

Tutorial on Generative Recommendation: Foundations and Frontiers



Xiaopeng Li¹



Yejing Wang¹



Wenlin Zhang¹



Ziwei Liu¹



Kaifeng Guo¹



Bo Chen²

Speaker:

Contributor: Honghui Bao², Sheng Zhang¹, Binhao Wang¹, Qinglin Jia², Kuo Cai², Qiang Luo², Ruiming Tang², Xiangyu Zhao¹

¹ City University of Hong Kong ² Kuaishou Technology



Survey Paper

*KDD 2026 | Jeju, Korea
August, 9, 2026*



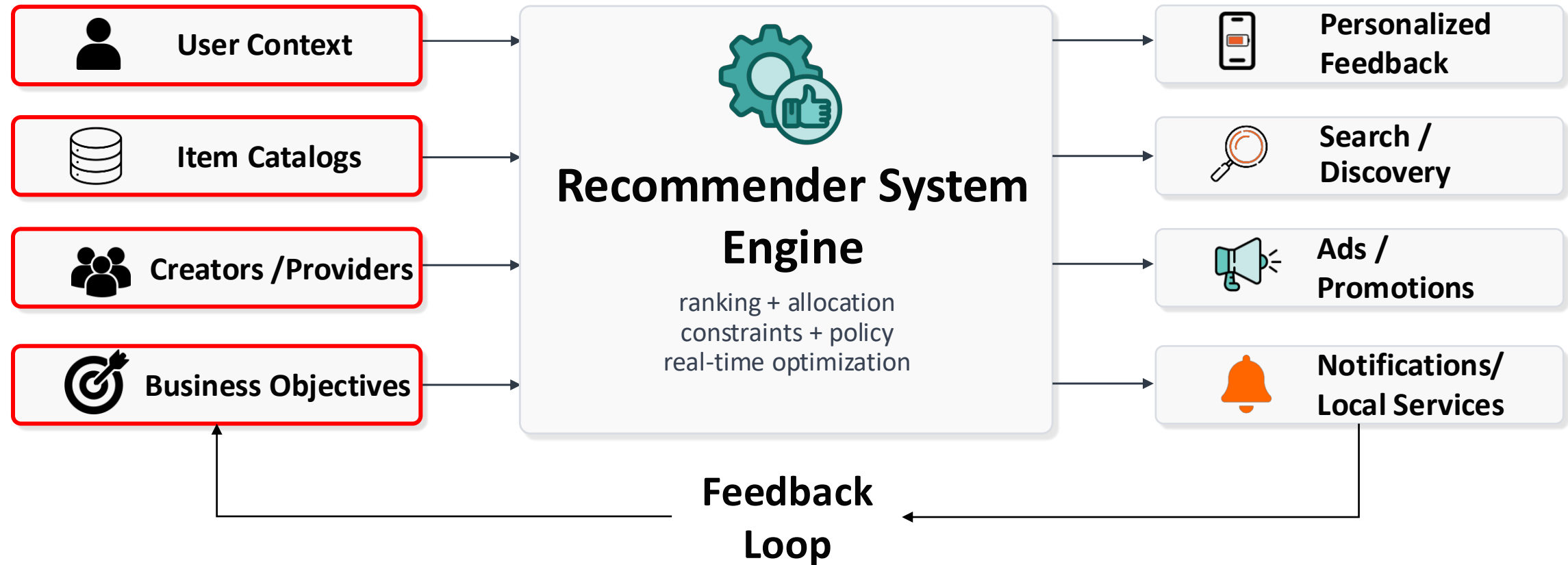
Project Page

Multi-objective Goals / Real Time Policy

User

Platform

Provider

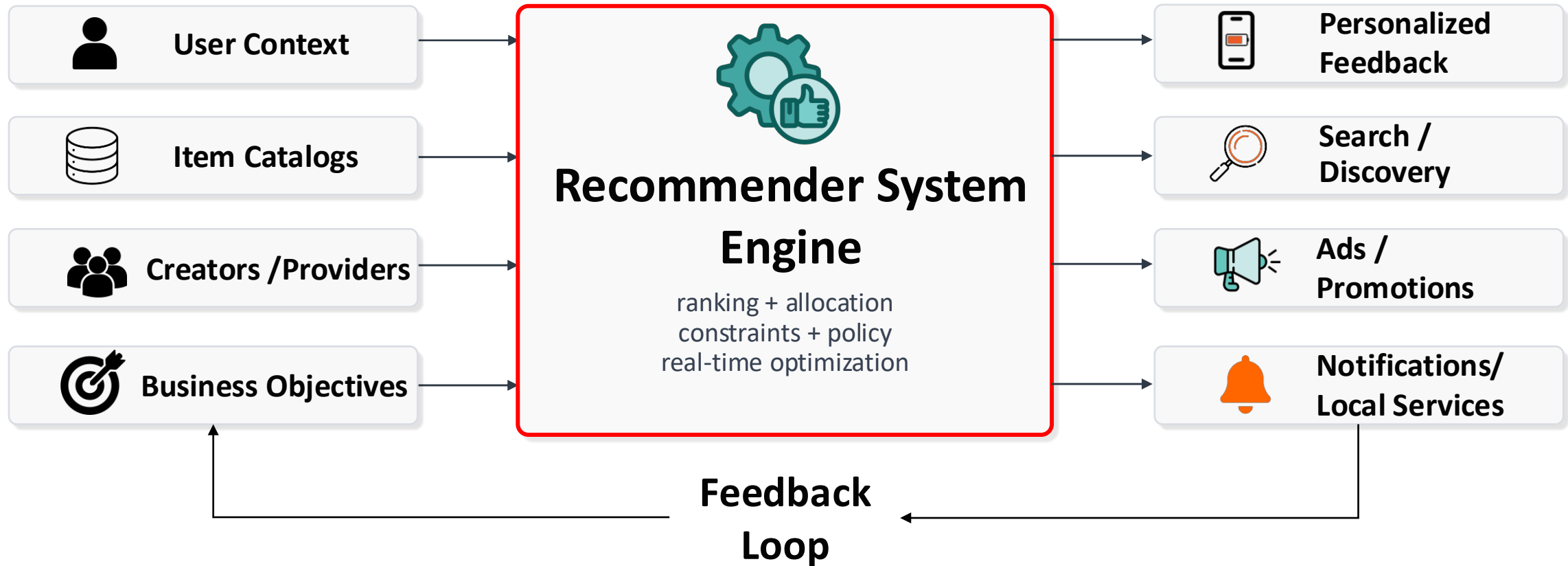


Multi-objective Goals / Real Time Policy

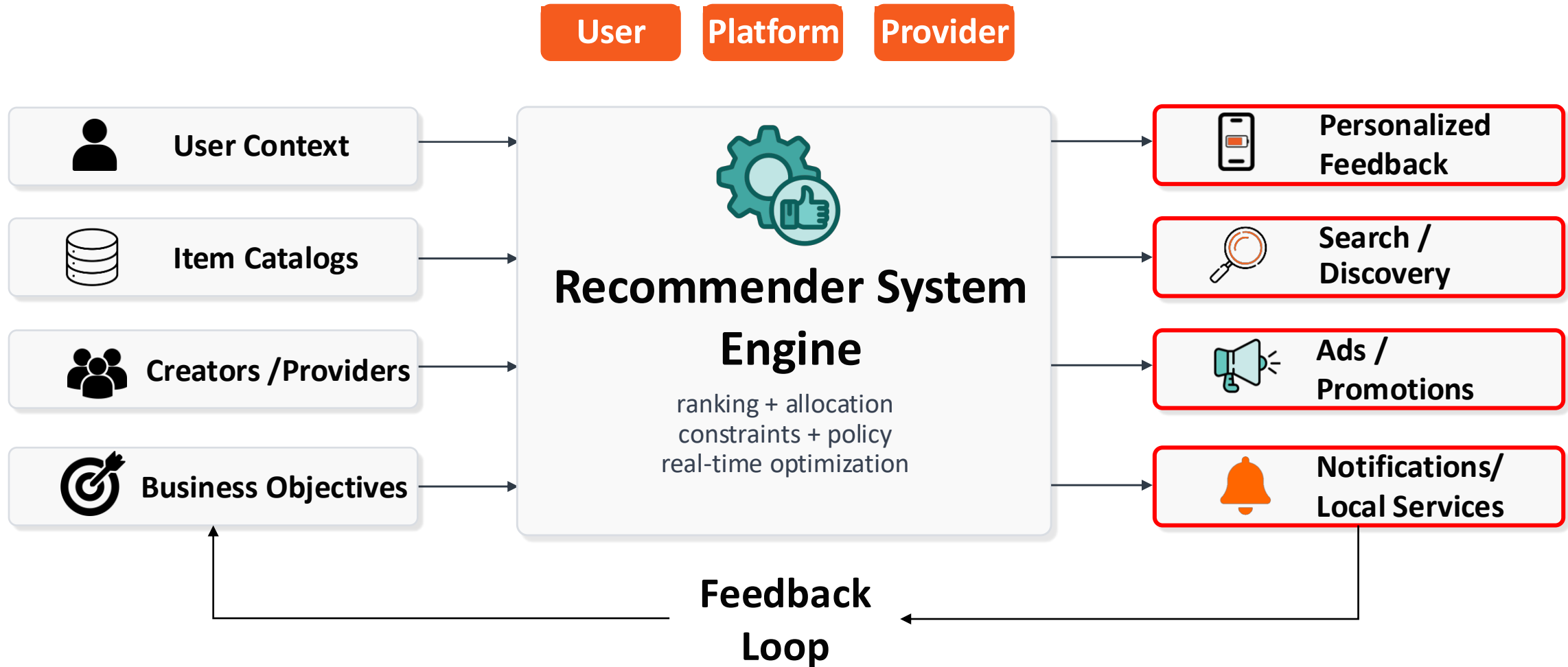
User

Platform

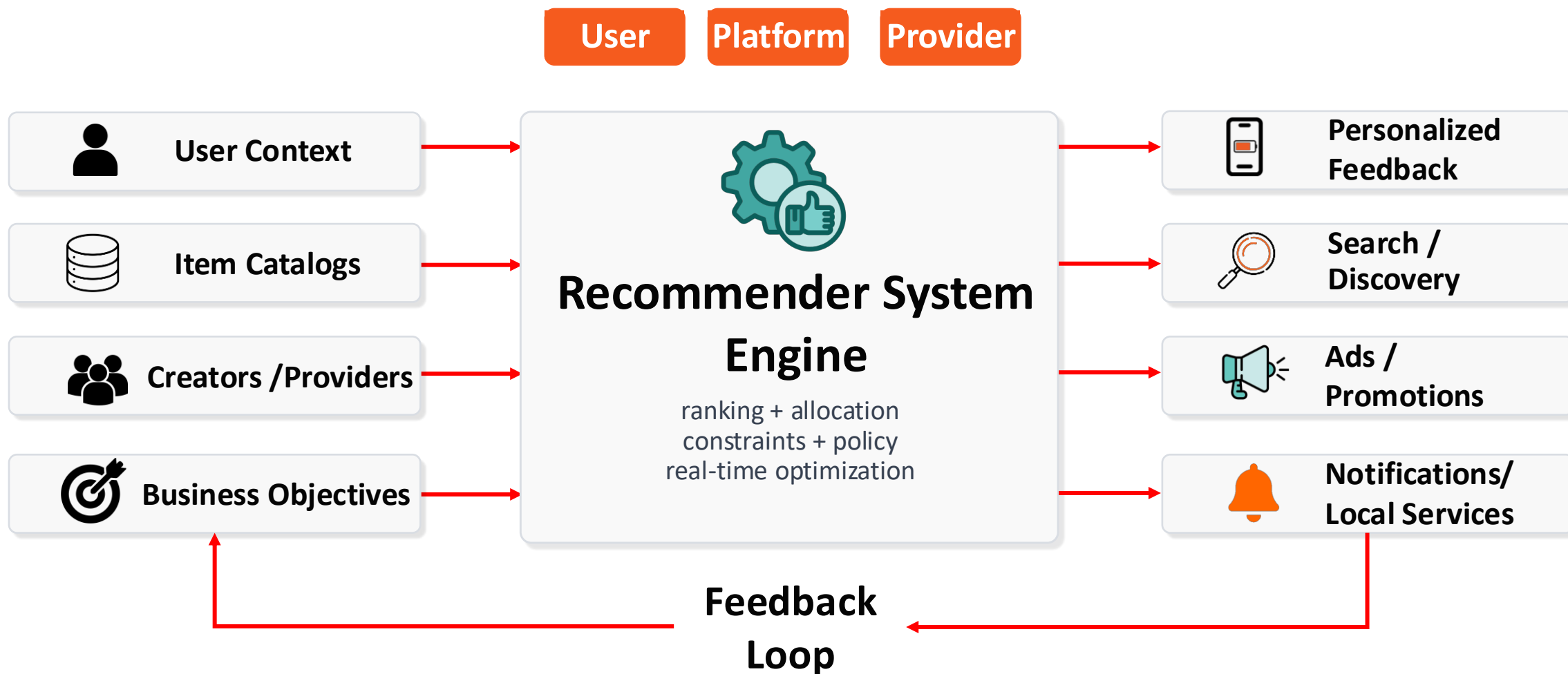
Provider



Multi-objective Goals / Real Time Policy



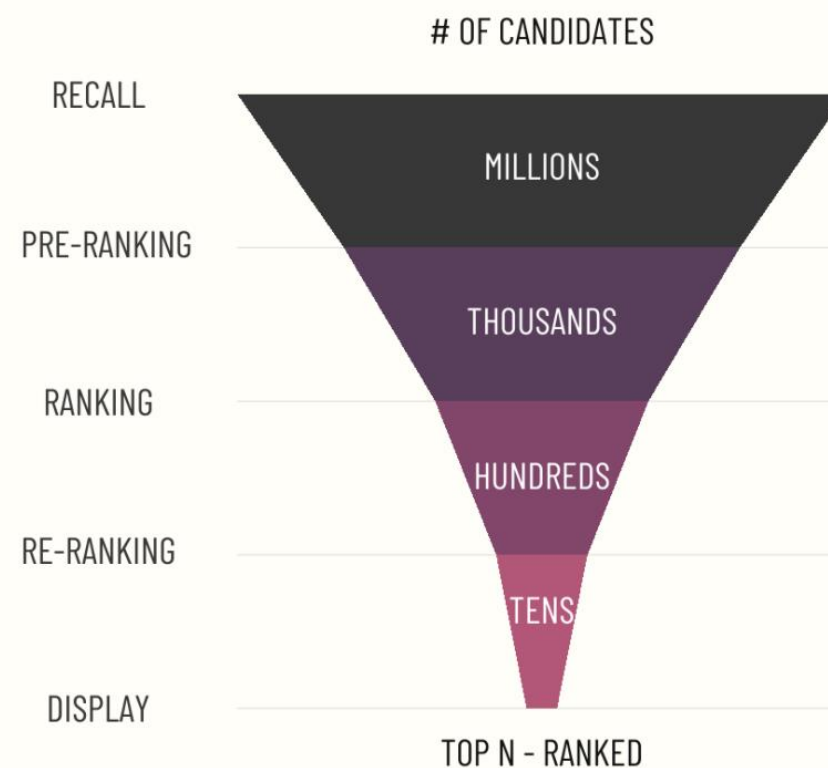
Multi-objective Goals / Real Time Policy



Current Industrial Recommendation Pipeline:

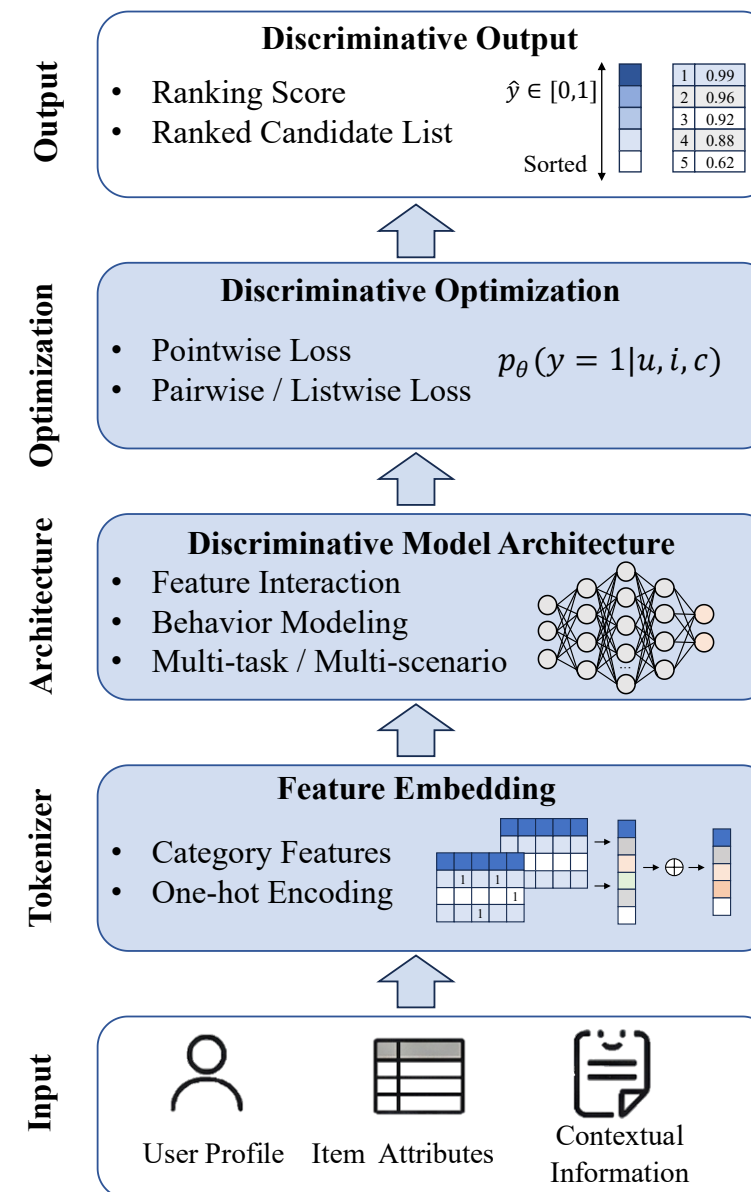
- Multi-stage Cascade Ranking System
 - Recall
 - Pre-Ranking
 - Ranking
 - Re-Ranking
- Not just engineering but decomposes a global recommendation problem into **local discrimination score problems**
