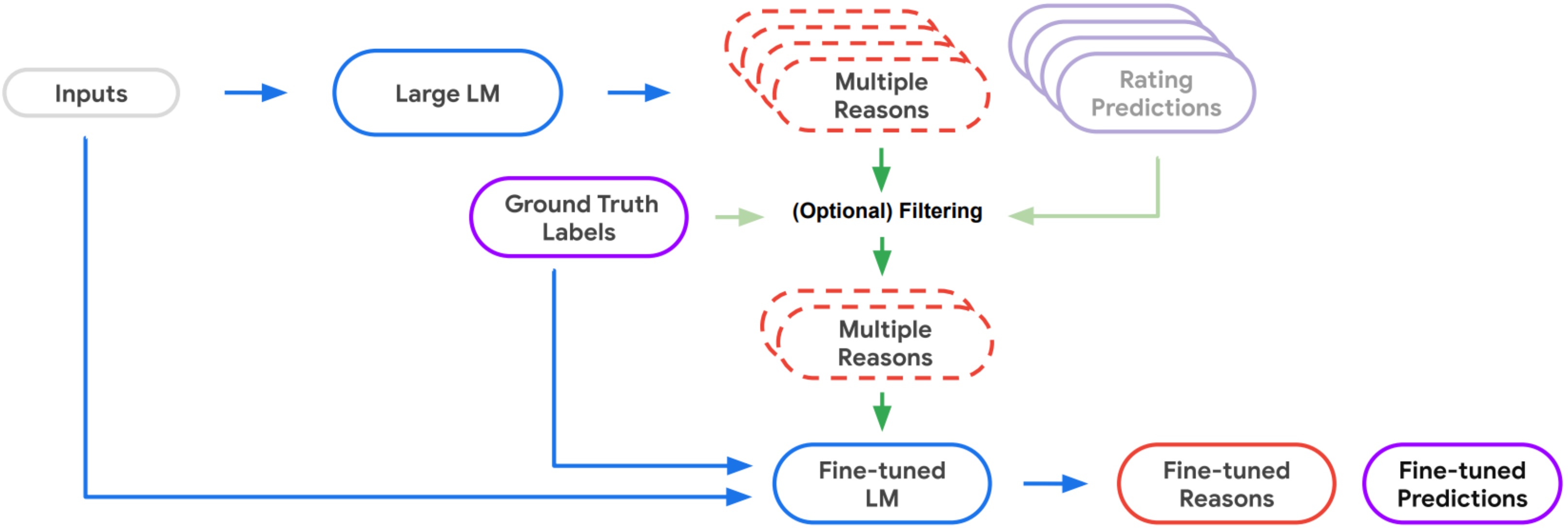
LatentR^3@arXiv’25,

ReaRec@arXiv25

Explicit Reasoning: RecSAVER

RecSAVER: enhance reasoning with created reasoning data

- ❑ **Objective**: leverage **explicit reasoning of LLMs** to enhance preference alignment
- ❑ **Challenges**:
  - ❑ **Lack of reasoning supervision data**
- ❑ **Solutions**: generate reasoning data via larger LLM and select effective reasoning data for tuning

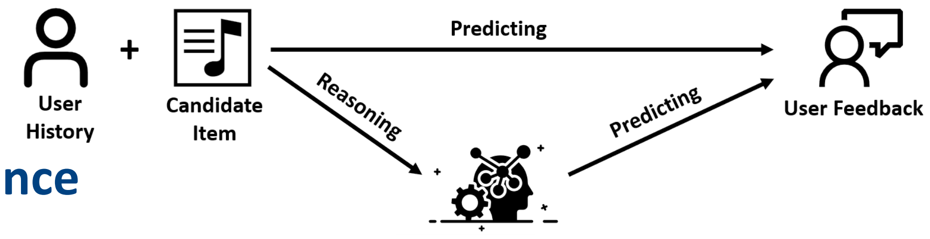


Tsai, Alicia Y., et al. "Leveraging llm reasoning enhances personalized recommender systems." ACL 2024.

Explicit Reasoning: Reason4Rec

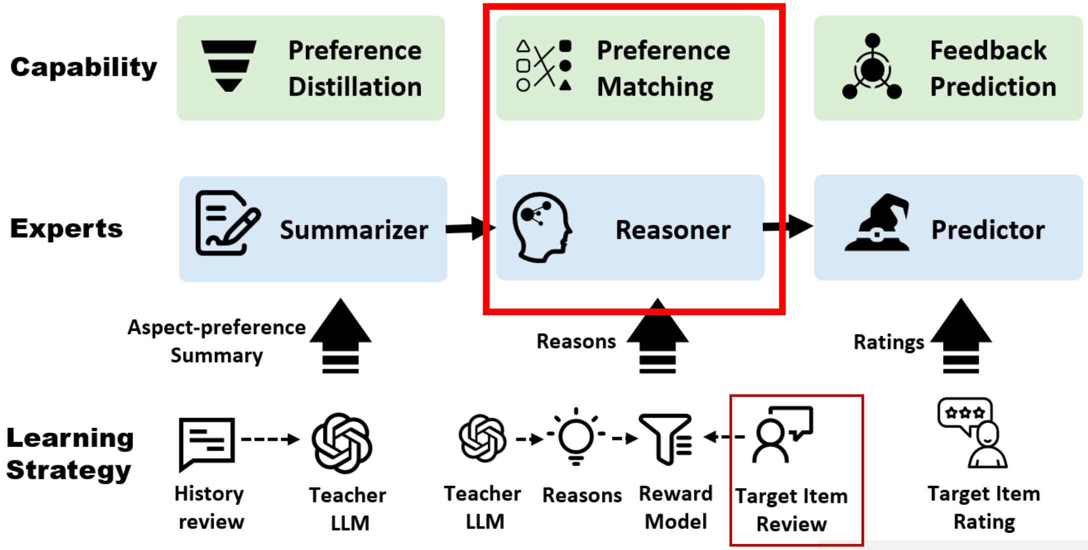
Reason4Rec: enhance reasoning with review data

- **Core**:
  - **Leverage the verbalized user feedback (review data) to enhance reasoning**



CoT Reasoning

- **Solutions**:
  - Propose a multi-step reasoning framework with **three collaborative experts** to reason user preference: 1) preference summarization, 2) reasoning preference matching, 3) final prediction
  - Aligned the reasoning process with users' true preferences derived from **verbalized user feedback (review data)**.



verbalized user feedback

Fang et al. Reason4Rec: Large Language Models for Recommendation with Deliberative User Preference Alignment, Arxiv 2025.

Explicit Reasoning: Reason4Rec

Type	Method	Music		Book		Yelp	
		MAE ↓	RMSE ↓	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓
CF-based	MF	0.6188	0.8142	0.6277	0.8565	0.7980	1.0711
Review-based	DeepCoNN	0.6034	0.8057	0.6221	0.8403	0.8312	1.0665
	NARRE	0.5799	0.7881	0.6242	0.8435	0.8177	1.0785
	DAML	0.5703	<u>0.7848</u>	0.6214	<u>0.8371</u>	<u>0.7964</u>	**1.0405**
LLM-based	GPT-4o	0.7438	1.1069	0.7591	1.1558	0.8766	1.3005
	Rec-SAVER	0.6463	0.9262	0.6645	0.9356	0.8295	1.1282
	EXP3RT	<u>0.5608</u>	0.8385	<u>0.6135</u>	0.9370	0.8306	1.2311
Ours	**Reason4Rec**	**0.5442**	**0.7722**	**0.6029**	**0.8345**	**0.7586**	<u>1.0418</u>

Method	Music		Book		Yelp	
	GPT	BLEURT	GPT	BLEURT	GPT	BLEURT
Rec-SAVER	75.60	0.3652	72.45	0.4233	<u>66.43</u>	0.4102
EXP3RT	<u>76.22</u>	<u>0.3840</u>	<u>73.60</u>	<u>0.4373</u>	64.28	<u>0.4275</u>
Reason4Rec	**80.53**	**0.4067**	**77.31**	**0.4731**	**72.70**	**0.4565**

Method	Avg. Inference Time (s)	Avg. Tokens Generated
Reason4Rec	5.86	147.78
Rec-SAVER	6.43	175.59
EXP3RT	5.62	150.74

- The proposed method achieves better prediction accuracy than all baselines
- The reasons generated by the proposed method are better aligned with user preferences.
- The inference cost of our method is comparable to that of the previous reason-enhanced LLMRec methods.

Fang et al. Reason4Rec: Large Language Models for Recommendation with Deliberative User Preference Alignment, Arxiv 2025.

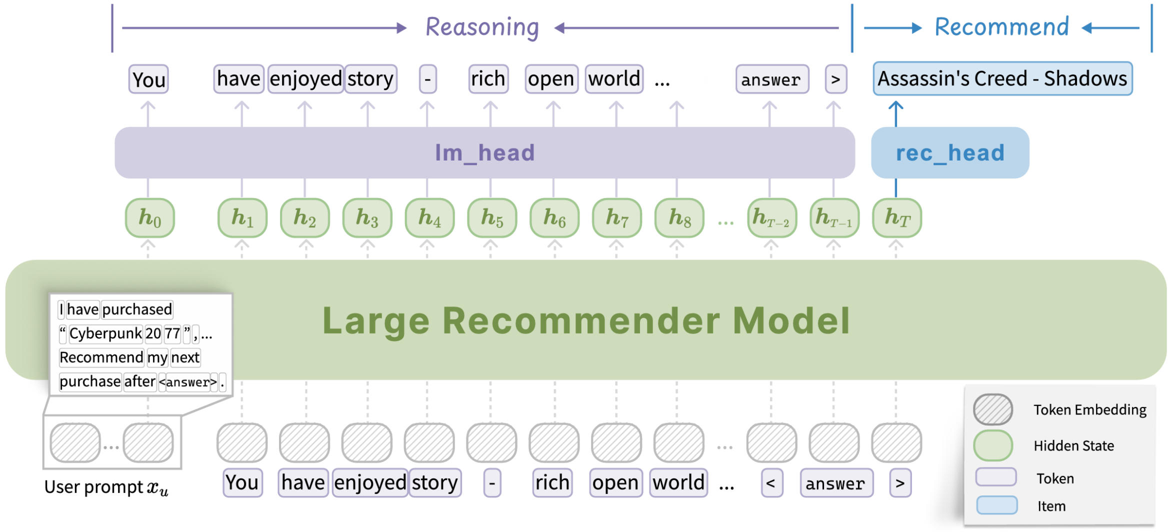
Explicit Reasoning: R²ec

R²ec: enhance reasoning without reinforced learning

Architecture

Decoder-only backbone with two heads

- lm_head for generating reasoning.
- rec_head for scoring items.



RecPO (based on GRPO)

1. Sample multiple reasoning trajectories per user via top-K + temperature sampling.
2. Assign a **fused reward** combining Discrete reward (NDCG) and Continuous reward (softmax over item embeddings)
3. Single policy update jointly optimizes reasoning and recommendation.

$$\mathcal{J}(\theta) = \mathbb{E}_{\{u,v^+\}\sim\mathcal{D},\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|x_u)} \frac{1}{G}\sum_{i=1}^{G}\Big[\sum_{t=1}^{T_i}\ell_\epsilon(r_{i,t}(\theta),A_i)+\delta_{i,i^*}\ell_\epsilon\big(r_{i,T+1}(\theta),A_i\big)\Big].$$

$$r_{i,t}(\theta) = \begin{cases} \dfrac{\pi_\theta(o_{i,t}|x_u,o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|x_u,o_{i,<t})}, & \text{if } t \leqslant T \text{ (reasoning)} \\ \dfrac{\pi_\theta(v^+|x_u,o_{i,\leqslant T})}{\pi_{\theta_{\text{old}}}(v^+|x_u,o_{i,\leqslant T})}, & \text{if } t = T+1 \text{ (recommendation)}. \end{cases}$$

Explicit Reasoning: R^2ec

R^2ec – Towards Large Recommender Models with Reasoning

1. On three Amazon domains (CDs, Games, Instruments) with 2 backbones (Gemma2-2b, Qwen2.5-3b), **+68.7% Hit@5 and +45.2% NDCG@20** over best baselines.

		Instruments						CDs and Vinyl						Video Games					
	Method	H@5	N@5	H@10	N@10	H@20	N@20	H@5	N@5	H@10	N@10	H@20	N@20	H@5	N@5	H@10	N@10	H@20	N@20
	GRU4Rec	0.0171	0.0135	0.0193	0.0142	0.0201	0.0144	0.0067	0.0037	0.0104	0.0041	0.0156	0.0051	0.0109	0.0070	0.0181	0.0093	0.0301	0.0123
	Caser	0.0109	0.0141	0.0115	0.0149	0.0127	0.0155	0.0045	0.0029	0.0067	0.0037	0.0089	0.0042	0.0124	0.0083	0.0191	0.0103	0.0279	0.0126
	SASRec	<u>0.0175</u>	<u>0.0144</u>	0.0201	<u>0.0162</u>	0.0223	<u>0.0210</u>	0.0076	0.0104	0.0081	0.0119	0.0086	0.0141	0.0129	0.0080	0.0206	0.0105	0.0326	0.0135
	TIGER	0.0171	0.0128	0.0184	0.0132	0.0193	0.0134	0.0067	0.0045	0.0097	0.0055	0.0156	0.0069	0.0123	0.0085	0.0222	0.0116	0.0323	0.0142
Qwen	BigRec	0.0052	0.0033	0.0111	0.0052	0.0189	0.0072	0.0045	0.0025	0.0089	0.0039	0.0141	0.0052	0.0008	0.0004	0.0016	0.0006	0.0128	0.0034
Qwen	D^3	0.0042	0.0020	0.0094	0.0037	0.0192	0.0062	0.0082	0.0057	0.0141	0.0076	0.0253	0.0104	0.0054	0.0028	0.0104	0.0044	0.0197	0.0067
Qwen	LangPTune	0.0127	0.0083	<u>0.0224</u>	0.0115	0.0348	0.0145	0.0074	0.0053	0.0156	0.0080	0.0208	0.0094	0.0049	0.0027	0.0088	0.0040	0.0140	0.0140
Qwen	**R^2ec**	**0.0237***	**0.0154***	**0.0374***	**0.0198***	**0.0615***	**0.0259***	**0.0513***	**0.0372***	**0.0647***	**0.0414***	**0.0818***	**0.0457***	**0.0288***	**0.0185***	**0.0532***	**0.0264***	**0.0827***	**0.0337***
Qwen	*% Improve.*	35.43%	6.94%	66.96%	22.22%	52.61%	23.33%	46.57%	58.30%	37.95%	51.09%	20.83%	40.62%	84.62%	76.19%	104.62%	87.23%	92.33%	50.45%
Gemma	BigRec	0.0068	0.0048	0.0101	0.0058	0.0130	0.0066	0.0030	0.0030	0.0052	0.0037	0.0119	0.0053	<u>0.0156</u>	<u>0.0105</u>	<u>0.0260</u>	0.0138	<u>0.0430</u>	0.0182
Gemma	D^3	0.0072	0.0038	0.0202	0.0080	0.0339	0.0114	0.0216	0.0129	0.0327	0.0164	0.0446	0.0194	0.0117	0.0068	0.0210	<u>0.0141</u>	0.0378	<u>0.0224</u>
Gemma	LangPTune	0.0130	0.0079	0.0221	0.0107	<u>0.0403</u>	0.0152	<u>0.0350</u>	<u>0.0235</u>	<u>0.0469</u>	<u>0.0274</u>	<u>0.0677</u>	<u>0.0325</u>	0.0068	0.0053	0.0120	0.0059	0.0195	0.0094
Gemma	**R^2ec**	**0.0264***	**0.0161***	**0.0397***	**0.0203***	**0.0615***	**0.0257***	**0.0573***	**0.0398***	**0.0804***	**0.0472***	**0.1042***	**0.0527***	**0.0326***	**0.0205***	**0.0531***	**0.0271***	**0.0835***	**0.0347***
Gemma	*% Improve.*	50.86%	11.81%	77.23%	25.31%	52.61%	22.38%	63.71%	69.36%	71.43%	72.26%	53.91%	62.15%	108.97%	95.24%	104.23%	92.20%	94.19%	54.91%

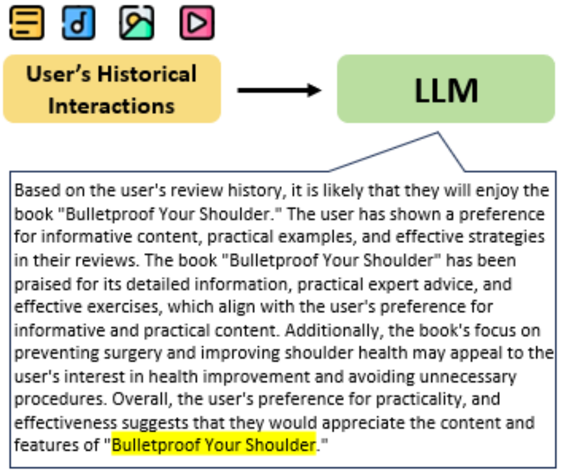
2. Removing reasoning causes ~15% performance drop, underscoring the value of "thinking."

	Instruments						CDs and Vinyl						Video Games					
Method	H@5	N@5	H@10	N@10	H@20	N@20	H@5	N@5	H@10	N@10	H@20	N@20	H@5	N@5	H@10	N@10	H@20	N@20
w/o Reasoning	0.0176	0.0121	0.0296	0.0153	0.0511	0.0200	0.0469	0.0321	0.0692	0.0393	0.0945	0.0456	0.0277	0.0174	0.0441	0.0227	0.0748	0.0303
w/o R_d	0.0198	0.0124	0.0338	0.0164	0.0560	0.0224	0.0521	0.0338	0.0766	0.0404	0.0974	0.0486	0.0302	0.0196	0.0487	0.0254	0.0798	0.0332
w/o R_c	<u>0.0244</u>	<u>0.0160</u>	<u>0.0394</u>	**0.0208**	<u>0.0605</u>	**0.0258**	<u>0.0543</u>	<u>0.0382</u>	<u>0.0774</u>	<u>0.0456</u>	<u>0.1012</u>	<u>0.0515</u>	<u>0.0316</u>	<u>0.0202</u>	**0.0534**	<u>0.0264</u>	<u>0.0814</u>	<u>0.0355</u>
R^2ec	**0.0264**	**0.0161**	**0.0397**	<u>0.0203</u>	**0.0615**	<u>0.0257</u>	**0.0588**	**0.0388**	**0.0804**	**0.0457**	**0.1086**	**0.0525**	**0.0326**	**0.0205**	<u>0.0531</u>	**0.0271**	**0.0853**	**0.0363**

Latent Reasoning

- Challenge of implicit reasoning: rely on explicit chain-of-thought (CoT) data.
  - Difficult to obtain **high-quality CoT data** for fine-tuning.
  - **High inference latency** during inference.

useless tokens generation

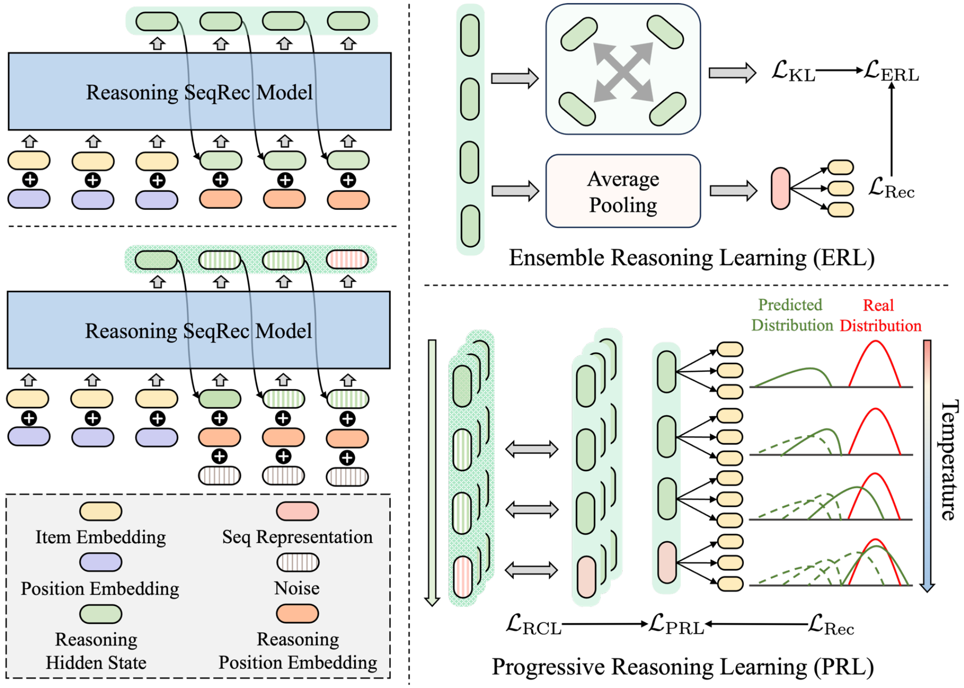


- Objective: Design a method that can fully utilize the reasoning capabilities of LLM while eliminating the need for CoT data.

Zhang, Yang, et al. "Reinforced Latent Reasoning for LLM-based Recommendation." arXiv preprint arXiv:2505.19092 (2025).

Latent Reasoning: ReaRec

- ReaRec: Latent Reasoning for sequential recommendation
  - Solution： Integrate some latent reasoning steps by treating last-layer hidden states as the latent thinking



- **Ensemble Reasoning Learning (ERL)**

  Obtain sequence representations from diverse reasoning views.

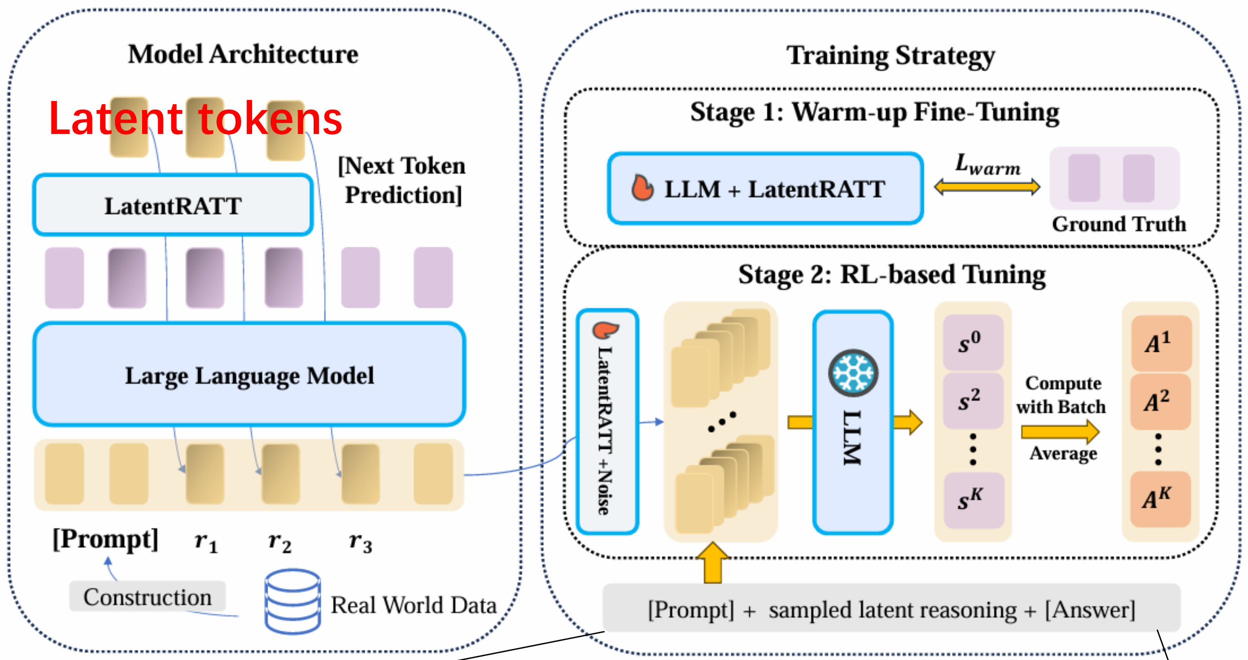
- **Progressive Reasoning Learning (PRL)**

  Temperature annealing mechanism progressively directs model reasoning to the optimal solution

Tang, Jiakai, et al. "Think before recommend: Unleashing the latent reasoning power for sequential recommendation." arXiv preprint arXiv:2503.22675 (2025).

Latent Reasoning: LatentR^3

- LatentR^3: Latent reasoning for LLM4rec with RL
  - **Model architecture**: add a module to aggregate last-layer hidden state to generate latent reasoning tokens.
  - **2-Stage Training Strategy**: Through SFT and a modified GRPO to fully unleash LLM's latent reasoning capabilities.



[Prompt] + sampled latent reasoning + [Answer]

$[p_0(y) \quad p_1(y) \dots \quad p_n(y)]$

Different thought tokens generate different p of the ground-truth item.

- RL designs (modified GRPO):
  - **Sampling via adding noise on latent reasoning**
  - **PPL(perplexity) as reward (efficient training).**
  - Batch-level advantage normalization.

$$A^k = \frac{s^k - \bar{s}_{batch}}{\|\mathbf{S}_{batch} - \bar{s}_{batch}\|}$$

Zhang, Yang, et al. "Reinforced Latent Reasoning for LLM-based Recommendation." arXiv 2025.

Latent Reasoning: LatentR^3

1) Effectiveness: All metrics of all datasets consistently surpass existing methods, demonstrating the **effectiveness** of the method.

2) Generalizability: Our latent reasoning method can be applied to different LLM-based methods.

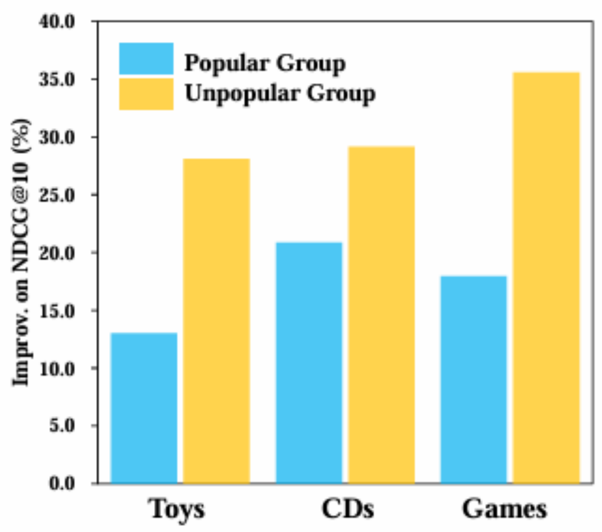


Table 1. Overall performance of baselines and LatentR^3

Dataset	Methods	Traditional			LLM-based				
		Caser	GRU4Rec	SASRec	AlphaRec	BIGRec	+LatentR^3	D^3	+LatentR^3
Toys	H@5	0.0251	0.0417	0.0601	0.0579	0.0701	0.0821	0.0830	**0.0898**
	H@10	0.0384	0.0564	0.0760	0.0893	0.0931	0.1107	0.1026	**0.1152**
	N@5	0.0170	0.0305	0.0458	0.0347	0.0508	0.0600	0.0610	**0.0670**
	N@10	0.0214	0.0352	0.0510	0.0448	0.0582	0.0693	0.0674	**0.0752**
CDs	H@5	0.0469	0.0481	0.0841	0.0479	0.0757	0.0934	0.1122	**0.1137**
	H@10	0.0689	0.0669	0.1054	0.0774	0.0929	0.1160	0.1272	**0.1327**
	N@5	0.0312	0.0365	0.0622	0.0278	0.0616	0.0754	0.0906	**0.0915**
	N@10	0.0382	0.0425	0.0691	0.0373	0.0672	0.0826	0.0955	**0.0977**
Games	H@5	0.0324	0.0322	0.0416	0.0558	0.0461	0.0580	0.0608	**0.0716**
	H@10	0.0538	0.0517	0.0633	0.0893	0.0709	0.0870	0.0860	**0.1006**
	N@5	0.0211	0.0207	0.0280	0.0397	0.0334	0.0413	0.0423	**0.0507**
	N@10	0.0280	0.0270	0.0350	0.0515	0.0414	0.0506	0.0505	**0.0601**
	RI	170.8%	121.0%	52.3%	77.5%	21.8%	-	10.4%	-

3) Larger improvements on unpopular items: The incorporation of reasoning is particularly beneficial in more challenging recommendation scenarios.

Zhang, Yang, et al. "Reinforced Latent Reasoning for LLM-based Recommendation." arXiv preprint arXiv:2505.19092 (2025).

Outline

- Introduction
- Development of LLMs
- **Technical Stacks of LLM4Rec**
  - Model Architecture and Pre-training
  - Model Post-training - accuracy
  - **QA & Coffee Break**
  - Model Post-training – efficiency and trustworthiness
  - Decoding and Deployment
- Open Problems
- Future Direction & Conclusions

Outline

QA & Coffee Break

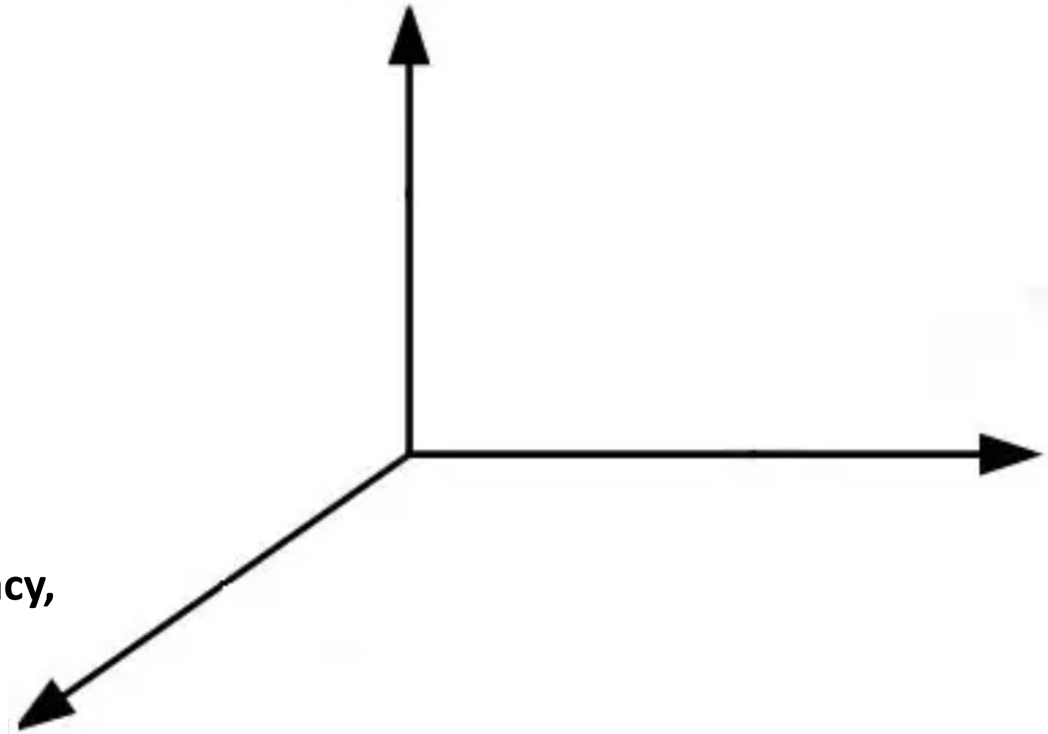
The tutorial will continue at 16:00

Model Post-training

Three dimensions:

Accuracy

Learn to capture user preference and generate items for accurate recommendation



Efficiency

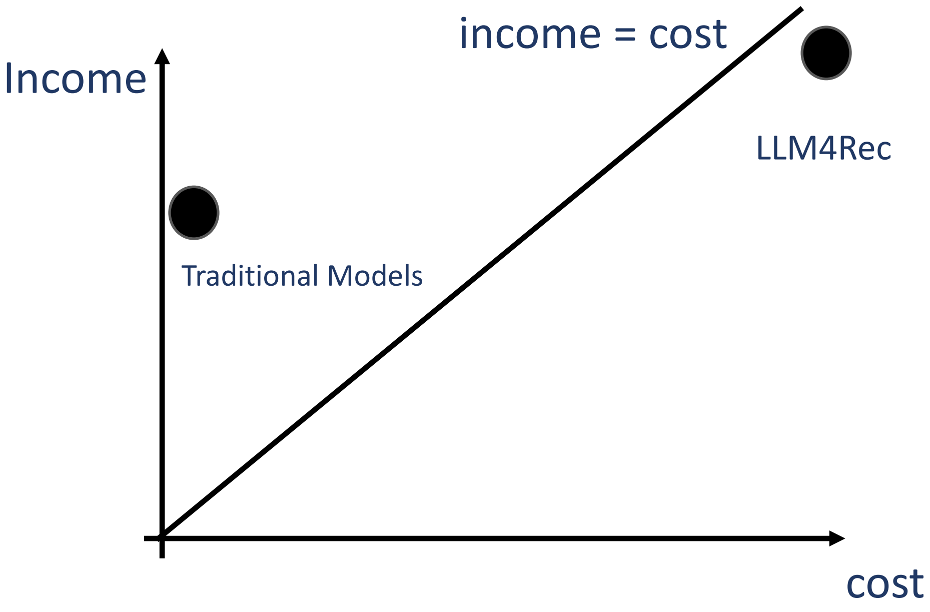
Data-efficient, parameter-effici post-training, etc

Trustworthiness

Beyond accuracy such as privacy, fairness, etc.

Efficiency Issue

- The income-cost trade-off is sensitive for recommendation
- Deployment cost of LLM4Rec is high



LLM Parameters: tens/hundreds of billions

Training and inference:

- High demand on GPUs/Memory
- Slow

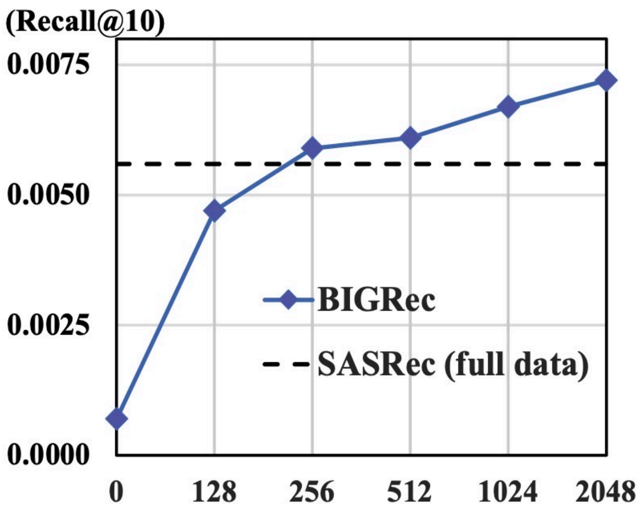
How to reduce the cost?

Post-training Efficiency

One exploration: Data-efficient training

Statistics from Tiktok [1] (per day)
- New videos: ~160M
- New interactions: ~942B

- ❑ Fine-tuning LLM is necessary
  - ❑ LLMs are not particularly trained on recommendation data
- ❑ LLM fine-tuning is expensive, e.g., high computational costs, time-consuming
- ❑ Few-shot fine-tuning is a promising solution
- ❑ Data pruning for efficient LLM-based recommendation
  - ❑ identify representative samples tailored for LLMs



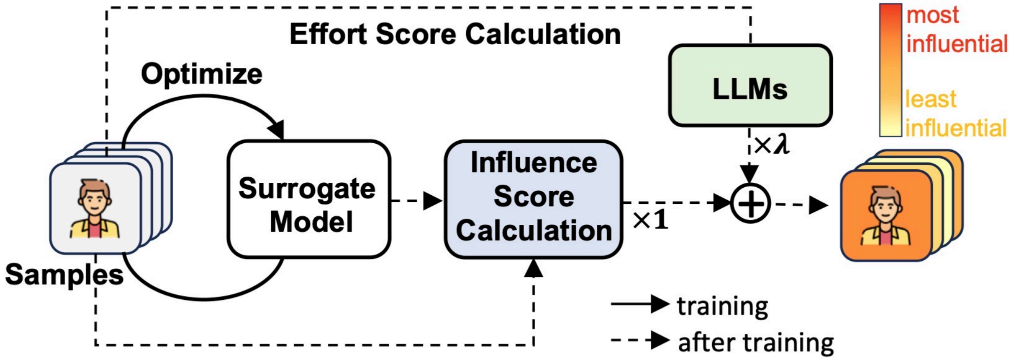
(a) Few-shot performance on MicroLens-50K.

Lin et al. Data-efficient fine-tuning for LLM-based recommendation. SIGIR'24.

Post-training Efficiency

One exploration: Data pruning

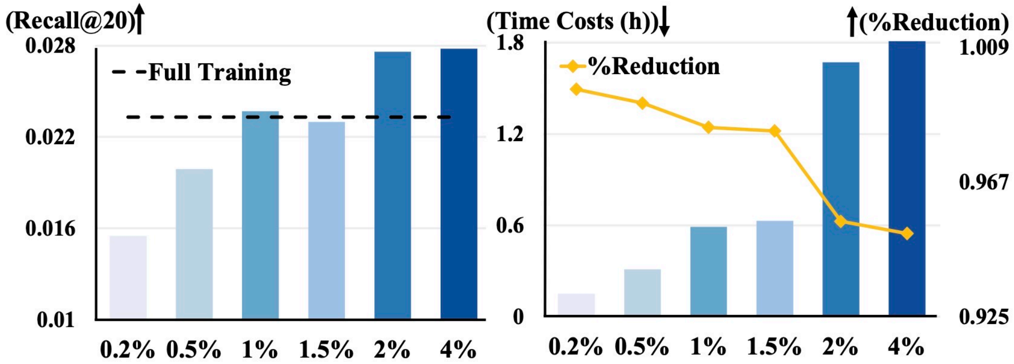
- Two objectives for data pruning
  - high accuracy: select the samples that can lead to higher performance -> influence score
  - high efficiency: emphasize the low costs of the data pruning process
    - surrogate model to improve efficiency
    - effort score to bridge between surrogate model and LLMs



- Experimental results
  - fine-tune with 1024 samples

	Games				
	R@10↑	R@20↑	N@10↑	N@20↑	Time↓
Full	0.0169	0.0233	0.0102	0.0120	36.87h
DEALRec	0.0181	0.0276	0.0115	0.0142	1.67h
% Improve.	**7.10%**	**18.45%**	**12.75%**	**18.33%**	**-95.47%**

  - Increasing samples from 0.2% to 4% of all training data



(a) Effect of r *w.r.t.* Recall (b) Effect of r *w.r.t.* time costs

Lin et al. Data-efficient fine-tuning for LLM-based recommendation. SIGIR'24.

Distillation for Inference Efficiency

One solution: Model distillation

Distill LLM's knowledge to smaller models and utilize small models for inference

- **Work#1: distill recommendation results**

Dataset	Model	HR@20	NDCG@20	Inference time
Games	DROS	0.0473	0.0267	1.8s
	BIGRec	0.0532	0.0341	2.3×10^4s
	Gain	+12.47%	+27.72%	-1.3×10^6%
Toys	DROS	0.0231	0.0144	1.6s
	BIGRec	0.0420	0.0207	1.1×10^4s
	Gain	+81.82%	+43.75%	-6.8×10^5%

The inference latency of BIGRec far exceeds that of DROS.

Dataset	Condition	Relative Ratio
Games	BIGRec > DROS	53.90%
	BIGRec < DROS	46.10%
MovieLens	BIGRec > DROS	40.90%
	BIGRec < DROS	59.10%
Toys	BIGRec > DROS	66.67%
	BIGRec < DROS	33.33%

BIGRec does not always outperform DROS.

- ❑ Distillation challenges:
  - ❑ 1）The teacher's knowledge may not always be reliable.
  - ❑ 2）The divergence in semantic space poses a challenge to distill the knowledge from embeddings.