- Cause **information loss, inconsistent goals, and accumulated errors**

Stages of a Cascade Ranking System



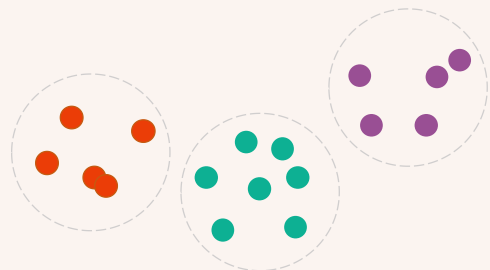
Discriminative Recommendation (GR):

- Structure follows “Embedding – MLP”
 - Input
 - Feature Embedding
 - Discriminative Model Structure
 - Optimization
 - Output
- Learns a **scoring function** over a given user, item, and context



Scoring pipelines create three structural bottlenecks:

1 Representation Bottleneck



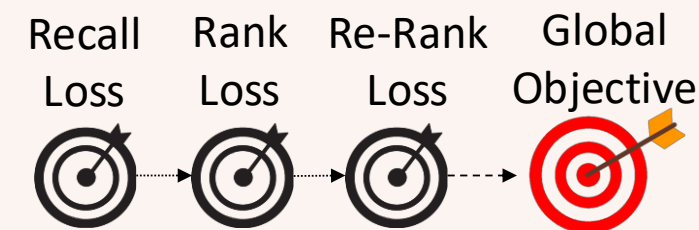
- Atomic item embeddings create semantic isolation, making cold-start and cross-domain transfer difficult.

2 Architecture Bottleneck



- Fragmented components limit scaling and MFU (<5%).
- Modest-sized models modest-sized models

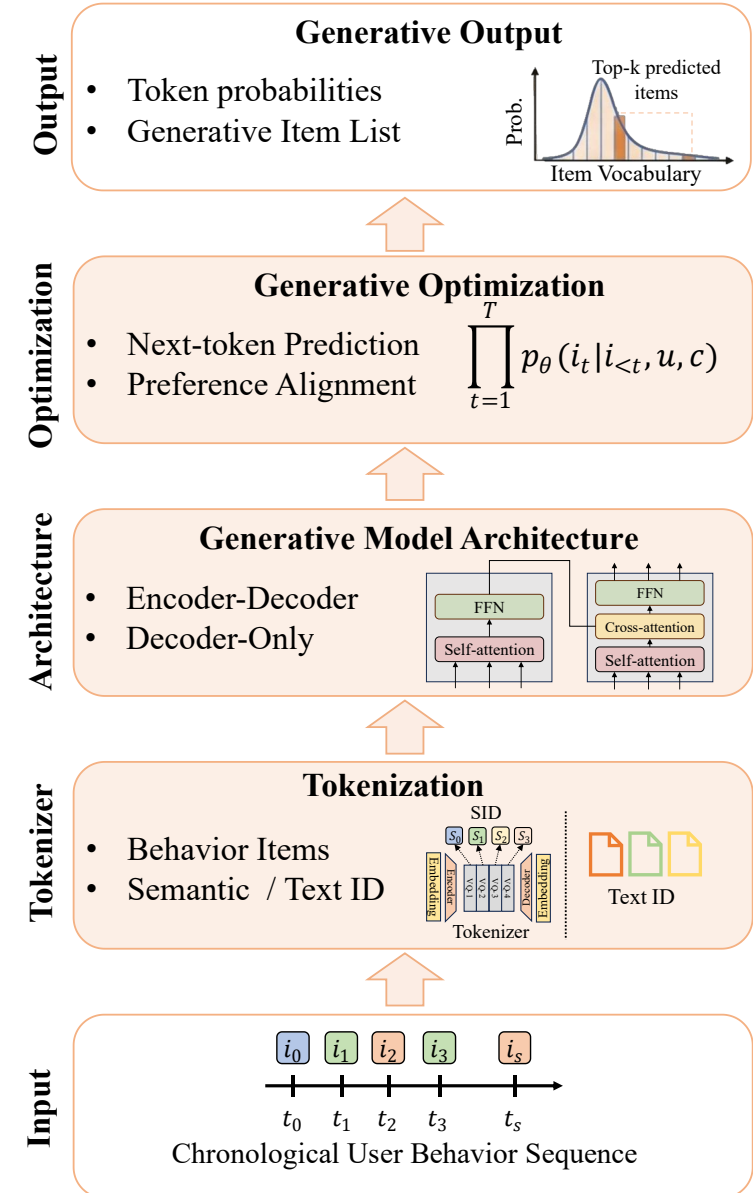
3 Optimization Bottleneck



- Stage-specific losses miss global item distributions and multi-stakeholder goals.
- Cumulative errors with progressive information loss

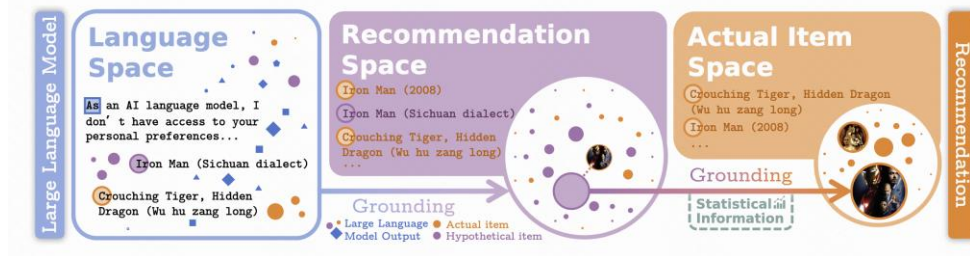
Generative Recommendation (GR):