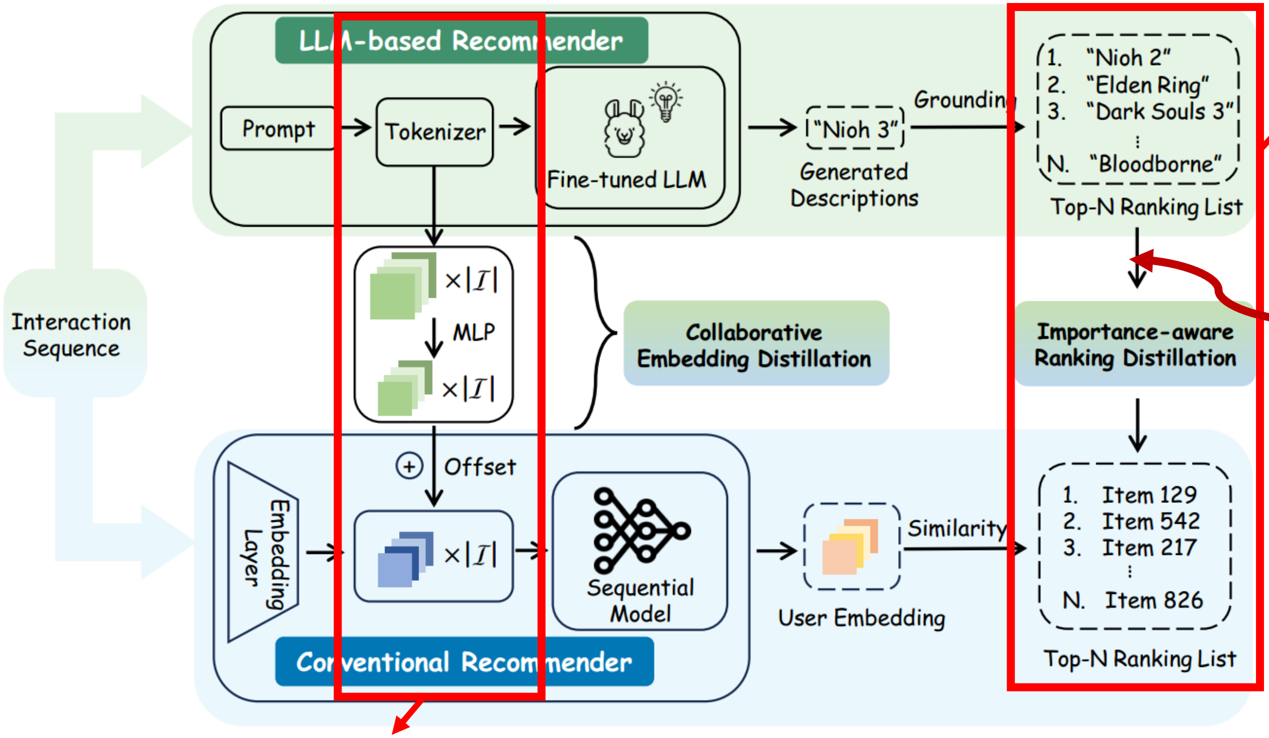
Yu Cui et al. "Distillation Matters: Empowering Sequential Recommenders to Match the Performance of Large Language Model" arXiv 2024

Distillation for Inference Efficiency

One solution: Model distillation

Distill LLM's knowledge to smaller models and utilize small models for inference

- **Work#1: distill recommendation results**



Supervised signals

- Importance-aware Ranking Distillation

  filter reliable and student-friendly knowledge by weighting instances

  - Confidence of LLMs

    The distance between the generated descriptions with the target item

  - Teacher-Student Consensus

    The items recommended by both teacher and student are more likely to be positive

  - Ranking Position

    Higher ranked items by teachers are more reliable

Collaborative Embedding Distillation

integrate knowledge from teacher and student

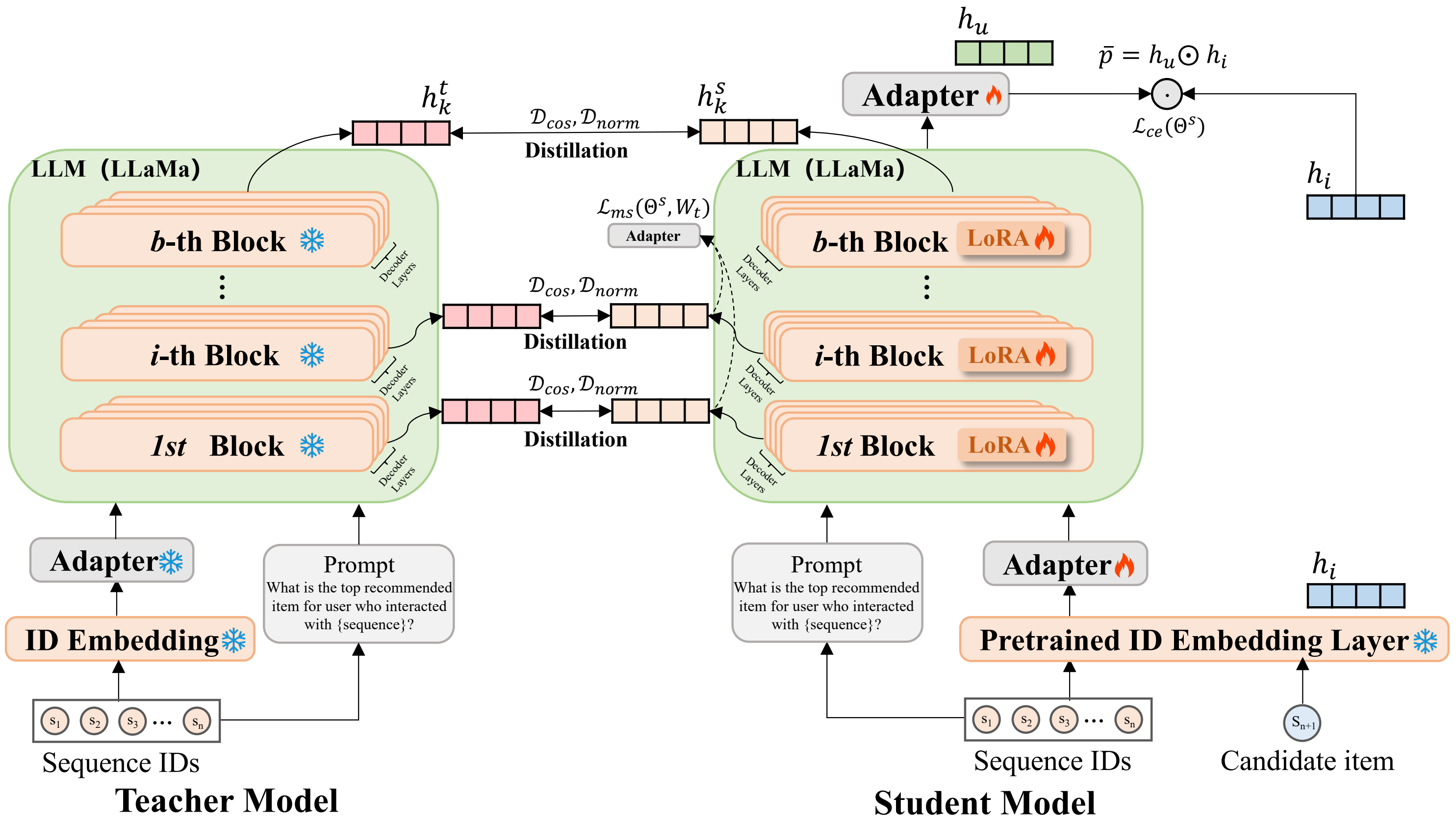
Yu Cui et al. "Distillation Matters: Empowering Sequential Recommenders to Match the Performance of Large Language Model" arXiv 2024

Distillation for Inference Efficiency

One solution: distillation

Distill LLM’s knowledge to smaller models and utilize small models for inference

- **Work#2: layer-wise knowledge distillation for smaller LM**



Motivation:

some layers of LLMs are redundant in the downstream recommendation task

Efficiency:

achieves up to 6.6x/8.0x speedup in terms of training/inference time costs against LLM-based recommendation models

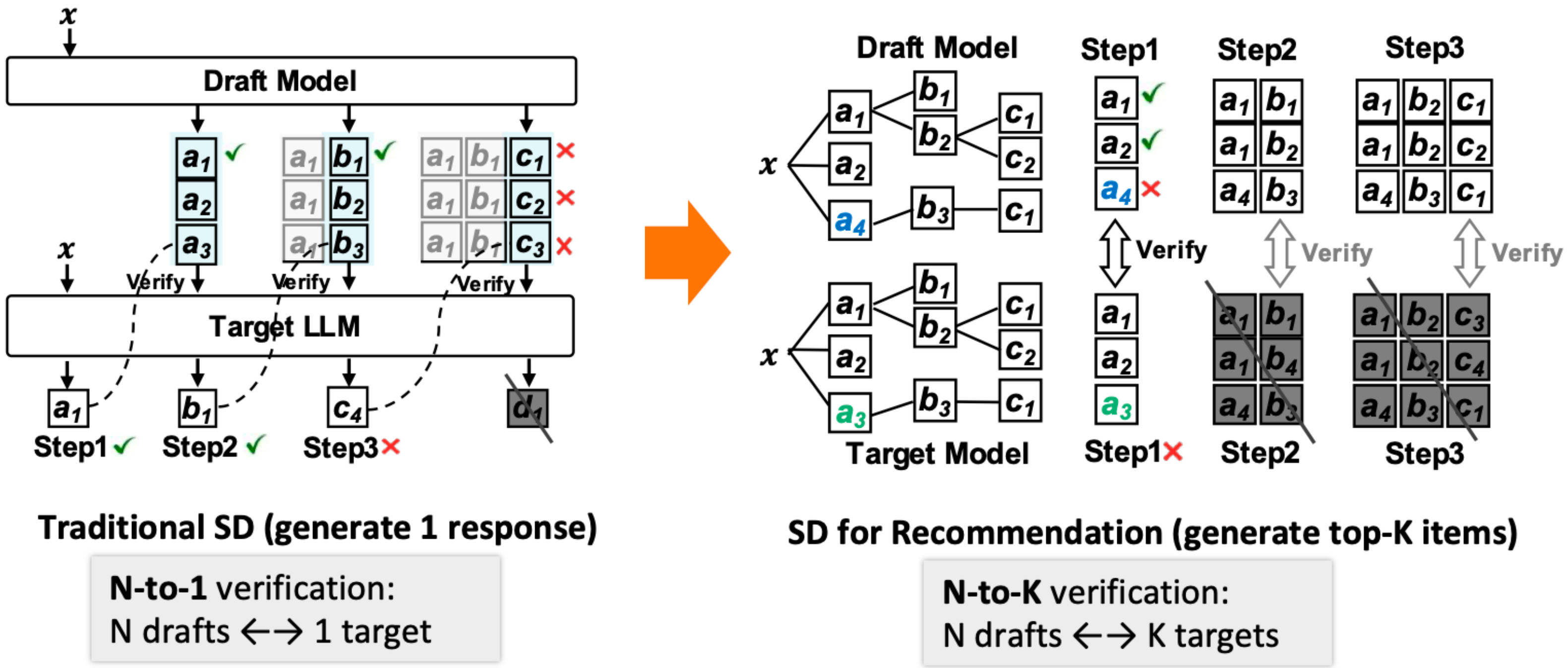
Wujiang Xu et al. “SLMRec: Empowering Small Language Models for Sequential Recommendation” ICLR 2025

Post Alignment for Inference Efficiency

One solution: speculative decoding

- **Work#1: speculative decoding for LLM-based recommendation**
  - **Core idea:** reduce the LLMs' autoregressive steps
  - **Speculative Decoding:** small model draft multiple tokens – LLM verify in parallel

Challenges From N-to-1 to N-to-K verification



Lin et al. Efficient Inference for Large Language Model-based Generative Recommendation. ICLR'25.

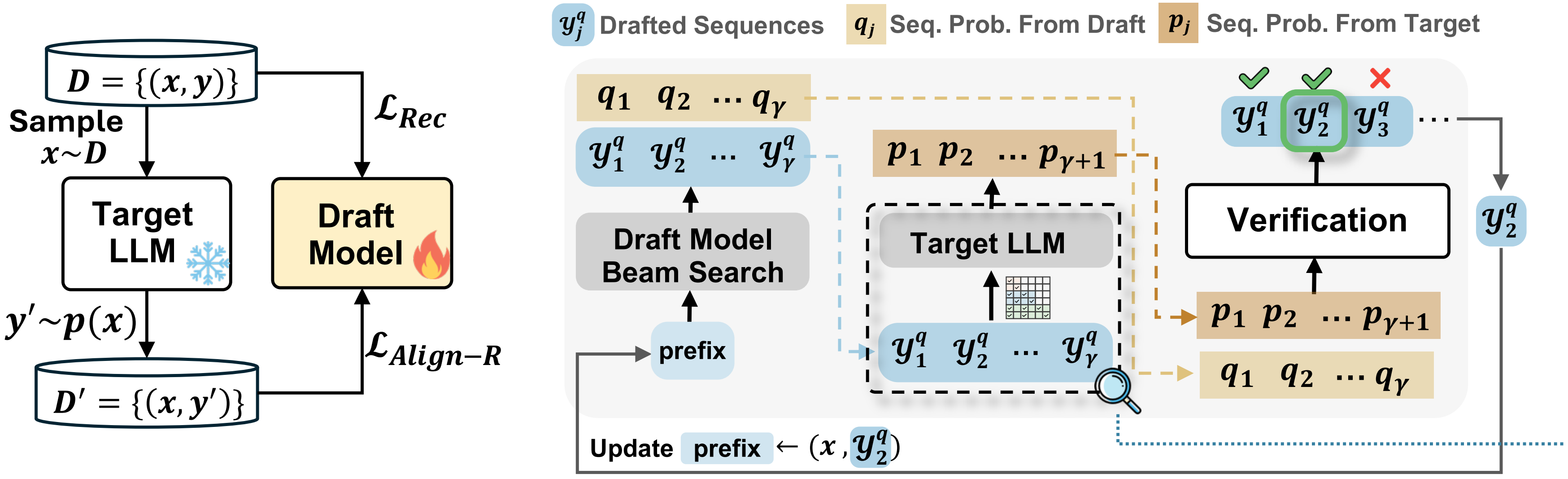
Post Alignment for Inference Efficiency

One solution: speculative decoding

- **Work#1: speculative decoding for LLM-based recommendation**

Solution

- **Strong alignment:** align the drafted sequences with the target top-K sequences
- **Relaxed verification:** ease the strict matching with maintained accuracy



(a) AtSpeed Training

(b) AtSpeed Inference

achieves 2.5x speedup with relaxed verification on top-20 recommendation

Lin et al. Efficient Inference for Large Language Model-based Generative Recommendation. ICLR'25.

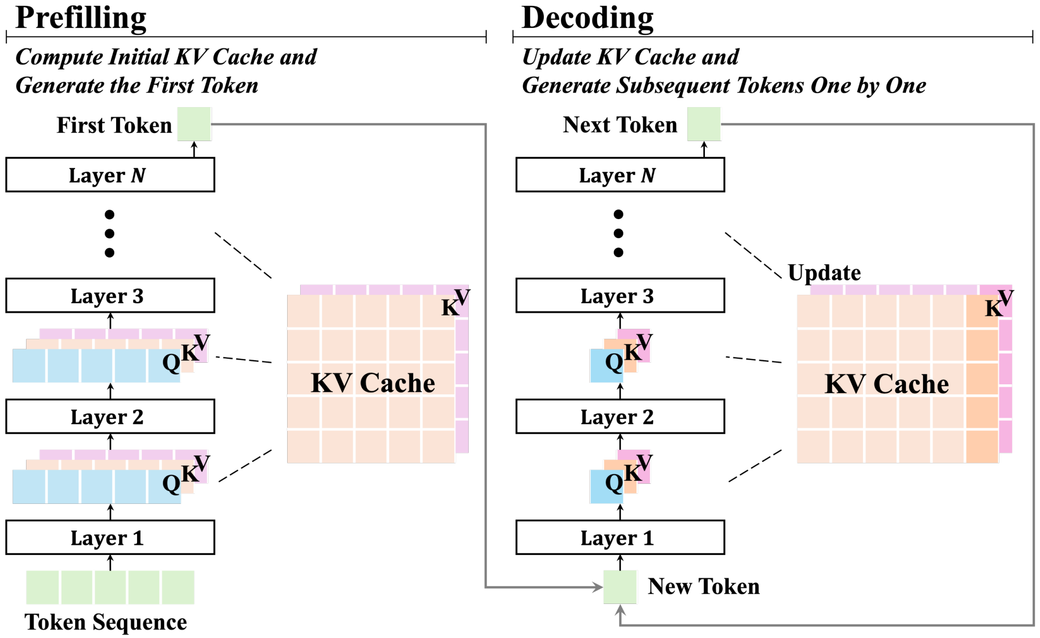
Post-training for Inference Efficiency

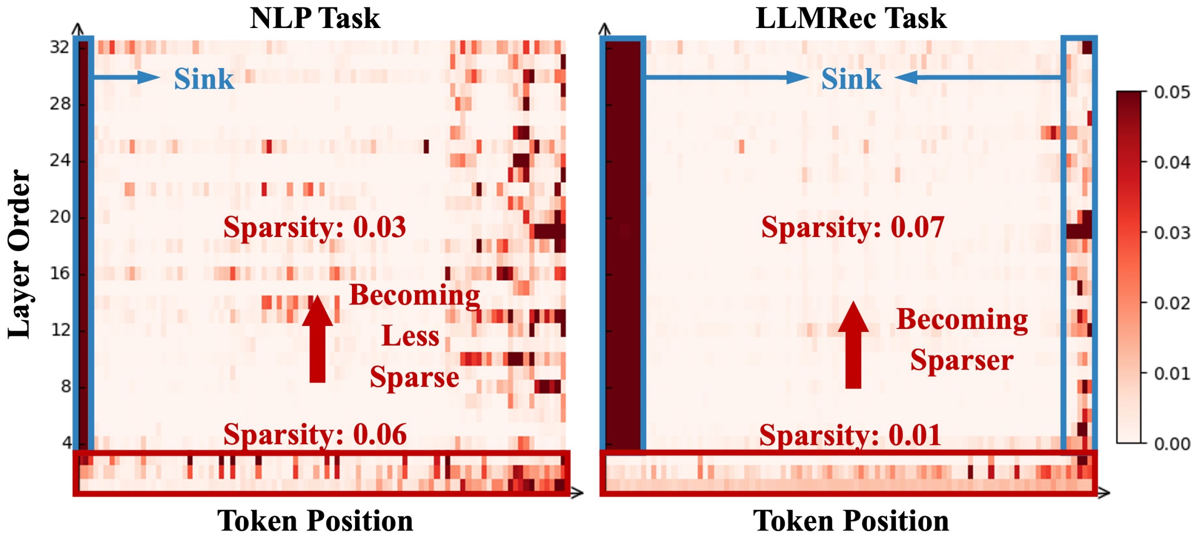
➢ Background

- LLMRec has achieved notable success, but it suffers from high inference latency due to massive computational overhead and memory pressure of KV Cache.

➢ Observation

- **Layer-wise attention sparsity inversion:** Early layers dense, later layers sparse.
- **Dual attention sinks phenomenon:** Attention scores concentrate on both head and tail tokens of input sequences.



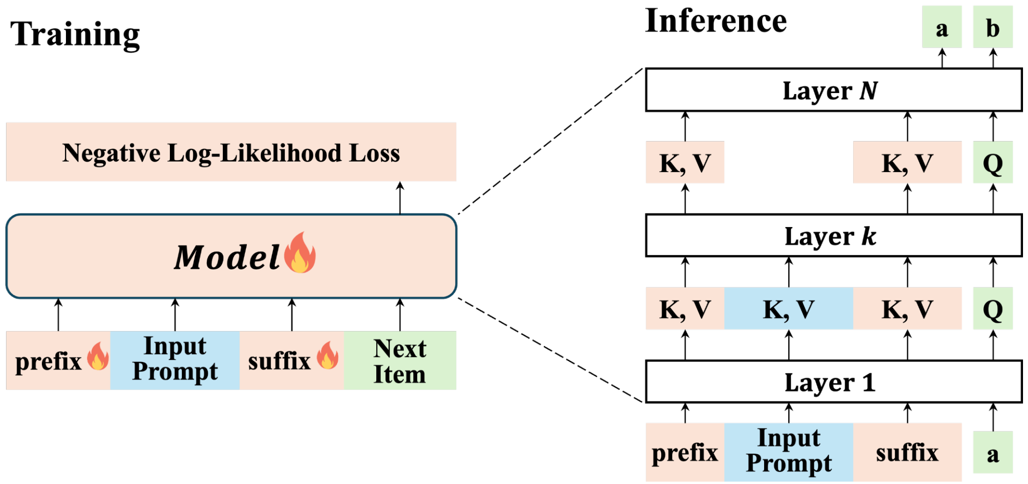


Yang et al. EARN: Efficient Inference Acceleration for LLM-based Genera

Post-training for Inference Efficiency

➢ Method

- **Training:** Learn to leverage the early layers to compress information into register tokens.
- **Inference:** After layer k, EARN removes the prompt tokens to achieve acceleration.



➢ Performance

- 3.79x speedup, 80.8% KV Cache reduction, better accuracy!

Dataset	Model	Method	Time Efficiency ω	Time Efficiency τ	Space Efficiency γ	Space Efficiency σ	Recommendation Effectiveness R@10	R@20	N@10	N@20
Beauty	Llama	Finetune	1.00	505.2	0.0	85.45	0.0145	0.0225	0.0084	0.0108
		SkipLayers	1.79	895.4	44.4	47.50	0.0013	0.0013	0.0013	0.0013
		POD	1.15	585.0	14.7	72.87	0.0045	0.0074	0.0032	0.0041
		500xCompressor	2.31	1168.6	74.8	21.55	0.0005	0.0006	0.0002	0.0003
		StreamingLLM	1.22	611.2	**96.4**	**3.09**	0.0005	0.0005	0.0004	0.0004
		SnapKV	1.20	600.7	94.5	4.73	0.0054	0.0061	0.0030	0.0032
		Gist	1.18	597.6	17.5	70.50	0.0048	0.0077	0.0028	0.0036
		AnLLM	1.24	625.1	92.2	6.70	0.0000	0.0000	0.0000	0.0000
		EARN	**3.79**	**1844.8**	80.5	16.68	**0.0167**	**0.0265**	**0.0095**	**0.0124**
	Qwen	Finetune	1.00	622.1	0.0	14.08	0.0145	0.0248	0.0087	0.0117
		SkipLayers	1.73	1056.1	58.0	5.92	0.0000	0.0000	0.0000	0.0000
		POD	1.09	679.3	8.4	12.90	0.0082	0.0127	0.0047	0.0061
		500xCompressor	2.56	1587.1	91.1	1.26	0.0003	0.0003	0.0001	0.0001
		StreamingLLM	1.05	652.6	**92.0**	**1.12**	0.0088	0.0147	0.0058	0.0075
		SnapKV	1.02	634.5	69.5	4.29	0.0097	0.0165	0.0058	0.0077
		Gist	1.15	715.4	20.0	11.30	0.0084	0.0161	0.0050	0.0074
		AnLLM	1.06	659.3	80.5	2.70	0.0000	0.0000	0.0000	0.0000
		EARN	**2.71**	**1662.6**	75.2	3.49	**0.0155**	**0.0265**	**0.0091**	**0.0122**

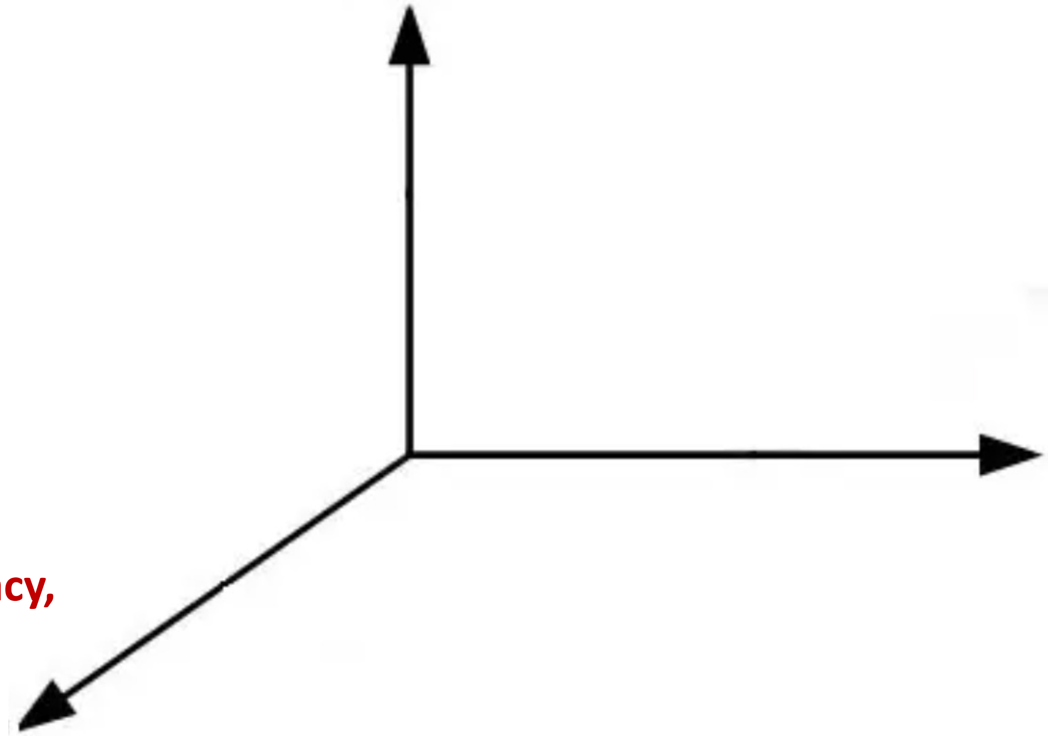
Yang et al. EARN: Efficient Inference Acceleration for LLM-based Generative Recommendation by Register Tokens. KDD'25.

Model Post-training

Three dimensions:

Accuracy

Learn to capture user preference and generate items for accurate recommendation



Efficiency

Data-efficient, parameter-effici post-training, etc

Trustworthiness

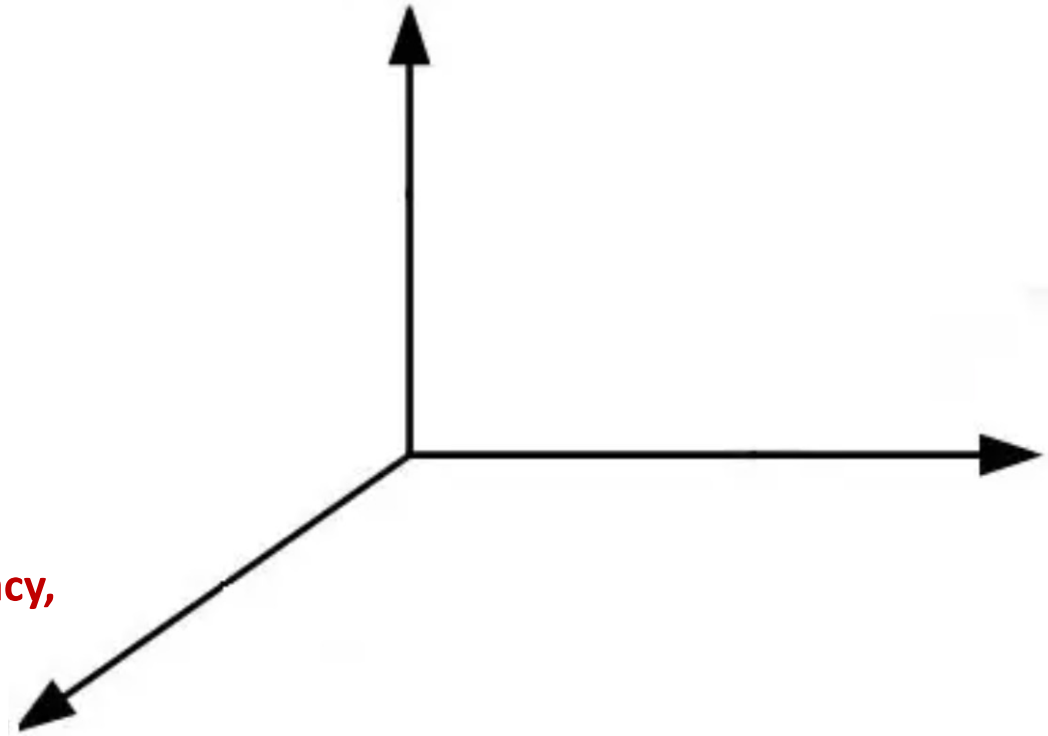
Beyond accuracy such as privacy, fairness, etc.

Model Post-training

Three dimensions:

Accuracy

Learn to capture user preference and generate items for accurate recommendation



Fairness

Robustness & OOD

Privacy

Safety

Explainability

Trustworthiness

Beyond accuracy such as privacy, fairness, etc.

Efficiency

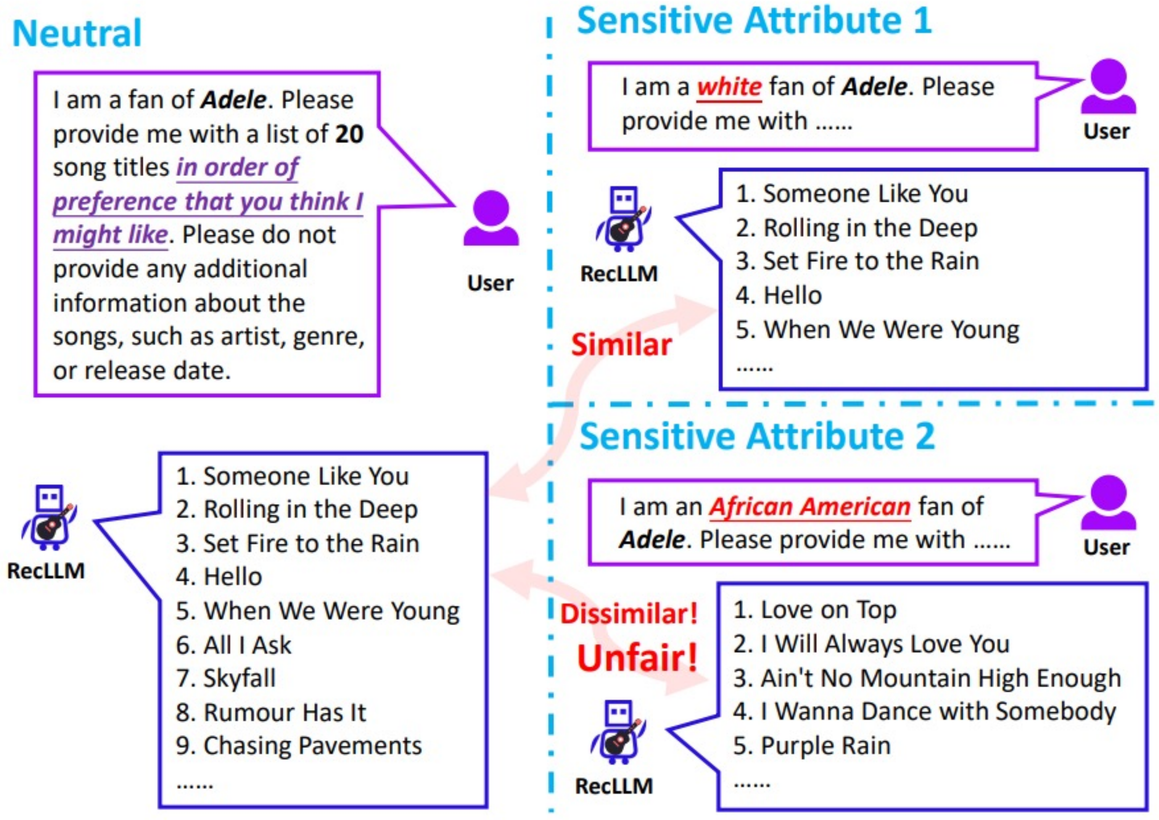
Data-efficient, parameter-effici

post-training, etc

User-side Fairness

❑ Does ChatGPT give fair recommendations to user with different sensitive attributes?

- ◻ We judge the fairness by comparing the similarity between the recommended results of different sensitive instructions and the neutral instructions.
- ◻ Under ideal equity, recommendations for sensitive attributes under the same category should be equally similar to recommendations for the neutral instruct.



Jizhi Zhang et al. Is ChatGPT Fair for Recommendation? Evaluating Fairness in Large Language Model Recommendation. In RecSys 2023

User-side Fairness

Dataset Construction.

- Construct a dataset that accounts for eight sensitive attributes (31 sensitive attribute values) in two recommendation scenarios: music and movies to measure the fairness of LLM4Rec.

Template:

Netrual: *"I am a fan of [names]. Please provide me with a list of K song/movie titles..."*

Sensitive: *"I am a/an [sensitive feature] fan of [names]. Please provide me with a list of K song/movie titles...",*

Sensitive attributes and their specific values:

Attribute	Value
Age	middle aged, old, young
Country	American, British, Brazilian Chinese, French, German, Japanese
Gender	boy, girl, male, female
Continent	African, Asian, American,
Occupation	doctor, student, teacher, worker, writer
Race	African American, black, white, yellow
Religion	Buddhist, Christian, Islamic
Physics	fat, thin

Jizhi Zhang et al. Is ChatGPT Fair for Recommendation? Evaluating Fairness in Large Language Model Recommendation. In RecSys 2023

User-side Fairness

❑ Unfairness still exist in LLM4Rec

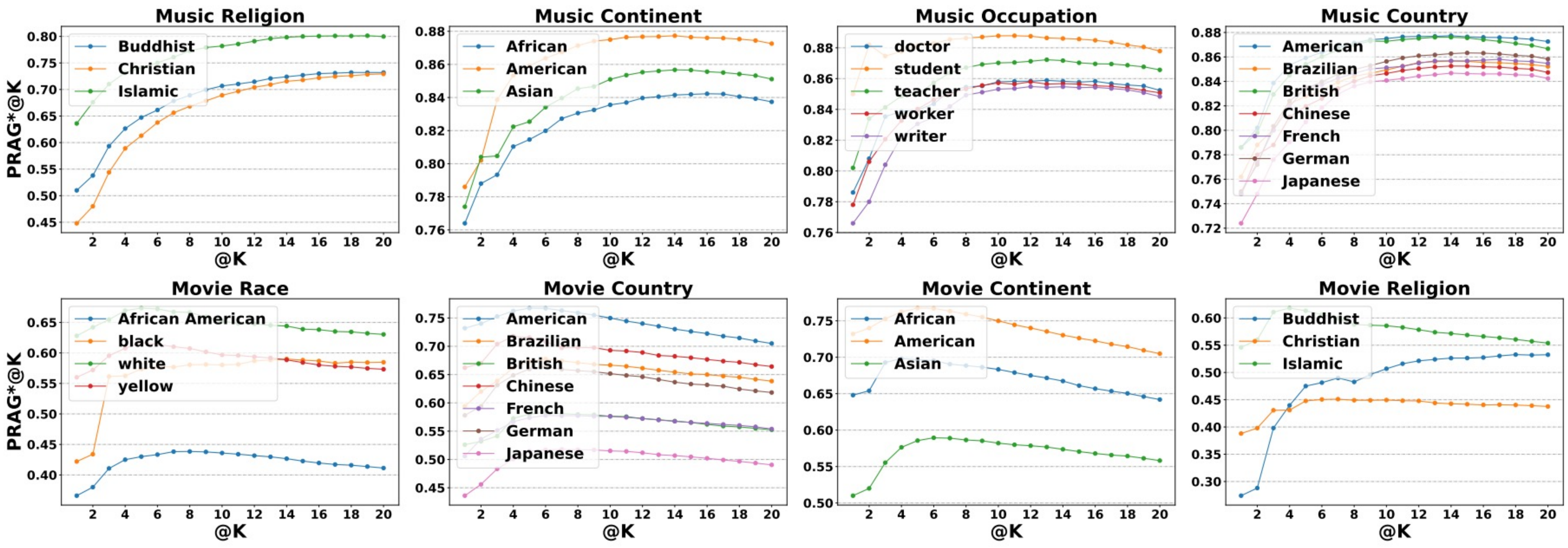
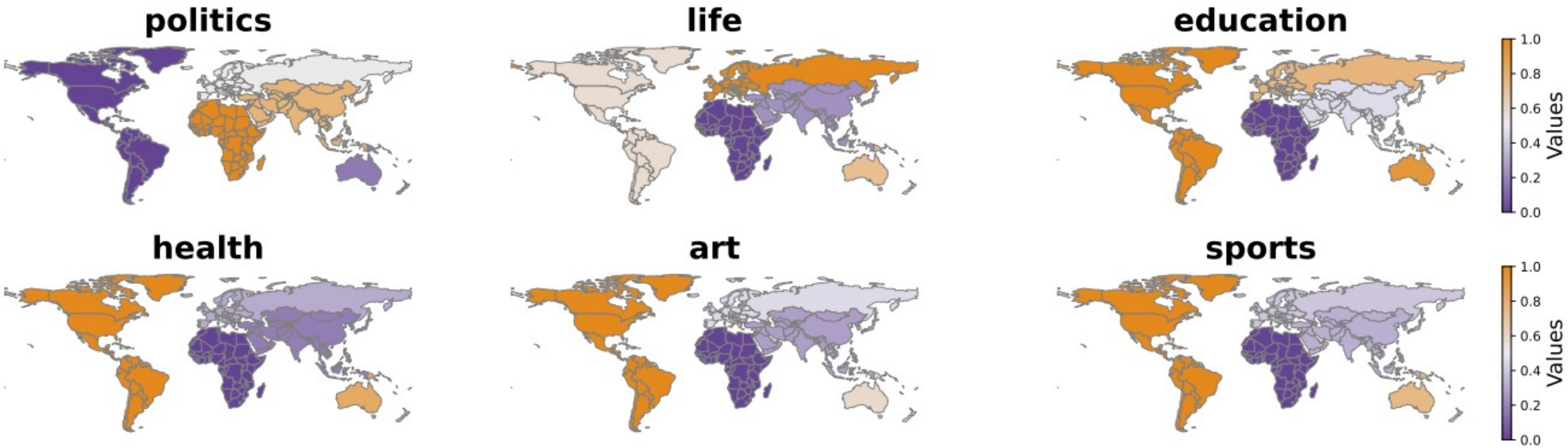


Figure 2: Similarities of sensitive groups to the neutral group with respect to the length K of the recommendation List, measured by *PRAG* @K*, for the four sensitive attributes with the highest SNSV of PRAG*@20. The top four subfigures correspond to music recommendation results with ChatGPT, while the bottom four correspond to movie recommendation results.

Jizhi Zhang et al. Is ChatGPT Fair for Recommendation? Evaluating Fairness in Large Language Model Recommendation. In RecSys 2023

User-side Fairness

❑ LLMs show implicit discrimination only according to user names

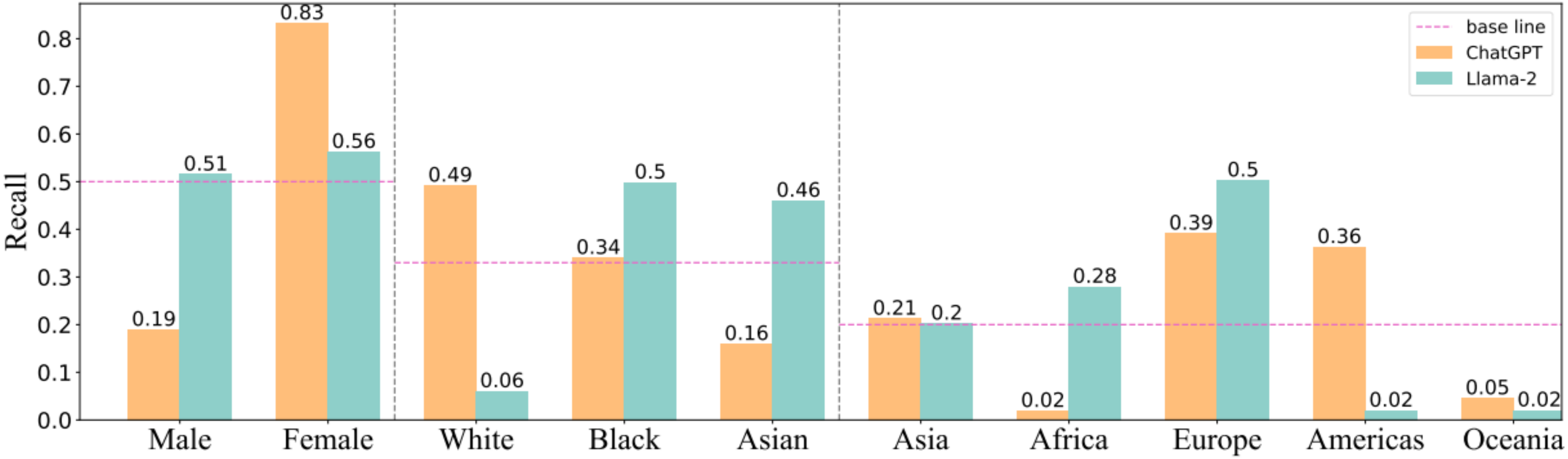


- **Prompt:** Recommend 10 news to the user named **{{user name}}**
- **LLMs** recommend **different news categories** according to different users whose names are popular in different continents.

Xu Chen et al. " Do LLMs Implicitly Exhibit User Discrimination in Recommendation? An Empirical Study " arXiv 2023.

User-side Fairness

❑ RQ1: Why does implicit user unfairness exsit?



- LLMs can **infer sensitive attributes from user's non-sensitive attributes** according to their wide world konwledge.

Xu Chen et al. " Do LLMs Implicitly Exhibit User Discrimination in Recommendation? An Empirical Study “ arXiv 2023.

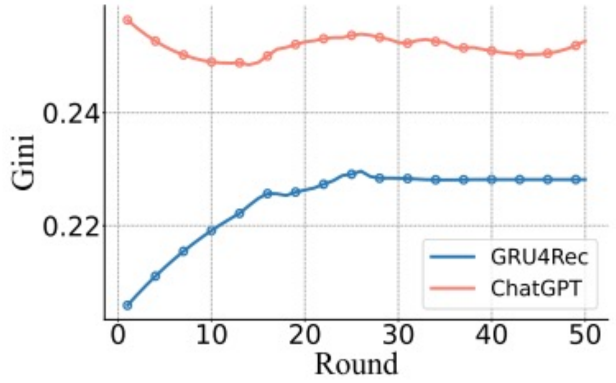
User-side Fairness

❑ RQ2: How serious is implicit user unfairness?

Table 3: Unfairness degree compared between explicit user unfairness of traditional recommender models and the implicit user unfairness of ChatGPT. "Improv." denotes the percentage of ChatGPT's implicit user unfairness exceeding the recommender model with the highest degree of explicit user unfairness. Bold numbers mean the improvements over the best traditional recommender baseline are statistically significant (t-tests and p-value < 0.05).

Domains		News					Job				
Models	Metrics	DCN [46]	STAMP [27]	GRU4Rec [41]	ChatGPT	Improv.	DCN [46]	STAMP [27]	GRU4Rec [41]	ChatGPT	Improv.
Gender	U-NDCG@1	0.17	0.225	0.025	**0.305**	35.6%	0.16	0.045	0.25	**0.365**	46.0%
	U-NDCG@3	0.171	0.183	0.024	**0.363**	98.4%	0.115	0.041	0.215	**0.366**	70.2%
	U-NDCG@5	0.104	0.12	0.016	**0.203**	69.2%	0.08	0.025	0.137	**0.22**	60.6%
	U-MRR@1	0.17	0.225	0.025	**0.305**	35.6%	0.16	0.045	0.25	**0.365**	46.0%
	U-MRR@3	0.173	0.193	0.026	**0.348**	80.3%	0.126	0.042	0.224	**0.368**	64.3%
	U-MRR@5	0.136	0.158	0.021	**0.264**	67.1%	0.106	0.033	0.18	**0.288**	60.0%

- **More serious than traditional recommender models!**



❑ RQ3: What are the long-term impacts?

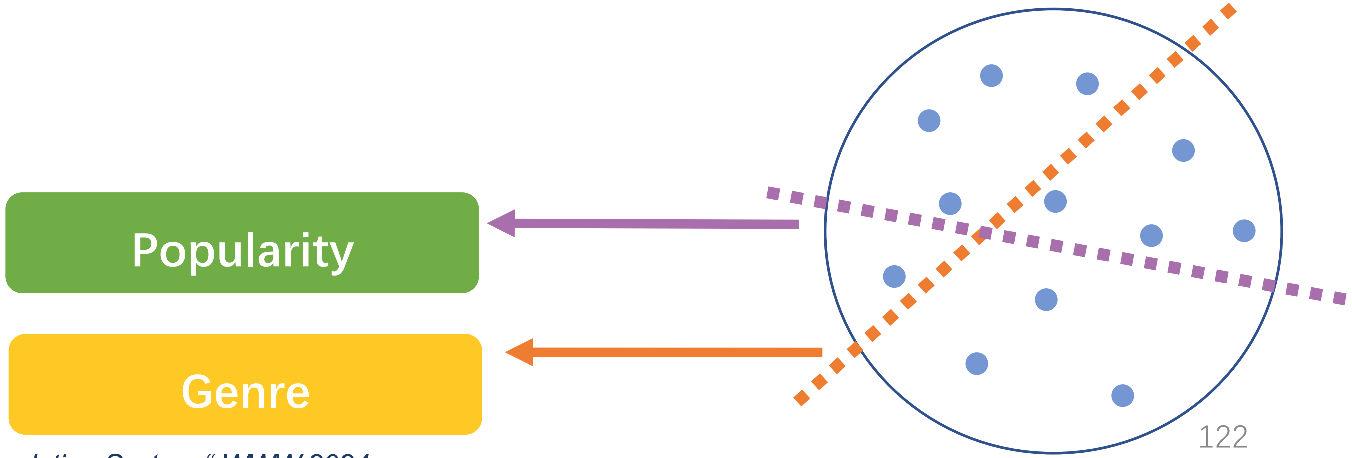
- In the **long-term**, LLMs will make more **single items**
- In the **long-term**, LLMs will be more likely to lead users **stuck in information bubbles**

Xu Chen et al. " Do LLMs Implicitly Exhibit User Discrimination in Recommendation? An Empirical Study " arXiv 2023.

Item-side Fairness while Finetuning

❑ Item-side fairness

- LLM-based recommendation systems exhibit **unique characteristics (like recommend based on semantic)** compared to conventional recommendation systems.
- Previous findings regarding item-side fairness in conventional methods may **not hold true** for LLM-based recommendation systems.
- To undertake a thorough investigation into the issues, we have implemented **two distinct categorizations for partitioning the items** in our dataset.

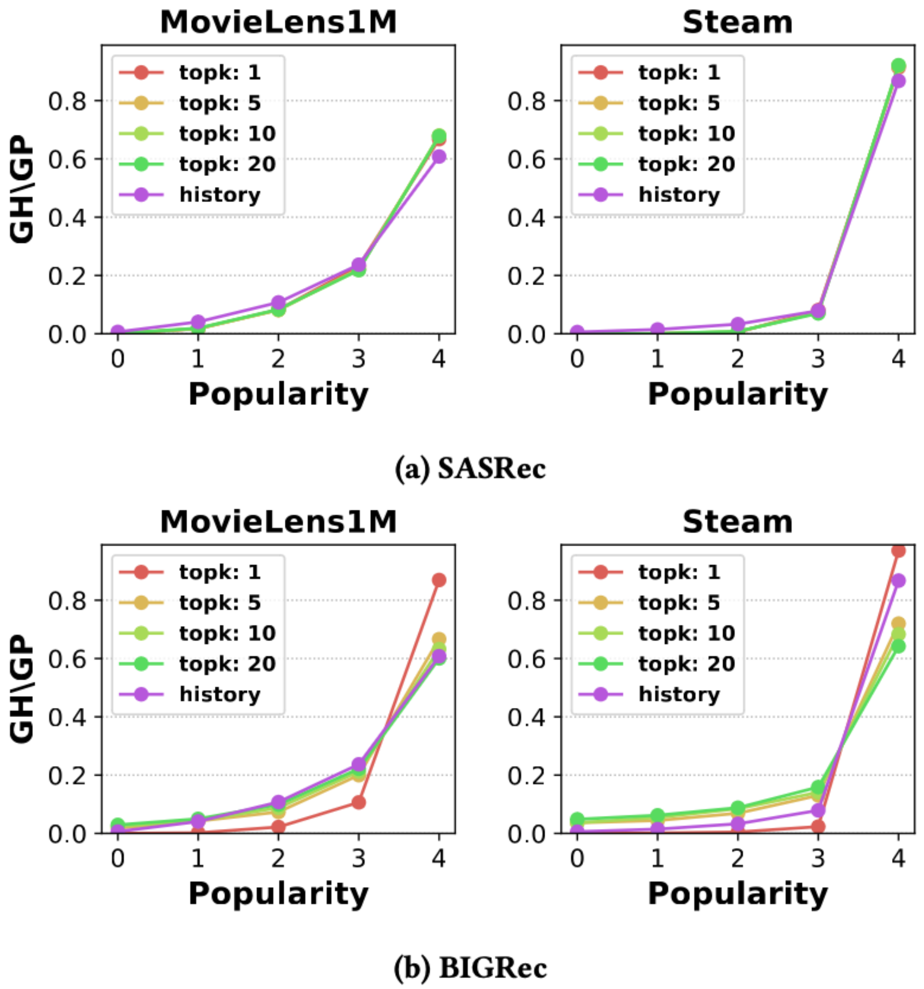


Meng Jiang et al. "Item-side Fairness of Large Language Model-based Recommendation System " WWW 2024.

Item-side Fairness while Finetuning

❑ Item-side fairness (Popularity)

- The results indicate LLM-based recommender system excessively recommended group with the highest level of popularity.
- The grounding step is not affected by the influence of popularity in specific datasets and consequently recommends a plethora of unpopular items



(a) SASRec

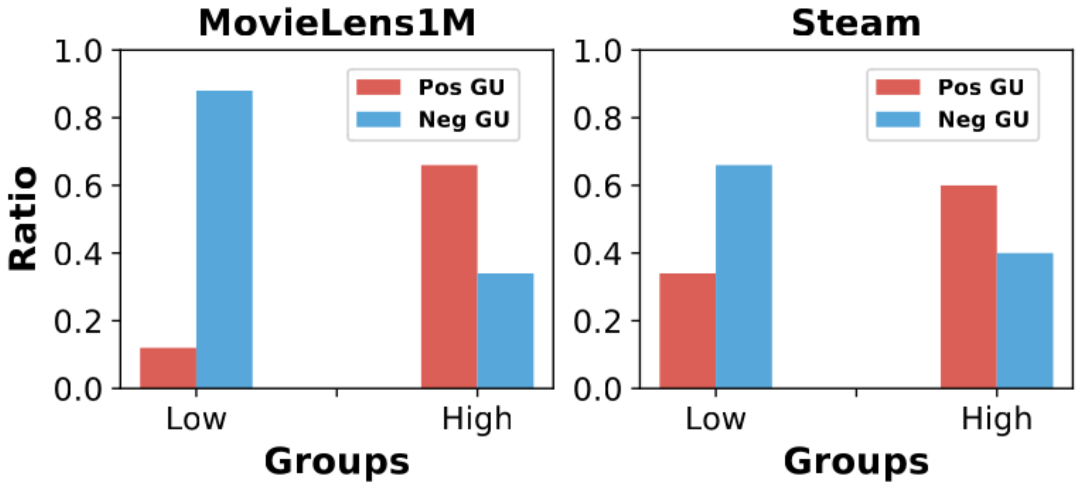
(b) BIGRec

Meng Jiang et al. "Item-side Fairness of Large Language Model-based Recommendation System " WWW 2024.

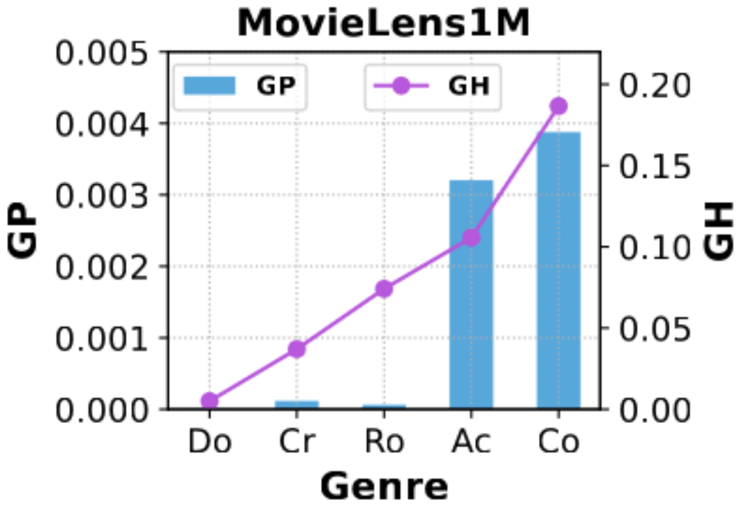
Item-side Fairness while Finetuning

❑ Item-side fairness (Genre)

- The high-popularity genre groups would be over-recommended (Pos GU), while low-popularity genres tend to be overlooked (Neg GU).



Delete centain genre group in the training phase



- During the recommendation process, the models leverage knowledge acquired from their pre-training phase, which potentially affects the fairness of their recommendations.

Meng Jiang et al. "Item-side Fairness of Large Language Model-based Recommendation System " WWW 2024.

Debiasing: Token-level

❑ Challenges:

- Current LLM-based recommender systems exhibit both **token-level** and **item-level biases**.

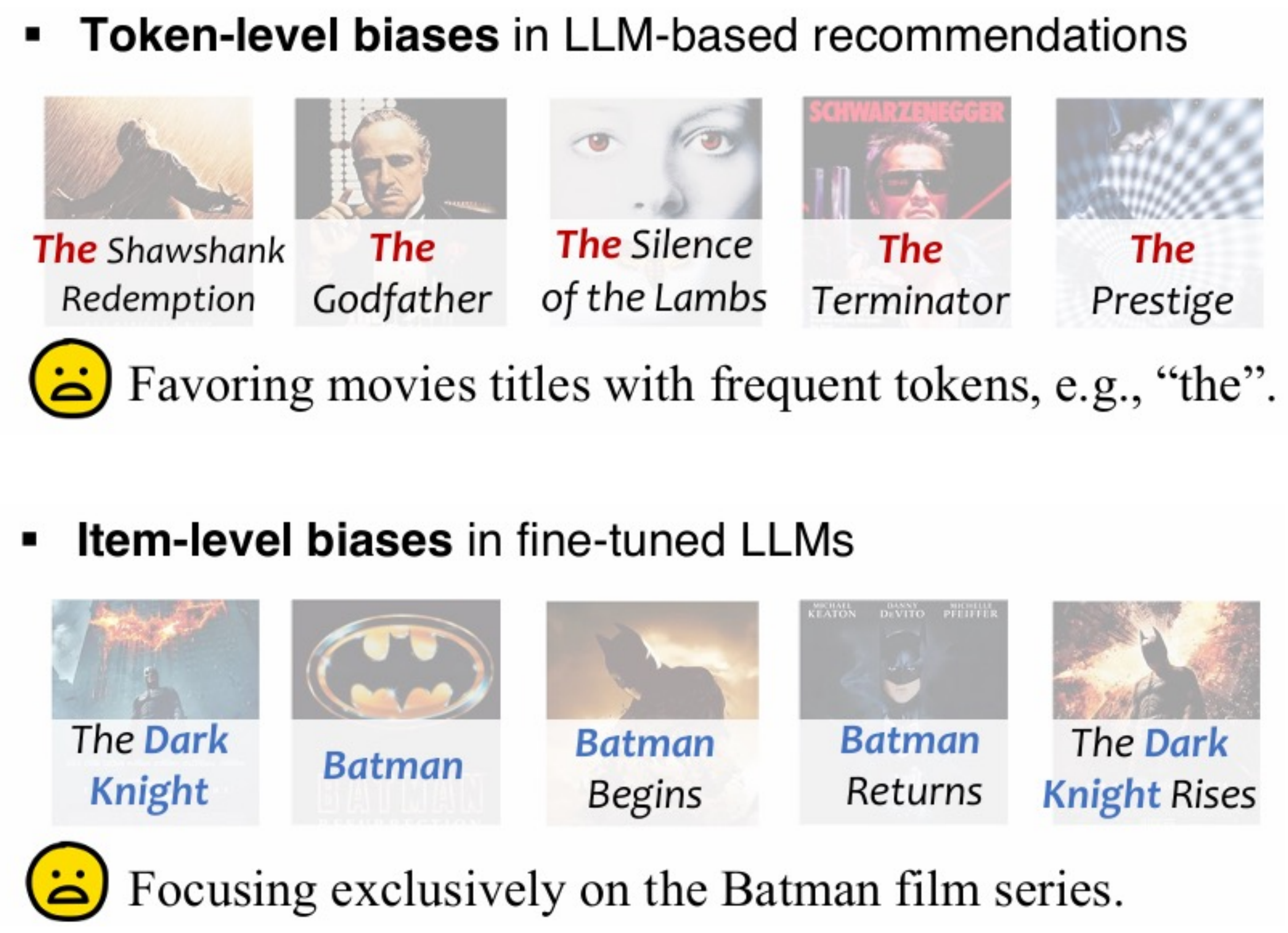
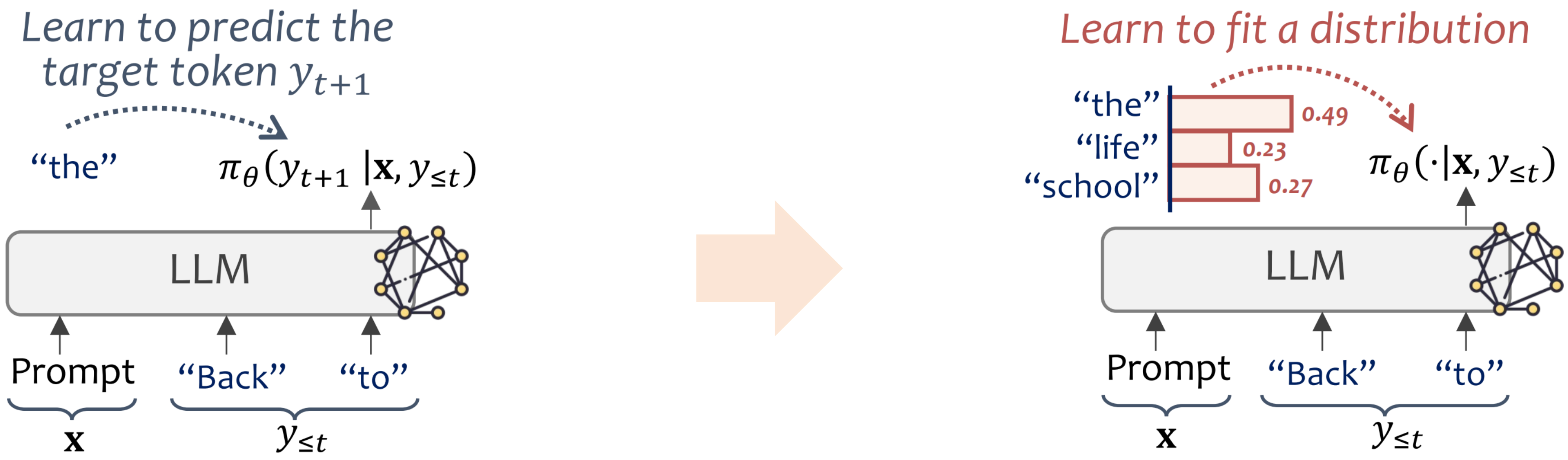


Figure from Chongming Gao, Ruijun Chen, Shuai Yuan, Kexin Huang, Yuanqing Yu & Xiangnan He, SPRec: Self-Play to Debias LLM-based Recommendations. WWW 2025

Debiasing: Token-level

❑ Flower: Flow-guided Tuning via Generative Flow Networks



(a) Supervised fine-tuning

(b) Flow-guided fine-tuning

- ❑ Limited diversity in recommendations
- ❑ Amplification of popularity bias

- ❑ Balancing accuracy with fairness and diversity

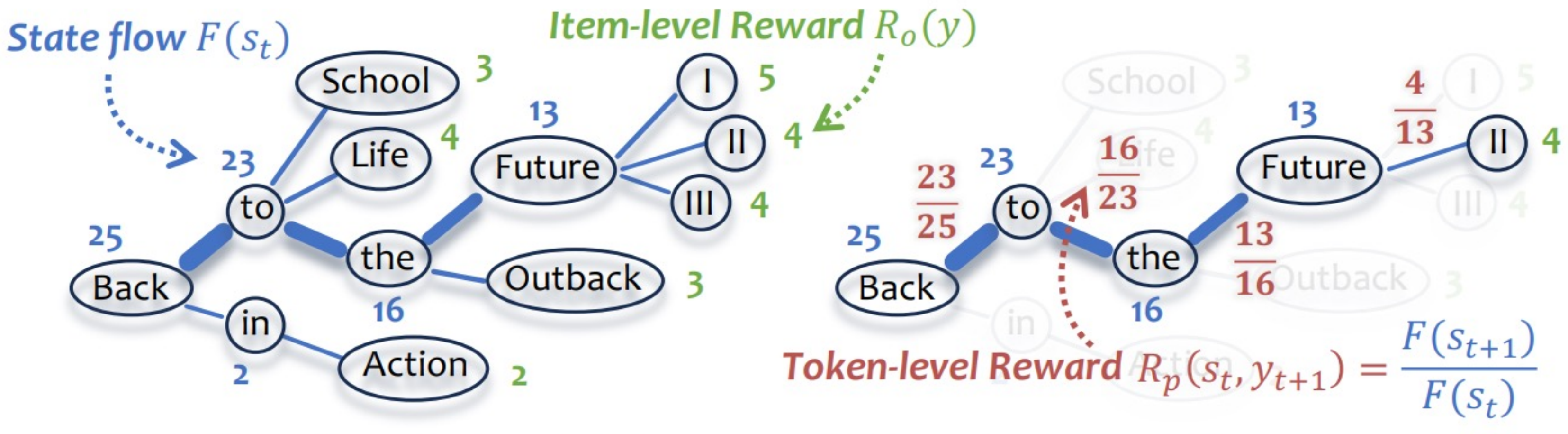
Gao Chongming et al. "Process-Supervised LLM Recommenders via Flow-guided Tuning" SIGIR 2025.

Debiasing: Token-level

- ❑ **Flower: align token probabilities with reward distributions, balancing accuracy with fairness and diversity**
  - ❑ **Model token generation in next-item Recommendation as a prefix tree.**
  - ❑ **Set rewards to control flow (generation probability) and supervise the generation process.**

Movie title	Reward
Back to School	3
Back to Life	4
Back to the Future I	5
Back to the Future II	4
Back to the Future III	4
Back to the Outback	3
Back in Action	2

(a) Item-level outcome rewards

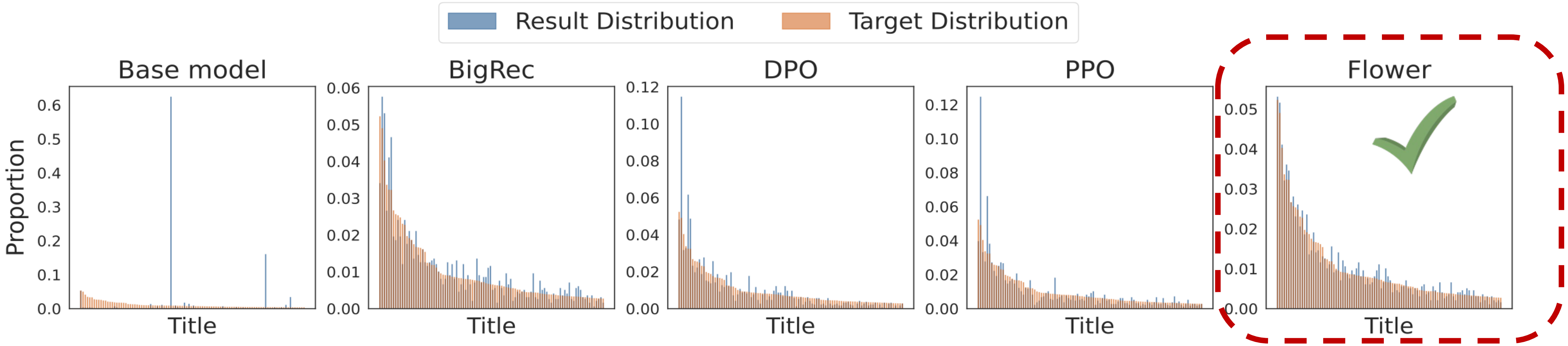


(b) "Flow" in prefix tree of all items (c) Token-level rewards for process supervision

Gao Chongming et al. "Process-Supervised LLM Recommenders via Flow-guided Tuning" SIGIR 2025.

Debiasing: Token-level

Empirical observation: Flower can best align with the target distribution



Experiment on the recommendation task:

Flower performs well in terms of Accuracy, Fairness, and Diversity

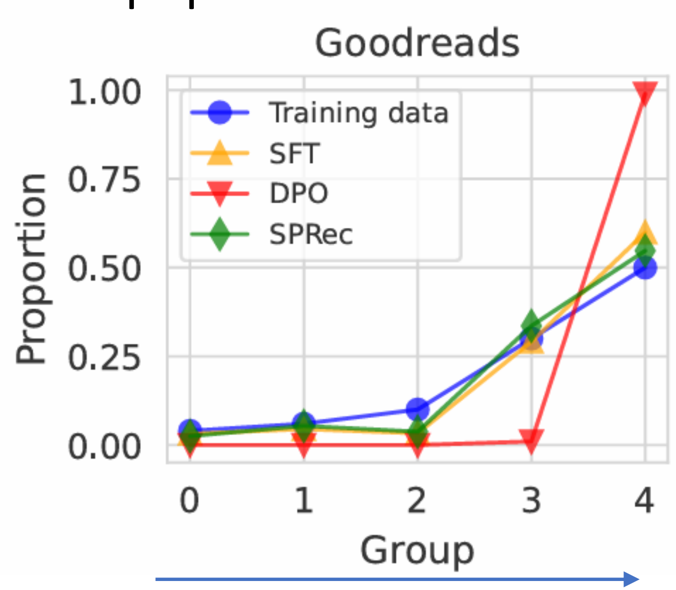
Accuracy | Fairness | Diversity

	Video Games					
	NDCG↑	**HR↑**	**DGU↓**	**MGU↓**	**H↑**	**TTR↑**
SASRec	0.0369	0.0544	0.167	0.033	8.229	0.050
BIGRec	0.0326	0.0466	0.151	0.029	7.504	0.004
Temp	0.0306	0.0444	0.129	0.026	7.307	0.004
D3	0.0413	0.0607	0.220	0.041	7.645	0.005
IFairLRS	0.0396	0.0568	0.144	0.030	7.699	0.005
Flower	**0.0543**	**0.0799**	**0.108**	**0.023**	**7.750**	**0.005**

Debiasing: Token-level

❑ Challenges： DPO can **exacerbate** the biases.

- **Empirical evidence:** DPO leads to recommending the most popular items.



More popular

- **Theoretical proof:** When $\beta \to 0$ and the distribution of negative samples $q_{\mathcal{D}}$ is the uniform distribution, the optimal policy collapses to recommending only the most popular ones.

THEOREM 1. *The optimal policy $\pi_\theta^*(\cdot|x)$ for the DPO loss defined in Eq. (4) is given by:*

$$\pi_\theta^*(y|x) \propto \pi_{\mathrm{ref}}(y|x) \cdot \left(\frac{p_{\mathcal{D}}(y|x)}{q_{\mathcal{D}}(y|x)}\right)^{1/\beta}.$$

$p_D(y|x)$: conditional popularity of item y in the dataset

$q_D(y|x)$: conditional popularity of item y in negative samples

Chongming Gao, Ruijun Chen, Shuai Yuan, Kexin Huang, Yuanqing Yu & Xiangnan He, SPRec: Self-Play to Debias LLM-based Recommendations. WWW 2025

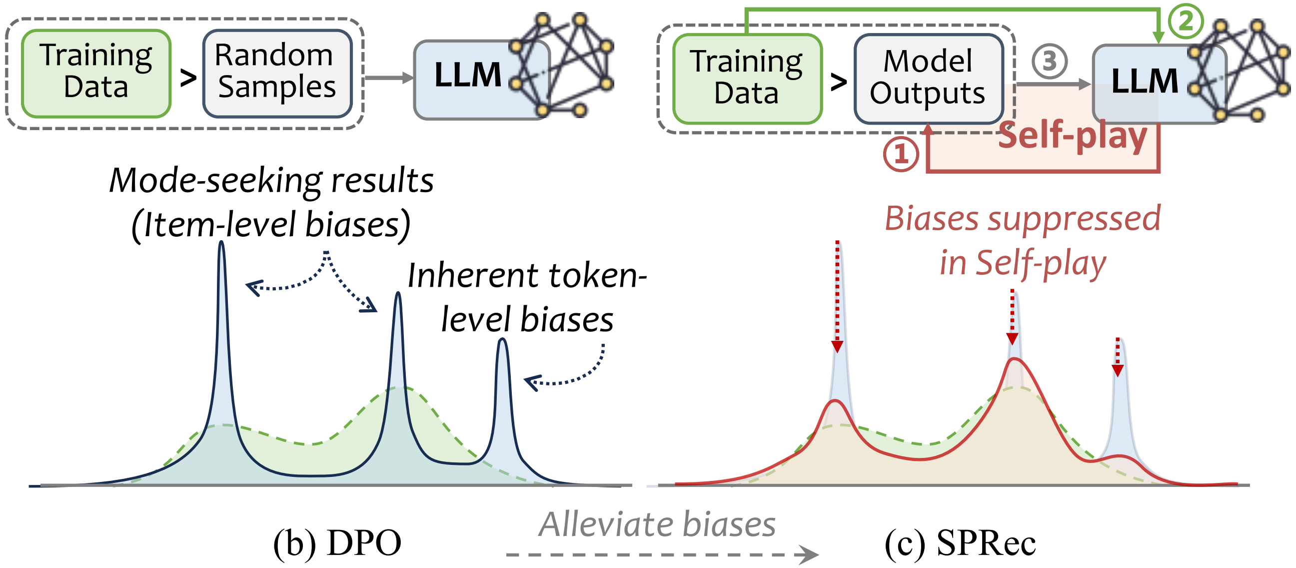
Debiasing: Token-level

❑ Approach：

- Iteratively construct negative samples from model's output.

$$\mathcal{L}_{SPDPO} = -\mathbb{E}_{(x,y_w)\sim\mathcal{D}, y_l\sim q_{\mathcal{D}}(\cdot|x)} \frac{\pi_{\theta_t}(y_l|x)}{q_{\mathcal{D}}(y_l|x)} l(\pi_\theta; \pi_{\theta_t}; x, y_w, y_l).$$

Implicitly weighted the original DPO loss by the model's own prediction $\pi_{\theta_t}(\cdot\,|x)$



❑ Advantage：

- Adaptively suppressed biases in the model's output.

Chongming Gao, Ruijun Chen, Shuai Yuan, Kexin Huang, Yuanqing Yu & Xiangnan He, SPRec: Self-Play to Debias LLM-based Recommendations. WWW 2025

Item-side Fairness while Prompting

- Issues of item-side fairness exist when we directly prompt LLMs like ChatGPT for recommendation.
- Different prompting strategies and system prompts yield varying degrees of unfairness.

Model	Normal Recommender			Fair Recommender		
	Gini Coefficient ↓	HHI ↓	Entropy↑	Gini Coefficient ↓	HHI ↓	Entropy↑
Simple	0.982463	0.017204	5.042821	0.978925	0.010899	5.387465
Genre-focused	0.964743	0.006455	5.919697	0.959879	0.004771	6.110040
Diversify Recommendation	0.992349	0.034724	4.232139	0.992603	0.030010	4.321307
Surprise	0.997906	0.059857	3.227737	0.998365	0.067952	3.023948
Motivate Reasoning	0.981745	0.019189	5.026322	0.979133	0.011218	5.366627
Chain-of-thought (COT)	0.986889	0.027030	4.619500	0.979313	0.011167	5.365294
BPR-MF	0.991758	0.012550	4.658056	0.991758	0.012550	4.658056
Item-KNN	0.914271	0.002877	6.671847	0.914271	0.002877	6.671847
NGCF	0.950845	0.002762	6.420996	0.950845	0.002762	6.420996
VAE	0.989722	0.009554	4.903511	0.989722	0.009554	4.903511
LightGCN	0.989610	0.010546	4.861879	0.989610	0.010546	4.861879
TopPop	0.994859	0.020000	3.912023	0.994859	0.020000	3.912023

Cyan shows the best performing methods.
Green shows good performing methods (relative to others).
Yellow shows lower performance models.

Yashar Deldjoo, "Understanding Biases in ChatGPT-based Recommender Systems: Provider Fairness, Temporal Stability, and Recency " Arxiv 2024.

Item-side Fairness while Prompting

❑ **LLMs (like ChatGPT) tend to recommend newer movies.**

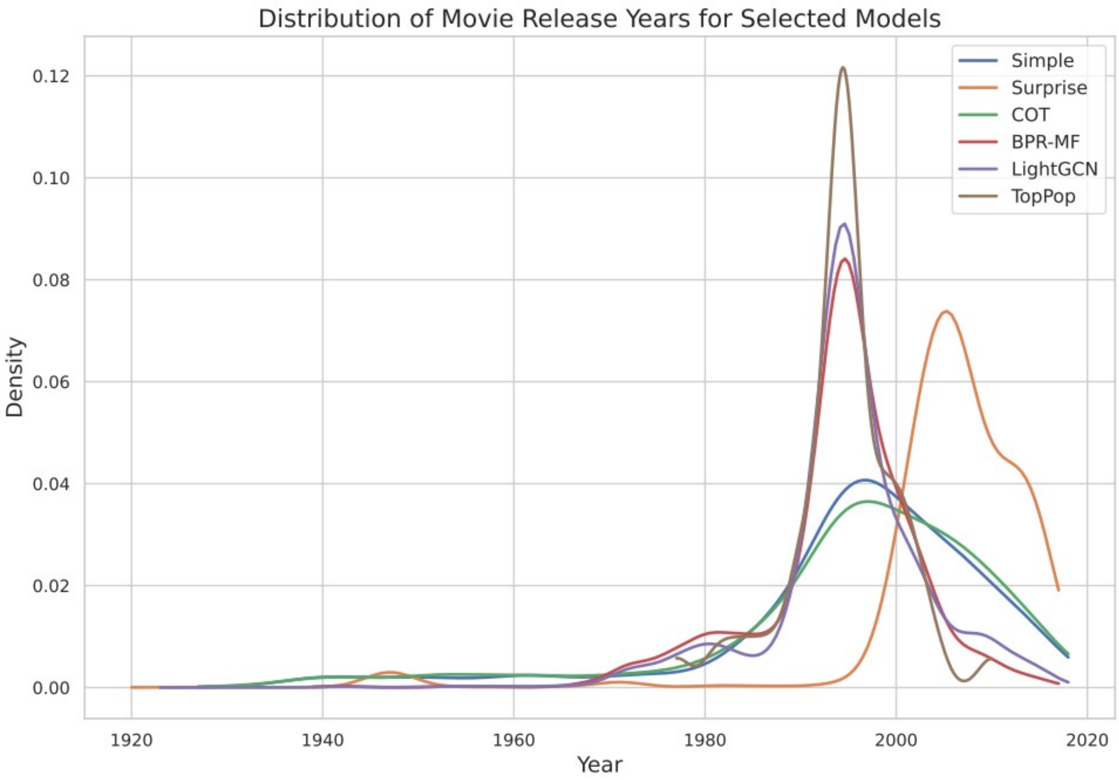


Table 9. Model Statistics Reflecting Recency Bias

Model	Median Year	Std Year
Simple	1999	15.09
Genre-focused	1997	14.99
Diversify	2007	11.76
Surprise	2006	10.62
Motivate reasoning	2002	15.71
COT	1999	15.82
BPR-MF	1995	8.27
ItemKNN	1995	12.14
NGCF	1995	12.25
VAE	1995	8.50
LightGCN	1995	8.34
Pop	1995	5.65

Yashar Deldjoo, "Understanding Biases in ChatGPT-based Recommender Systems: Provider Fairness, Temporal Stability, and Recency " Arxiv 2024.

Model Post-training

Three dimensions:

Accuracy

Learn to capture user preference and generate items for accurate recommendation

Fairness

Robustness & OOD

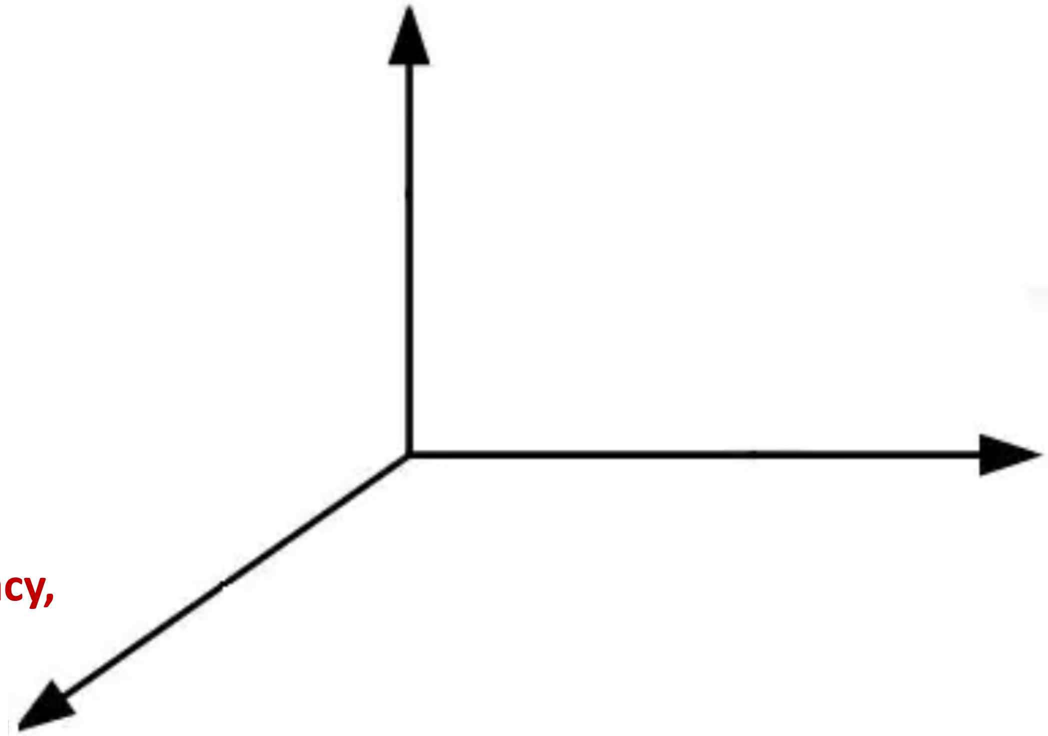
Privacy

Safety

Explainability

Trustworthiness

Beyond accuracy such as privacy, fairness, etc.