- Rather than scoring or ranking, GR directly generate item identifiers through generative models
- Structure paradigm shift:
 - Input
 - Item Tokenization
 - Generative Model Structure
 - Generative Optimization
 - Output

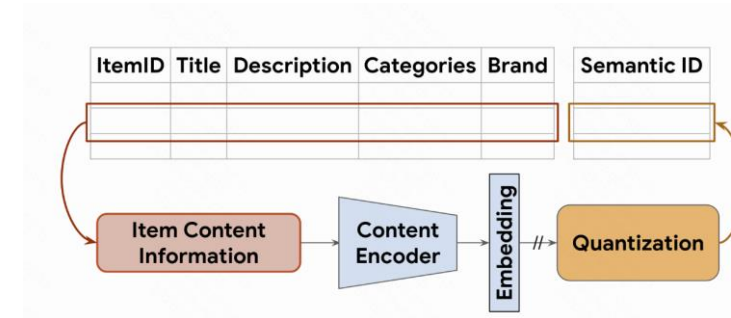


Advantage of GR Tokenization:

1. Operating at the **semantic level**



Reusing LLM vocabulary – BIGRec



Semantic ID – TIGER

2. More **compact vocabulary** design

Amazon-Reviews-2023

~48,200,000 Item IDs/ Tokens



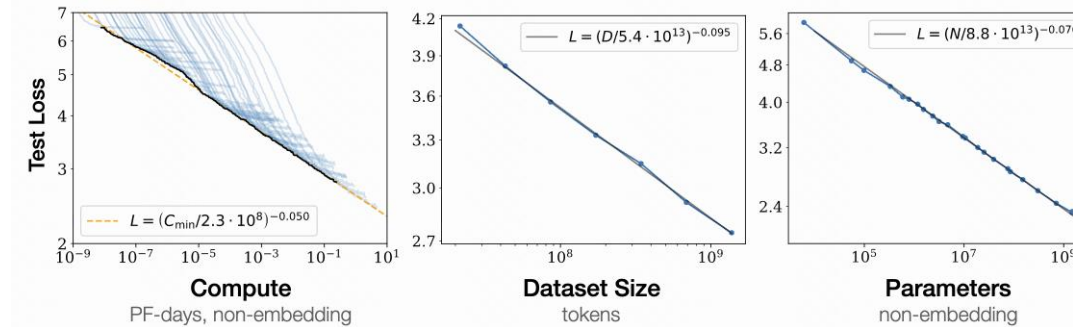
Llama-4 ~ 200,000 Tokens



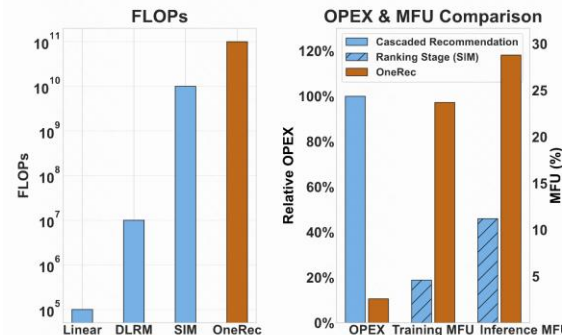
Qwen-3.5 ~ 250,000 Tokens

Advantage of GR Architecture:

1. Model size scaling advantages



2. High MFU and Maximizing hardware computation



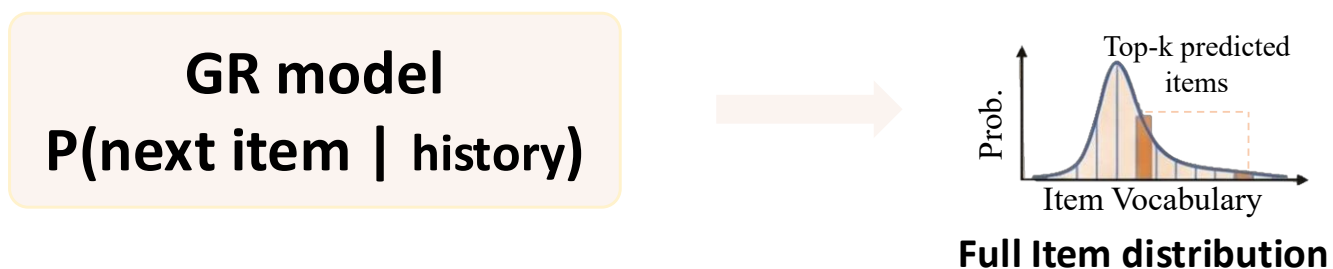
3. Leverage NLP Technologies

- PEFT, Reasoning, Agentic, etc.

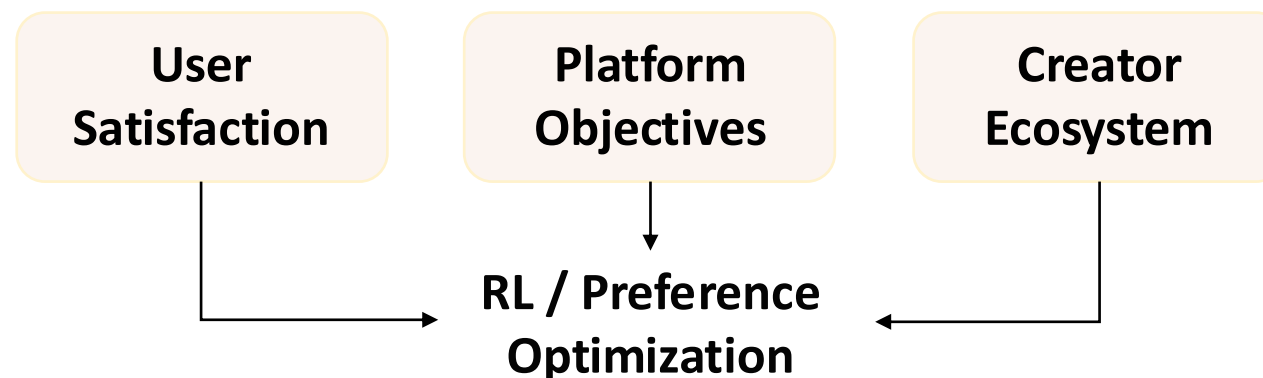
Advantage of GR Optimization:

1. Next-token prediction over item tokens

- No longer only to separate positives from negatives.



2. Preference alignment



1 Introduction (9:00 - 9:15)



Xiaopeng Li



2 Tokenization (9:15 - 9:45)



Xiaopeng Li

3 Model Structure (9:45 - 10:30)



Ziwei Liu



15mins

4 Optimization (10:45 - 11:15)



Wenlin Zhang



Kaifeng Guo

5 Application (11:15 - 11:30)



Yejing Wang



6 Challenge & Conclusion (11:30 - 11:45)



Bo Chen

- **What is Tokenization:**

- Convert “human-readable data” to “machine-readable formats”

Text, Image, Action  Sequence of Tokens

- **Language Tokenization:**

- Process of breaking text into small units, called tokens, that a language model can understand and process.

“I love recommender systems.”  ["I", " love", " recommend", "er", " systems", "."]
[40, 3021, 8542, 261, 6067, 13]

- **Recommendation Tokenization:**

- Process of converting users, items, and interactions into discrete tokens that a generative model can understand and predict.

We divide the categories of current Tokenization method in three types:

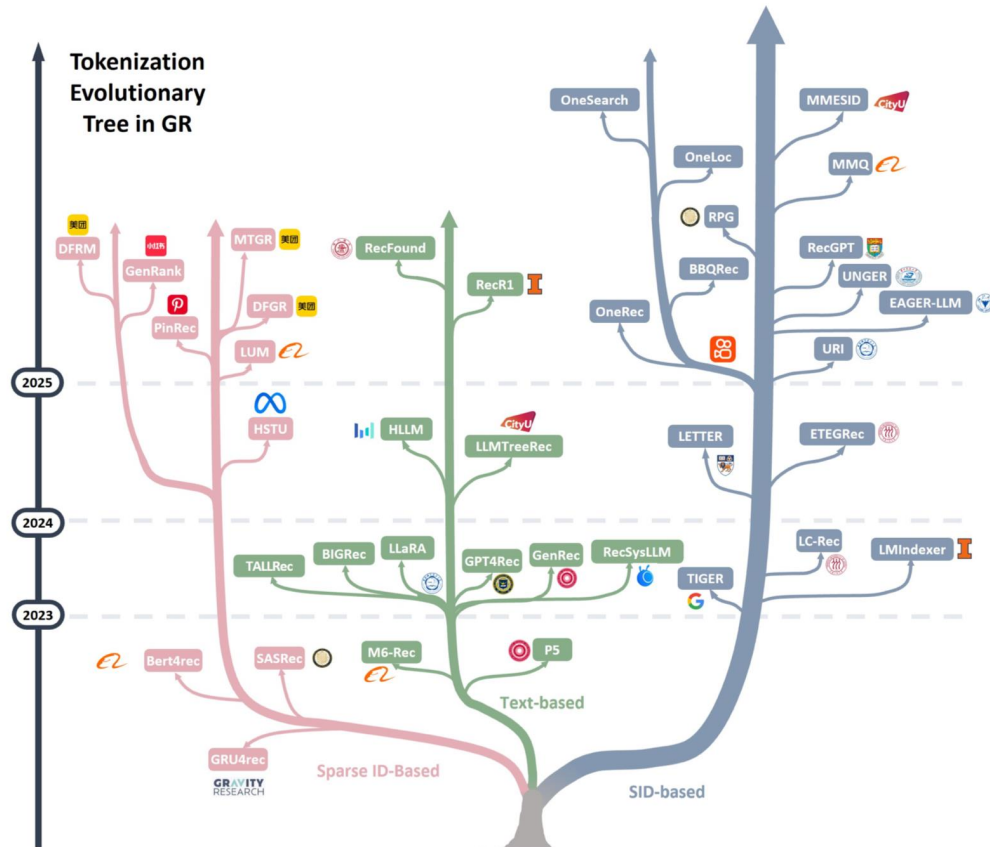


Figure 3. The evolution of tokenizer paradigms, showing the transition from sparse ID-based and text-based models toward semantic ID approaches, with affiliations noted beside each work.

- **Sparse ID-based**
 - Random unique IDs;
 - Massive vocabulary.
- **Text ID-based**
 - Natural language descriptions;
- **Semantic ID-based**
 - Fixed-length quantized sequences;
Compact & Semantic.

- **Background**

- Traditional Foundation: Rooted in the classic "Embedding & MLP" paradigm, assigning unique, parameterized vectors to each sparse ID (e.g., user/item ID).
- Generative Adaptation: Reformulates recommendation as a sequential transduction task. User behaviors are tokenized into chronological sequences for causal autoregressive modeling.