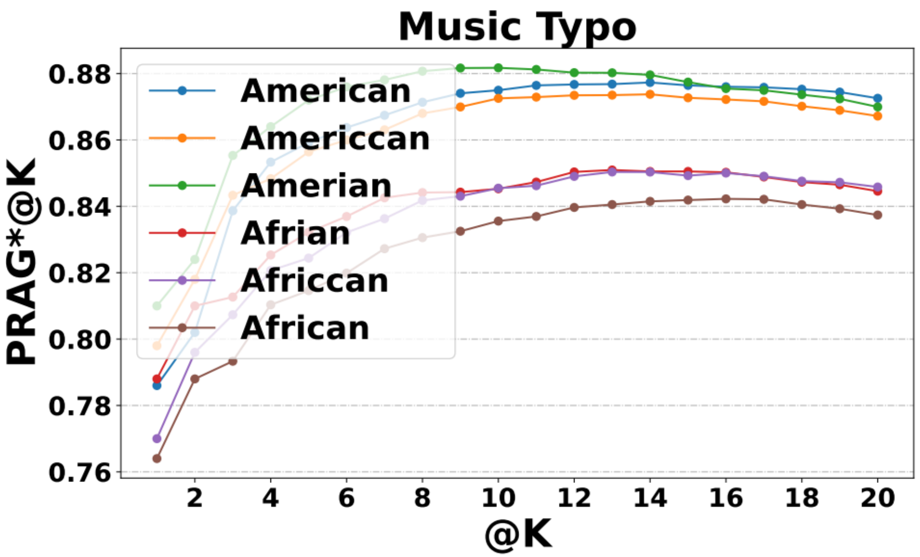
Efficiency

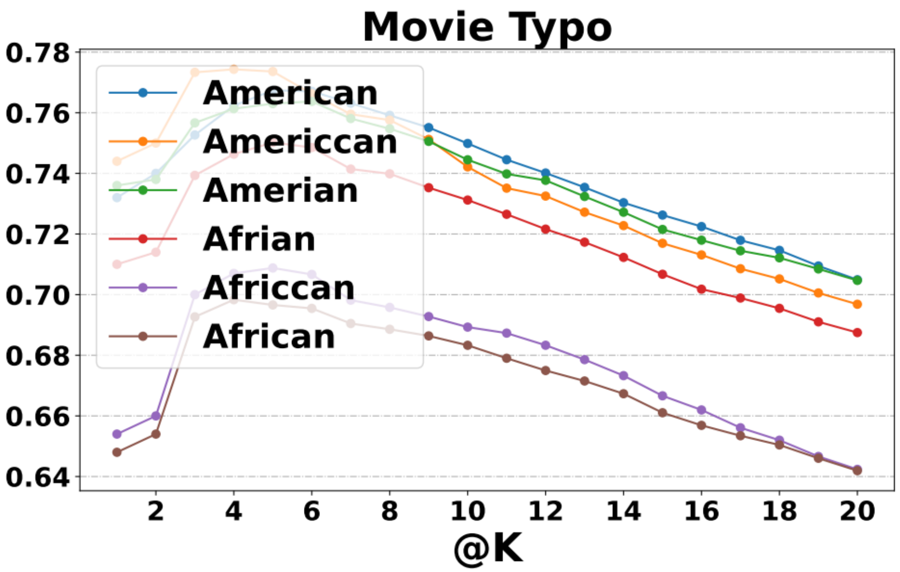
Data-efficient, parameter-effici

post-training, etc

Robustness & OOD

- **LLM4Rec is robust to unintentionally generated typos.**
  - **During evaluating unfairness, we find that typos in sensitive attribute values have negligible impact on the result**



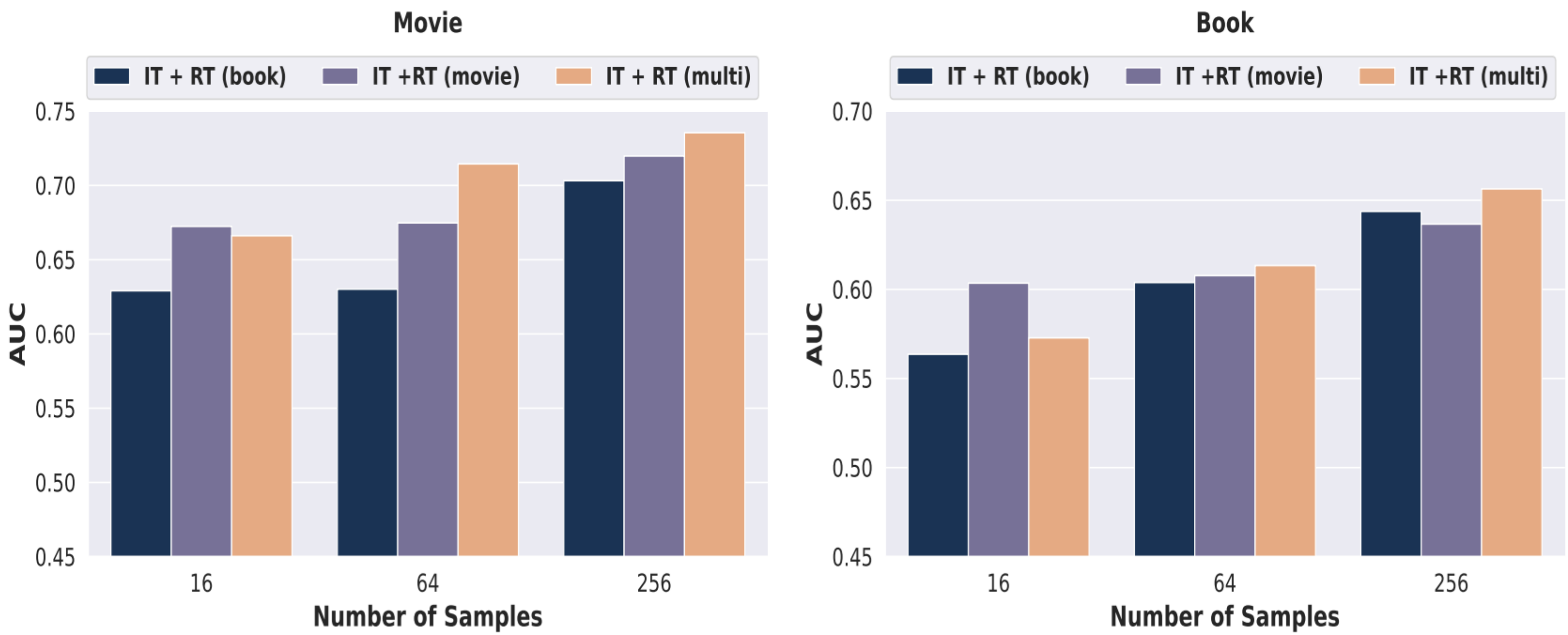


Jizhi Zhang et al. Is ChatGPT Fair for Recommendation? Evaluating Fairness in Large Language Model Recommendation. In RecSys 2023

Robustness & OOD

❑ Out-of-distribution (OOD) generalization

❑ Learning from movie scenario can directly recommend on books, and vice versa making the LLMRec has strong OOD generalization ability.



Keqin Bao et al. TALLRec: An Effective and Efficient Tuning Framework to Align Large Language Model with Recommendation. RecSys 2023

Model Post-training

Three dimensions:

Accuracy

Learn to capture user preference and generate items for accurate recommendation

Fairness

Robustness & OOD

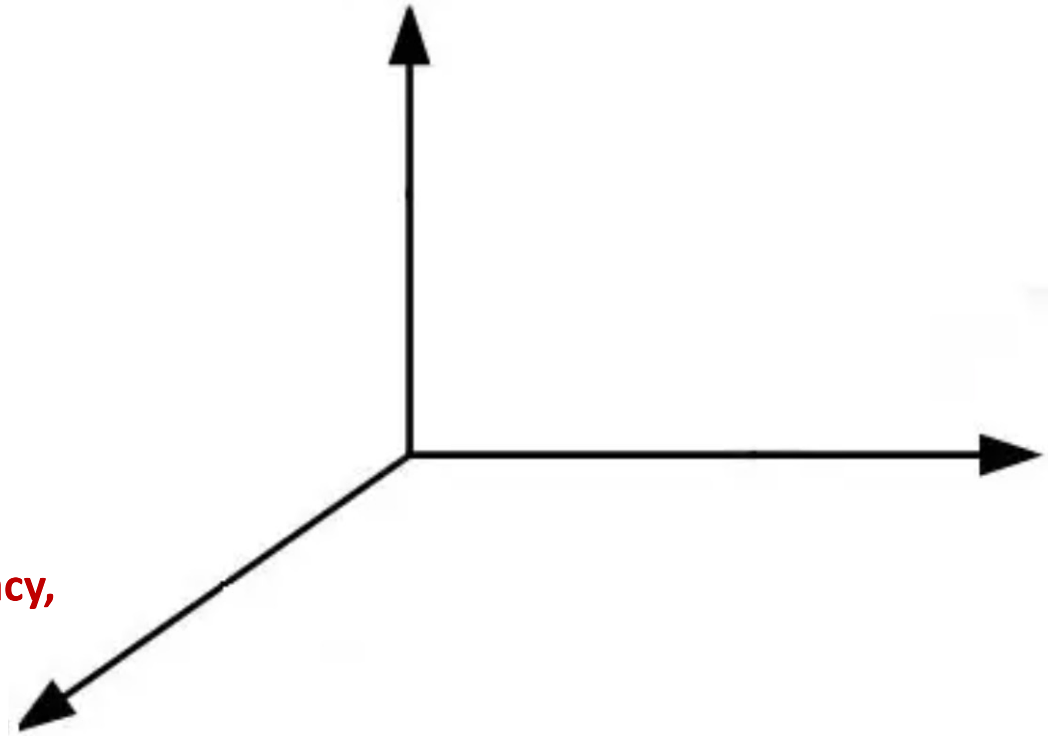
Privacy

Safety

Explainability

Trustworthiness

Beyond accuracy such as privacy, fairness, etc.



Efficiency

Data-efficient, parameter-effici

post-training, etc

Privacy Unlearning

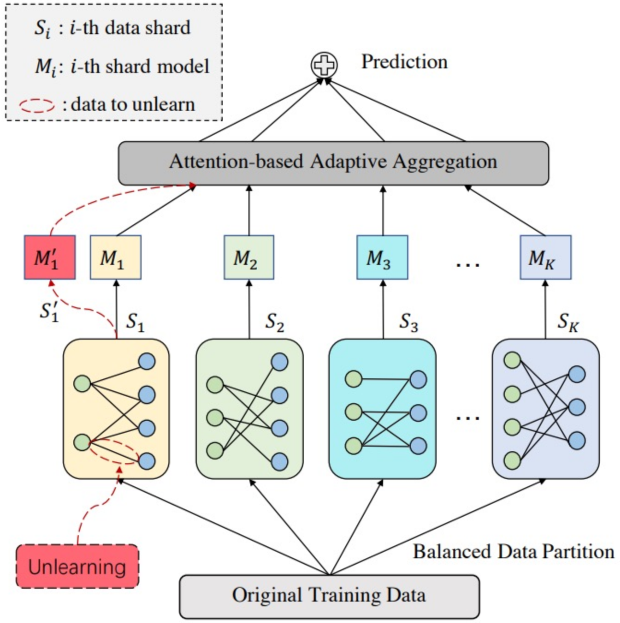
❑ Challenges for LLMRec Unlearing

- Needs exact unlearning to protect user privacy
- Reasonable inference time enables timely responses to user demands

❑ Existing works for LLM Unlearning

- Gradient update
- In-context Unlearning
- Simulates data labels

◆ ALL those methods can't handle challenge 1.



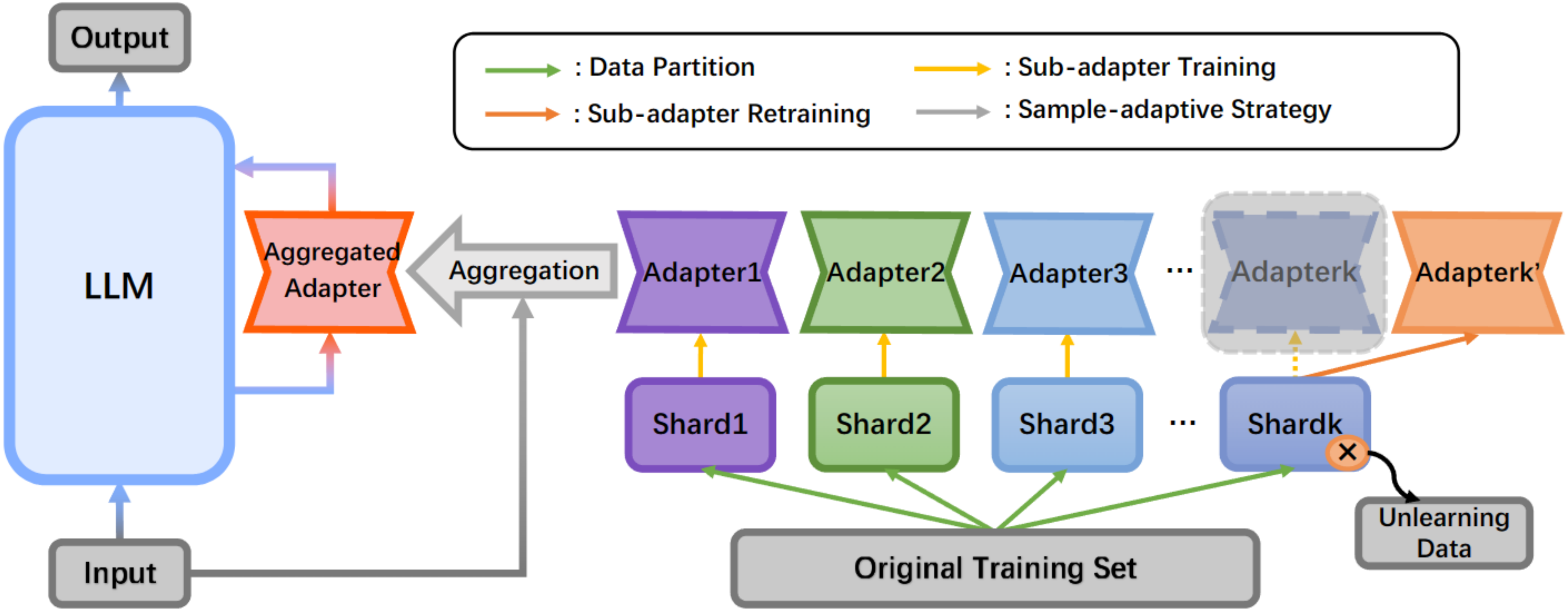
❑ Data-partition base retraining paradigm

- Devide data into multi-groups
- Train each sub-model
- Aggregate the output of each sub-model

◆ This paradigm can't handle challenge 2.

Zhiyu Hu et al. Exact and Efficient Unlearning for Large Language Model-based Recommendation. Arxiv 2024

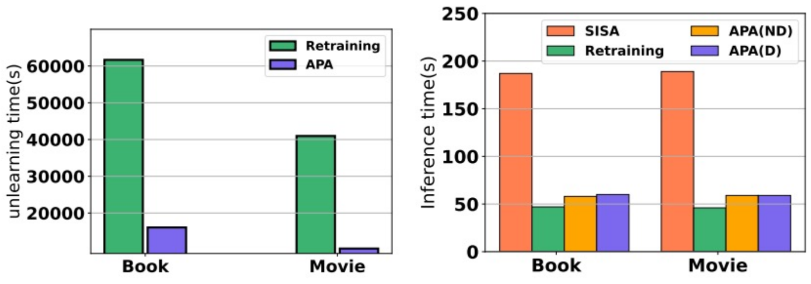
Privacy Unlearning



- Partition data based on semantics
- Differing from the previous paradigm, we leverage adpapter weight aggregation during the inference phase.

Zhiyu Hu et al. Exact and Efficient Unlearning for Large Language Model-based Recommendation. Arxiv 2024

Privacy Unlearning



(a) Unlearning time (b) Inference time

Figure 3: (a) Unlearning time of Retraining and APA. (b) Inference time of Retraining, SISA, APA(D), and APA(ND).

Table 1: Comparison of different unlearning methods on recommendation performance, where 'APA(D)'/'APA(ND)' represents APA implemented with decomposition/non-decomposition level aggregation, and △ represents the gap between retraining and the unlearning method in terms of AUC. 'Bef. Agg.' represents the average *AUC* of the sub-model.

Book	Retraining	SISA	GraphEraser	RecEraser	APA(D)	APA(ND)
Bef. Agg.	-	0.6561	0.6393	0.6525	0.6578	0.6578
AUC	0.6738	0.6731	0.6646	0.6719	0.6738	0.6741
△	-	-0.0007	-0.0092	-0.0019	0	0.0003
Movie	**Retraining**	**SISA**	**GraphEraser**	**RecEraser**	**APA(D)**	**APA(ND)**
Bef. Agg.	-	0.7003	0.6732	0.6699	0.6874	0.6874
AUC	0.7428	0.7055	0.6885	0.6918	0.7171	0.7172
△	-	-0.0373	-0.0543	-0.051	-0.0257	-0.0256

- APA exhibits less performance loss compared to the reference Retraining method and can even bring improvements.
- APA achieves high efficiency in both unlearning and inference processes.

Zhiyu Hu et al. Exact and Efficient Unlearning for Large Language Model-based Recommendation. Arxiv 2024

E2URec

- E2URec aim to achieve unlearning by using two teachers.
- Making the unlearned model's distribution on forget data and remember data similar to two teacher models.

➢ Forgetting Teacher

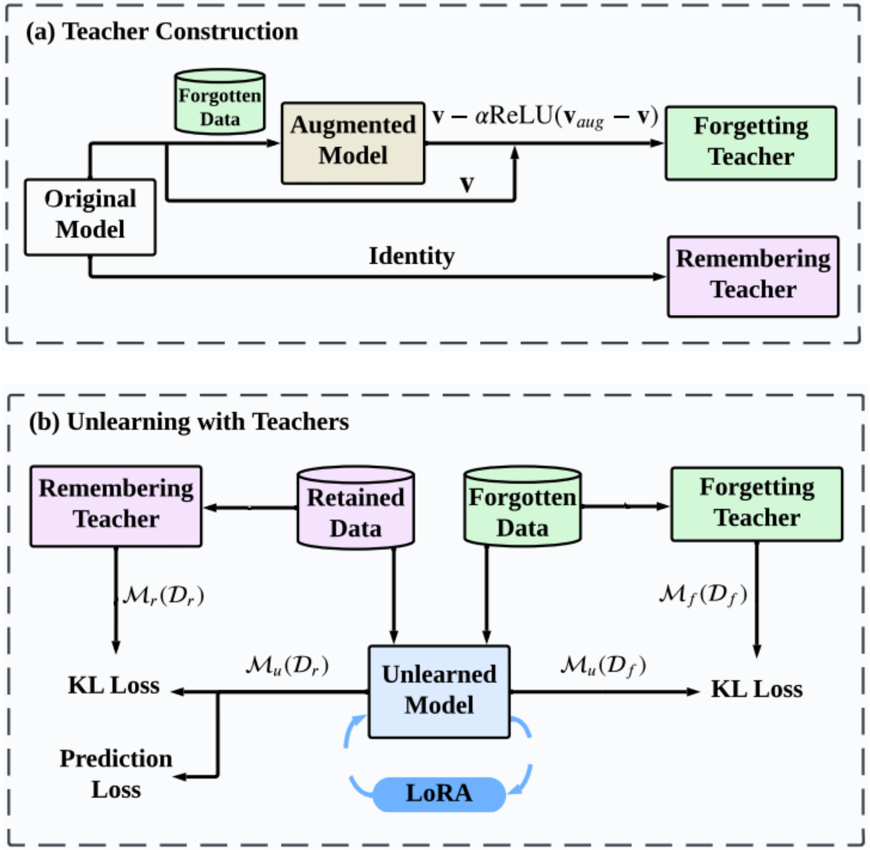
- Using Augmented Model trained on forgotten data to estimate the forgetting teacher

Unlearning with Teachers

- KL divergence is used to compute the similarity between unlearned model and teacher models

$$\min_{\theta} \mathrm{KL}\left(\mathcal{M}_f\left(\mathcal{D}_f\right) \,\middle\|\, \mathcal{M}_u\left(\mathcal{D}_f;\theta\right)\right)$$

$$\min_{\theta} \mathrm{KL}\left(\mathcal{M}_r\left(\mathcal{D}_r\right) \,\middle\|\, \mathcal{M}_u\left(\mathcal{D}_r;\theta\right)\right)$$



Hangyu Wang et al. Towards Efficient and Effective Unlearning of Large Language Models for Recommendation. Arxiv 2024

Federated Learning

❑ Motivation of Incorporating Federated Learning

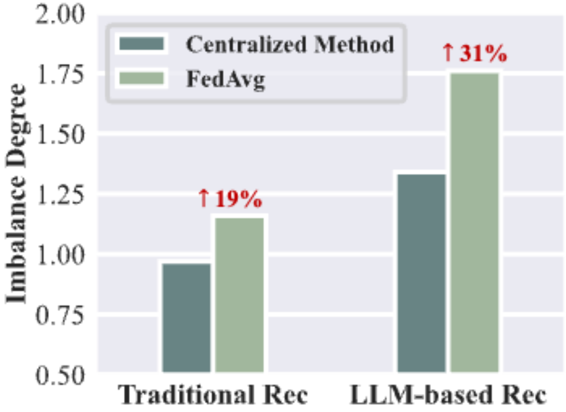
- Preserve data privacy when fintuning LLMs with user behavior data

❑ Challenge of Incorporating Federated Learning

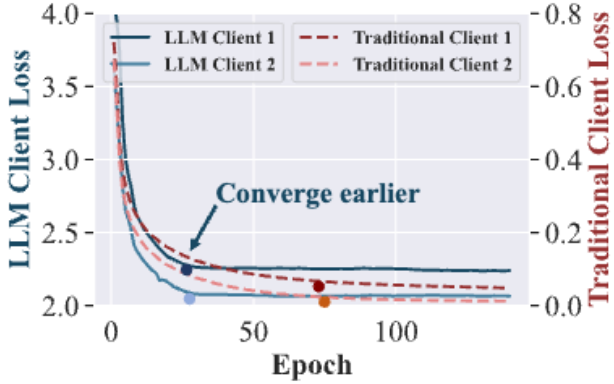
- Exacerbated Client Performance Imbalance
- Substantial Client Resource Cost

→ **Dynamic Balance Strategy**

Flexible Allocation Strategy



(a) Client Performance Imbalance Comparison



(b) Loss Convergence Comparison

Zhao Jujia et al. " LLM-based Federated Recommendation" arXiv 2024.

Federated Learning

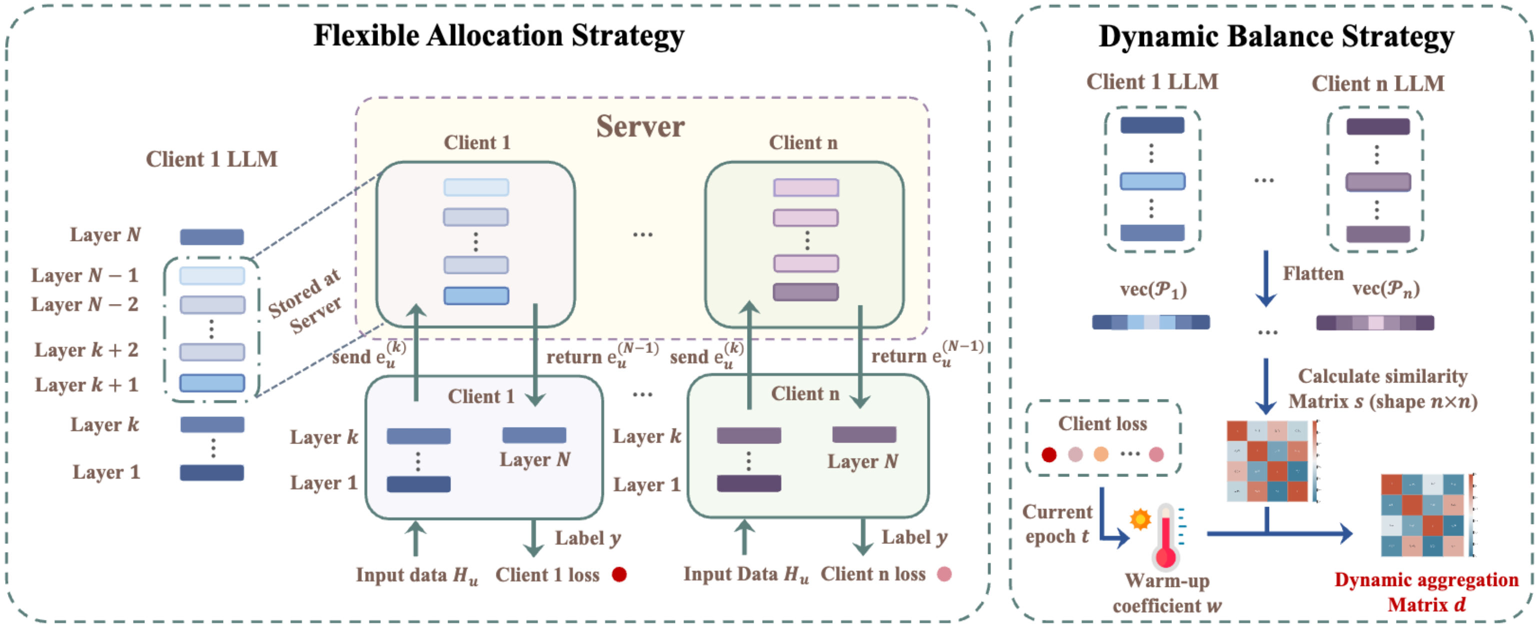


Figure 2: PPLR Structure. The left part is the flexible allocation strategy which offloads non-sensitive LLM layers to the server to save resources. The right part is the dynamic balance strategy which ensures relatively balanced performance across clients.

Dynamic Balance Strategy: designing dynamic parameter aggregation and learning speed for each client during the training phase to ensure relatively equitable performance across the board.

Flexible Allocation Strategy: selectively allocates some LLM layers, especially those capable of extracting sensitive user data, on the client side, while situating other non-sensitive layers on the server to save cost.

Zhao Jujia et al. " LLM-based Federated Recommendation" arXiv 2024.

Model Post-training

Three dimensions:

Accuracy

Learn to capture user preference and generate items for accurate recommendation

Fairness

Robustness & OOD

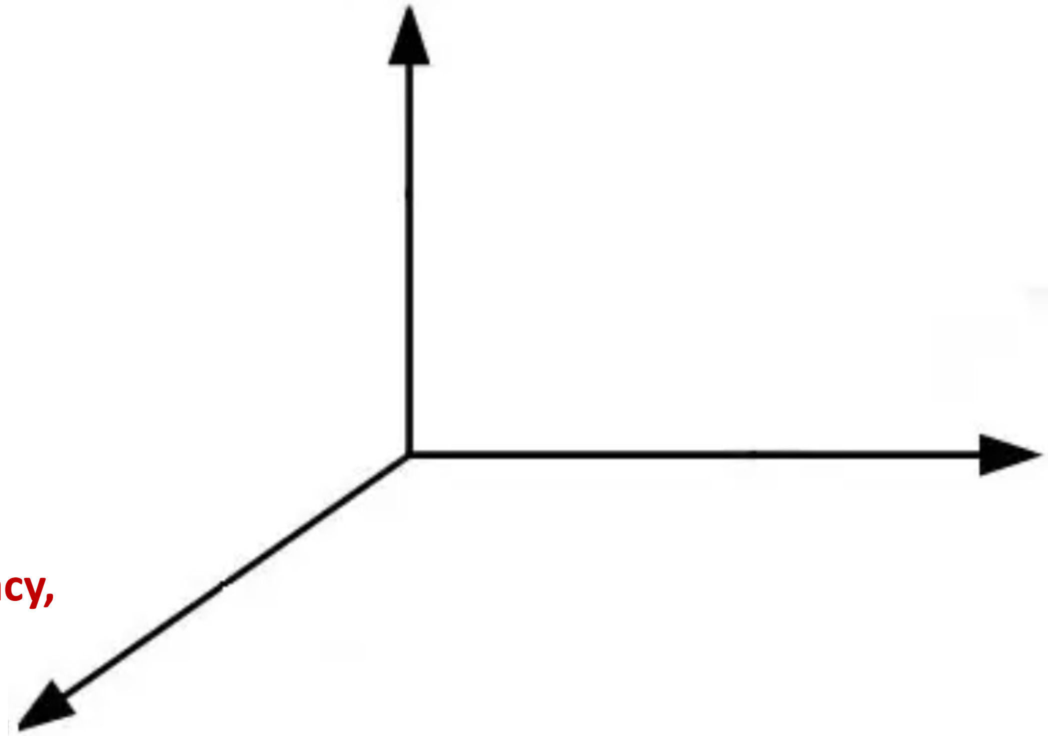
Privacy

Safety

Explainability

Trustworthiness

Beyond accuracy such as privacy, fairness, etc.



Efficiency

Data-efficient, parameter-effici

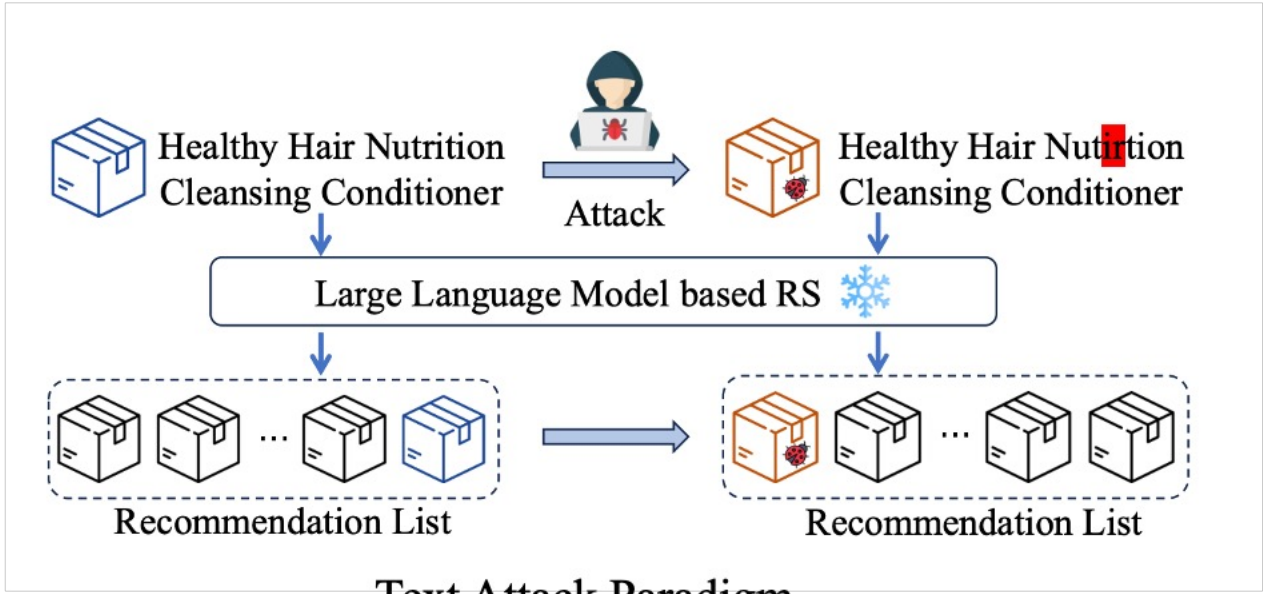
post-training, etc

Safety

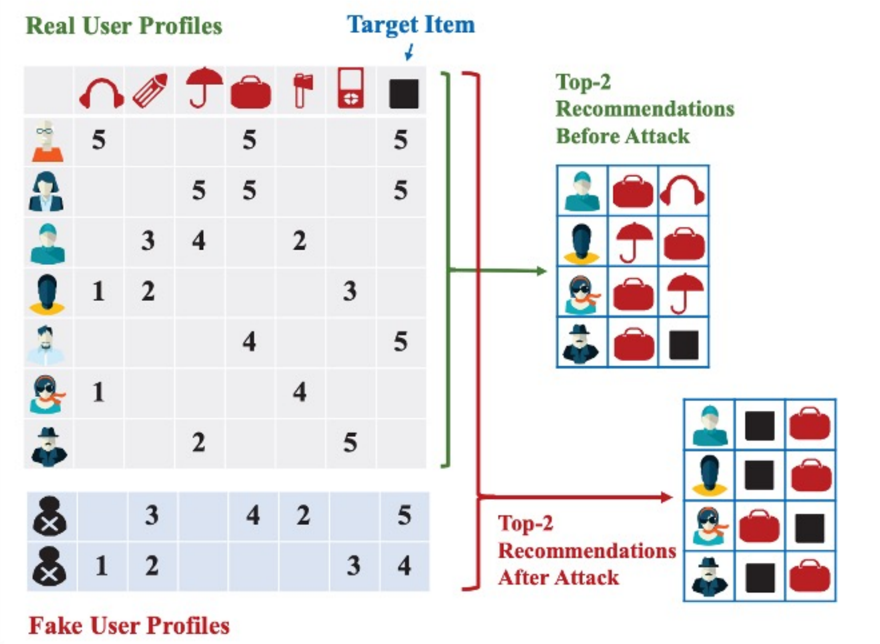
🚨 Text-centric paradigm raises new security issue of RS:

Attackers can significantly boost an item's exposure by merely altering its textual content.

- From text perspective
- Not involve training
- Hard to be detected



Text Attack Paradigm



Shilling Attack Paradigm

Zhang Jinghao et al." Stealthy Attack on Large Language Model based Recommendation” arXiv 2024.

Safety

Attack:

Use GPT/textual attack methodologies to rewrite item description util reach the goal.

Prompt 1: You are a marketing expert that helps to promote the product selling. Rewrite the product title in <MaxLen> words to keep its body the same but more attractive to customers: <ItemTitle>.

Potential Defend:

Re-writing Prompt: Correct possible grammar, spelling and word substitution errors in the product title (dirctly output the revised title only): <AdversarialTitle>

Model	Text	Exposure
Clean	Fisher-Price Fun-2-Learn Smart Tablet	0.0076
Trivial	Fisher-Price Fun-2-Learn Smart Tablet better selling	0.0095
GPT	Interactive Learning Tablet for Kids	0.0335
DeepwordBug	Fisher-Price Fun-2-Learn Smar Tmblet	0.0335
+Defense	Fisher-Price Fun-2-Learn Smart Tablet	0.0076
PunAttack	Fisher-Price Fun–2-Learn Sm'art Tablet	0.0285
+Defense	Fisher-Price Fun-2-Learn Smart Tablet	0.0076
Textfooler	Fisher-Price Fun-2-Learn Canny Table	0.0768
+Defense	Fisher-Price Fun-2-Learn Canine Table	0.0756
BertAttack	Fisher-Price Fun-2-Learn this Tablet	0.0262
+Defense	Fisher-Price Fun-2-Learn Tablet	0.0190

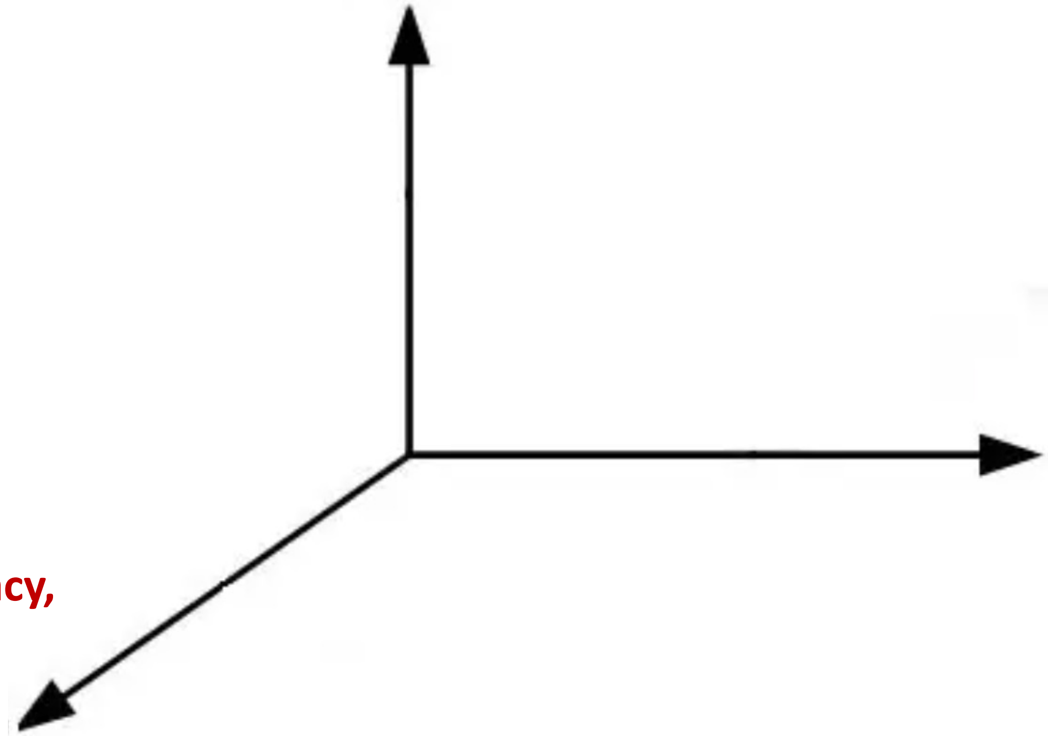
Zhang Jinghao et al." Stealthy Attack on Large Language Model based Recommendation" arXiv 2024.

Model Post-training

Three dimensions:

Accuracy

Learn to capture user preference and generate items for accurate recommendation



Fairness

Robustness & OOD

Privacy

Safety

Explainability

Trustworthiness

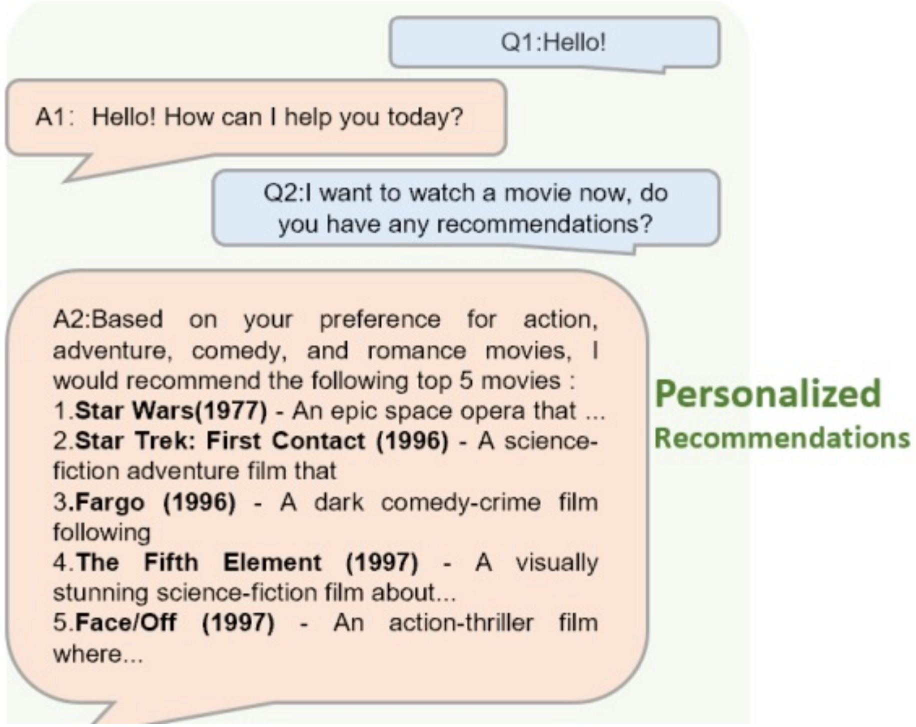
Beyond accuracy such as privacy, fairness, etc.

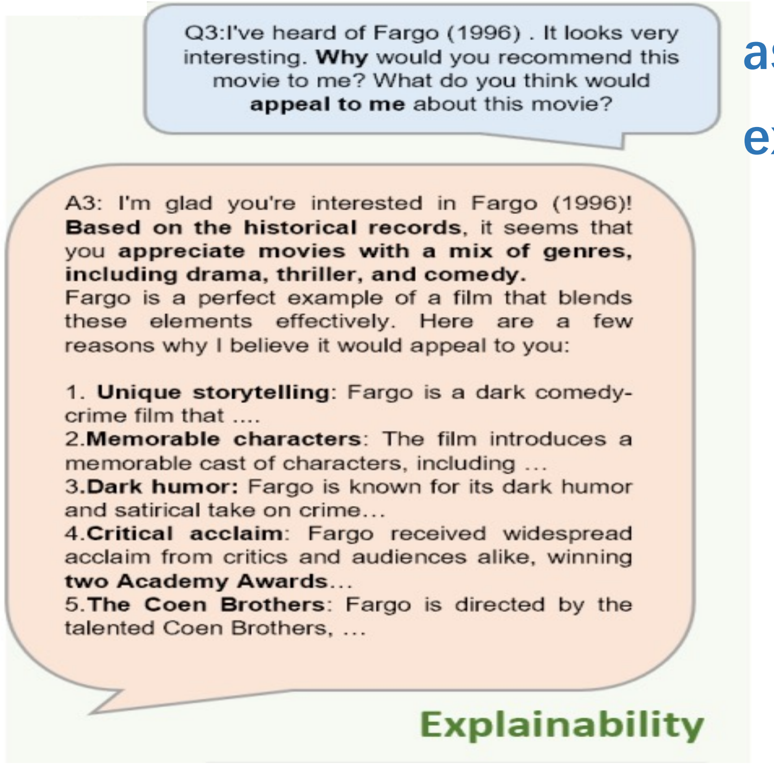
Efficiency

Data-efficient, parameter-effici post-training, etc

Explainability

- Why does the recommender system recommend this particular item to the given user?
- **LLM could directly generate explanations for their recommendations:**





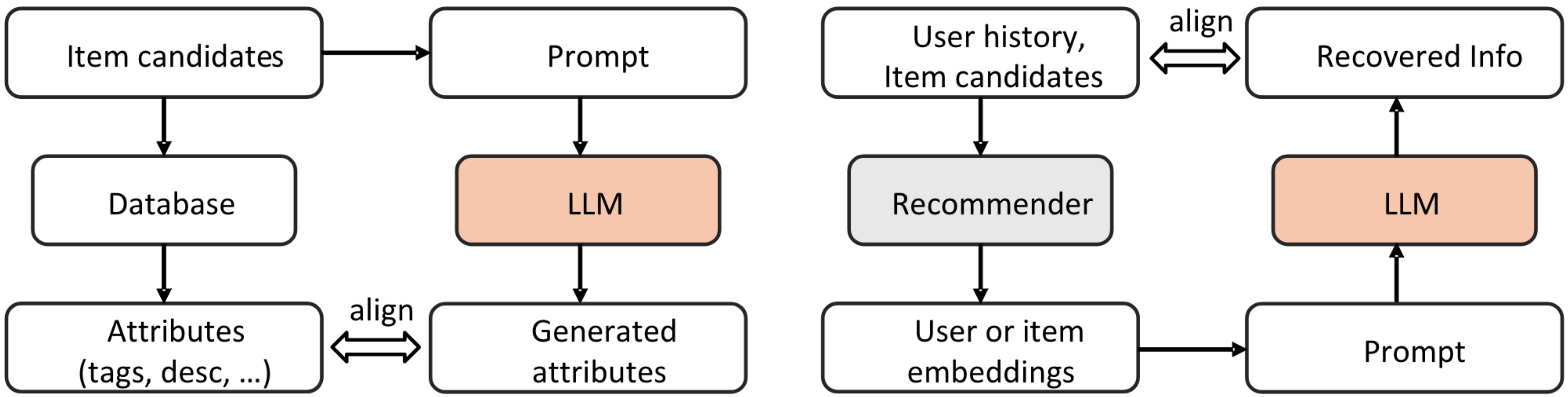
ask for explanation

[1] Gao Yunfan, et al. "Chat-rec: Towards interactive and explainable llms-augmented recommender".