- **Motivation**

- Zero ID Collision
- Rich Feature Interaction
- Industrial Readiness

- **Limitation of DLRMs:**

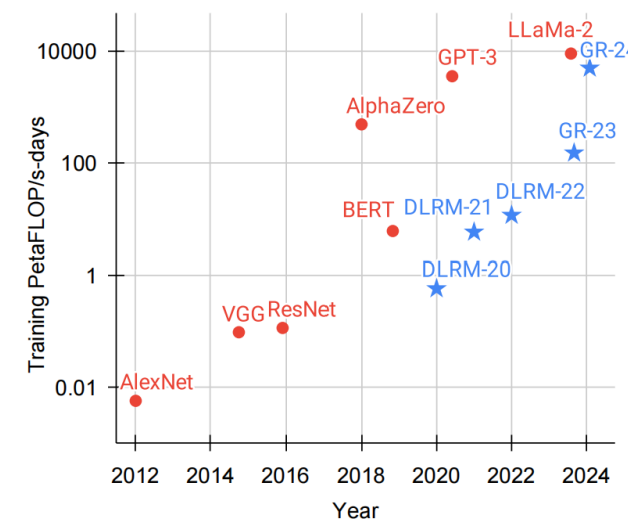
- Despite extensive features and vast data, most Deep Learning Recommendation Models scale poorly with compute

- **Key Challenges:**

- Features lack explicit structures in recommendation systems
- Billion-scale dynamic vocabulary (vs. 100K static in language)
- Computational cost: recommendation systems handle orders of magnitude more tokens than language models

- **Scale Gap:**

- GPT-3 trained on 300B tokens over 1-2 months, while platforms process billions of user actions daily



Generative Recommenders (GRs)

- Reformulate ranking and retrieval as sequential transduction tasks
- Unify heterogeneous features into single time series
- Interleave items and actions: $[\text{item}_1, \text{act}_1, \dots, \text{item}_n, \text{act}_n]$

HSTU Architecture:

- Pointwise aggregated attention (replaces softmax)

$$U(X), V(X), Q(X), K(X) = \text{Split}(\phi_1(f_1(X)))$$

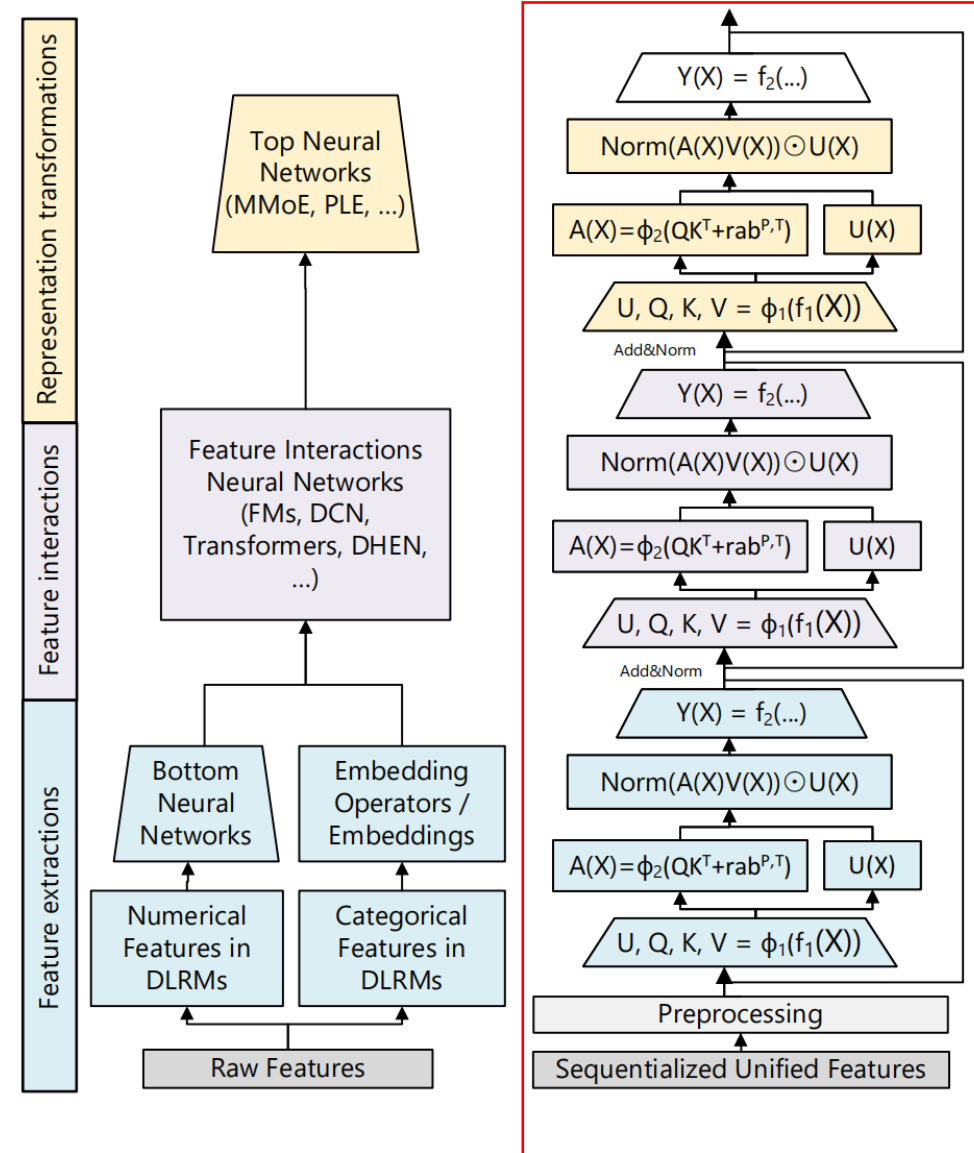
$$A(X)V(X) = \phi_2 \left(Q(X)K(X)^T + \text{rab}^{p,t} \right) V(X)$$

$$Y(X) = f_2(\text{Norm}(A(X)V(X)) \odot U(X))$$

- Captures preference intensity for non-stationary vocabulary
- $5.3\times$ to $15.2\times$ faster than FlashAttention2-based Transformers

Efficiency Innovations:

- Stochastic Length (SL): reduces sequence sparsity
- M-FALCON: batched inference for ranking candidates



• Offline Performance:

- HSTU outperforms SASRec by up to 65.8% in NDCG on public datasets
- $1.5\times-2\times$ faster wall-clock time with 50% less HBM usage

• Online A/B Tests:

- 12.4% improvement on main engagement metrics
- Deployed on multiple surfaces with billions of users

• Scaling Law:

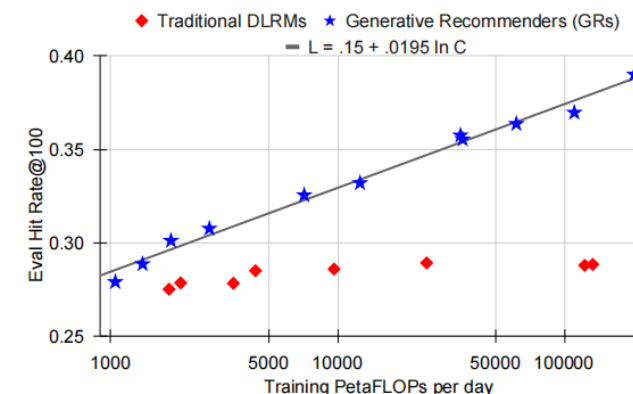
- First demonstration that scaling law applies to recommendations
- Model quality scales as power-law of training compute
- Successfully scaled to 1.5 trillion parameters (GPT-3/LLaMa-2 scale)

Table 4. Evaluations of methods on public datasets in multi-pass, full-shuffle settings.

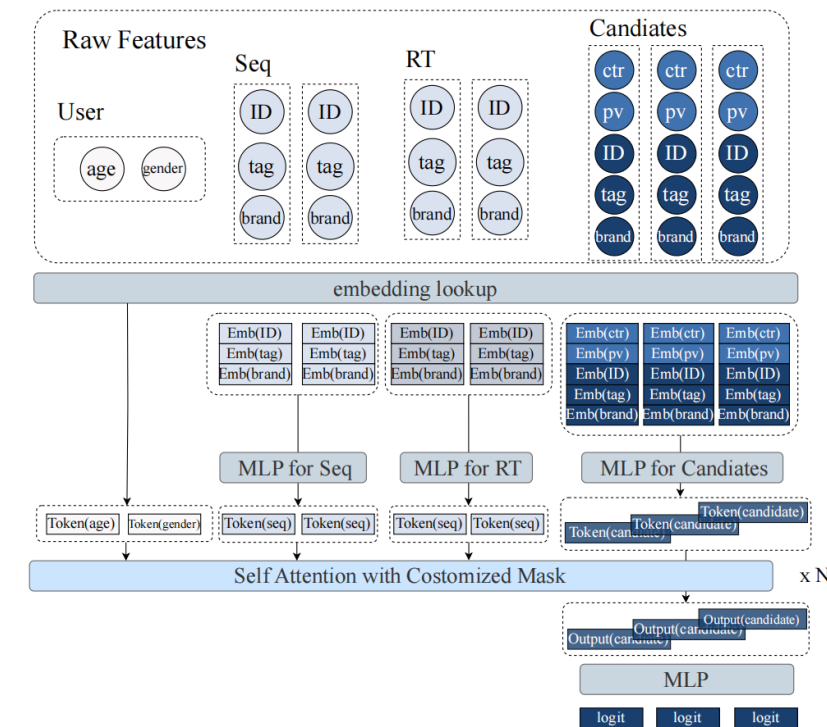
	Method	HR@10	HR@50	HR@200	NDCG@10	NDCG@200
ML-1M	SASRec (2023)	.2853	.5474	.7528	.1603	.2498
	HSTU	.3097 (+8.6%)	.5754 (+5.1%)	.7716 (+2.5%)	.1720 (+7.3%)	.2606 (+4.3%)
	HSTU-large	.3294 (+15.5%)	.5935 (+8.4%)	.7839 (+4.1%)	.1893 (+18.1%)	.2771 (+10.9%)
ML-20M	SASRec (2023)	.2906	.5499	.7655	.1621	.2521
	HSTU	.3252 (+11.9%)	.5885 (+7.0%)	.7943 (+3.8%)	.1878 (+15.9%)	.2774 (+10.0%)
	HSTU-large	.3567 (+22.8%)	.6149 (+11.8%)	.8076 (+5.5%)	.2106 (+30.0%)	.2971 (+17.9%)
Books	SASRec (2023)	.0292	.0729	.1400	.0156	.0350
	HSTU	.0404 (+38.4%)	.0943 (+29.5%)	.1710 (+22.1%)	.0219 (+40.6%)	.0450 (+28.6%)
	HSTU-large	.0469 (+60.6%)	.1066 (+46.2%)	.1876 (+33.9%)	.0257 (+65.8%)	.0508 (+45.1%)

Table 7. Offline/Online Comparison of Ranking Models.

Methods	Offline NEs		Online metrics	
	E-Task	C-Task	E-Task	C-Task
DLRM	.4982	.7842	+0%	+0%
DLRM (DIN+DCN)	.5053	.7899	—	—
DLRM (abl. features)	.5053	.7925	—	—
GR (interactions only)	.4851	.7903	—	—
GR	.4845	.7645	+12.4%	+4.4%



- **Scaling Law Dilemma:**
 - Generative recommendation shows excellent scalability but requires abandoning meticulously constructed cross features
- **Critical Challenge for Sparse ID:**
 - How to retain sparse ID-based cross features (user-item interactions) while achieving GRM scalability?
 - Removing cross features severely damages performance, cannot be compensated by scaling
- **Industrial Requirements:**
 - High QPS and low latency constraints
 - Need to process billions of user actions with sparse ID sequences



• Sparse ID Tokenization Strategy:

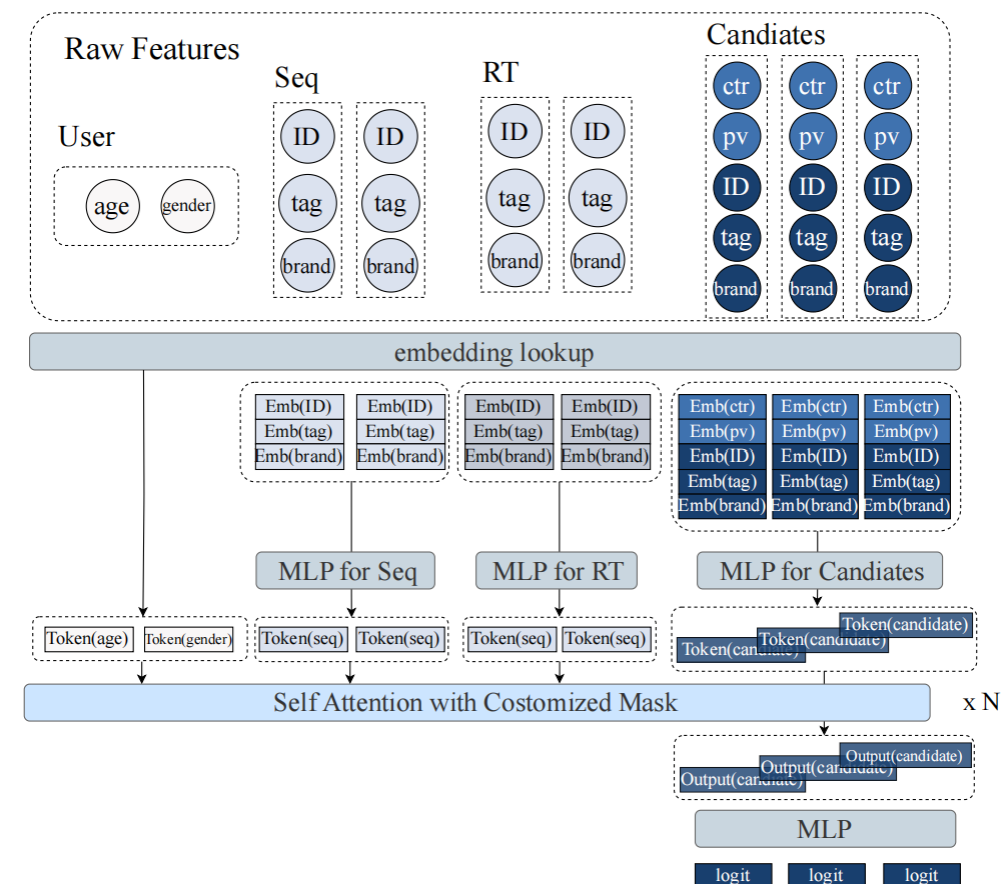
- Converts user features (U), sequence features ($\rightarrow S$, $\rightarrow R$), and candidate features (C, I) into unified token sequences
- Aggregates candidates by user: $\mathbb{D} = [\mathbf{U}, \vec{S}, \vec{R}, [\mathbf{C}, \mathbf{I}]_1, \dots, [\mathbf{C}, \mathbf{I}]_K]$
- Preserves all sparse ID-based cross features in candidate tokens

• Architecture Innovations:

- HSTU-based encoder with Group-Layer Normalization (GLN) for different semantic spaces
- Dynamic masking: Full attention for static features, autoregressive for real-time behaviors, diagonal mask for candidates

• Efficiency Optimization:

- Dynamic hash tables for sparse embedding entries
- Embedding ID de-duplication and automatic table merging



- **Background: The Limitation of Sparse IDs**

- Semantic Void: Random IDs lack inherent meaning; models rely solely on collaborative signals
- Cold-Start Bottleneck: Struggles to represent long-tail or newly introduced items.
- The Paradigm Shift: Tokenizing items via natural language (e.g., titles, attributes) using LLM pre-trained vocabularies

- **Motivation & Advantages**

- World Knowledge Integration: Leverages the LLM's dense semantic space and reasoning capabilities
- Zero/Few-Shot Generalization: Naturally resolves cold-start issues and enables cross-domain transfer
- Interpretability & Interactivity: Provides human-readable explanations and supports conversational recommendations

- **Limitations:**

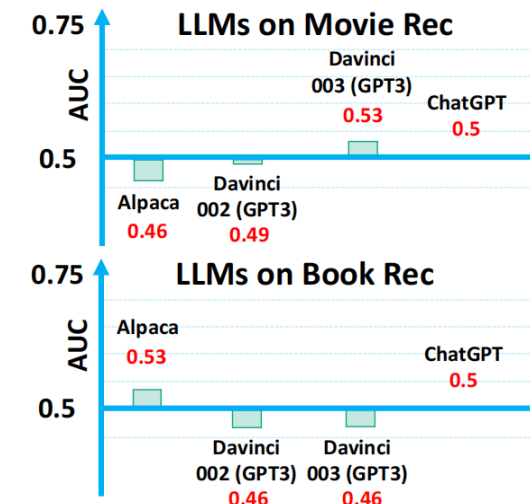
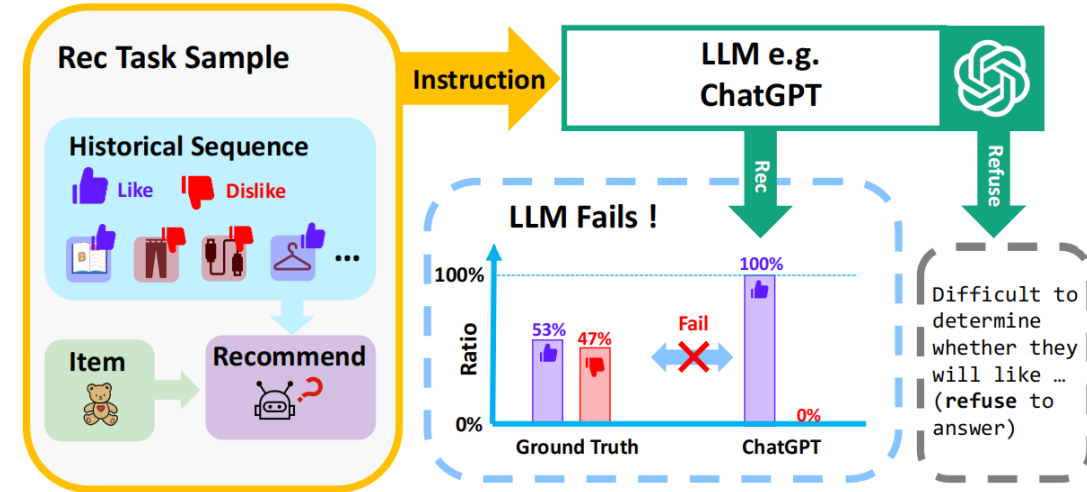
- Random IDs lack semantic information
- Cold-start problem for long-tail items
- Cannot leverage LLM's world knowledge

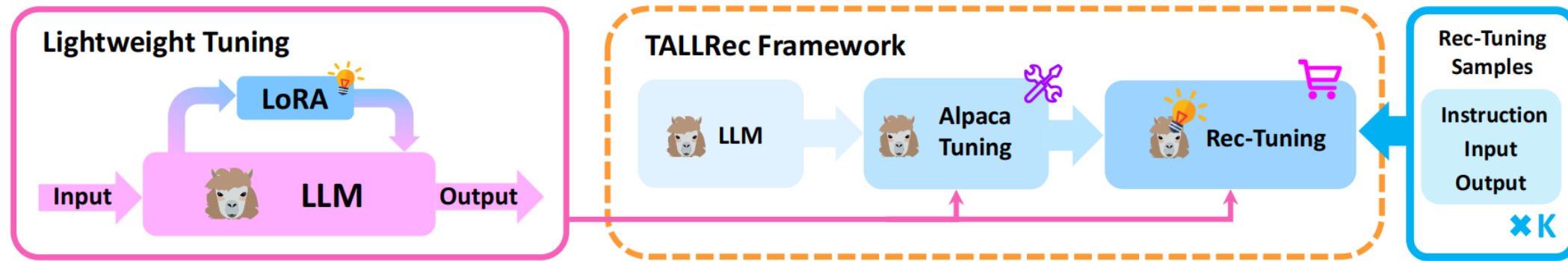
- **Critical Finding:**

- LLMs with In-context Learning perform at random level (AUC ≈ 0.5)
- ChatGPT either refuses to answer or always predicts "like"

- **Core Challenge:**

- Huge gap between language tasks and recommendation
- Need to align LLMs with recommendation without sparse IDs





- **Text-based Tokenization:**

- Replace sparse IDs with textual descriptions
- Format: "attribute name: attribute value" (e.g., "Title: iPhone 17")
- Leverage LLM's pre-trained vocabulary (~30K tokens vs millions of sparse IDs)

- **Two-Stage Tuning:**

- Alpaca Tuning: General instruction following capability
- Rec-Tuning: Recommendation-specific alignment with LoRA

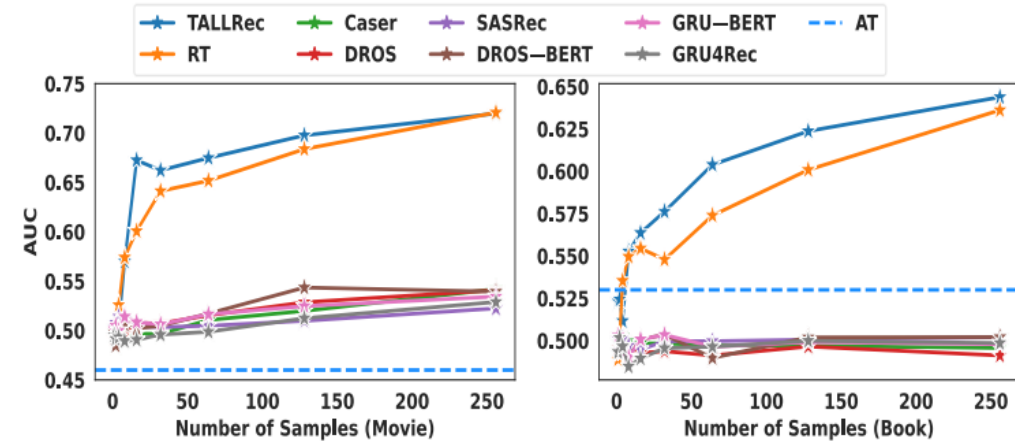
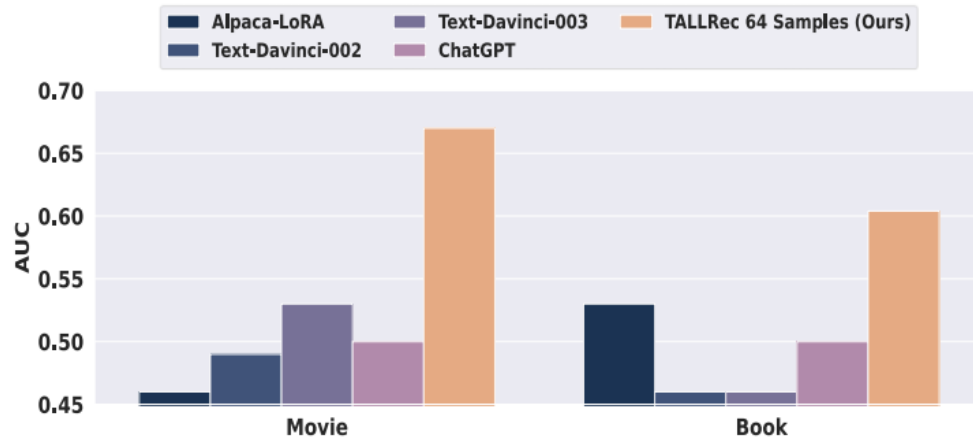
- **Efficiency:**

- Only tune 0.1% parameters via LoRA
- Trainable on single RTX 3090 (24GB)

Table 3: Performance comparison between conventional sequential recommendation baselines and TALLRec under different few-shot training settings. The reported result is the AUC multiplied by 100, with boldface indicating the highest score. ‡: significantly better than all baselines with t-test $p < 0.01$.

	Few-shot	GRU4Rec	Caser	SASRec	DROS	GRU-BERT	DROS-BERT	TALLRec
movie	16	49.07	49.68	50.43	50.76	50.85	50.21	67.24 ‡
	64	49.87	51.06	50.48	51.54	51.65	51.71	67.48 ‡
	256	52.89	54.20	52.25	54.07	53.44	53.94	71.98 ‡
book	16	48.95	49.84	49.48	49.28	50.07	50.07	56.36
	64	49.64	49.72	50.06	49.13	49.64	48.98	60.39 ‡
	256	49.86	49.57	50.20	49.13	49.79	50.20	64.38 ‡

- **Effectiveness (64-shot setting):**
 - TALLRec: AUC = 0.67 (Movie), 0.60 (Book)
 - Traditional models (SASRec, GRU4Rec): AUC $\approx$ 0.50
 - In-context Learning (ChatGPT): AUC $\approx$ 0.50 (random)



- **Effectiveness (64-shot setting):**

- TALLRec: AUC = 0.67 (Movie), 0.60 (Book)
- Traditional models (SASRec, GRU4Rec): AUC $\approx$ 0.50
- In-context Learning (ChatGPT): AUC $\approx$ 0.50 (random)

- **Few-shot Capability:**

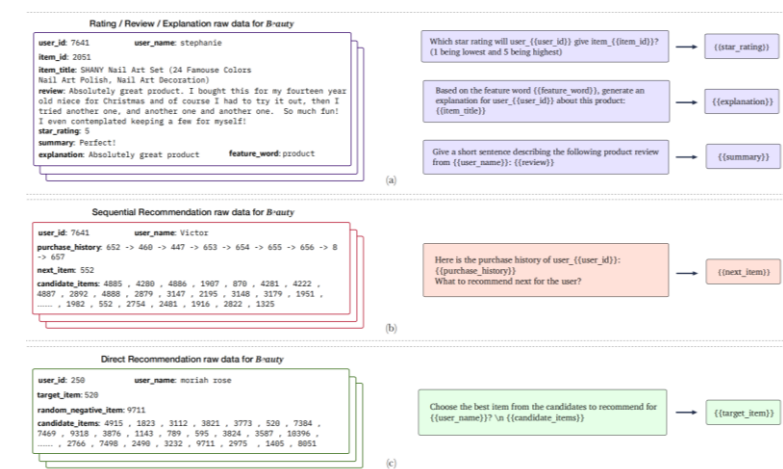
- Strong performance with only 64 training samples & Outperforms sparse ID-based models

- **Cross-domain Generalization:**

- Movie $\rightarrow$ Book transfer shows strong generalization
- Text-based ID enables semantic knowledge transfer across domains

- **Limitations:**

- Task-Specific Silos: Traditional Sparse ID models require distinct, customized architectures for different tasks (e.g., rating vs. sequence)
- Semantic Deficiency: Sparse IDs are random integers, lacking inherent semantic meaning and world knowledge
- Cold-Start & Transferability: Sparse ID embeddings fail on long-tail items and cannot transfer knowledge across domains



- **Core Question:**

- Replacing random IDs with the rich semantics of natural language?

- **Text-based Representation:**
 - Replace Sparse IDs with textual attributes (e.g., "Title: iPhone", "Brand: Apple")
- **Personalized Prompts:**
 - Formulate diverse tasks into unified, instruction-based natural language templates
- **Shared Architecture:**
 - Use a single Encoder-Decoder (T5) backbone for all tasks, eliminating Sparse ID embedding tables
- **Unified Objective:**
 - Optimize all tasks with a single Language Modeling (text-to-text) loss

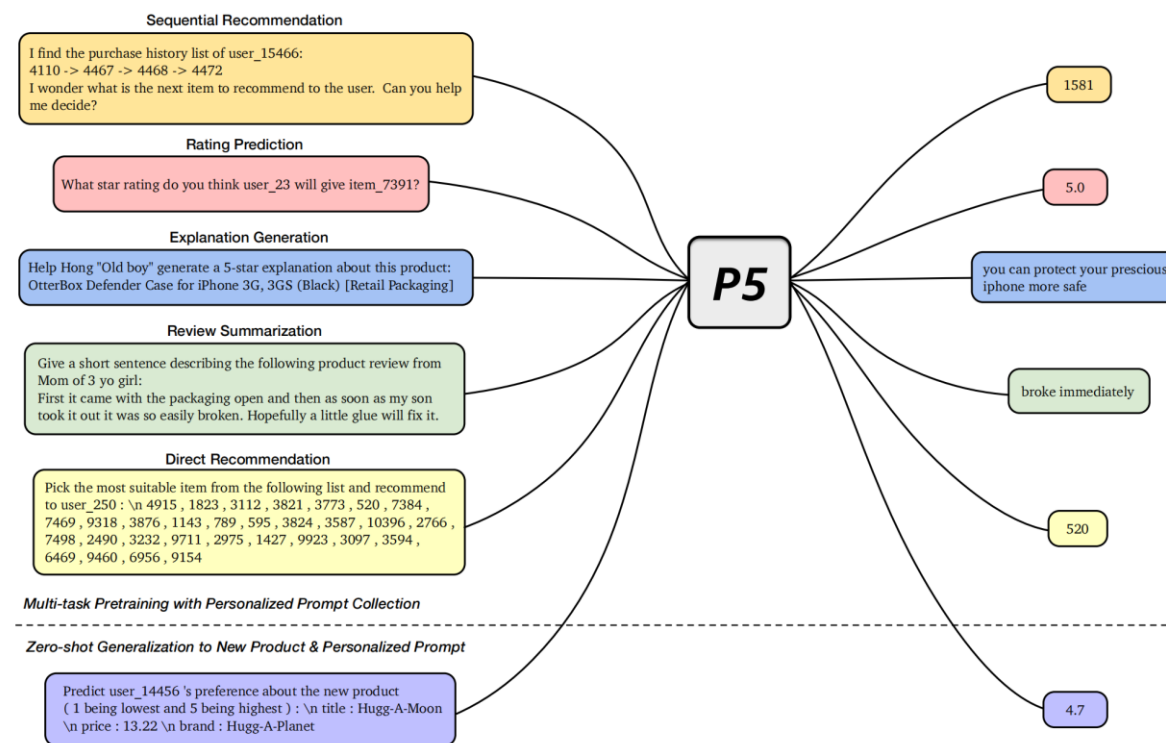


Table 3: Performance comparison on sequential recommendation.

Methods	Sports				Beauty				Toys			
	HR@5	NDCG@5	HR@10	NDCG@10	HR@5	NDCG@5	HR@10	NDCG@10	HR@5	NDCG@5	HR@10	NDCG@10
Caser	0.0116	0.0072	0.0194	0.0097	0.0205	0.0131	0.0347	0.0176	0.0166	0.0107	0.0270	0.0141
HGN	0.0189	0.0120	0.0313	0.0159	0.0325	0.0206	0.0512	0.0266	0.0321	0.0221	0.0497	0.0277
GRU4Rec	0.0129	0.0086	0.0204	0.0110	0.0164	0.0099	0.0283	0.0137	0.0097	0.0059	0.0176	0.0084
BERT4Rec	0.0115	0.0075	0.0191	0.0099	0.0203	0.0124	0.0347	0.0170	0.0116	0.0071	0.0203	0.0099
FDSA	0.0182	0.0122	0.0288	0.0156	0.0267	0.0163	0.0407	0.0208	0.0228	0.0140	0.0381	0.0189
SASRec	0.0233	0.0154	0.0350	0.0192	0.0387	0.0249	0.0605	0.0318	0.0463	0.0306	0.0675	0.0374
S ³ -Rec	0.0251	0.0161	0.0385	0.0204	0.0387	0.0244	0.0647	0.0327	0.0443	0.0294	0.0700	0.0376
P5-S (2-3)	0.0272	0.0169	0.0361	0.0198	<u>0.0503</u>	<u>0.0370</u>	<u>0.0659</u>	<u>0.0421</u>	0.0648	0.0567	0.0709	0.0587
P5-B (2-3)	<u>0.0364</u>	<u>0.0296</u>	<u>0.0431</u>	<u>0.0318</u>	0.0508	0.0379	0.0664	0.0429	0.0608	0.0507	0.0688	0.0534
P5-S (2-13)	0.0258	0.0159	0.0346	0.0188	0.0490	0.0358	0.0646	0.0409	<u>0.0647</u>	<u>0.0566</u>	<u>0.0705</u>	<u>0.0585</u>
P5-B (2-13)	0.0387	0.0312	0.0460	0.0336	0.0493	0.0367	0.0645	0.0416	0.0587	0.0486	0.0675	0.0536

- Unified Task Performance: P5 matches or exceeds task-specific Sparse ID baselines across 5 task families.

<u>Toys -> Beauty</u>			
Case 1	Input: Based on the word color , help user_4549 write a 5-star explanation for this new product : \n title : Bain De Terre Serum Anti-Frizz Recovery Complex 1.7 oz. \n price : 3.86 \n brand : Bain de Terre	Target Output: I have color treated fine hair but lots of it	P5's Output: The colors are bright
Case 2	Input: Based on the word shampoo , help user_1491 write a 1-star explanation for this new product : \n title : CLEAR SCALP & HAIR BEAUTY Volumizing Root Boost Nourishing Shampoo, 12.9 Fluid Ounce \n price : 1.66 \n brand : Clear	Target Output: I tried the shampoo and it hardly foams up	P5's Output: Not worth the price
<u>Beauty -> Sports</u>			
Case 3	Input: Based on the word tool , help user_3877 write a 5-star explanation for this new product : \n title : SOG Specialty Knives & Tools TF6-CP Trident Knife with Straight Edge Assisted Folding 3.75-Inch Steel Tanto Shape Blade and GRN Handle, Satin Finish \n price : 57.6 \n brand : SOG Specialty Knives	Target Output: this is an excellent tool	P5's Output: This is a great tool tool
Case 4	Input: Based on the word saw , help user_326 write a 2-star explanation for this new product : \n title : Rome's #1705 Square Pie Iron with Steel and Wood Handles \n price : 20.26 \n brand : Rome Industries	Target Output: After looking closely we saw 2 of the rivots that are supposed to hold it on were missing	P5's Output: I saw no visible improvement
<u>Sports -> Toys</u>			
Case 5	Input: Based on the word game , help user_29782 write a 3-star explanation for this new product : \n title : Deal Breaker Coaster Card Game \n price : 13.95 \n brand : Zobmondo!!	Target Output: I think this game could prove interesting if played while drinking	P5's Output: I'm a big fan of this game and it is a great game
Case 6	Input: Based on the word cars , help user_2121 write a 5-star explanation for this new product : \n title : Ridemakerz Marvel Universe Spiderman and Venom Xtreme Customz Swap Set \n price : 14.89 \n brand : RIDEMAKERZ	Target Output: These are great looking little plastic cars	P5's Output: I have several cars and this one is the best

Directions	Z-1 & Z-4	Z-2 & Z-3	Z-5 & Z-7		Z-6	
	Accuracy	MAE	BLUE4	ROUGE1	BLUE4	ROUGE1
Toys -> Beauty	0.7922	0.8244	1.8869	61.1919	5.4609	66.4931
Toys -> Sports	0.8682	0.6644	0.7405	60.9575	2.2601	62.0353
Beauty -> Toys	0.8073	0.7792	0.0929	41.3061	11.8046	64.8701
Beauty -> Sports	0.8676	0.6838	0.0346	39.7191	6.6409	66.9222
Sports -> Toys	0.8230	0.7443	0.0687	42.9310	13.3408	69.7910
Sports -> Beauty	0.8057	0.8102	0.0790	41.0659	13.1690	66.7687

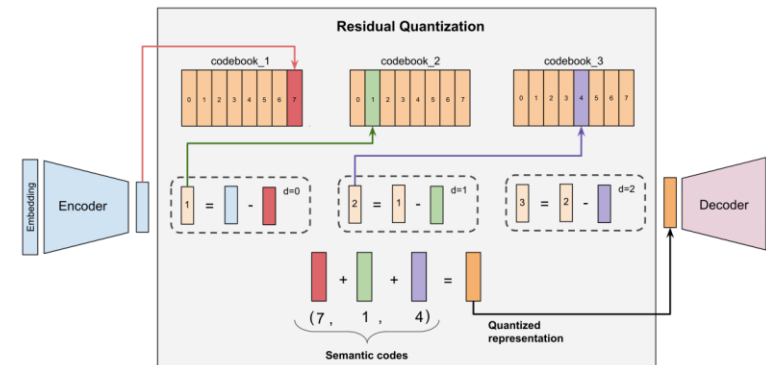
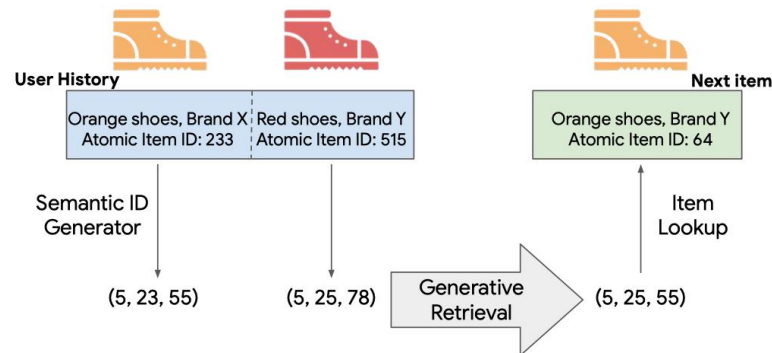
- Unified Task Performance: P5 matches or exceeds task-specific Sparse ID baselines across 5 task families.
- Cross-Domain Transfer: Successfully recommends and explains items in unseen domains (e.g., Toys to Beauty).
- Semantic Advantage: Text IDs enable LLMs to utilize world knowledge for cold-start and cross-domain scenarios (Impossible for Sparse IDs).