[2] Junling Liu, et al. "Is ChatGPT a Good Recommender? A Preliminary Study".

Finetune LLM for Rec Explanation

- Design different tasks to finetune LLM for Recommendation Explaination
- Besides finetuning for recommendation performance, RecExplainer finetunes LLM on different task related to recommendation explaination, such as Item discrimination and history reconstruction.



Yuxuan Lei et al. RecExplainer: Aligning Large Language Models for Recommendation Model Interpretability Arxiv 2023

Outline

- Introduction
- Development of LLMs
- **Technical Stacks of LLM4Rec**
  - Model Architecture and Pre-training
  - Model Post-training
  - QA & Coffee Break
  - Model Post-training
  - **Decoding and Deployment**
- Open Problems
- Future Direction & Conclusions

LLM4Rec Decoding

Open-ended decoding over full human vocabulary

Gap ⟺

Constrained decoding over existing items

Key challenges in recommendation:

- Bias and homogeneity issue
- Local optima
- inefficiency

Large Foundation Model Stack

Decoding
Beam Search, greedy decoding ......

Post-training
RLHF, DPO, SFT for (safety) alignment
RL for reasoning enhancement

Pre-training
Next-token prediction for Content understanding

Architecture
Self-attention Transformer

Satisfy the decoding needs for recommendation →

Teach the model for recommendation →

NA →

Model expressiveness for recommendation →

LLM4Rec Stack

Decoding
Speculative decoding, Parallel decoding,

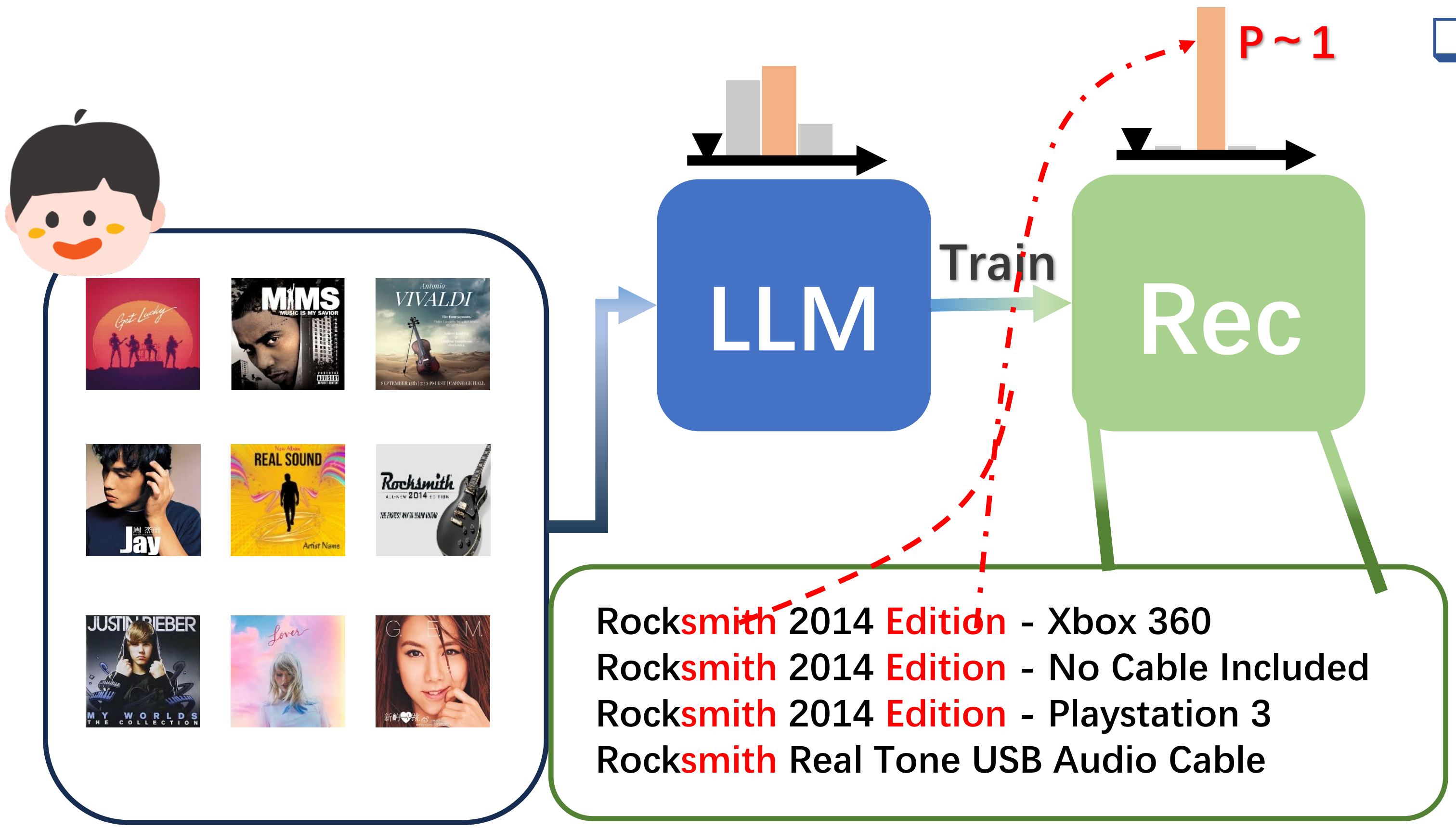
Post-training
Behavior understanding, preference alignment and reasoning to pursue accuracy, efficiency, and trustworthiness.

Pre-training
N/A

Architecture
Item tokenizer, Memory for LLMs

LLM4Rec Decoding

- ❑ Language decoding ≠ Recommendation decoding



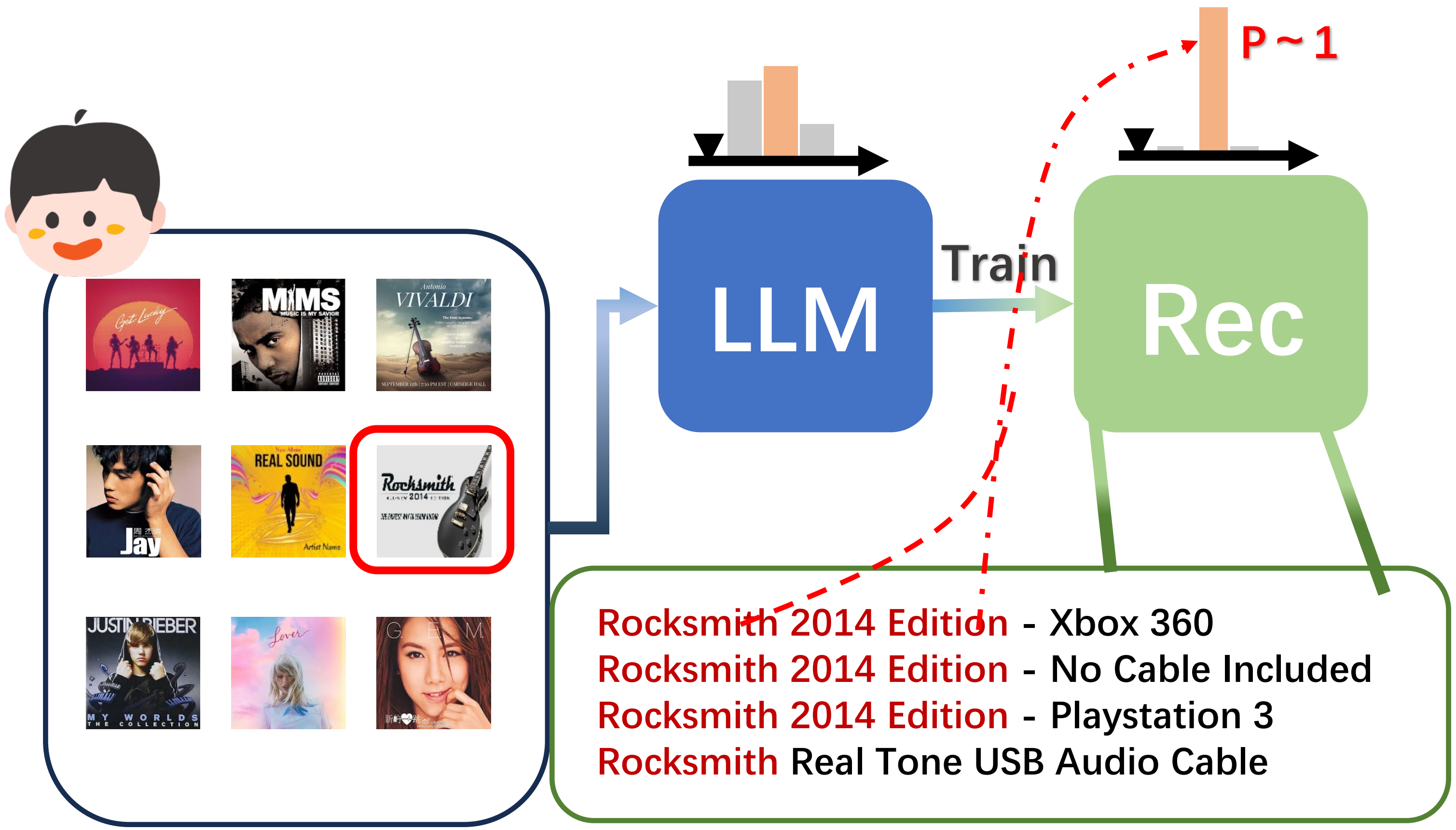
- ❑ **Amplification Bias**
  - ❑ The generation probabilities of some tokens are **close to 1** under the condition of already generated tokens. (e.g. "smith" & "Edition")
  - ❑ Length normalization tends to enhance scores for items contains more of those tokens

→ Remove length normalization

Keqin Bao et al. " Decoding Matters: Addressing Amplification Bias and Homogeneity Issue for LLM-based Recommendation" EMNLP 2024.

LLM4Rec Decoding

- Language decoding $\neq$ Recommendation decoding



- **Homogeneity Issue**
  - Recommend items with similar content and structures
  - Frequently repeats item features based on past user interactions

→ Use a text-free model to assist the LLM to decode

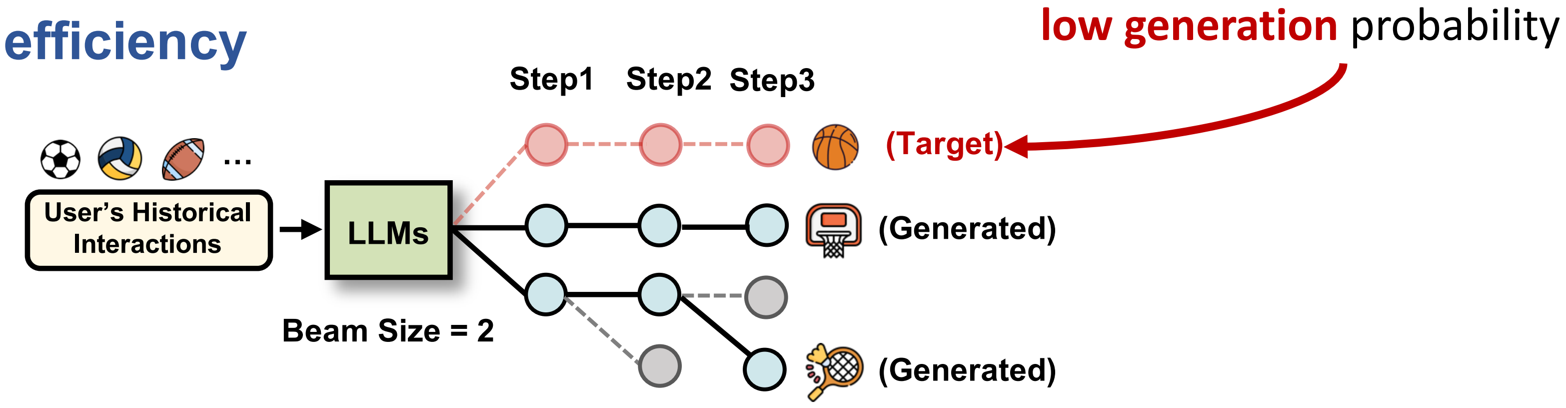
$$L_{TF}(h_t|h_{\le t-1}) = \log\left(\frac{\sum_{i\in I_{h_{<t},h_t}} p_{TF}(I_i)}{\sum_{i\in I_{h_{<t}}} p_{TF}(I_i)}\right)$$

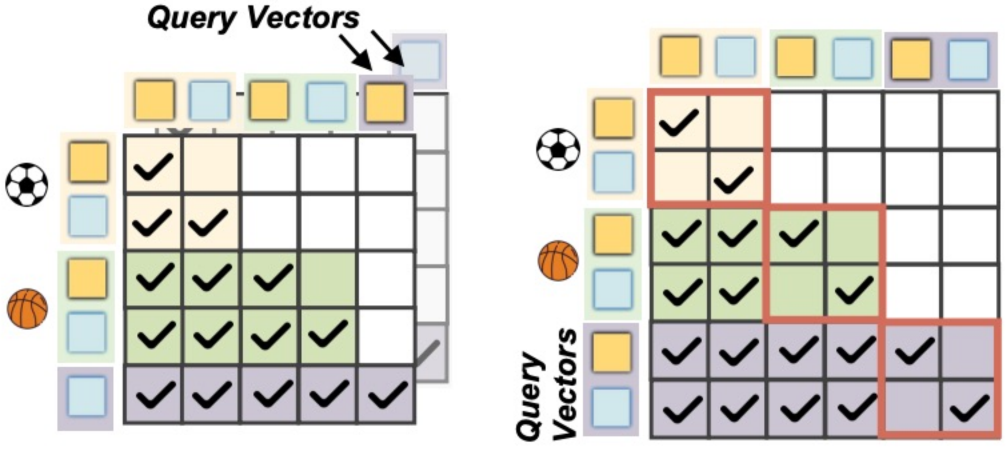
$$S_{TF}(h_{\le t}) = S_{TF}(h_{\le t-1}) + L_{TF}(h_t|h_{\le t-1})$$

Keqin Bao et al. " Decoding Matters: Addressing Amplification Bias and Homogeneity Issue for LLM-based Recommendation" EMNLP 2024.

LLM4Rec Decoding: Parallel Decoding

- Work #1: SETRec
- Autoregressive decoding with beam search suffer from
  - Local optima issue, inefficiency





(a) Original Attention Mask (b) Sparse Attention Mask

Simultaneous generation - decode all tokens in parallel

Sparse attention: in-item tokens cannot ”see” each other

Query vector: generate token for each specific dimension

Fusion: Weighted sum over generated CF token and semantic tokens.

Lin et al. Order-agnostic Identifier for Large Language Model-based Generative Recommendation. SIGIR 2025

LLM4Rec Decoding: Parallel Decoding

Table 1. Overall performance of baselines and SETRec instantiated on T5

		All				Warm				Cold				Inf. Time (s)
Dataset	**Method**	**R@5**	**R@10**	**N@5**	**N@10**	**R@5**	**R@10**	**N@5**	**N@10**	**R@5**	**R@10**	**N@5**	**N@10**	**All Users**
Toys	**DreamRec**	0.0020	0.0027	0.0015	0.0018	0.0027	0.0039	0.0020	0.0024	0.0066	0.0168	0.0045	0.0082	912
	E4SRec	0.0061	0.0098	0.0051	0.0064	0.0081	0.0128	0.0065	0.0082	0.0065	0.0122	0.0056	0.0078	**55**
	BIGRec	0.0008	0.0013	0.0007	0.0009	0.0014	0.0019	0.0011	0.0013	0.0278	0.0360	0.0196	0.0223	2,079
	IDGenRec	0.0063	0.0110	0.0052	0.0069	0.0109	0.0161	0.0081	0.0102	0.0318	0.0589	0.0236	0.0335	658
	CID	0.0044	0.0082	0.0040	0.0053	0.0065	0.0128	0.0049	0.0071	0.0059	0.0111	0.0047	0.0066	810
	SemID	0.0071	0.0108	0.0061	0.0074	0.0086	0.0153	0.0075	0.0100	0.0307	0.0507	0.0220	0.0292	1,215
	TIGER	0.0064	0.0106	0.0060	0.0076	0.0091	0.0147	0.0080	0.0102	0.0315	0.0555	0.0228	0.0314	448
	LETTER	0.0081	0.0117	0.0064	0.0077	0.0109	0.0155	0.0083	0.0101	0.0183	0.0395	0.0115	0.0190	448
	SETRec	**0.0110***	**0.0189***	**0.0089***	**0.0118***	**0.0139***	**0.0236***	**0.0112***	**0.0147***	**0.0443***	**0.0812***	**0.0310***	**0.0445***	60

1) Best effectiveness on warm-start items and **generalize well** on **cold-start items.**

2) High efficiency compared to the auto-regressive generation.

		All		Warm		Cold	
		R@10	**N@10**	**R@10**	**N@10**	**R@10**	**N@10**
1.5B	**LETTER**	0.0093	0.0064	0.0126	0.0085	0.0416	0.0239
	E4SRec	0.0108	0.0072	0.0144	0.0096	0.0235	0.0111
	SETRec	**0.0188**	**0.0120**	**0.0236**	**0.0151**	**0.0883**	**0.0507**
3B	**LETTER**	0.0109	0.0072	0.0151	0.0097	0.0471	0.0236
	E4SRec	0.0096	0.0061	0.0129	0.0081	0.0218	0.0103
	SETRec	**0.0195**	**0.0123**	**0.0258**	**0.0159**	**0.0964**	**0.0571**
7B	**LETTER**	0.0099	0.0061	0.0137	0.0081	0.0406	0.0216
	E4SRec	0.0088	0.0057	0.0114	0.0072	0.0133	0.0065
	SETRec	**0.0194**	**0.0115**	**0.0239**	**0.0140**	**0.1016**	**0.0613**

Continuously increasing

3) Promising scalability on cold-start items

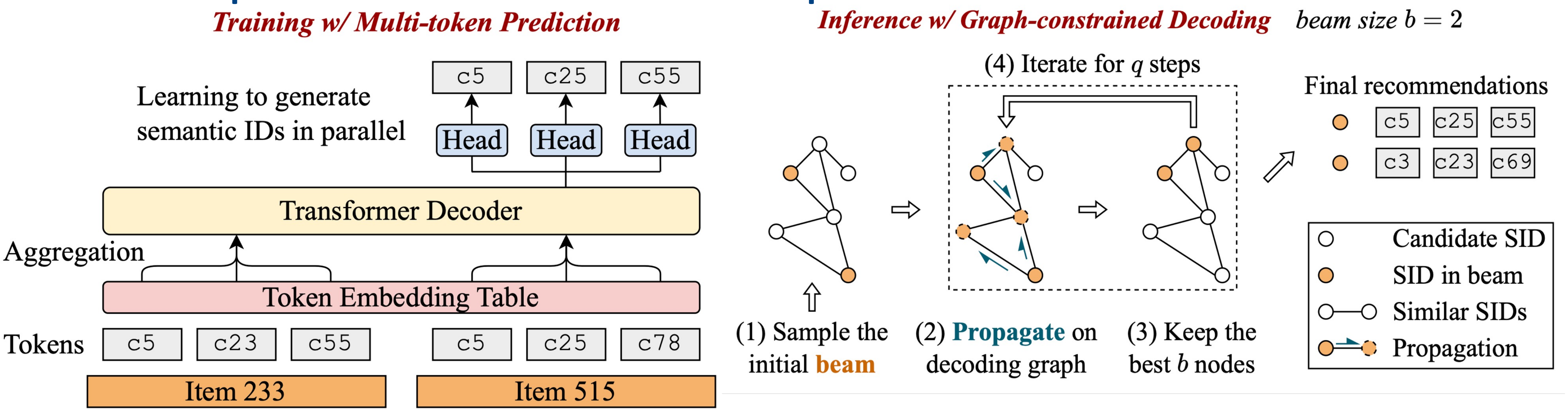
as the model size is scaled up.

Lin et al. Order-agnostic Identifier for Large Language Model-based Generative Recommendation. SIGIR 2025

LLM4Rec Decoding: Parallel Decoding

Work #2: RPG

Multi-token prediction loss: Generate tokens in parallel



Hou et al. Generating Long Semantic IDs in Parallel for Recommendation. KDD 2025.

LLM4Rec Decoding: Boost cold-start

Work #1: SpecGR

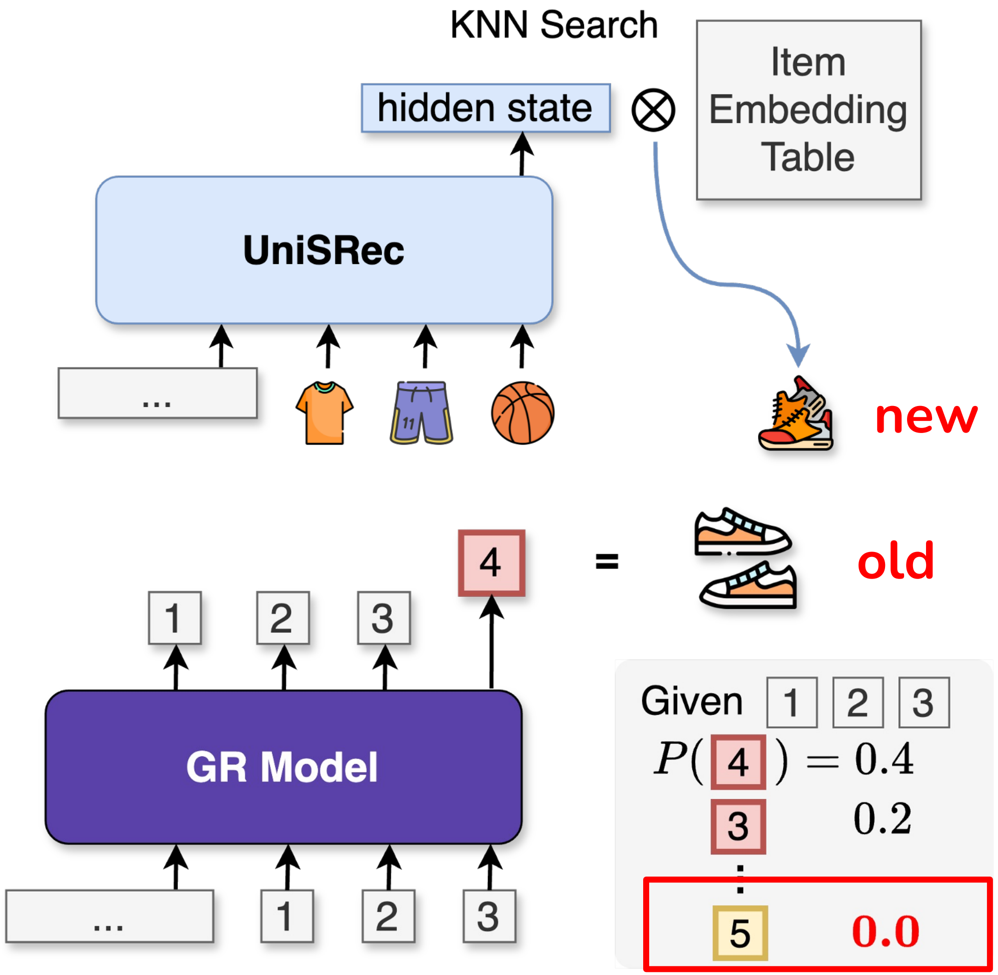
- **Cold-start Recommendation**

We want the model to recommend new items to users without retraining

- **It is difficult for generative models to generate new items**

Because it assigns very low score for unseen semantic ID patterns (i.e., items)

How can we achieve good cold-start performance?



Ding et al., *Inductive Generative Recommendation via Retrieval-based Speculation. 2024 arxiv (slides borrowed from Yupeng Hou)*

LLM4Rec Decoding: Boost cold-start

Work #1: SpecGR

- Method

An inductive model as a **drafter** to propose items, then use generative model (e.g., LLMs) as a **verifier** to accept or reject candidate items (the drafter isn't necessarily "cheaper" in this setting, just inductive)

💡 **Why does this work?**

- Inductive **drafter** — candidate items contain new items
- Generative **verifier** — accept or reject candidate items using the model's strong capability in understanding semantic IDs

Ding et al., Inductive Generative Recommendation via Retrieval-based Speculation. 2024 arxiv (slides borrowed from Yupeng Hou)

LLM4Rec Decoding: Boost cold-start

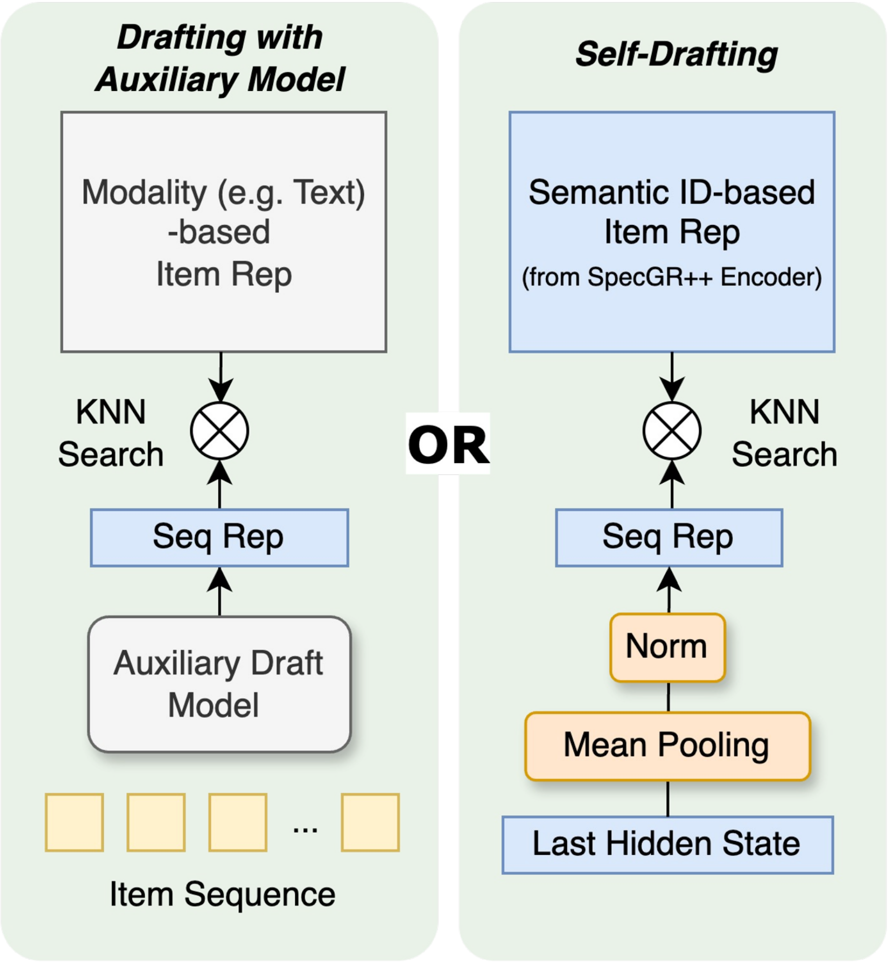
1. Inductive Drafting

- Either use an inductive model such as UniSRec or its own encoder module

2. Target-Aware Verification

- Use joint token likelihood for verification
- Ignore the identifier token for unseen items

$$\mathrm{V}(x_t, X) = \begin{cases} \frac{1}{l}\sum_{i=1}^{l} \log P(c_i^t \mid c_{<i}^t, X) & \text{if } x \in \mathcal{I}, \\ \frac{1}{l-1}\sum_{i=1}^{l-1} \log P(c_i^t \mid c_{<i}^t, X) & \text{if } x \in \mathcal{I}^* \setminus \mathcal{I}, \end{cases}$$



Ding et al., Inductive Generative Recommendation via Retrieval-based Speculation. 2024 arxiv (slides borrowed from Yupeng Hou)

Outline

- Introduction
- Development of LLMs
- **Technical Stacks of LLM4Rec**
  - Model Architecture and Pre-training
  - Model Post-training
  - QA & Coffee Break
  - Model Post-training
  - **Decoding and Deployment**
- Open Problems
- Future Direction & Conclusions

LLM4Rec Deployment

- **On top of stack, how can we utilize LLMs in recommender system?**
  - **Non-agent: LLM as recommender model**
  - **Agent: LLM as agent for recommender system**
  - **…**

	Large Foundation Model Stack		LLM4Rec Stack
Deployment	Conversation, Math, Chat…	LLM utilization for recommendation tasks →	Various LLM roles: LLM as recommender, LLM as user simulator, LLM as rec agent
Decoding	Beam Search, greedy decoding ……	Satisfy the decoding needs for recommendation →	Speculative decoding, Parallel decoding,
Post-training	RLHF, DPO, SFT for (safety) alignment RL for reasoning enhancement	Teach the model for recommendation →	Behavior understanding, preference alignment and reasoning to pursue accuracy, efficiency, and trustworthiness.
Pre-training	Next-token prediction for Content understanding	NA →	N/A
Architecture	Self-attention Transformer	Model expressiveness for recommendation →	Item tokenizer, Memory for LLMs

LLM4Rec Deployment

- **LLMs not only as recommender, but can also act as an agent**
- **LLM-empowered Agents for Recommendation**
  - Agent as User Simulator
    - Main idea: using agents to simulate user behavior for real-world recommendation.
    - RecAgent[1], Agent4Rec[2]
  - Agent for Recommendation
    - Main idea: harnessing the powerful capabilities of LLMs, such as reasoning, reflection, planning and tool usage, for recommendation.
    - RecMind[3], InteRecAgent[4], BiLLP[5], Multi-Agent Collaboration[6]

[1] Lei Wang et al. "When Large Language Model based Agent Meets User Behavior Analysis: A Novel User Simulation Paradigm" arXiv 2023.

[2] Zhang An et al. "On Generative Agents in Recommendation" arXiv 2023.

[3] Wang Yancheng et al. "RecMind: Large Language Model Powered Agent For Recommendation" arXiv 2023.

[4] Xu Huang et al. "Recommender AI Agent: Integrating Large Language Models for Interactive Recommendations" arxiv 2023.

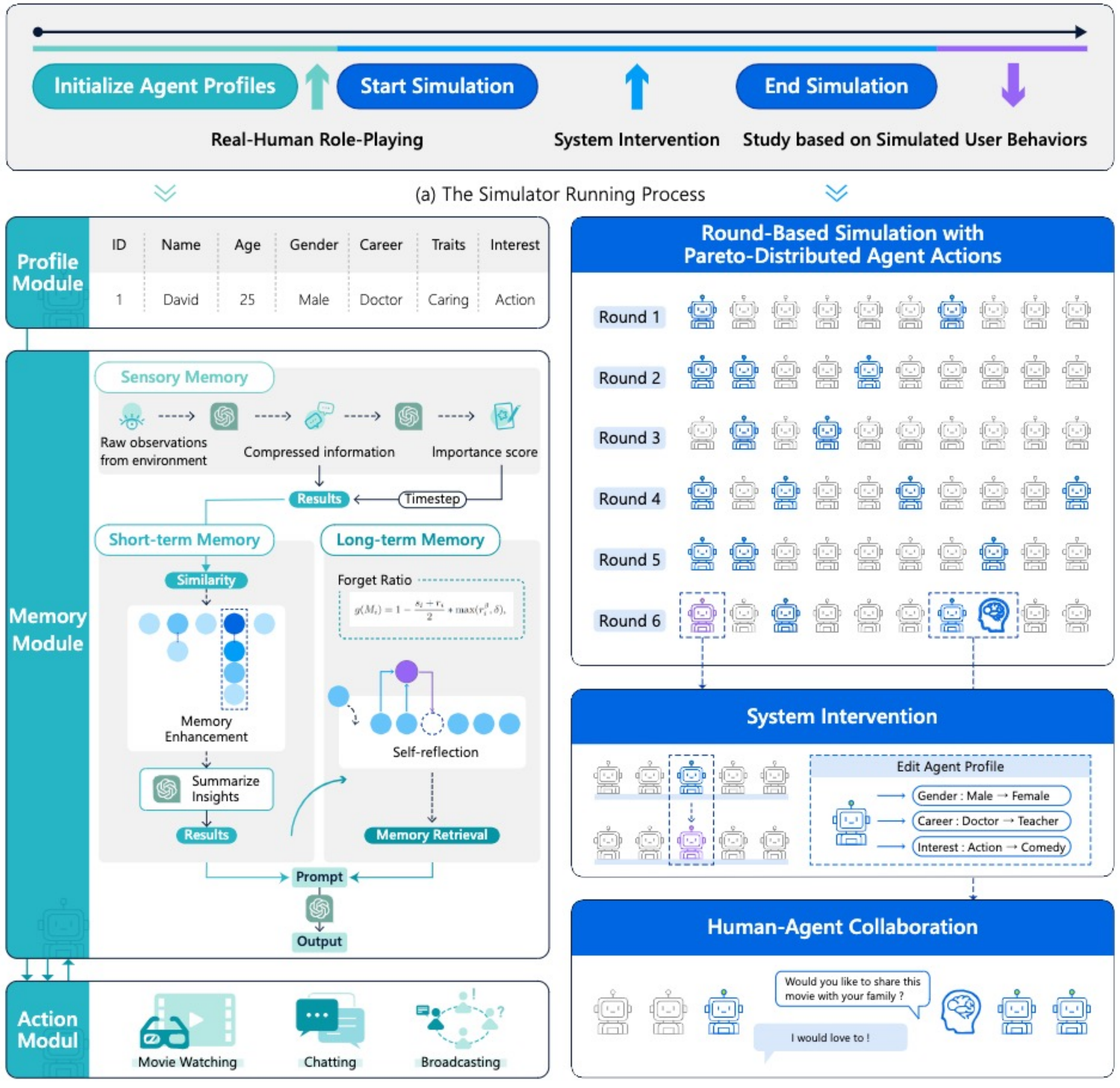
[5] Wentao Shi et al. 2023. Large Language Models are Learnable Planners for Long-Term Recommendation. in SIGIR 2024.

[6] Jiabao Fang et al. A Multi-Agent Conversational Recommender System. Arxiv 2024

Deployment as Agent: RecAgent

❑ LLM-based agent for user simulation

- User simulation is a fundamental problem in human-centered applications.
- Traditional methods struggle to simulate complex user behaviors.
- LLMs show potential in human-level intelligence and generalization capabilities.



(b) The Agent Framework
(c) The Simulator Implementations

Lei Wang et al. " User Behavior Simulation with Large Language Model based Agents" arXiv 2023.

Deployment as Agent: RecAgent

❑ Recommendation Behaviors

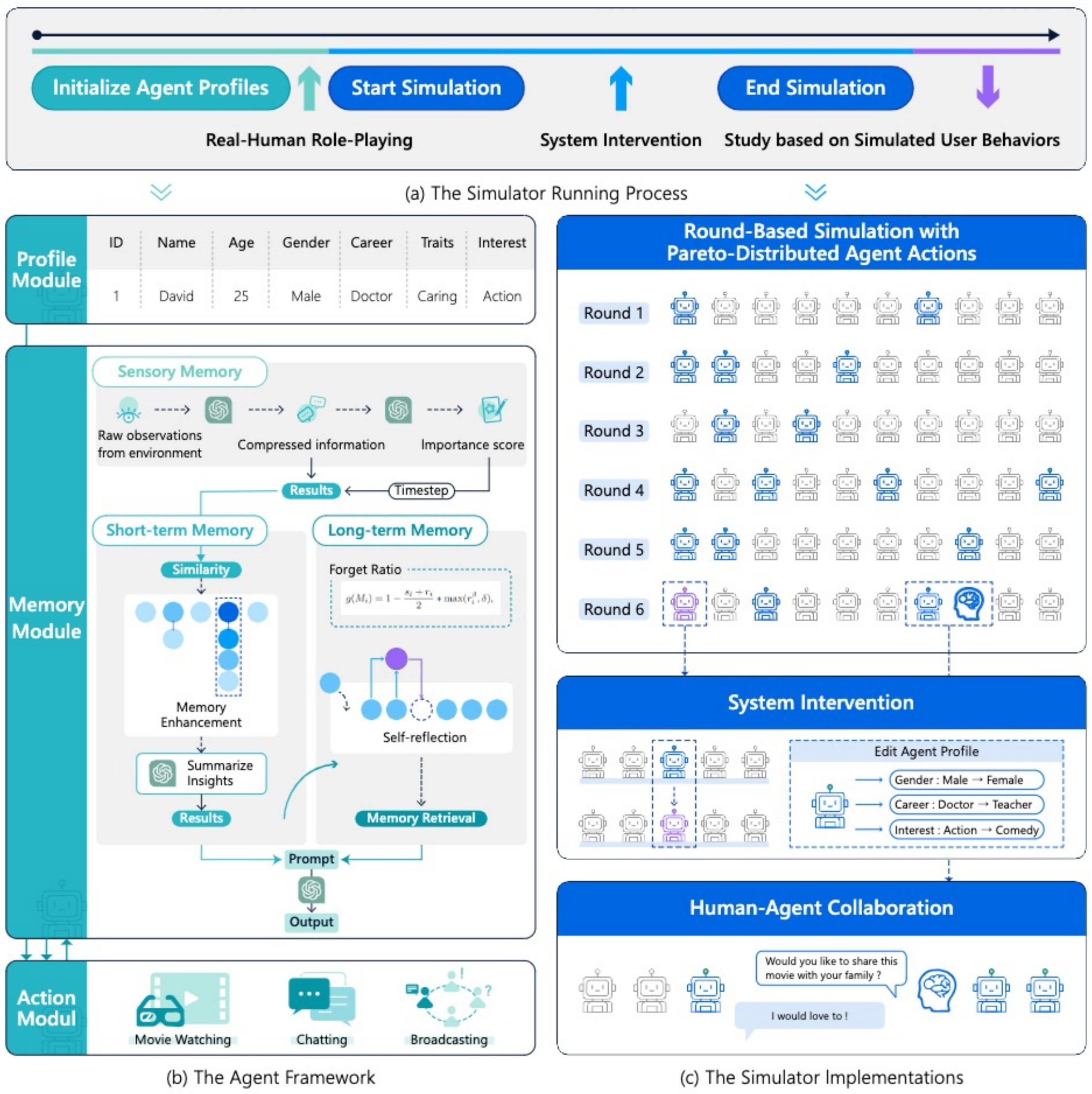
Agent chooses to **search or receive recommendations**, selects movies, and **stores** feelings after watching.

❑ Chatting Behaviors

Two agents **discuss and stored** the conversation in their memories.

❑ Broadcasting Behaviors

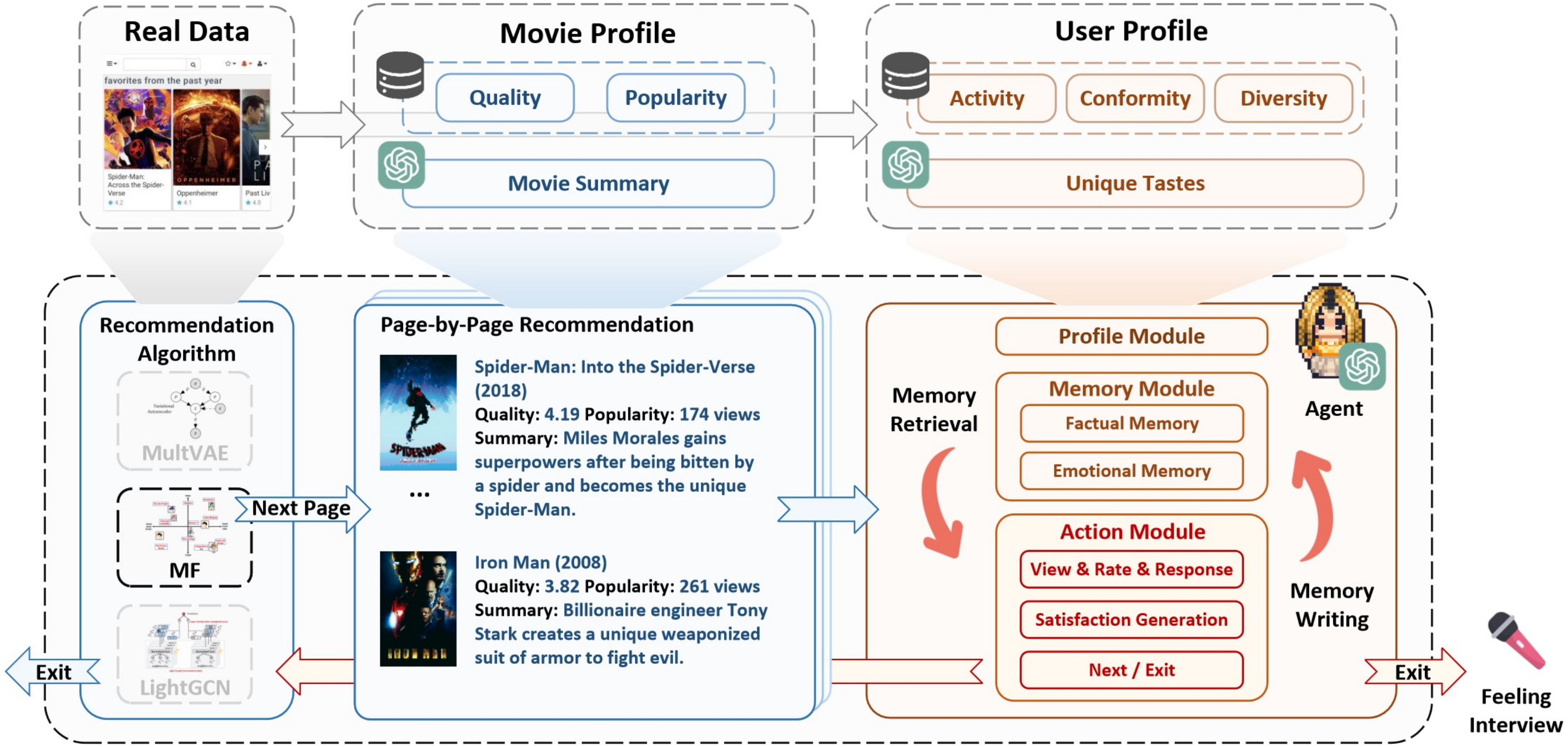
An agent **posts** a message on social media, **received by friends** and stored in their memories.



Lei Wang et al. " User Behavior Simulation with Large Language Model based Agents" arXiv 2023.

Deployment as Agent: Agent4Rec

- Agent4Rec, a simulator with 1,000 LLM-empowered generative agents.
- Agents are trained by the MovieLens-1M dataset, embodying varied social traits and preferences.
- Each agent interacts with personalized movie recommendations in a page-by-page manner and undertakes various actions such as watching, rating, evaluating, exiting, and interviewing.



Zhang An et al. "On Generative Agents in Recommendation" arXiv 2023.

Deployment as Agent: Agent4Rec

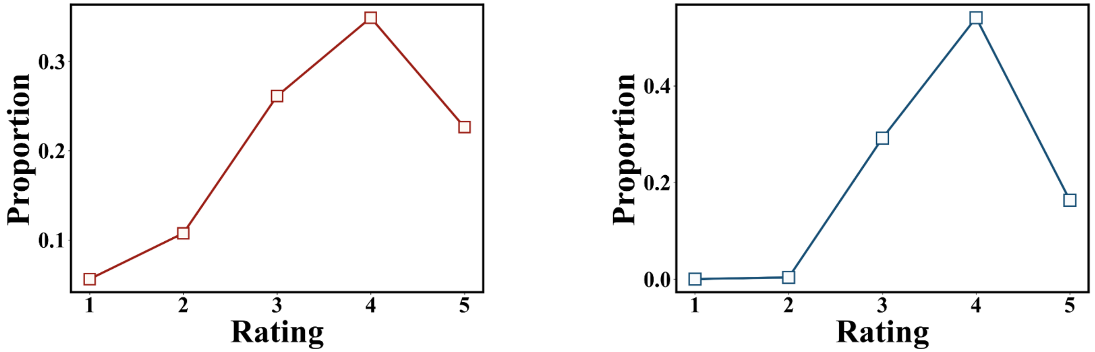
- To what extent can LLM-empowered generative agents truly simulate the behavior of genuine, independent humans in recommender systems?

- User Taste Alignment

Table 1: User taste discrimination.

1:m	Accuracy	Recall	Precision	F1 Score
1:1	**0.6912***	0.7460	**0.6914***	**0.6982***
1:2	0.6466	0.7602	0.5058	0.5874
1:3	0.6675	0.7623	0.4562	0.5433
1:9	0.6175	**0.7753***	0.2139	0.3232

- Rating Distribution Alignment



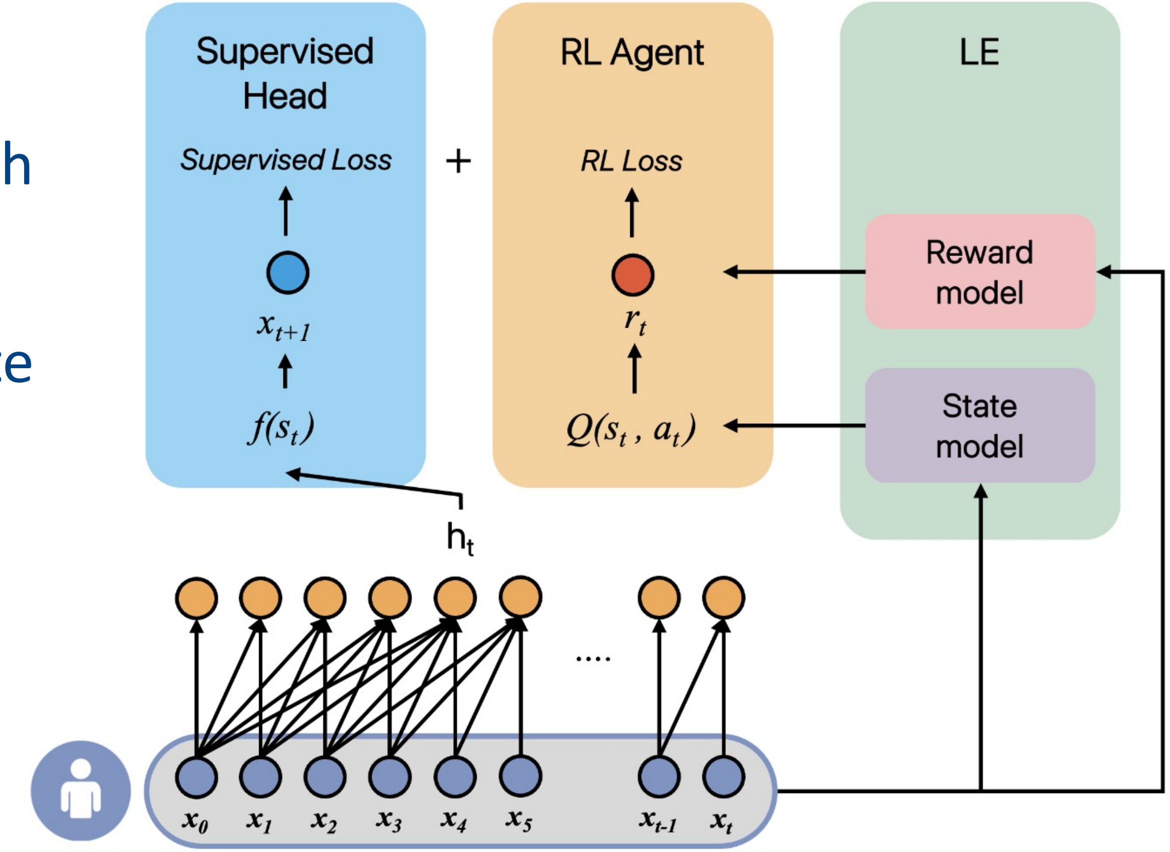
(a) Distribution on MovieLens (b) Agent-simulated distribution

Zhang An et al. "On Generative Agents in Recommendation" arXiv 2023.

Deployment as Agent: Environment Simulation

- **LLM as environment simulator**
  - Act as a state model that produces high quality states
  - Function as a reward model to simulate user feedback on actions

- **Application: interactive training with RL-based recommender models**



Jie Wang et al. "Reinforcement Learning-based Recommender Systems with Large Language Models for State Reward and Action Modeling" arXiv 2024.

Deployment as Agent for Rec

❑ LLM-empowered Agents for Recommendation

- ❑ Agent as User Simulator
  - Main idea: using agents to simulate user behavior for real-world recommendation.
  - RecAgent[1], Agent4Rec[2]
- ❑ Agent for Recommendation
  - Main idea: harnessing the powerful capabilities of LLMs, such as reasoning, reflection, planning and tool usage, for recommendation.
  - RecMind[3], InteRecAgent[4], BiLLP[5], Multi-Agent Collaboration[6]

[1] Lei Wang et al. "When Large Language Model based Agent Meets User Behavior Analysis: A Novel User Simulation Paradigm" arXiv 2023.

[2] Zhang An et al. "On Generative Agents in Recommendation" arXiv 2023.

[3] Wang Yancheng et al. "RecMind: Large Language Model Powered Agent For Recommendation" arXiv 2023.

[4] Xu Huang et al. "Recommender AI Agent: Integrating Large Language Models for Interactive Recommendations" arxiv 2023.

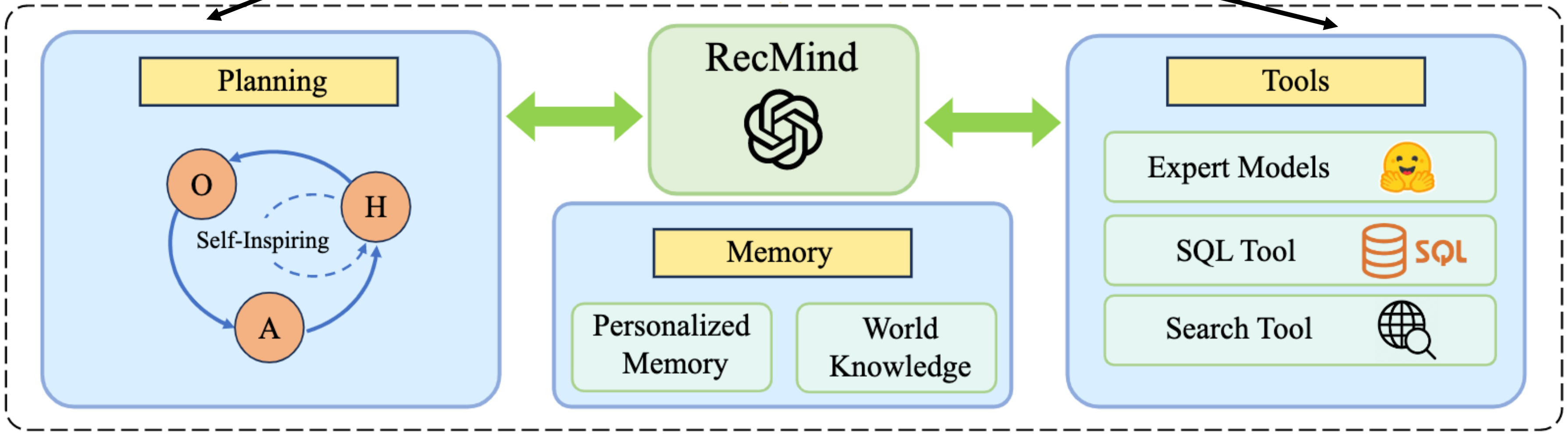
[5] Wentao Shi et al. 2023. Large Language Models are Learnable Planners for Long-Term Recommendation. in SIGIR 2024.

[6] Jiabao Fang et al. A Multi-Agent Conversational Recommender System. Arxiv 2024

Deployment as Agent: RecMind

❑ LLM-based agent for recommendation

- ❑ Traditional methods train and fine-tune models on **task-specific** datasets, struggle to leverage **external knowledge** and lack generalizability across tasks and domains.
- ❑ Existing LLM4Rec methods primarily rely on internal knowledge in LLM weights.
- ❑ RecMind **fully utilizes strong <u>planning</u> and <u>tool-using</u> abilities** of LLMs for recommendation.

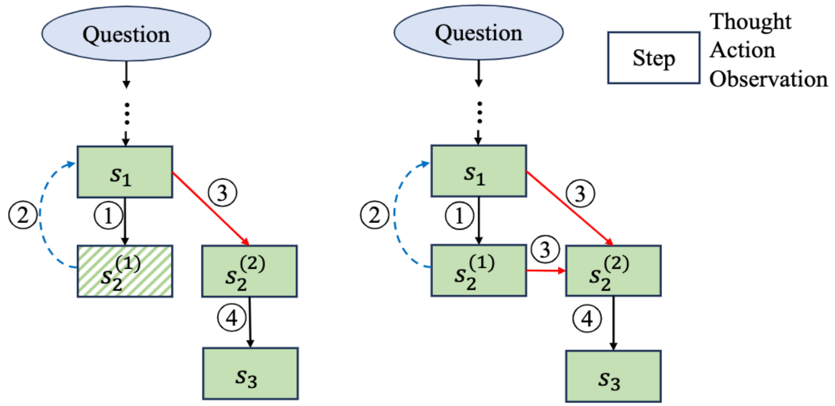


Yancheng Wang et al. "RecMind: Large Language Model Powered Agent For Recommendation" arXiv 2023.

Deployment as Agent: RecMind

❑ Planning ability

- ❑ To break complex tasks into smaller sub-tasks.
- ❑ **Self-inspiring** to integrates multiple reasoning paths.



(a) Tree-of-Thoughts (DFS) (b) Self-Inspiring

❑ Tool-using ability

- ❑ **Database tool** to access domain-specific knowledge.
- ❑ **Search tool** to access real-time information.
- ❑ **Text summarization** tool to summarize lengthy texts.

❑ Evaluation

- **Precision-oriented tasks** (rating prediction, direct recommendation, and sequential recommendation).
- **Explainability-oriented tasks** (explanation generation and review summarization).

❑ Result

RecMind can achieve performance comparable to the **fully trained P5** model.

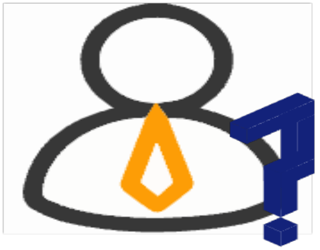
Table 3: Performance comparison in sequential recommendation on Amazon Reviews (Beauty) and Yelp.

Methods	Beauty				Yelp			
	HR@5	NDCG@5	HR@10	NDCG@10	HR@5	NDCG@5	HR@10	NDCG@10
S^3-Rec	0.0387	0.0244	**0.0647**	0.0327	0.0201	0.0123	0.0341	0.0168
SASRec	0.0401	0.0264	0.0643	0.0319	0.0241	0.0175	0.0386	0.0215
P5 (pre-trained expert,few-shot)	**0.0459**	**0.0347**	0.0603	**0.0411**	**0.0565**	**0.0389**	**0.0702**	**0.0441**
ChatGPT (zero-shot)	0.0089	0.0053	0.0103	0.0060	0.0102	0.0062	0.0143	0.0089
ChatGPT (few-shot)	0.0179	0.0124	0.0256	0.0125	0.0217	0.0116	0.0320	0.0165
RecMind-CoT (zero-shot)	0.0182	0.0139	0.0297	0.0160	0.0368	0.0239	0.0554	0.0316
RecMind-CoT (few-shot)	0.0349	0.0187	0.0486	0.0302	0.0427	0.0305	0.0590	0.0380
RecMind-ToT (BFS, zero-shot)	0.0297	0.0172	0.0368	0.0249	0.0379	0.0251	0.0538	0.0322
RecMind-ToT (BFS, few-shot)	0.0387	0.0235	0.0522	0.0327	0.0447	0.0319	0.0624	0.0337
RecMind-ToT (DFS, zero-shot)	0.0299	0.0168	0.0359	0.0241	0.0358	0.0240	0.0519	0.0324
RecMind-ToT (DFS, few-shot)	0.0365	0.0211	0.0497	0.0355	0.0455	0.0328	0.0622	0.0349
RecMind-SI (zero-shot)	0.0339	0.0200	0.0469	0.0310	0.0396	0.0281	0.0569	0.0340
RecMind-SI (few-shot)	0.0415	0.0289	0.0574	0.0375	0.0471	0.0342	0.0635	0.0407

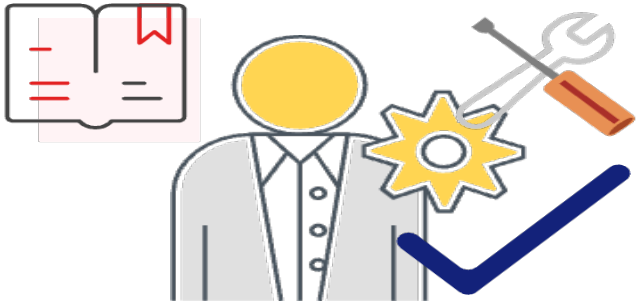
Yancheng Wang et al. "RecMind: Large Language Model Powered Agent For Recommendation" arXiv 2023.

Deployment as Agent: ToolRec

- Traditional challenges: conventional recommender systems (CRS) lack commonsense knowledge about users and items, “narrow expert”
- LLM Advantages: LLMs excel in commonsense reasoning and leveraging external tools



CRS



LLMs with RSs

Current LLMs with RSs Limitations:

- Hallucinations …
- Misalignment between language tasks and recommendation tasks …

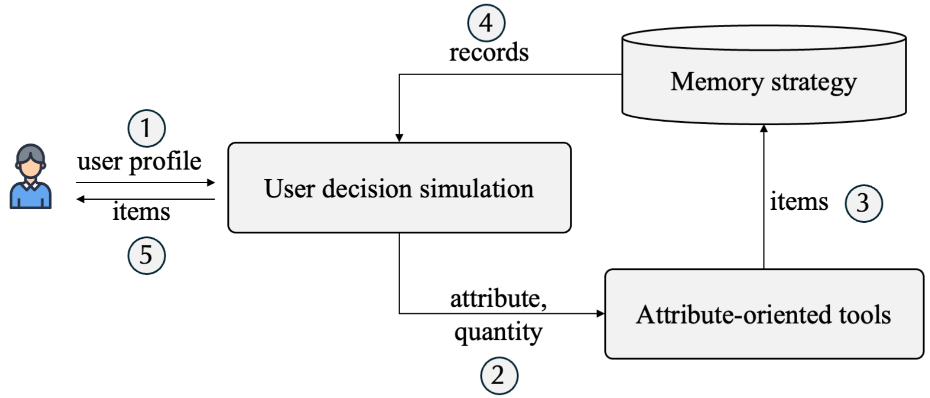
- Our Key: Use LLMs to understand current contexts and preferences, and apply attribute-oriented tools to find suitable items

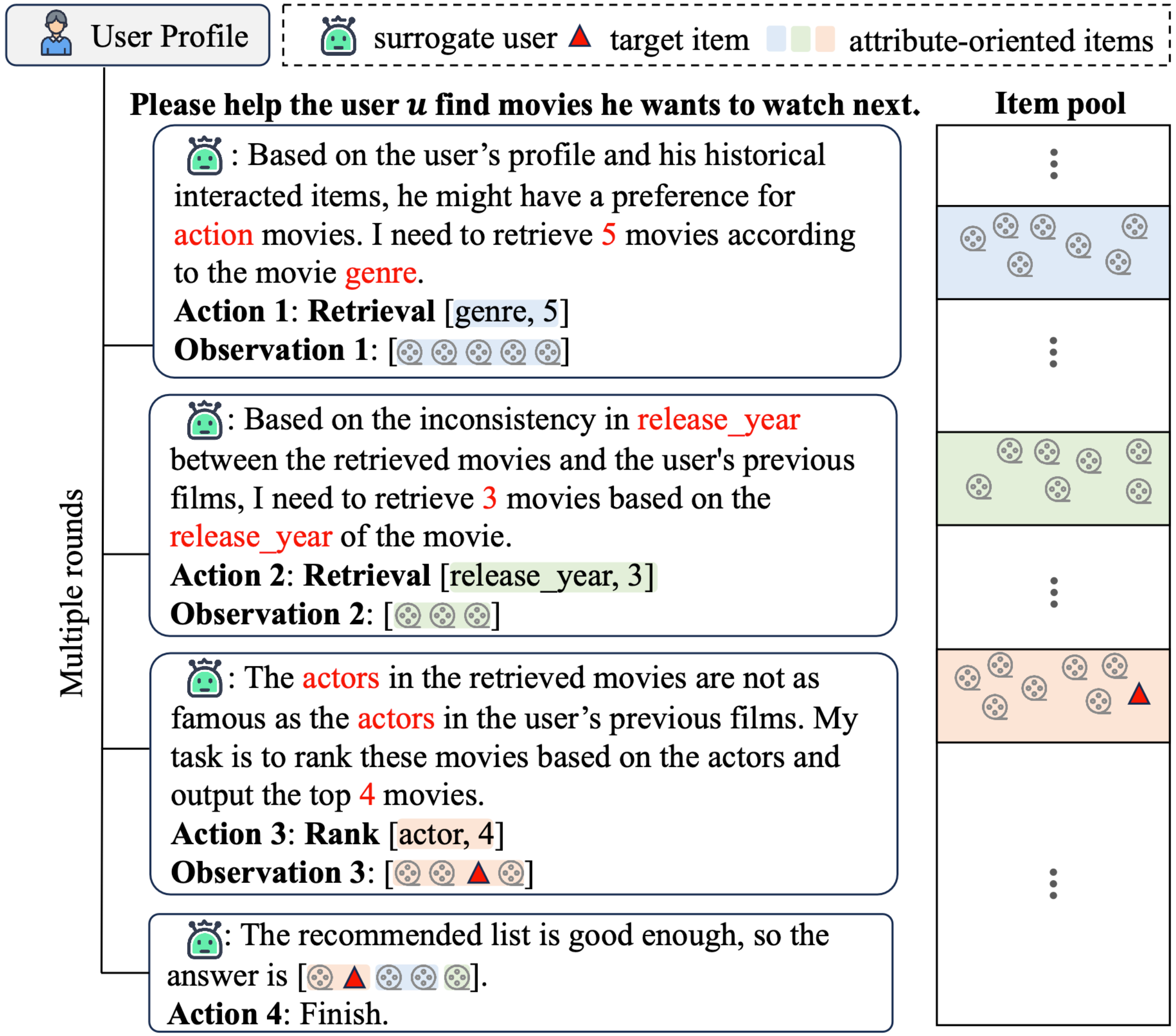
Let Me Do It For You: Towards LLM Empowered Recommendation via Tool Learning. In SIGIR 2024

Deployment as Agent: ToolRec

Methodology:

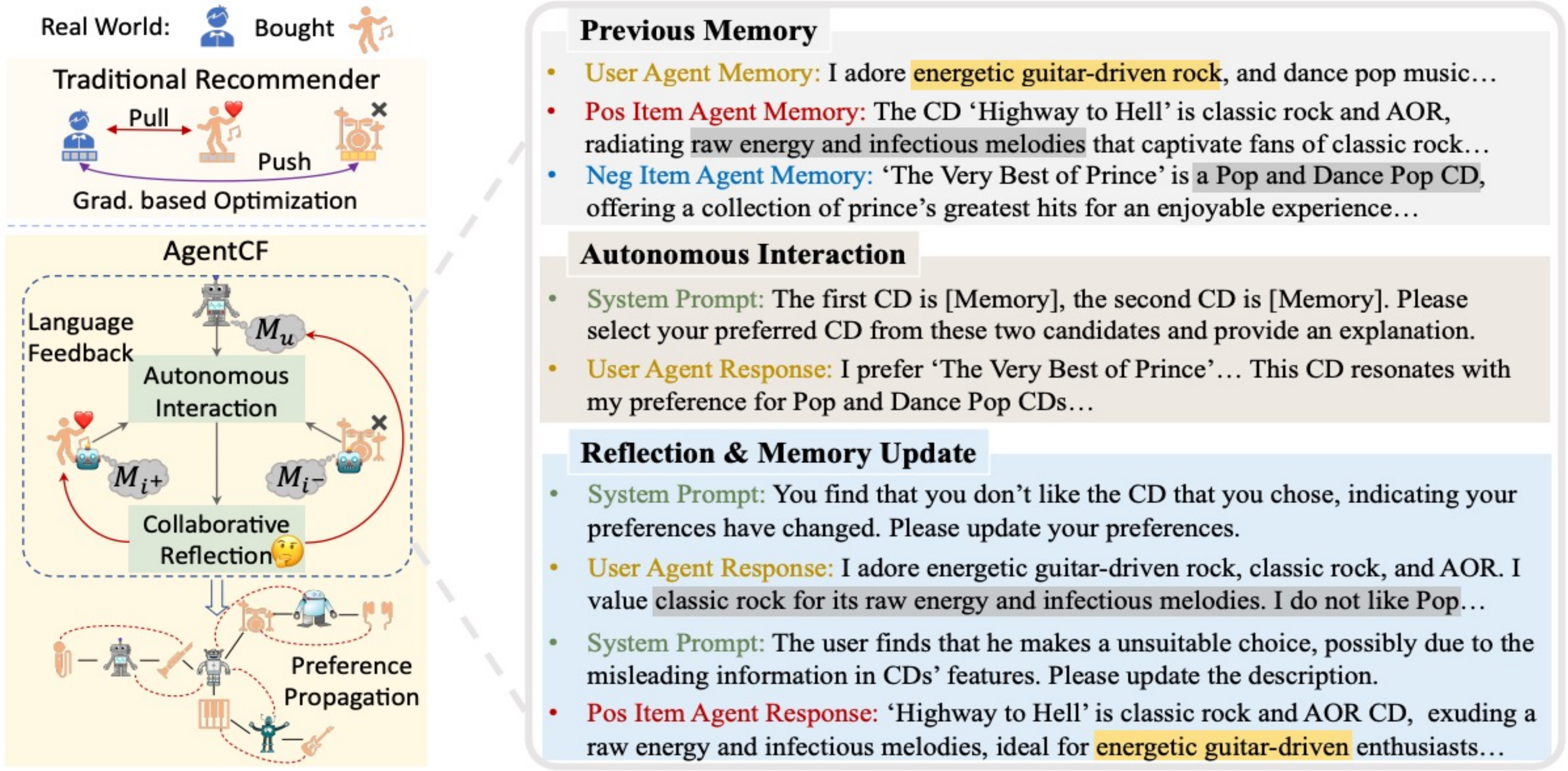
- **LLMs** as the central controller, simulating the user decision.
- **Attribute-oriented Tools**: rank tools and retrieval tools.
- **Memory strategy** can ensure the correctness of generated items and cataloging candidate items.





Let Me Do It For You: Towards LLM Empowered Recommendation via Tool Learning. In SIGIR 2024

Deployment as Agent: AgentCF

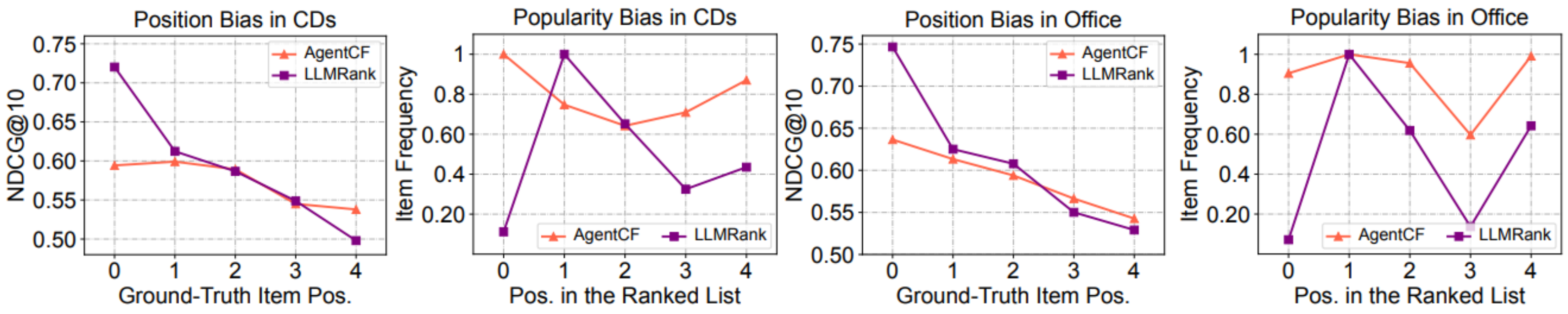


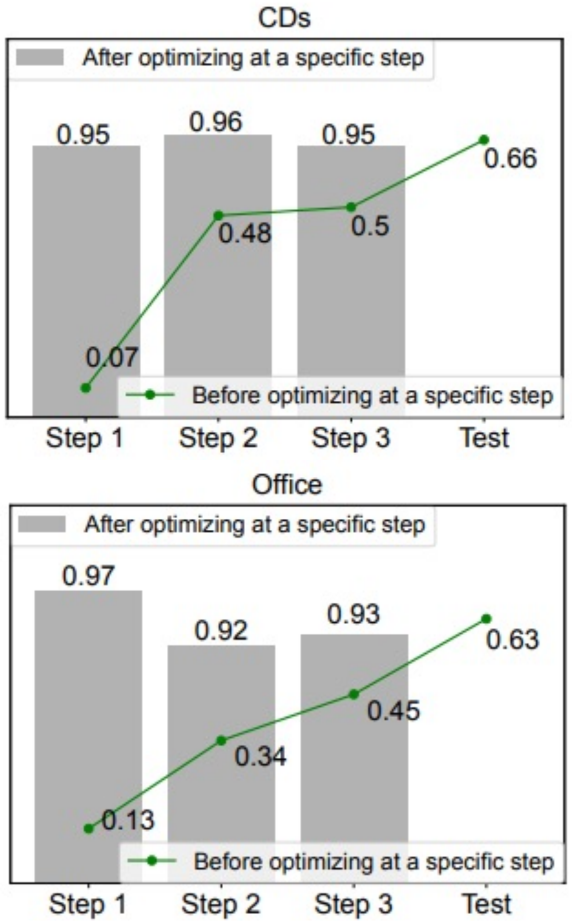
- ❑ Use Agent to simulate both user/items
- ❑ Provide a collaborative reflection optimizing mechanism to optimize the user/item agents, and mutual update of user and item memory.

Junjie Zhang et al. 2023. Agentcf: Collaborative learning with autonomous language agents for recommender systems. in WWW 2024

Deployment as Agent: AgentCF

Method	CDs$_{sparse}$			CDs$_{dense}$			Office$_{sparse}$			Office$_{dense}$		
	N@1	N@5	N@10	N@1	N@5	N@10	N@1	N@5	N@10	N@1	N@5	N@10
BPR$_{full}$	0.1900	0.4902	0.5619	0.3900	0.6784	0.7089	0.1600	0.3548	0.4983	0.5600	0.7218	0.7625
SASRec$_{full}$	0.3300	0.5680	0.6381	0.5800	0.7618	0.7925	0.2500	0.4106	0.5467	0.4700	0.6226	0.6959
BPR$_{sample}$	0.1300	0.3597	0.4907	0.1300	0.3485	0.4812	0.0100	0.2709	0.4118	0.1200	0.2705	0.4576
SASRec$_{sample}$	0.1900	0.3948	0.5308	0.1300	0.3151	0.4676	0.0700	0.2775	0.4437	**0.3600**	**0.5027**	**0.6137**
Pop	0.1100	0.2802	0.4562	0.0400	0.1504	0.3743	0.1100	0.2553	0.4413	0.0700	0.2273	0.4137
BM25	0.0800	0.3066	0.4584	0.0600	0.2624	0.4325	0.1200	0.2915	0.4693	0.0600	0.3357	0.4540
LLMRank	0.1367	0.3109	0.4715	0.1333	0.3689	0.4946	0.1750	0.3340	0.4728	0.2067	0.3881	0.4928
AgentCF$_{B}$	0.1900	0.3466	0.5019	0.2067	0.4078	0.5328	0.1650	0.3359	0.4781	0.2067	0.4217	0.5335
AgentCF$_{B+R}$	**0.2300**	**0.4373**	**0.5403**	**0.2333**	0.4142	**0.5405**	0.1900	0.3589	0.5062	0.1933	0.3916	0.5247
AgentCF$_{B+H}$	0.1500	0.4004	0.5115	0.2100	**0.4164**	0.5198	**0.2133**	**0.4379**	**0.5076**	0.1600	0.3986	0.5147



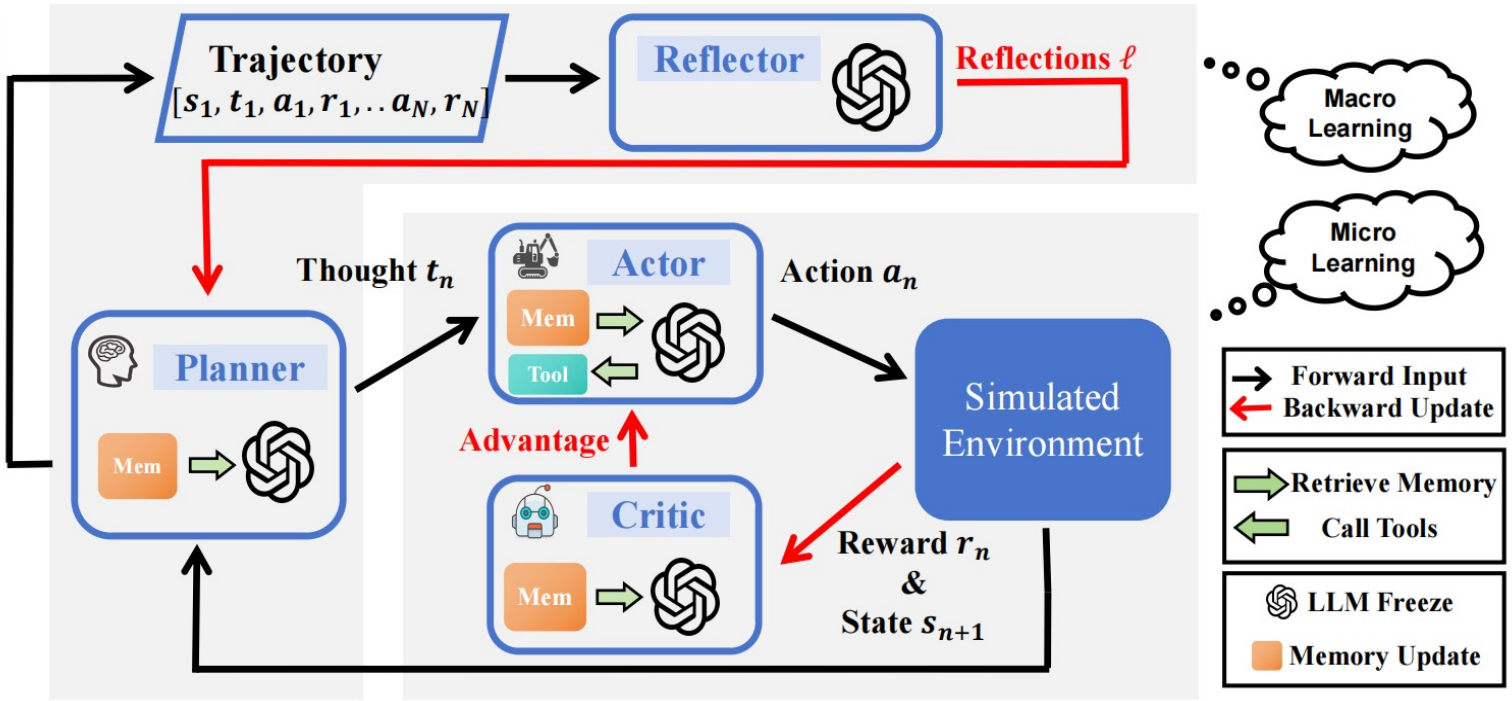


- Better performance and less influenced by bias than directly instructing LLM to rerank
- Collaborative Reflection is effective to optimize the agent’s ability to distinguish positive/negative items

Junjie Zhang et al. 2023. Agentcf: Collaborative learning with autonomous language agents for recommender systems. in WWW 2024

Deployment as Agent: BiLLP

- Use LLM to make plans for long-term recommendations
- Utilize a bi-level learnable mechanism to learn macro-level guidance and micro-level personalized recommendation policies.

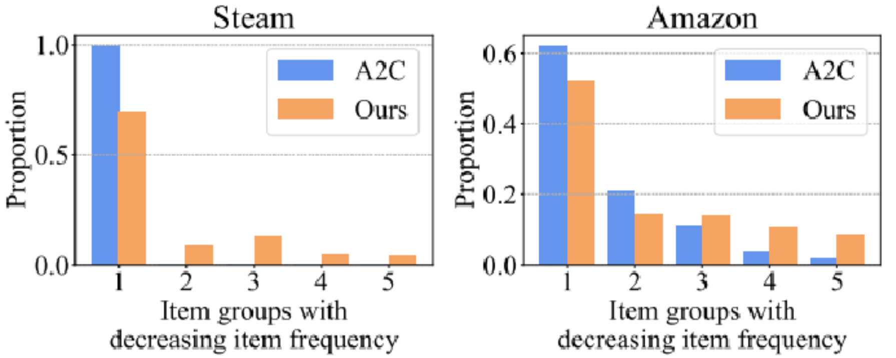


Wentao Shi et al. 2023. Large Language Models are Learnable Planners for Long-Term Recommendation. in SIGIR 2024

Deployment as Agent: BiLLP

Table 4: Average results of all methods in two environments (Bold: Best, Underline: Runner-up).

Methods	Steam			Amazon		
	Len	R_{each}	R_{traj}	Len	R_{each}	R_{traj}
SQN	2.183 ± 0.177	3.130 ± 0.050	6.837 ± 0.517	4.773 ± 0.059	4.303 ± 0.017	20.570 ± 0.245
CRR	4.407 ± 0.088	3.263 ± 0.427	14.377 ± 1.658	3.923 ± 0.162	4.537 ± 0.103	17.833 ± 1.129
BCQ	4.720 ± 0.343	3.997 ± 0.068	18.873 ± 1.092	4.847 ± 0.721	4.367 ± 0.053	21.150 ± 2.893
CQL	5.853 ± 0.232	3.743 ± 0.147	21.907 ± 0.299	2.280 ± 0.185	4.497 ± 0.039	10.263 ± 0.882
DQN	4.543 ± 0.693	4.500 ± 0.069	20.523 ± 3.618	4.647 ± 0.498	4.290 ± 0.083	19.923 ± 1.909
A2C	9.647 ± 0.848	4.367 ± 0.069	42.180 ± 3.937	7.873 ± 0.310	4.497 ± 0.026	35.437 ± 1.453
DORL	9.467 ± 0.862	4.033 ± 0.098	38.300 ± 4.173	7.507 ± 0.174	4.510 ± 0.014	33.887 ± 0.655
ActOnly	5.567 ± 0.160	<u>4.537 ± 0.021</u>	25.250 ± 0.637	6.383 ± 0.176	4.490 ± 0.008	28.660 ± 0.761
ReAct	11.630 ± 0.741	**4.559 ± 0.047**	52.990 ± 2.925	7.733 ± 0.450	<u>4.603 ± 0.033</u>	35.603 ± 1.806
Reflexion	<u>12.690 ± 1.976</u>	4.523 ± 0.026	<u>57.423 ± 8.734</u>	<u>8.700 ± 0.535</u>	**4.670 ± 0.073**	<u>40.670 ± 2.954</u>
BiLLP	**15.367 ± 0.119**	4.503 ± 0.069	**69.193 ± 1.590**	**9.413 ± 0.190**	4.507 ± 0.012	**42.443± 0.817**

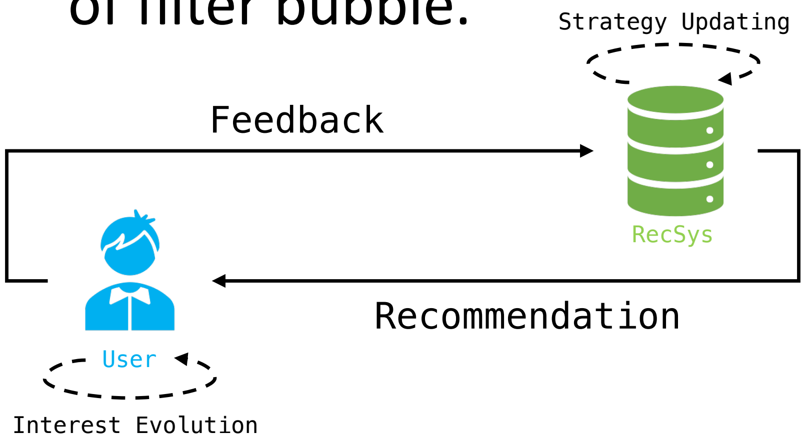


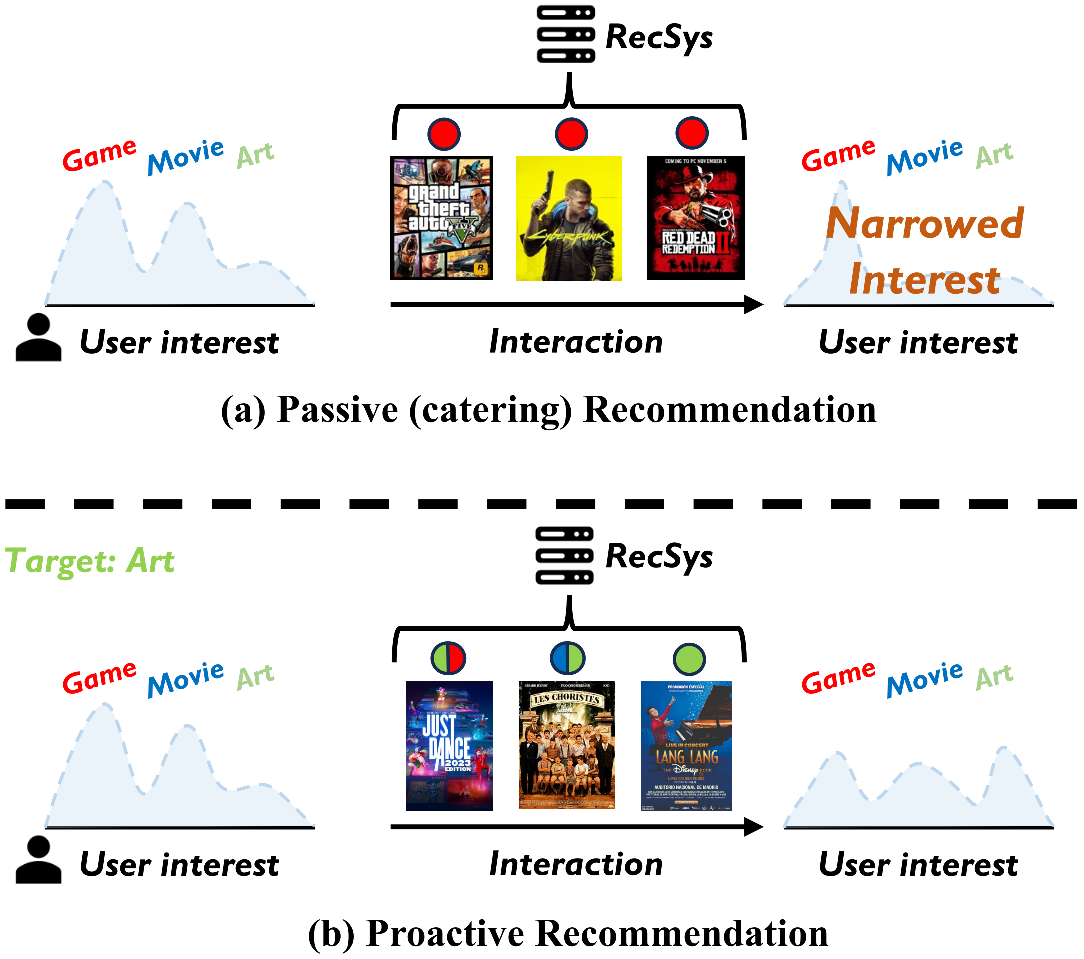
- Better long-term performance than traditional RL-based methods
- Better planning capabilities on long-tail items.