- **Background**

- Atomic item IDs share no semantics → poor transfer to long-tail / unseen items
- Embedding tables & ANN indices scale linearly with catalog size
- Generative rec needs compact, unique, structured item identifiers

- **Motivation**

- **First** SID-based generative recommender: next-item = SID sequence generation
- Sentence-T5 + RQ-VAE → coarse-to-fine Semantic ID
- Shows SID potential for cold-start expansion and diversity control



- **Background:** Item tokenization is necessary for LLM-based generative recommendation.
 - There is a gap between the language space of LLMs and recommendation data.
 - Varying length of item content make it hard to generate the exact item.

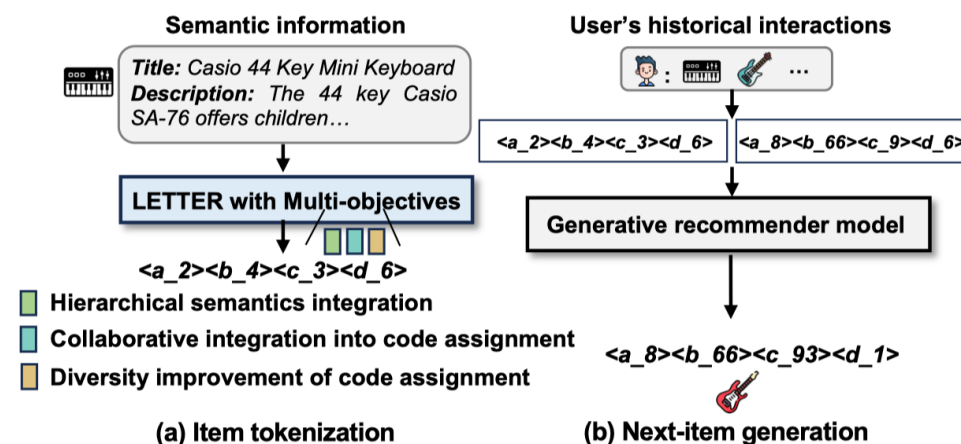


Figure 1: Overview of utilizing LETTER for LLM-based generative recommendation.

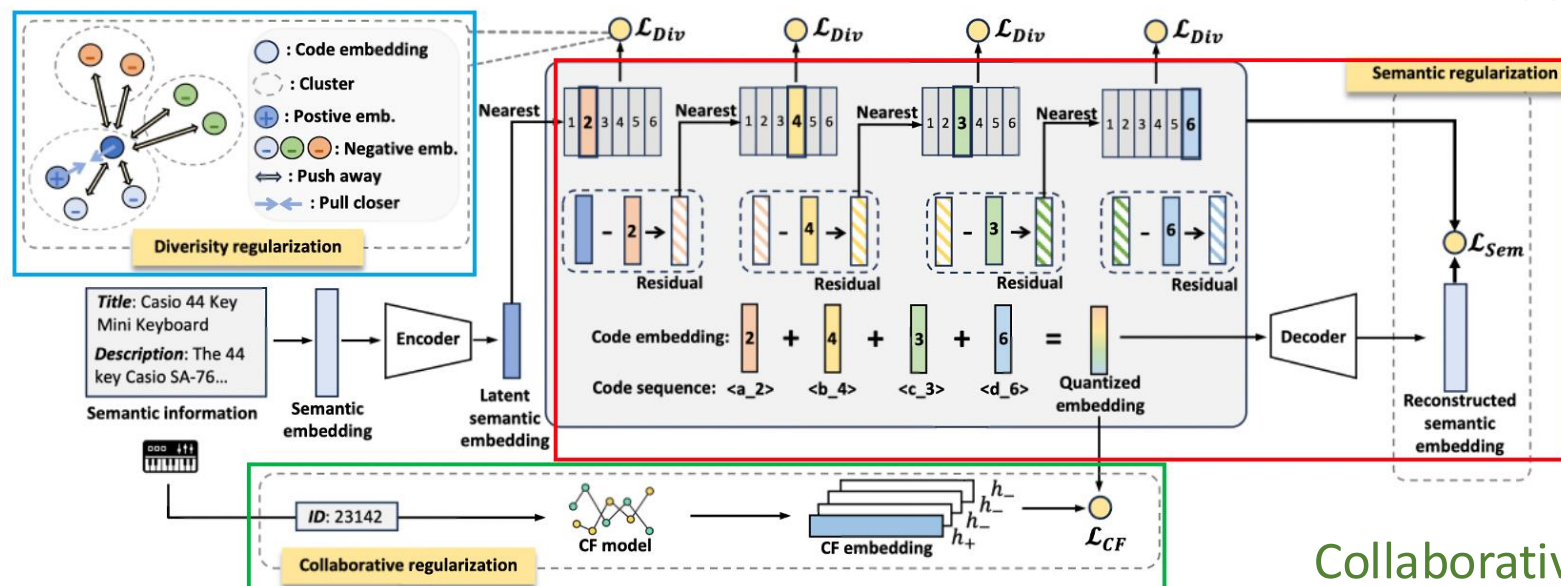
- **Motivation:**
 - **ID identifier:** assign each item with a **unique numerical string**
 - Inefficiency in encoding semantic information
 - **Textual identifier** : directly leverage item semantic information such as titles, attributes as item identifiers.
 - Non-hierarchical semantics
 - Lack of collaborative signals
 - **SID-based Identifier:** adopt auto-encoders to encode item semantics into hierarchical code sequences as identifiers
 - Lack of collaborative signals
 - Code assignment bias

Objective:

- **Semantic integration:** Integrating hierarchical semantic with the token sequence
- **Collaborative signals:** Incorporating collaborative signals into the token assignment
- **Diversity:** Mitigating code assignment bias brought by token assignment.

Diversity regularization

Semantic regularization

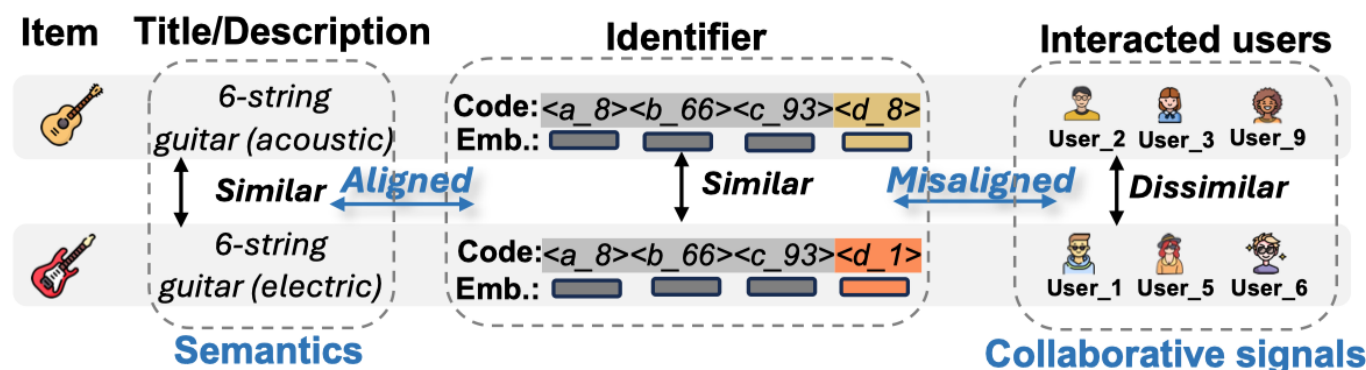


Collaborative regularization

• Collaborative Regularization

- **Motivation:** Alleviating the misalignment between item identifiers and collaborative signals
- **Idea:** Using embeddings from a well-trained CF Model (e.g. SASRec) to align with the quantized embedding in RQ-VAE

- **Loss form**
$$\mathcal{L}_{CF} = -\frac{1}{B} \sum_{i=1}^B \frac{\exp(\langle \hat{\mathbf{z}}_i, \mathbf{h}_i \rangle)}{\sum_{j=1}^B \exp(\langle \hat{\mathbf{z}}_i, \mathbf{h}_j \rangle)},$$

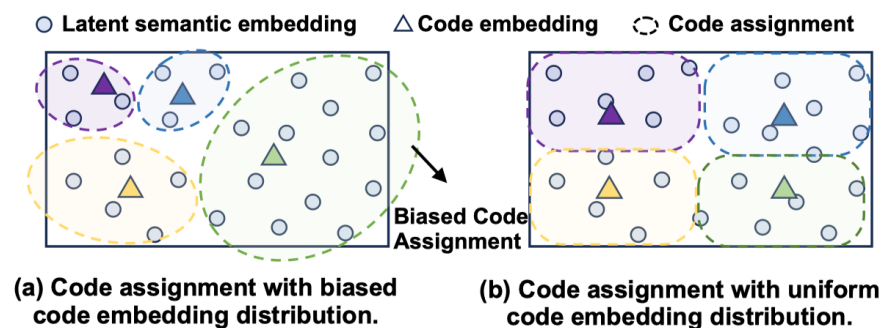


Misalignment between item identifiers and collaborative signals. “Emb.” denotes “Embeddings”.

• Diversity Regularization

- **Motivation:** Mitigating code assignments bias brought by a biased code embedding distribution
- **Idea:** Evenly distribute code embeddings across their representation space in each layer
- **Loss form**

$$\mathcal{L}_{\text{Div}} = -\frac{1}{B} \sum_{i=1}^B \frac{\exp(\langle \mathbf{e}_{c_l}^i, \mathbf{e}_+ \rangle)}{\sum_{j=1}^{N-1} \exp(\langle \mathbf{e}_{c_l}^i, \mathbf{e}_j \rangle)}, \quad (4)$$



Comparison between Code assignment with biased distribution and uniform distribution of code embeddings.

• Motivation:

- Retrieval / ranking / re-ranking are fragmented; communication & storage dominate compute
- Hundreds of objectives optimized independently → cross-stage conflicts
- Cascaded systems cannot fully exploit scaling laws, MoE, or RL

• Contribution:

- **First** Industrial SID-based Method.
- Deployable enc-dec generative recommender that unifies retrieval and ranking
- Collaborative-aware RQ-KMeans tokenizer with multimodal + collaborative signals

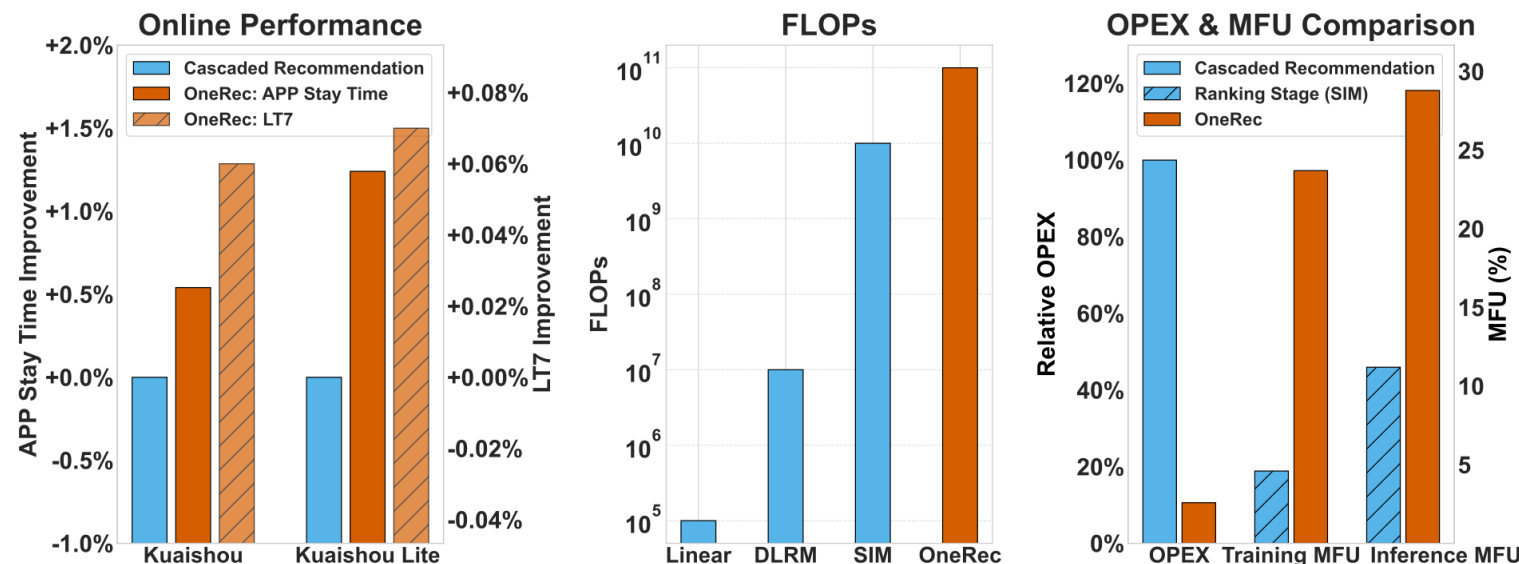


Figure 1 | Online performance, FLOPs, OPEX, and MFU comparison.