Wentao Shi et al. 2023. Large Language Models are Learnable Planners for Long-Term Recommendation. in SIGIR 2024

Deployment as Agent: Proactive Rec

- Passive Recommendation:
  - Passively cater to user interests.
  - Causing filter bubbles.
- Proactive Recommendation:
  - Actively guide user interests towards a predefined target.
  - Provide a solution to jump out of filter bubble.



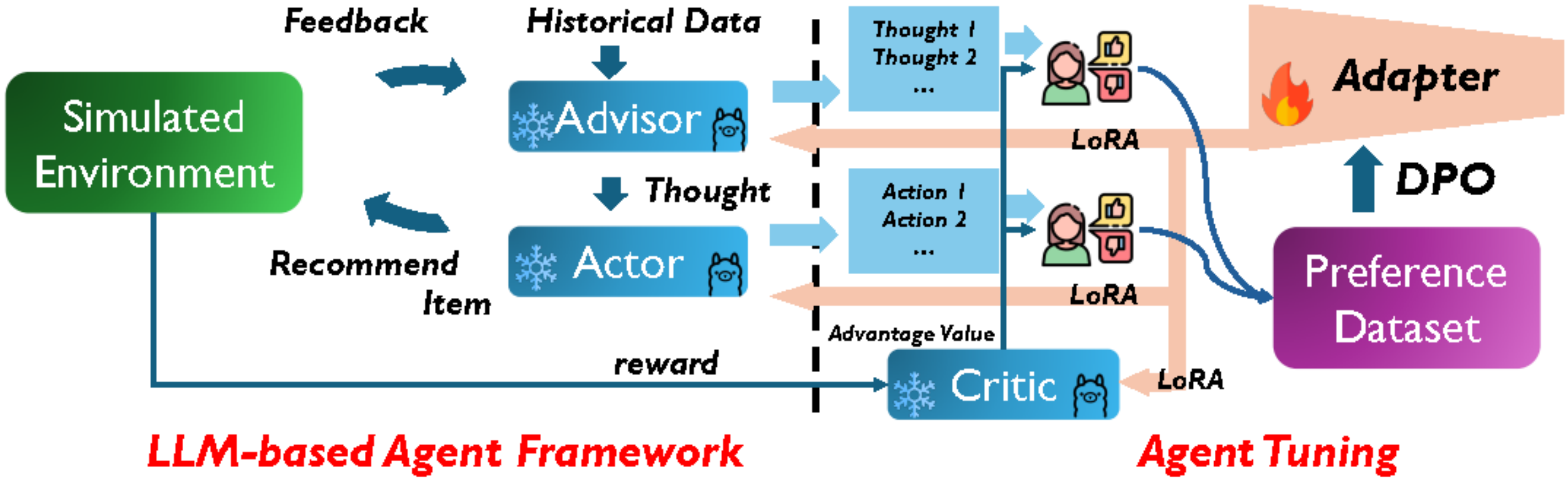


Shuxian Bi et al. 2024. Proactive Recommendation with Iterative Preference Guidance. in WWW 2024

Deployment as Agent: Proactive Rec

T-PRA, a novel agent that is both **adaptive** and **strategic** for proactive recommendation task.

- **1. Actor-Advisor Framework (Flexibility):**
  - **Advisor (Slow-Thinker):** Strategically plans the recommendation path.
  - **Actor (Fast-Thinker):** Makes immediate recommendations based on the Advisor's guidance.
- **2. Critic-Guided Optimization (Long-Term Vision):**
  - An LLM-based **Critic** evaluates the long-term value of actions by calculating Advantage Value.

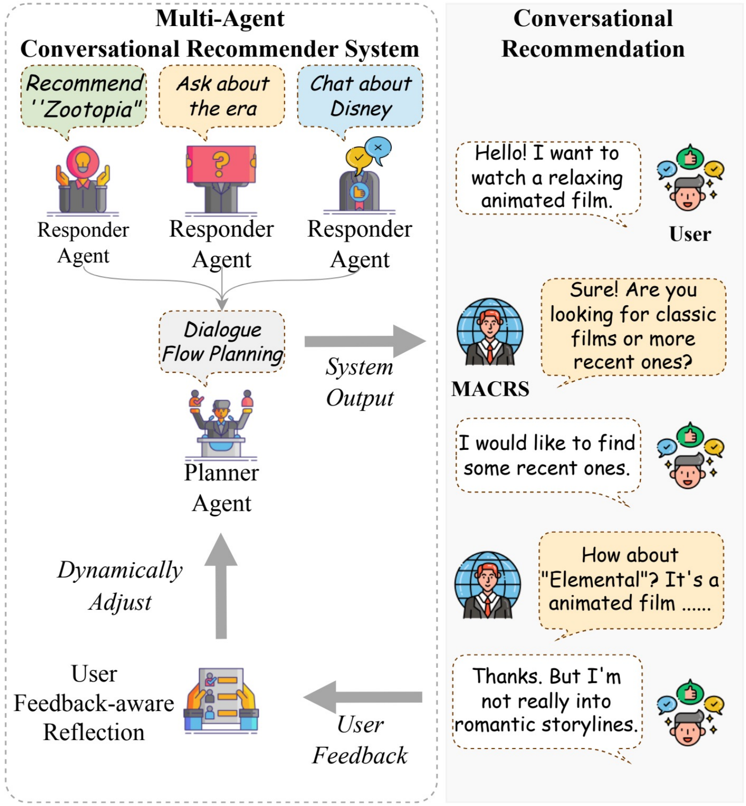


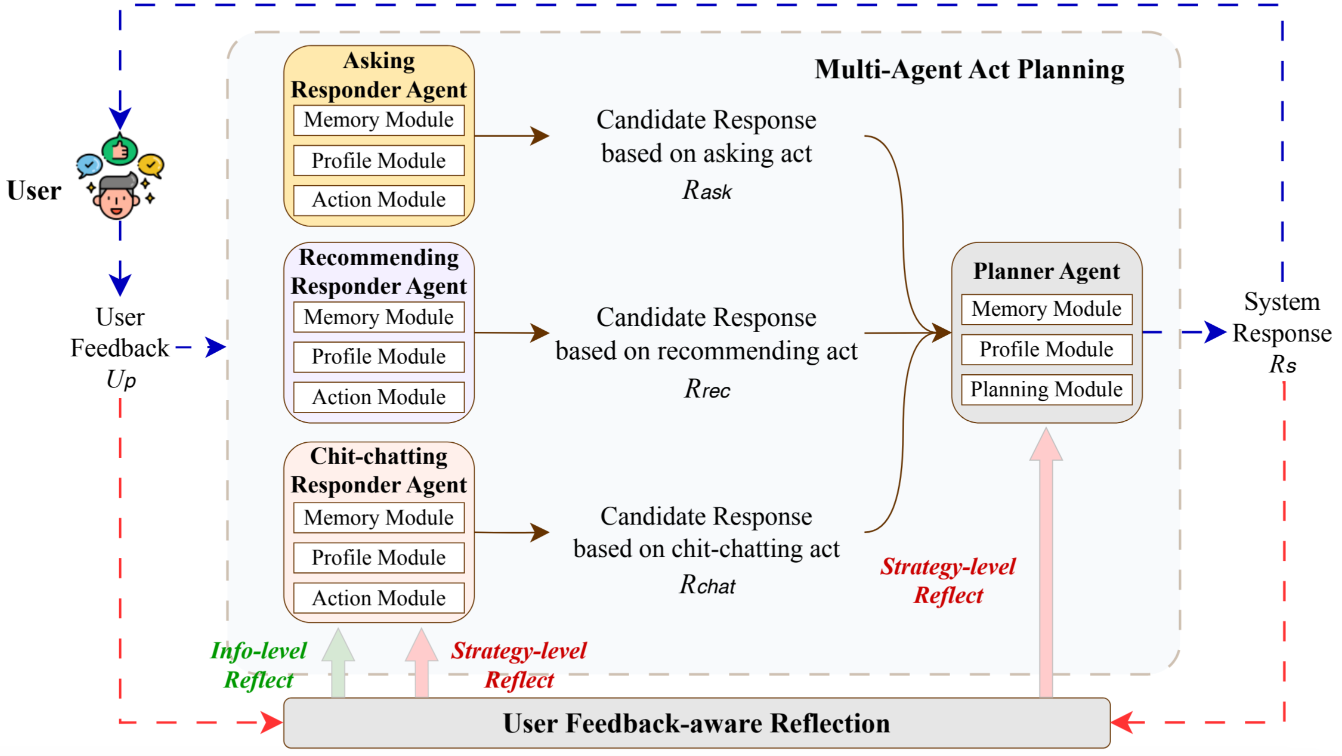
Mingze Wang et al. 2025. Tunable LLM-based Proactive Recommendation Agent (T-PRA). in ACL 2025

Multi-Agent Conversational Rec

❑ Different Agents Collaborate together for Conversational Recommendation

- The responder agent and planner agent collaboratively generate appropriate responses, while the reflection mechanism provides feedback and refined guidance to these agents

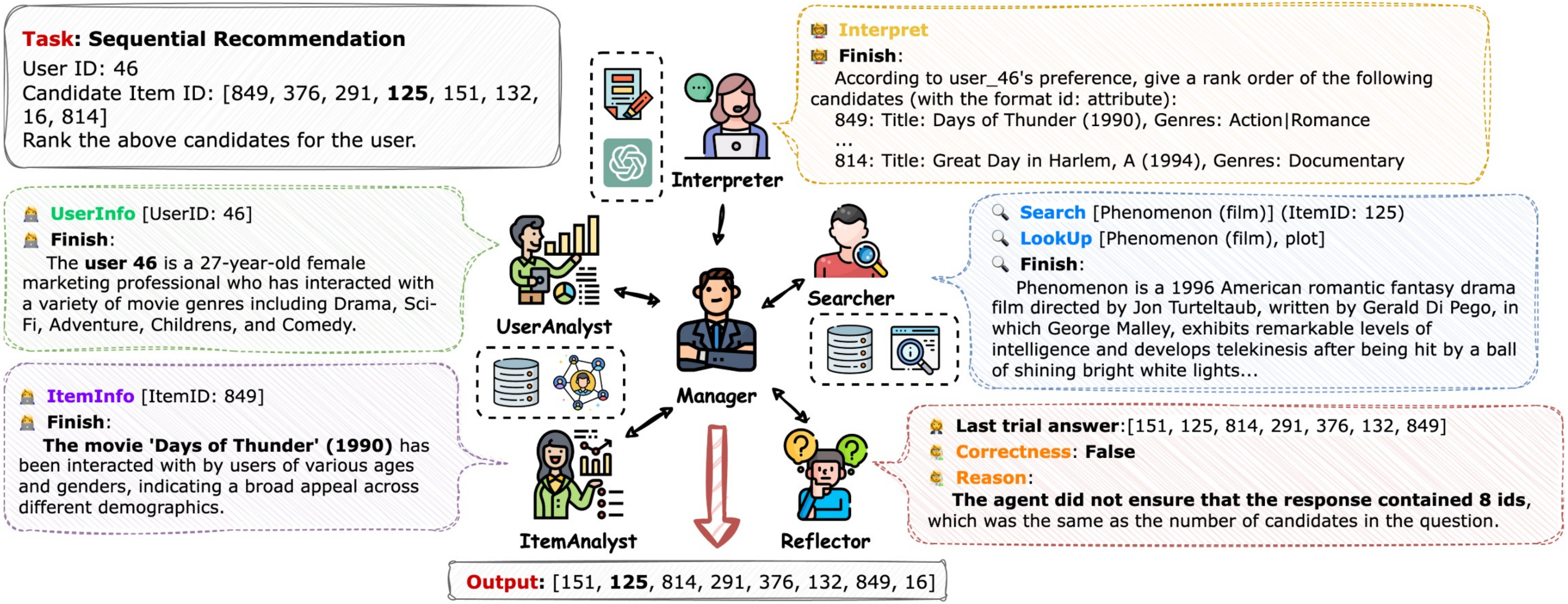




Jiabao Fang et al. A Multi-Agent Conversational Recommender System. Arxiv 2024

Multi-Agent Collaboration for Rec

- Different agents can collaborate together for information delivery.



Zhefan Wang et al. Multi-Agent Collaboration Framework for Recommender Systems. SIGIR 2024

Outline

- Introduction
- Development of LLMs
- Technical Stacks of LLM4Rec
- **Open Problems**
  - Heterogeneous Modeling
  - Lifelong Modeling
  - Evaluation
- Future Direction & Conclusions

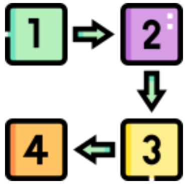
Open Problems & Challenges

Heterogenous Modeling

In-domain, in-platform user behaviors

Open-domain, cross-platform user behaviors

Lifelong Modeling

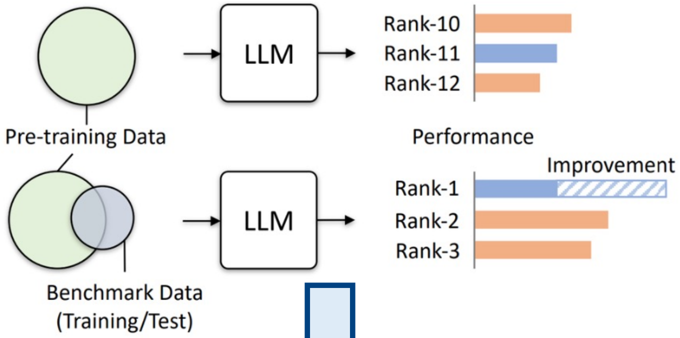


Short-term user behavior modeling

Lifelong user behavior modeling

Evaluation

LLM: Trained on many data, text-focused, language



Evaluation?

RecSys research: interactions, offline, anonymous data

Outline

- Introduction
- Development of LLMs
- Technical Stacks of LLM4Rec
- **Open Problems**
  - **Heterogeneous Modeling**
  - Lifelong Modeling
  - Evaluation
- Future Direction & Conclusions

Heterogenous Modeling

- Users are anticipated to engage with items in different domains.
- Sparse domain: needs information from domain with rich user interactions.
- Raise the need of heterogenous behavior modeling for users

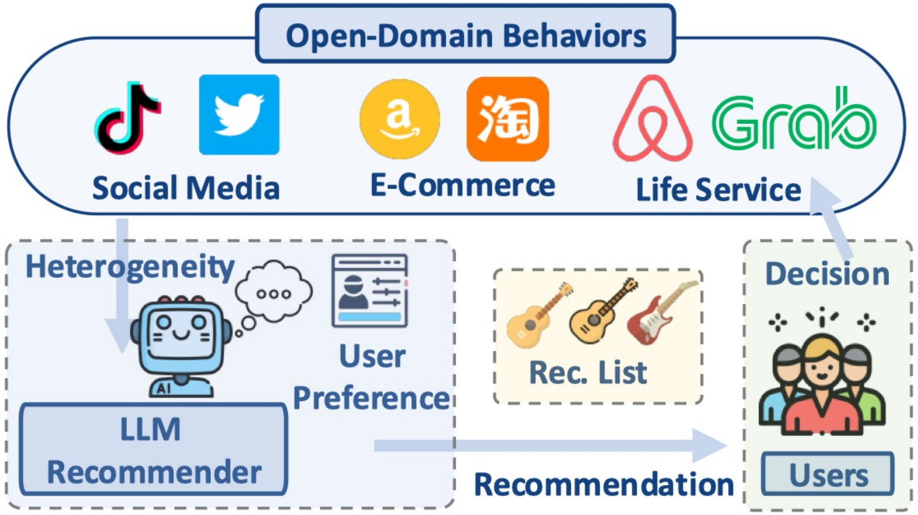


Figure 1: Overview of heterogeneous user modeling in open-domain environments.

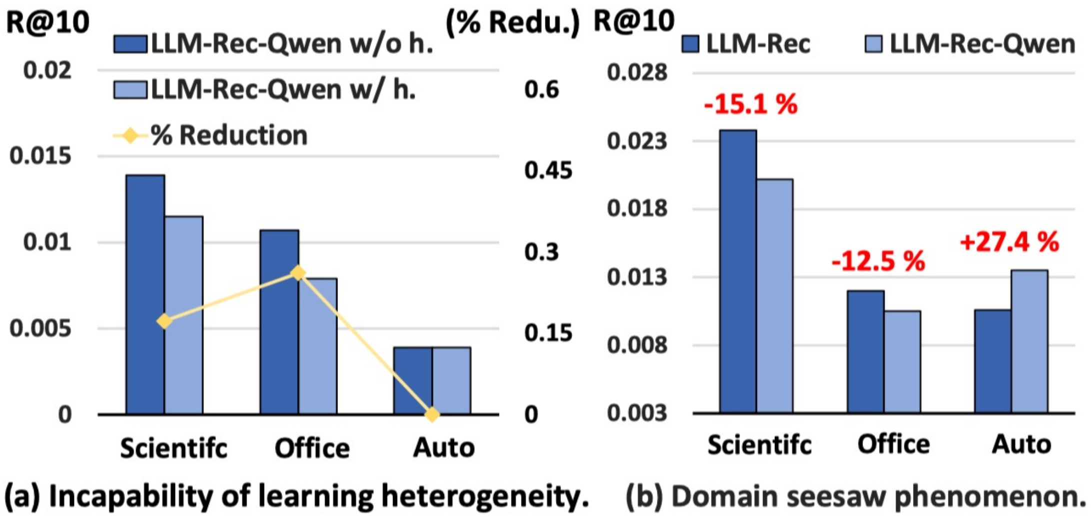


Figure 2: Illustration of the incapability of directly utilizing LLMs to learn heterogeneity and the domain seesaw phenomenon in three domains. "h." denotes "heterogeneity".

Key challenges: Domain seesaw phenomenon

Bao et al. "Heterogeneous User Modeling for LLM-based Recommendation" RecSys 2025.

Heterogenous Modeling

Solution #1: compress heterogeneous user behaviors into special token.

Compression enhancer

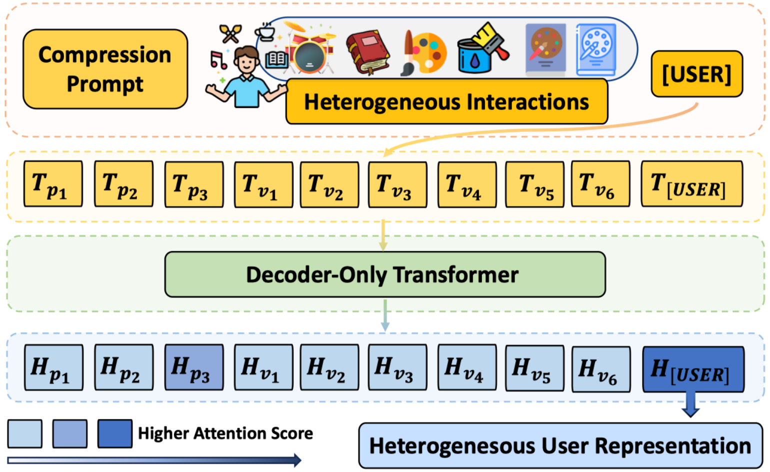


Illustration of compression, which moves from input to model to representation, where "T" represents a token, "H" represents token's last-layer hidden-vector.

- **Compression prompt:** Positively guiding LLMs to compress heterogeneous interactions.
- **User Token:** Unbiased information carrier.
- **Masking Mechanism:** Boosting transferable information extraction and understanding.

Robustness enhancer

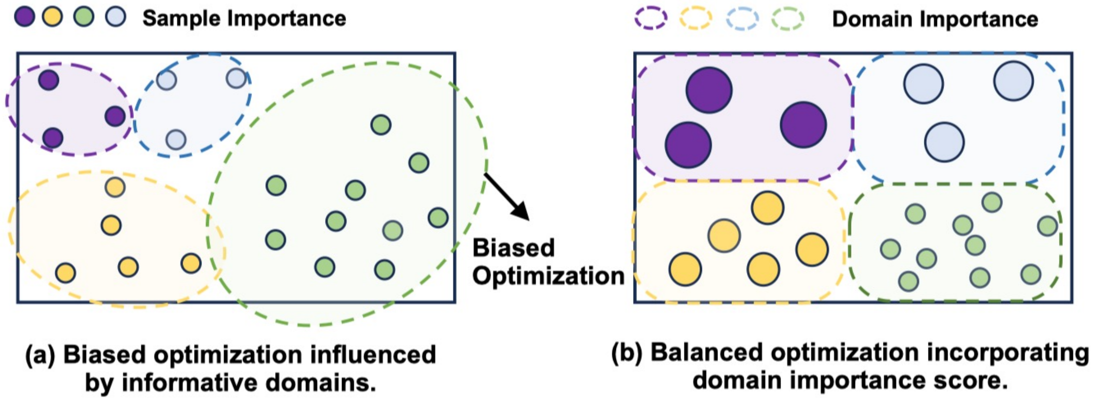


Illustration of biased optimization, dominated by informative domains and balanced optimization incorporating domain importance.

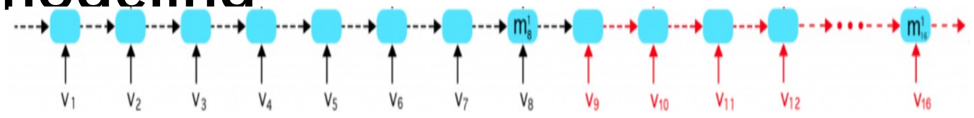
- **Domain importance:** Measuring domain optimization importance.
- **Domain smoothing:** Facilitating training stability.

Bao et al. "Heterogeneous User Modeling for LLM-based Recommendation" RecSys 2025.

Lifelong Modeling

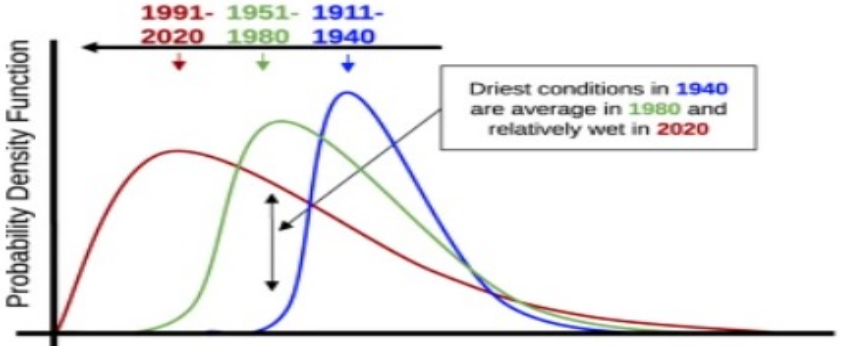
- Users are anticipated to engage with the recommender system continuously
- Raise the need of lifelong behavior modeling for users

Lifelong sequential behavior modeling



- The length of historical interaction sequences grows significantly, easily exceeding 1000
- How to model such long sequence effectively?

Continual learning

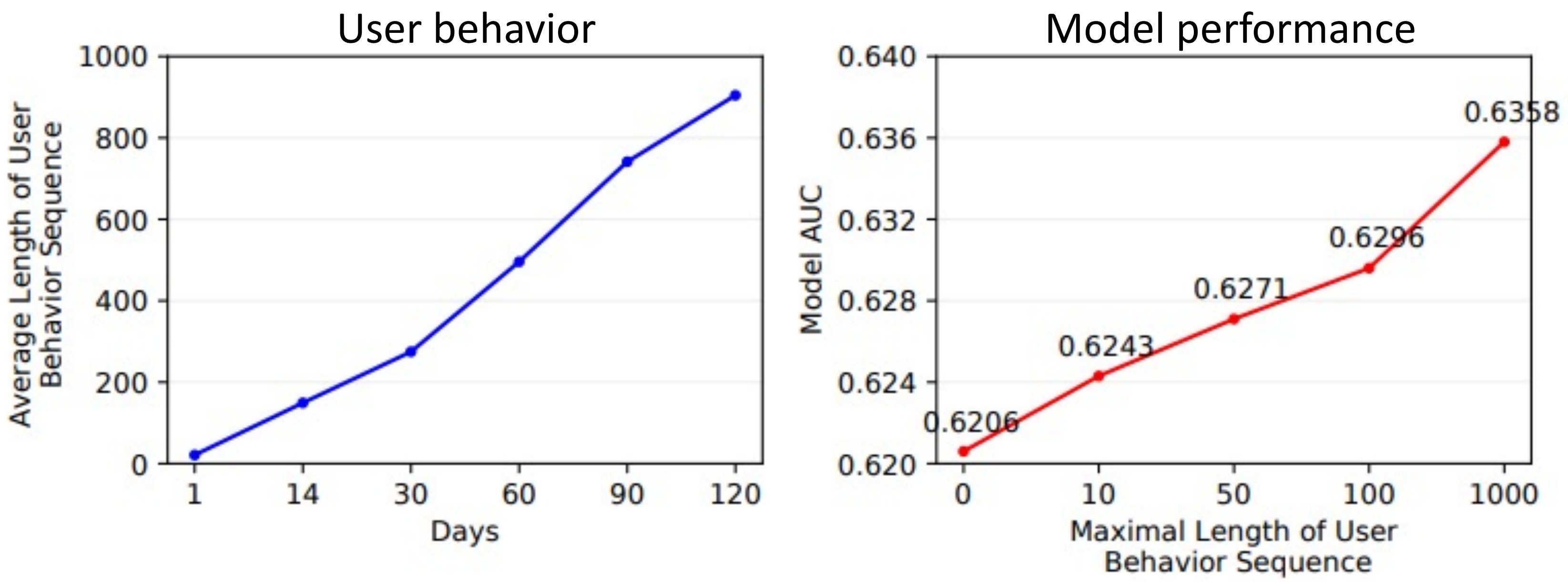


- User interests drift with time going
- How to continuously/incremental learn user interests?

Lifelong Modeling

Lifelong sequential behavior modeling:

- **A longer history signifies richer personalization information, and modeling this can lead to heightened prediction accuracy.**



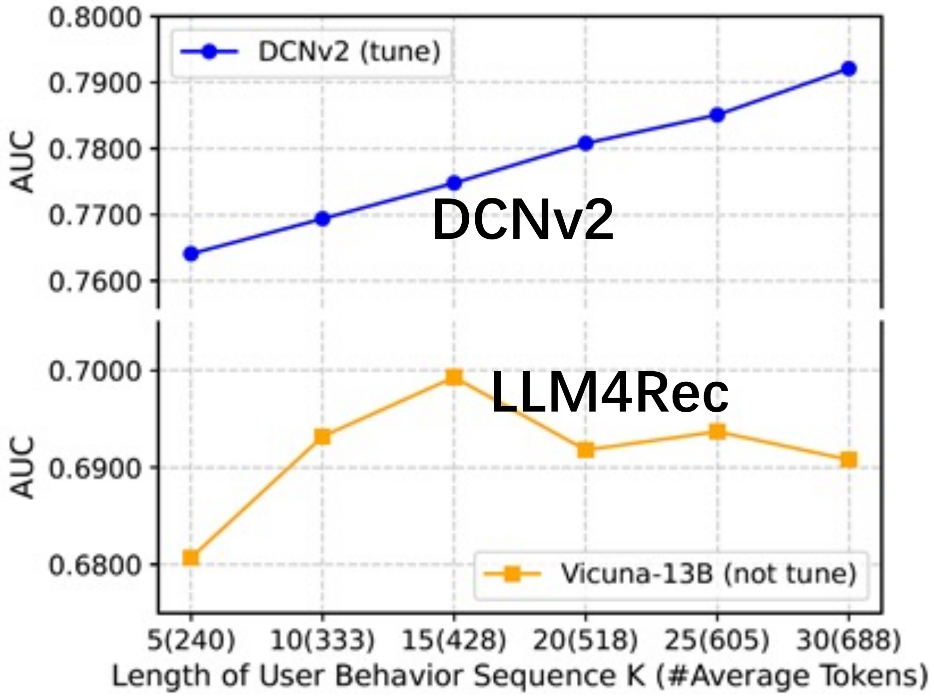
An example in the advertising system in Alibaba.

Qi Pi et al. Practice on Long Sequential User Behavior Modeling for Click-Through Rate Prediction. In KDD 2019.

Lifelong Modeling

Lifelong sequential behavior modeling:

LLM cannot effectively model long user Behavior sequence



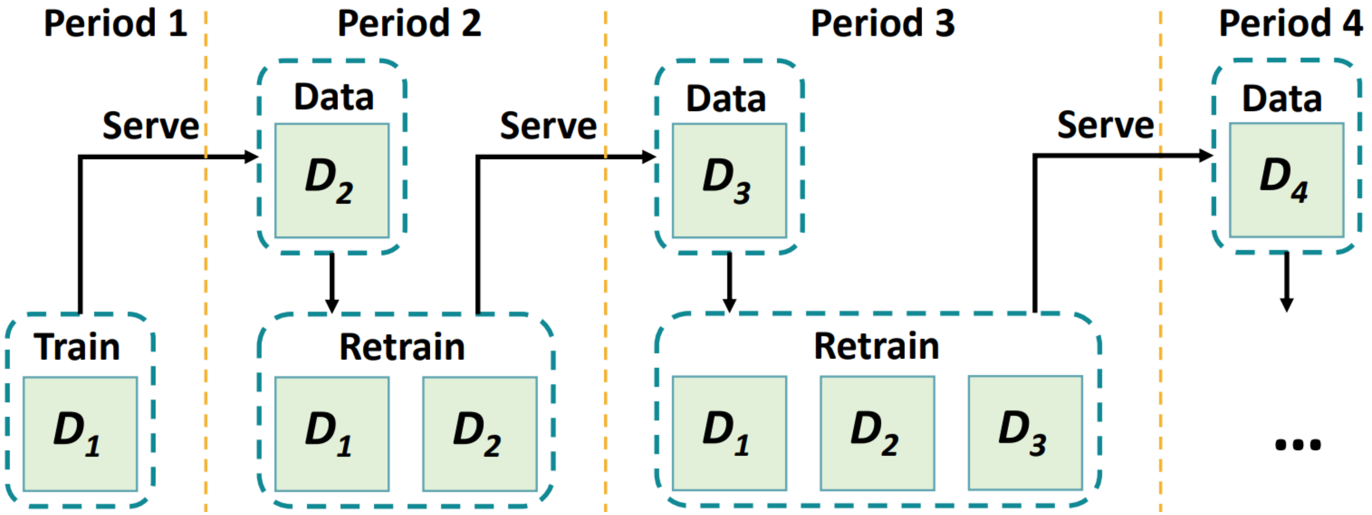
- Extending user behavior sequences doesn't necessarily enhance recommendation performance, even if the input length is far below the length limit of LLMs (e.g., Vicuna-13B has an upper limit of 2048 tokens).

Li et al. ReLLa: Retrieval-enhanced Large Language Models for Lifelong Sequential Behavior Comprehension in Recommendation. WWW 2024.

Lifelong Modeling

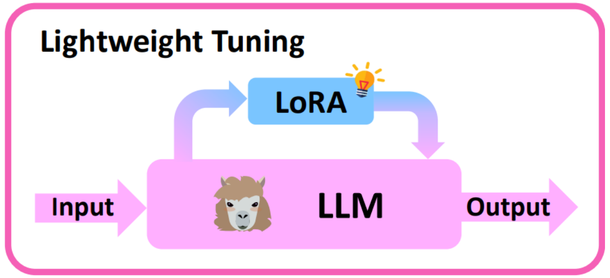
Continual learning：

- How to incrementally learn user interests?
- There is work [1] studying the common used methods: periodic retraining



Just retrain LoRA

(TALLRec)

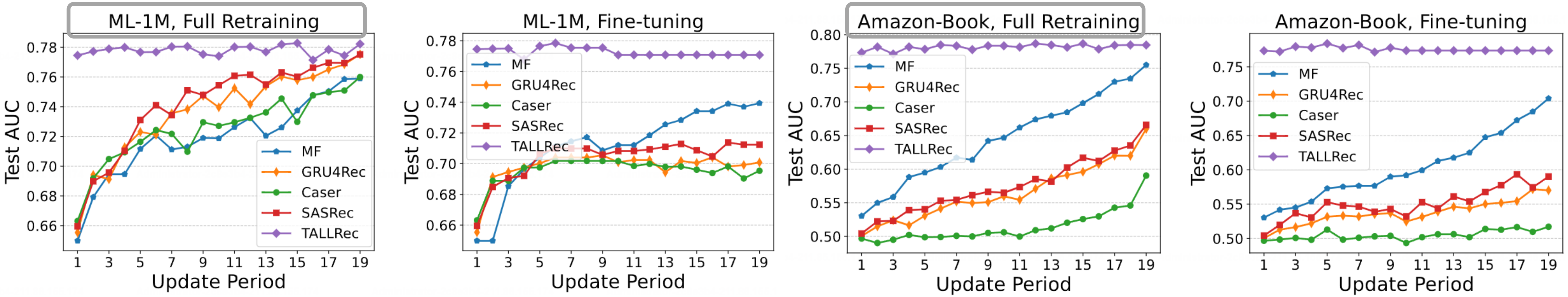


[1] Shi et al. Preliminary Study on Incremental Learning for Large Language Model-based Recommender Systems. In arXiv 2023.

Lifelong Modeling

Continual learning：

Work#1: The effectiveness of full-retraining and fine-tuning for TALLRec



- ❑ Periodically update TALLRec does not bring significant performance improvements.
- ❑ LLM4Rec may struggle to capture short-term preferences in the latest data with traditional periodic updates, limiting performance improvement.

[1] Shi et al. Preliminary Study on Incremental Learning for Large Language Model-based Recommender Systems. In arXiv 2023.

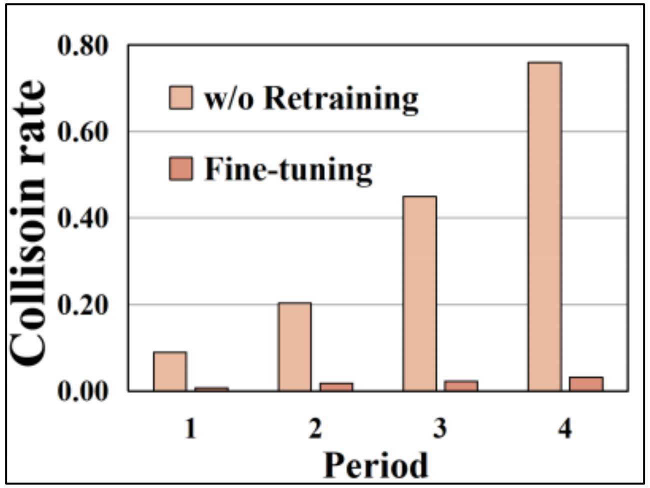
Lifelong Modeling

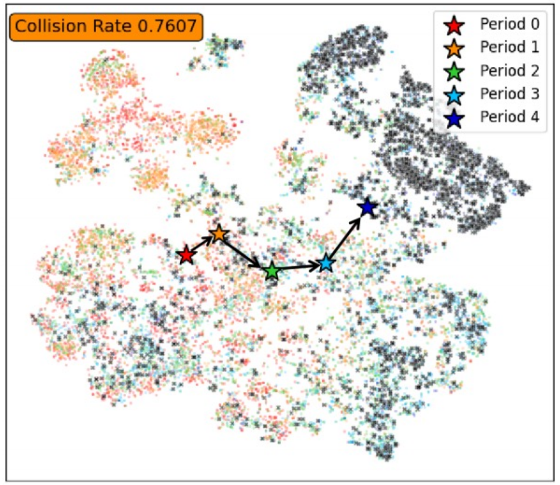
Work#2: Continual Learning with learnable tokenizer

Critical problems in tokenizer retraining (in codebook-based LLM4rec):

1. **Identifier Collision:** Frozen tokenizers fail to distinguish new items due to distribution shifts, assigning the same identifier to different items.

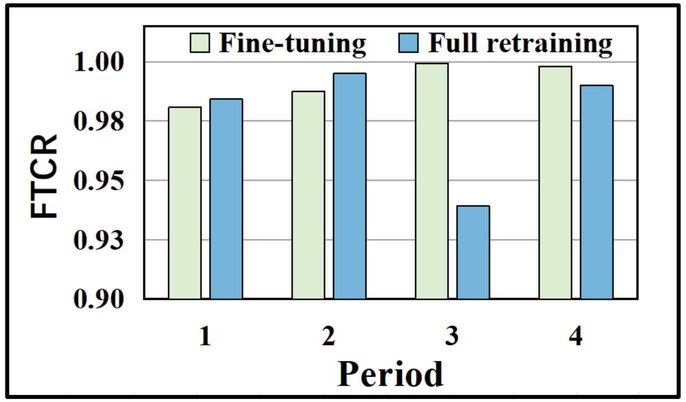
2. **Identifier Shift:** Retraining the tokenizer changes existing item identifiers due to parameter updates, disrupting the RecLLM's understanding of historical items, requiring costly full retraining to realign identifiers.





Left: Identifier collision rates with the frozen tokenizer across periods

Right: Item featrues distribution shifts

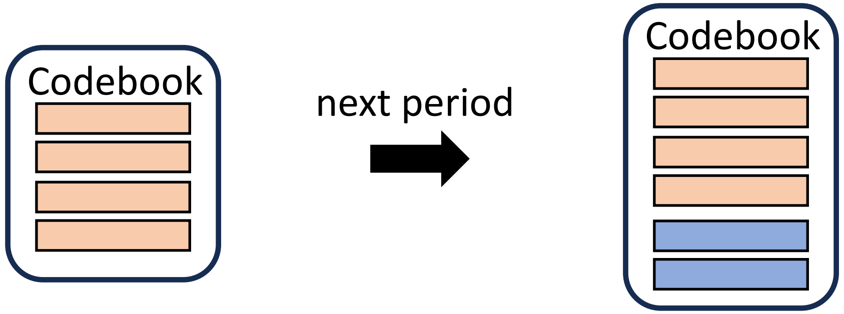


Change ratios of the first token in identifiers before and after tokenizer retraining

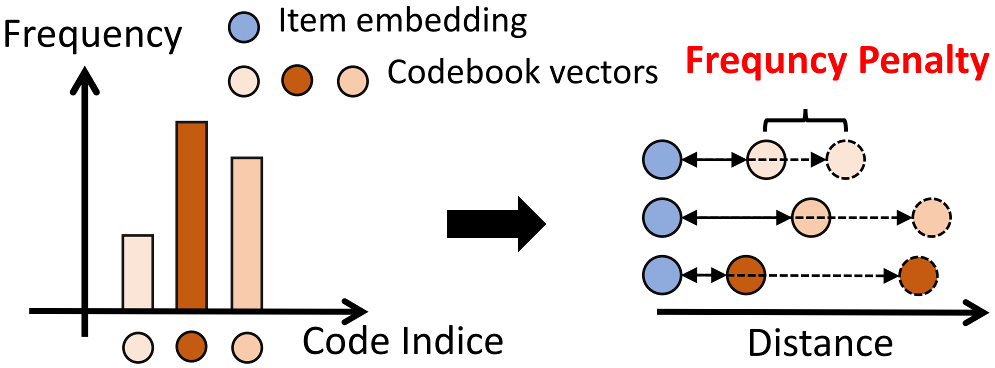
Retraining of RecLLMs with learnable tokenizers

Method:

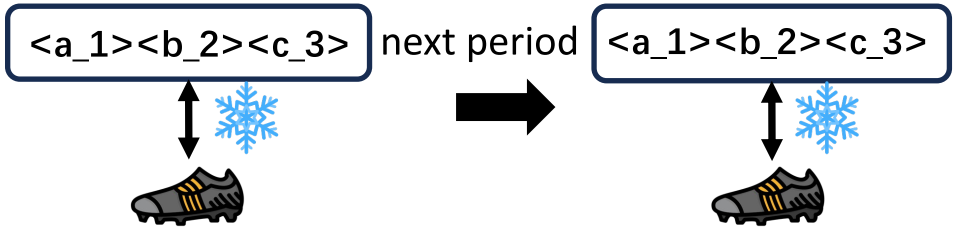
Dynamic Codebook Expansion addresses identifier collision issue by dynamically expanding the codebook at each retraining period.



Frequency-Based Diversity Constraint mitigates code assignment bias and enhances identifier distinguishability by calculating the assignment frequency of codebook vectors and penalizing overused codes during codebook quantization.



Historical Identifier Freezing ensures identifier invariance across retraining periods by storing and locking previously assigned identifiers.

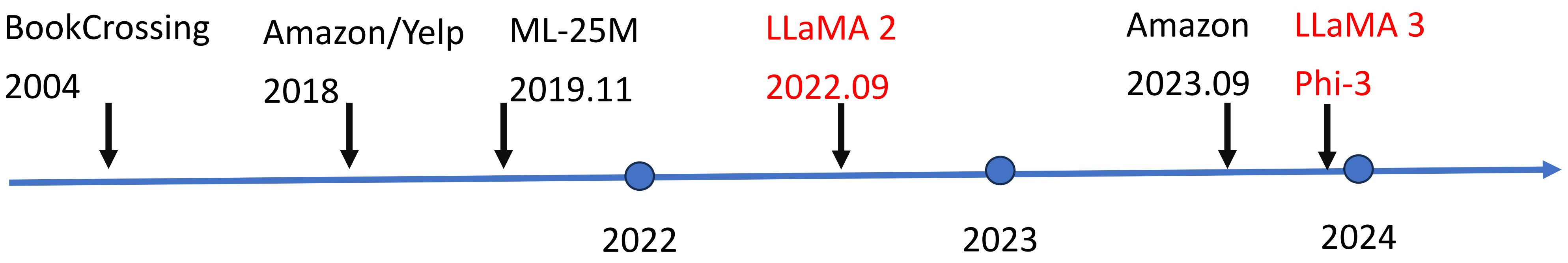


Outline

- Introduction
- Development of LLMs
- Technical Stacks of LLM4Rec
- **Open Problems**
  - Heterogeneous Modeling
  - Lifelong Modeling
  - **Evaluation**
- Future Direction & Conclusions

Evaluation: Data Issues

- **Challenge#1: Lack of data for evaluation**
  - **Most of benchmarks are proposed ahead of pre-training stage of LLMs, e.g., ChatGPT, LLaMA.**



  - The information of recommendation datasets (e.g., reviews,) may be include in LLMs.
  - Existing works usually did not discuss this.
  - Evaluations on the data that is not include in pretraining data of LLMs.

Evaluation: Data Issues

❑ Challenge#1: Lack of data for evaluation

- ❑ Insufficient features
  - ❑ Lack of raw feature
    - ❑ Anonymous (e.g., just feature ID)
    - ❑ Lack of content (e.g., video content)
  - ❑ Currently, many works just utilize titles

- **Underutilization of LLM capabilities;**
- **Underassessment of the effectiveness of LLM4Rec**

- ❑ Data homogeneity
  - ❑ content homogeneity: mostly from E-commerce platform / entertaining content or places
  - ❑ biased user distributions: mostly from China and U.S.

- **Not comprehensive evaluation**
- **Biased evaluation**

Evaluation: Interactive RecSys

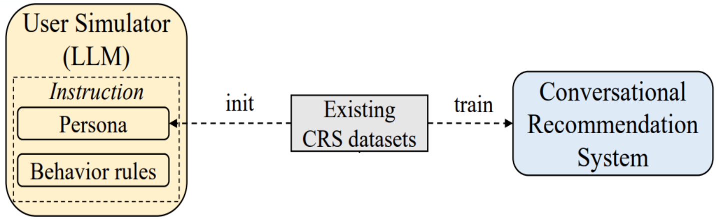
❑ Challenge#2: Evaluate interactive recommendation

❑ Conversational recommendation

- ❑ provide personalized recommendation via multi-turn dialogs in natural language
- ❑ focus on conversational quality and recommendation quality

- Issues of traditional evaluation:
  - Simulated users are overly simplified representations of human users
  - Conversations are often vague about the user preference, but not focus on exactly match the ground-truth item
  - Evaluation protocol is based on fixed conversations, but the conversation could be diverging.

- New evaluation: simulation with LLM-based agents?
  - Challenges: how to design simulators is still an open problem.



Xiaolei Wang et al. Rethinking the Evaluation for Conversational Recommendation in the Era of Large Language Models. EMNLP 2023

Evaluation: Interactive RecSys

- **Challenge#2: Evaluate interactive recommendation**
  - Long-term recommendation
    - Multi-turn user-system interactions
    - Focus on long-term user engagement, e.g., user retention
  - How to evaluate long-term engagement is a big challenge.
    - We have not feedback about the unseen interaction trajectory
    - Evaluation with agent-based simulator is a potential solution

Evaluation: platform and protocol

- **Platform: OpenP5**
  - **Develop, train, and evaluate LLM-based recommenders**
    - Customized **Dataset**
    - Customized **item indexing** methods
    - Personalized **prompt** collection
    - Extensibility of **multi-task learning**
    - New **backbone** methods

Shuyuan Xu et al. OpenP5: An Open-Source Platform for Developing, Training, and Evaluating LLM-based Recommender Systems. SIGIR'24

Outline

- Introduction
- Background: LM & LM4Rec
- Development of LLMs
- Progress of LLM4Rec
- Open Problems
- **Future Direction & Conclusions**

Generative Recommendation Paradigm

❑ Generative AI for recommendation

- Personalized **content generation**, including item repurposing and creation.
  - **Application**: News, fashion products, micro-videos, virtual products in games, etc.

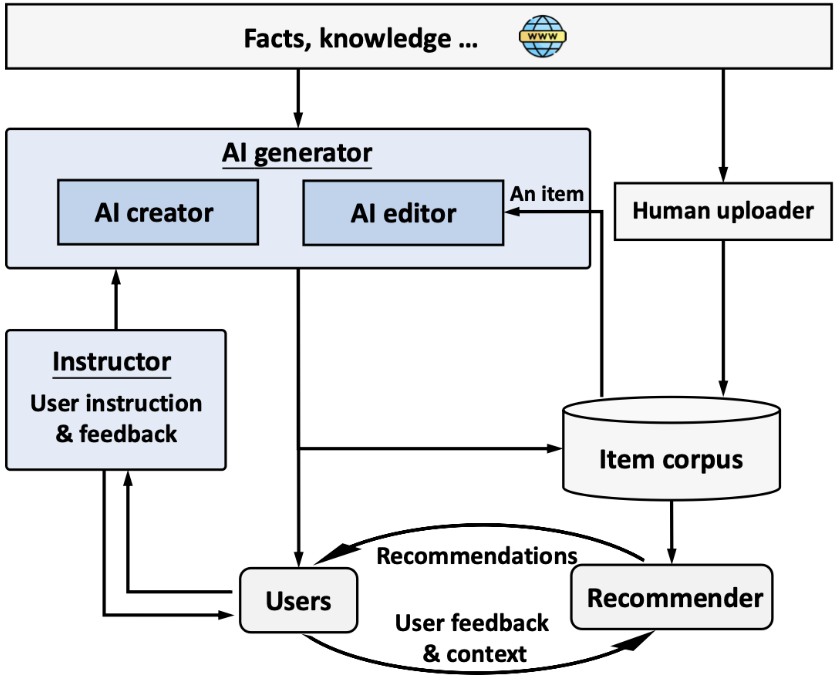


Figure 4: A demonstration of GeneRec. The instructor collects user instructions and feedback to guide content generation. The AI editor aims to repurpose existing items in the item corpus while the AI creator directly creates new items.

Instructor:

- Pre-process user instructions and feedback to guide the content generation of the AI generator.

AI Editor:

- Refine or repurpose existing items according to personalized user instructions and feedback.
- External facts and knowledge might be used for content generation.

AI Creator:

- Generate new items based on personalized user instructions and feedback.

AI Checker:

- Generation quality checks.
- Trustworthiness checks.

Applicable to many domains, including images, micro-videos, movies, news, books, and even products (for manufacture).

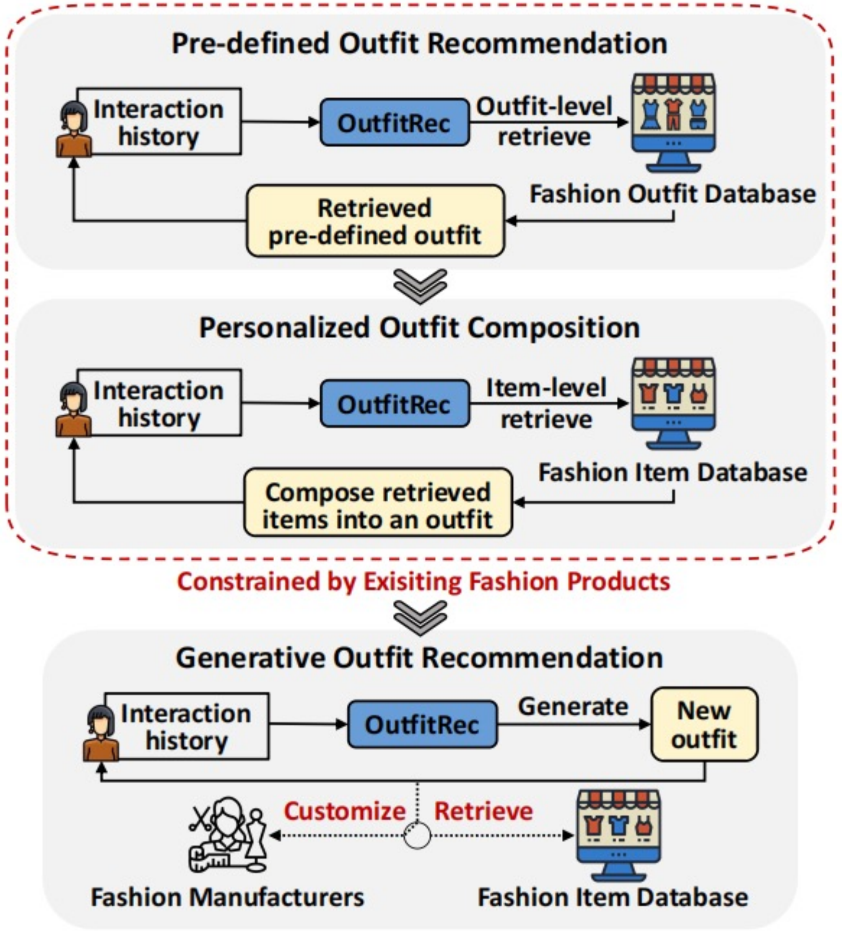
Wenjie Wang et al. Generative Recommendation: Towards Next-generation Recommender Paradigm. arxiv 2023

203

Generative Recommendation Paradigm

❑ Generative Recommendation in Fashion Domain

The Evolution of Fashion Outfit Recommendation

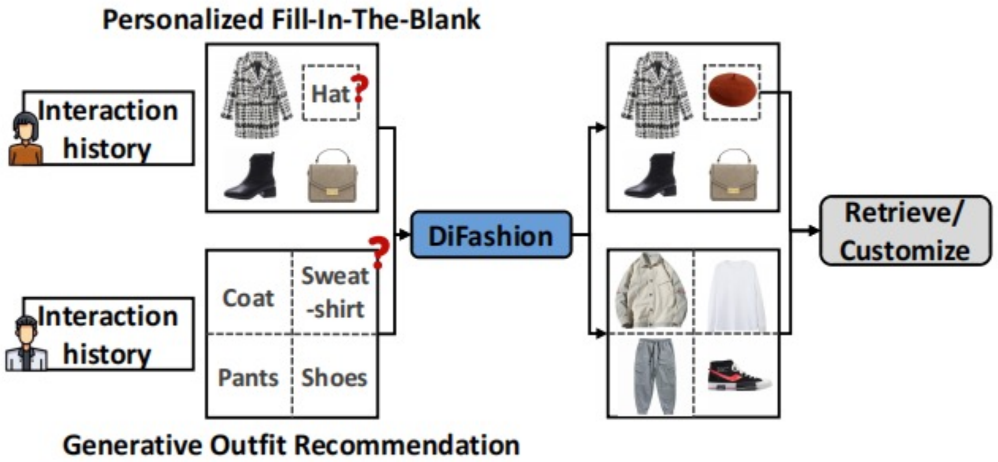


New Task

Generative Outfit Recommendation

Objective: generating a set of new personalized fashion products to compose a visually compatible outfit catering to users' fashion tastes.

Practical Implementation: retrieve or customize

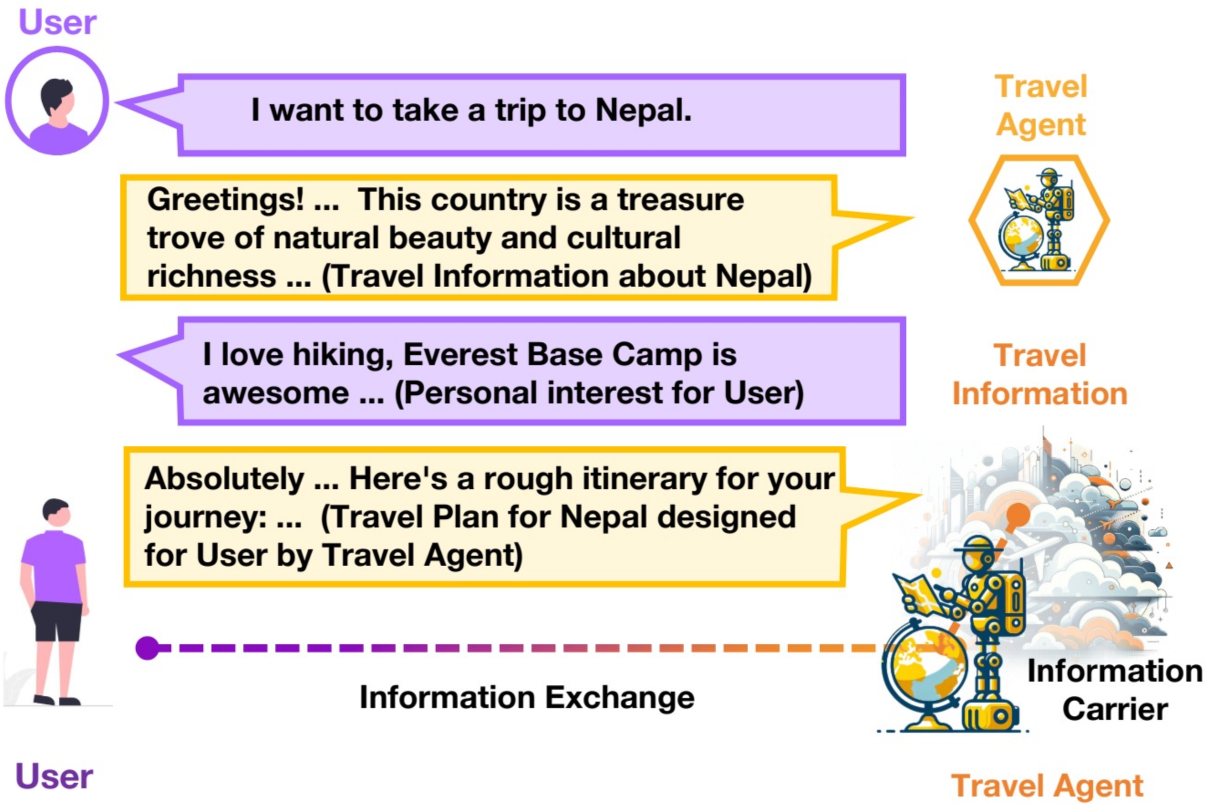


Yiyan Xu et al. Diffusion Models for Generative Outfit Recommendation. In SIGIR '24.

204

Recommender for Agent Platform

- Existing agent platforms such as GPTs (OpenAI), Poe (Quora), and DouBao (ByteDance) possess a vast number of LLM-based agents.
- How to recommend LLM-based Agent to the user?



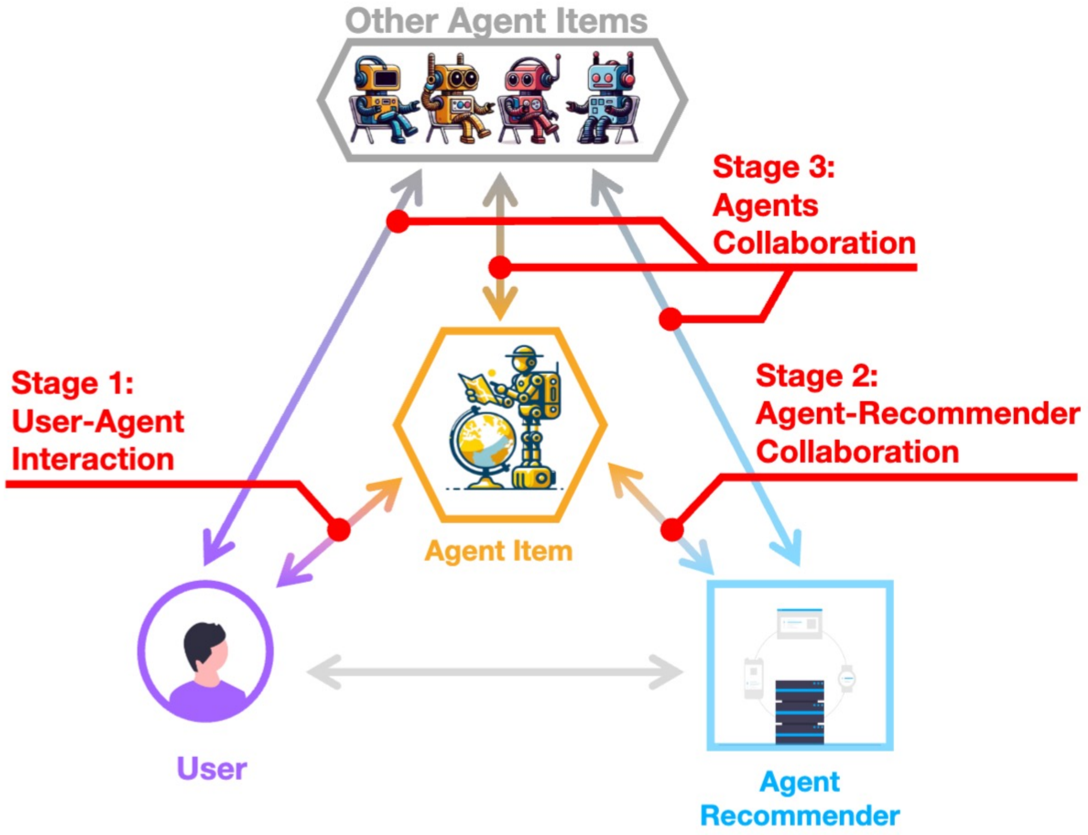
Different from Items in Traditional Recommender System, LLM-based Agent holds the potential to extend the format of information carriers and the way of information exchange.

-> Formulate new Information System

-> New Rec paradigm Rec4Agentverse

Jizhi Zhang et al. Prospect Personalized Recommendation on Large Language Model-based Agent Platform Arxiv 2024

Rec4Agentverse



Three stages of Rec4Agentverse . The bidirectional arrows depicted in the Figure symbolize the flow of information.

- User-Agent interaction stage:

Information flows between the user and Agent Item.

- Agent-Recommender Collaboration stage:

Information flows between Agent Item and Agent Recommender.

- Agents Collaboration stage:

Information flows between Agent Items.

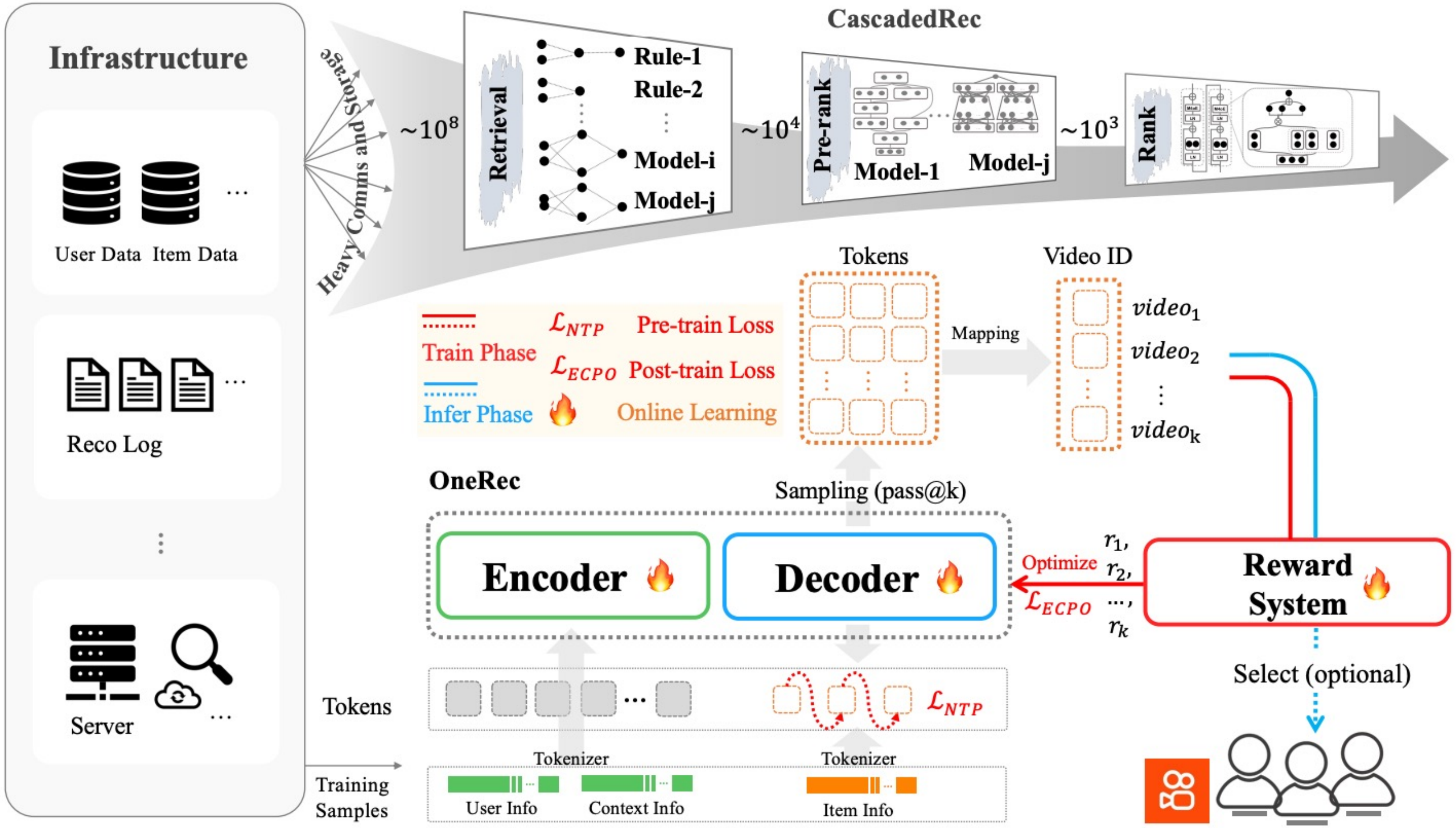
Jizhi Zhang et al. Prospect Personalized Recommendation on Large Language Model-based Agent Platform Arxiv 2024

End-to-end Generative Rec

- **Traditional multi-stage recommendation**

Key limitations:

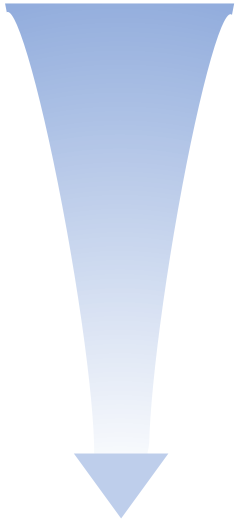
- *Fragmented Compute*
- *Objective collisions*
  - *Conflicts from Diverse Objectives*
  - *Cross-Stage Modeling Conflicts*
- *Lag Behind AI Evolution*



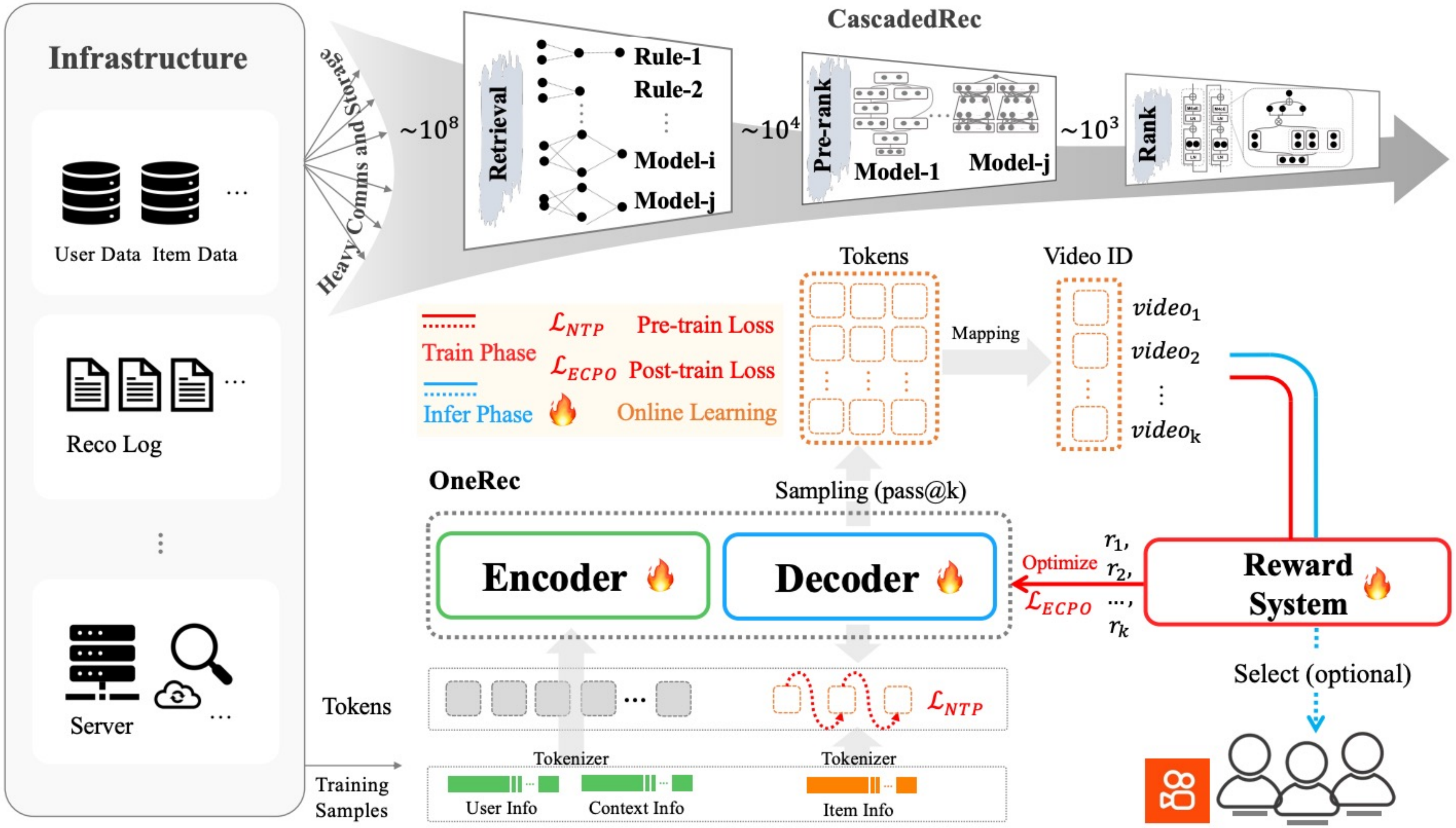
Kuaishou OneRec Team. OneRec Technical Report. 2025 Arxiv

End-to-end Generative Rec

Traditional **multi-stage** recommendation



End-to-end **single-stage** recommendation



Kuaishou OneRec Team. OneRec Technical Report. 2025 Arxiv

End-to-end Generative Rec

➢ End-to-end single-stage recommendation

Architecture

- **Tokenizer**: semantic identifier, RQ-VAE, align with CF signals
- **Encoder**: user static pathway, short-term pathway, positive-feedback pathway, and lifelong pathway
- **Decoder**: learnable beginning vector, autoregressive generate video semantic ID

Pre-training

- **Data info**: multi-scale user behavior representations as input. The pre-training objective involves predicting sequences of target items for users.
- **Data scale**: Exposure of **300 billion tokens** during pre-training.
- **Model size**: The **OneRec-0.935B** model

Post-training

- **Strategy**: Reject Sampling Fine-Tuning (RSFT) and Reinforcement Learning (RL). For RSFT, Onerec filters out the bottom 50% of exposure sessions based on play duration.

Kuaishou OneRec Team. OneRec Technical Report. 2025 Arxiv

End-to-end Generative Rec

➢ **Performance: accuracy and efficiency**

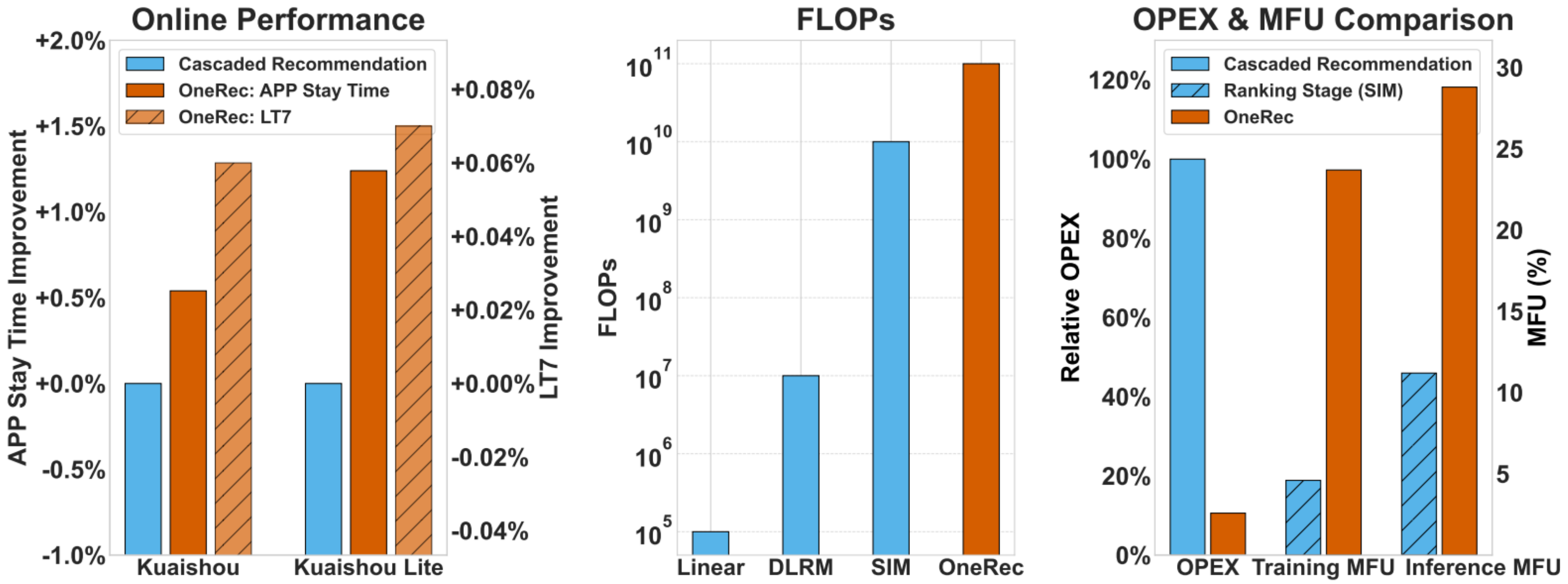


Figure 1 | Online performance, FLOPs, OPEX, and MFU comparison.

*The model's training and inference MFU is only **4.6% and 11.2% on flagship GPUs**, respectively, which is substantially lower than the efficiency observed in large language models (LLMs), where the MFU is approximately 40% on H100*

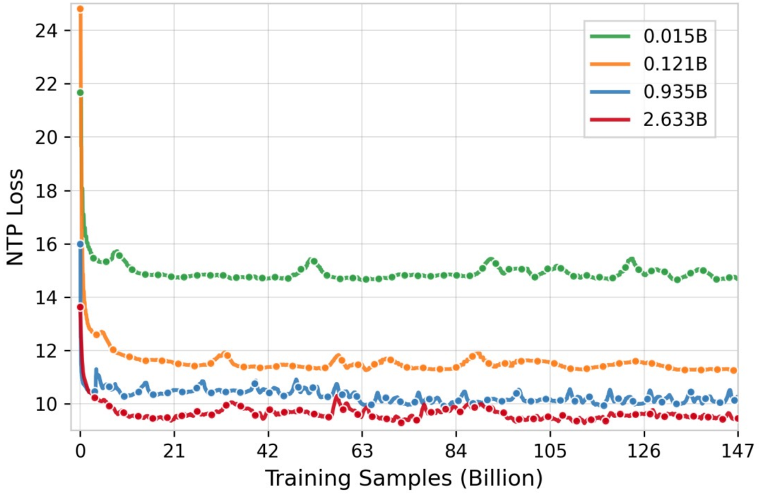
Kuaishou OneRec Team. OneRec Technical Report. 2025 Arxiv

End-to-end Generative Rec

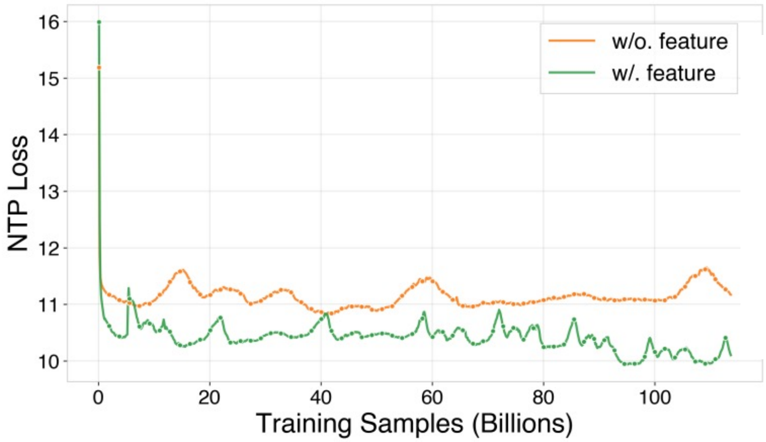
➢ Performance: scalability

• Parameter scaling

Model	Layers	Hid. Dim	FFN Hid. Dim	Attn. Heads	Experts (Tot/Act)	MoE Loc.
OneRec-0.015B (Dense)	4	128	256	4	N/A	N/A
OneRec-0.121B (Dense)	8	1024	2048	8	N/A	N/A
OneRec-0.935B (MoE)	8	1024	2048	8	24 / 2	Decoder
OneRec-2.633B (MoE)	24	1024	2048	8	24 / 4	Enc & Dec



• Feature scaling



Metric	w/o. feature	w/. feature	Impr.
lvtr	0.4940	0.5500	11.34%
vtr	0.8730	0.8901	1.96%
ltr	0.0391	0.0441	12.79%
wtr	0.0190	0.0224	17.89%
cmtr	0.0919	0.1010	9.90%
P-score	0.0749	0.0966	28.88%

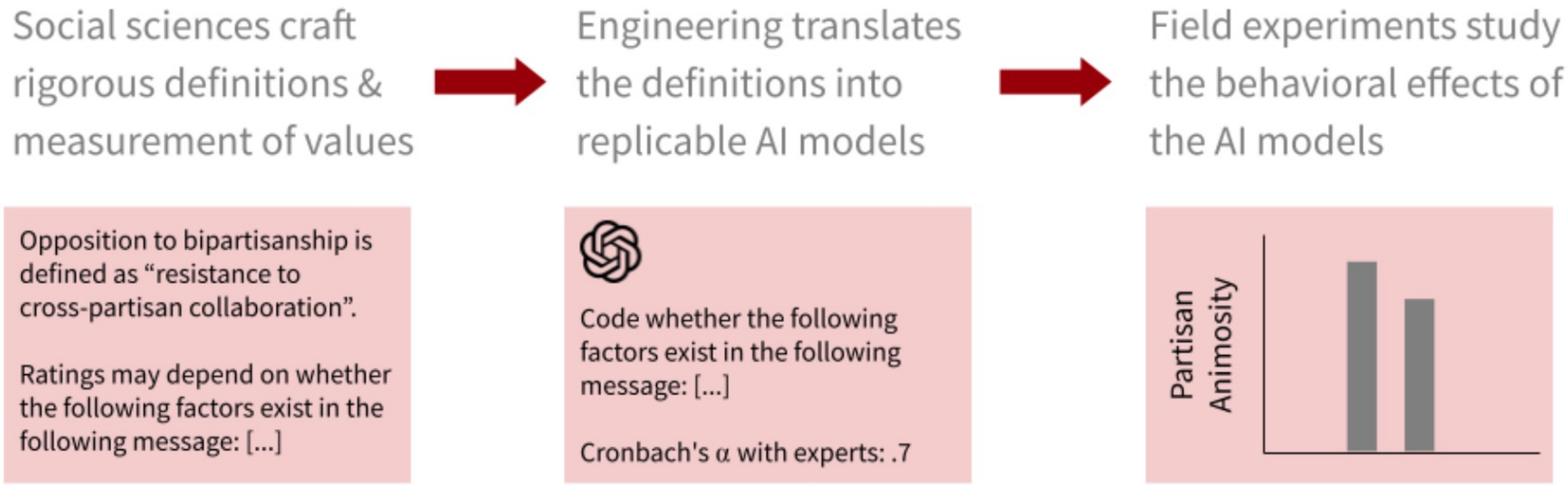
• Codebook scaling

Metric	Size=8K	Size=32K	Impr.
lvtr	0.5118	0.5245	2.48%
vtr	0.9384	0.9491	1.14%
ltr	0.0298	0.0299	0.34%
wtr	0.0153	0.0154	0.65%
cmtr	0.0650	0.0664	2.15%
P-score	0.2516	0.2635	4.75%

Kuaishou OneRec Team. OneRec Technical Report. 2025 Arxiv

Social Value Alignment of LLMRec

- **Social media AI already embed values** --- maximize each user's individual experience---as predicted through likes in RecSys
- **It can harm societal values** --- wellbeing, social capital, mitigating harm to minoritized groups, democracy, and maintaining pro-social norms.
- **Could we directly encode societal values into RecSys?**



Michael S. Bernstein, et al. Tuning Our Algorithmic Amplifiers: Encoding Societal Values into Social Media AIs.

Thanks for Your Listening !

Tutorial on Large Language Models for Recommendation

Find our slides at

https://generative-rec.github.io/tutorial-sigir25/

Survey: A Survey of Generative Search and Recommendation in the Era of Large Language Models

https://arxiv.org/pdf/2404.16924

Tutorial

Survey

智荐阁

Follow our WeChat account "智荐阁"!

Hiring@USTC LDS: Hiring tenure-track faculties, postdoc, PhD/Master student, Intern

Lab for Data Science: http://data-science.ustc.edu.cn/main.htm