- Tokenizer:

- miniCPM-V-8B $\rightarrow$ QFormer compress $\rightarrow$ collaborative alignment $\rightarrow$ 3-level RQ-KMeans

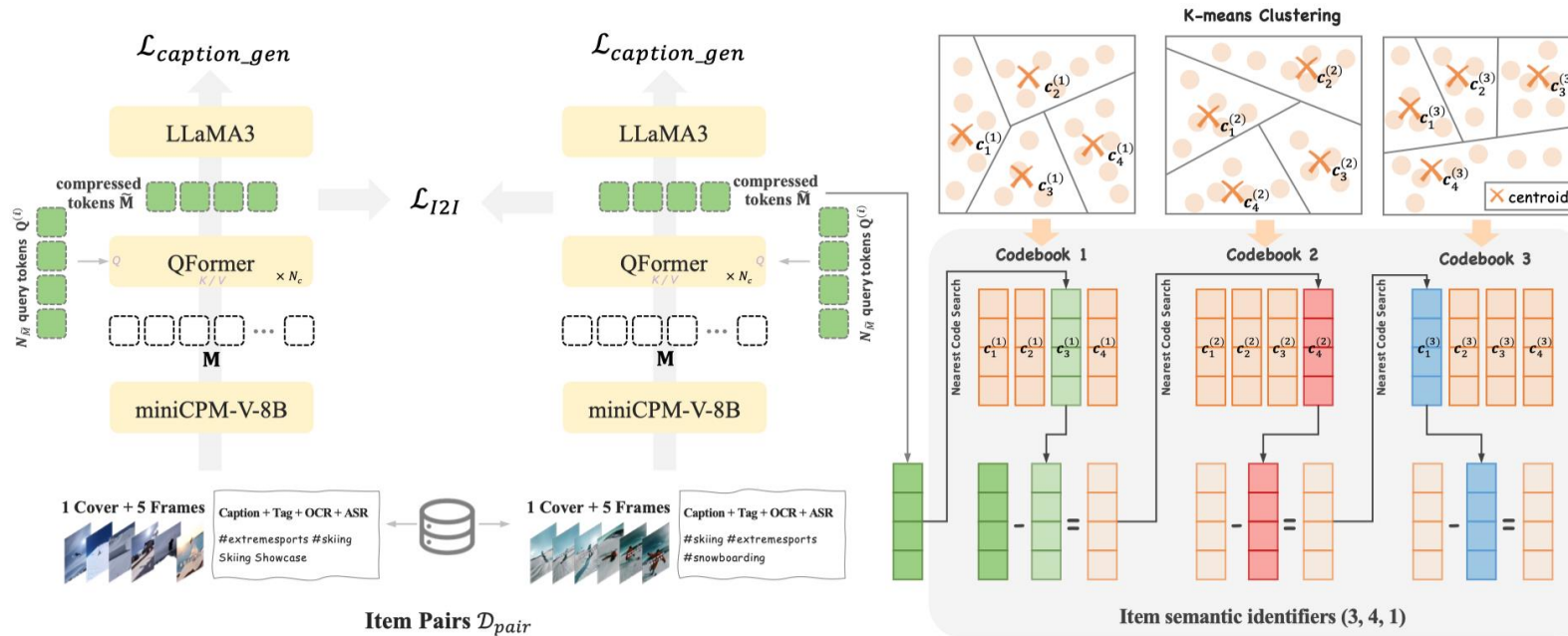


Figure 3 | Illustration of our tokenizer implementation. We first align multimodal representations of item pairs with high collaborative similarity to obtain collaborative multimodal representations, then tokenize these representations into discrete semantic IDs using RQ-Kmeans.

- Full Architecture:
 - Multi-path encoder (static / short-term / positive / lifelong) + MoE decoder $\rightarrow$ 3-token SID

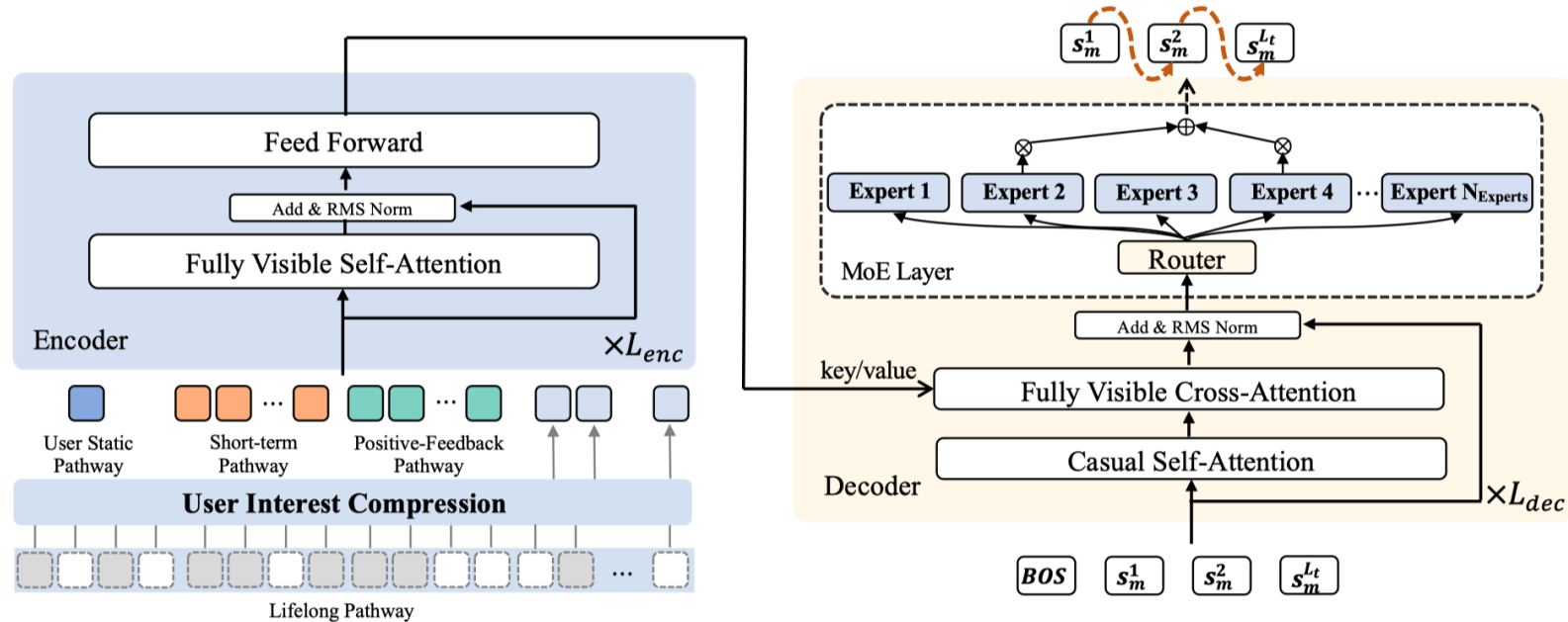


Figure 4 | Illustration of our encoder-decoder architecture.

Motivation:

- Short SIDs (4 tokens) have limited capacity; extending to 32–64 tokens makes AR decoding expensive
- Independent parallel prediction can produce invalid SID combinations
- Exhaustively scoring all items reintroduces catalog-scale online compute

Contribution:

- **First** OPQ-based SIDs up to 64 tokens with no cross-position AR dependency
- Multi-Token Prediction (MTP): all SID positions in one backbone forward pass

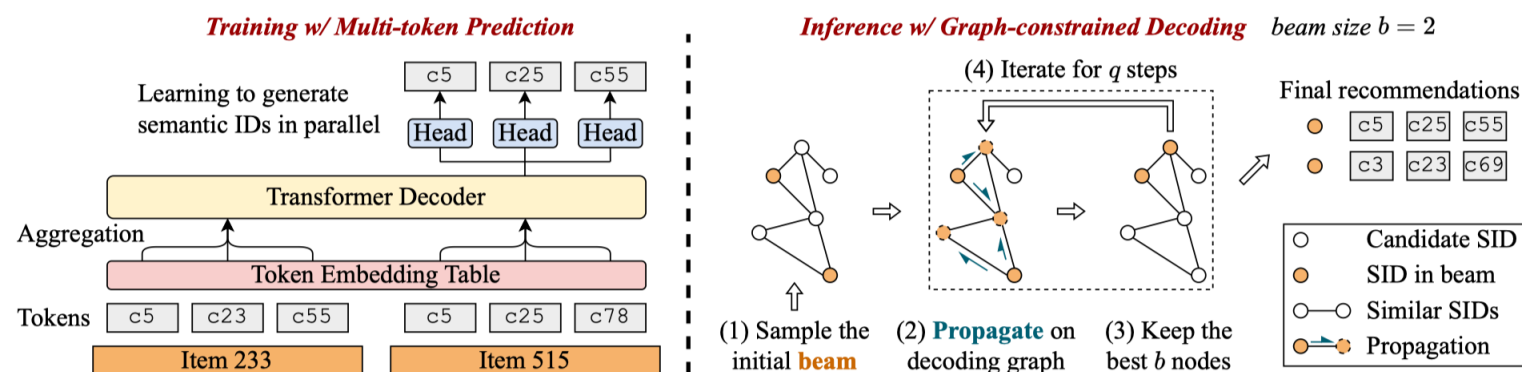


Figure 1: The overall framework of RPG.

ETEGRec

• Motivation:

- Two-stage methods (TIGER/LETTER): tokenizer frozen after pre-train → reconstruction $\neq$ recommendation objective
- Naively updating both components → unstable SID and objective drift

• Contribution:

- **First** joint training: RQ-VAE tokenizer + generative recommender co-adapt
- Alternating optimization stabilizes tokenizer–recommender co-evolution

Table 1: Comparison of ETEGRec with several related studies on item Tokenization and generative recommendation. “EL” means the length of item identifiers are equal. “IA” denotes interaction-aware. “TI” denotes tokenization integration.

Methods	Item Tokenization			Generative Recommendation	
	Learning	EL	IA	Token Sequence	TI
GPTRec [16]	Heuristic	✓	✓	Pre-processed	✗
CID [9]	Heuristic	✗	✗	Pre-processed	✗
TIGER [19]	Pre-learned	✓	✗	Pre-processed	✗
LETTER [30]	Pre-learned	✓	✗	Pre-processed	✗
ETEGRec	End-to-end	✓	✓	Gradually Refined	✓

SID Challenge - Interpretability

OneRec-Think

• Motivation:

- Existing generative recommenders generate SIDs directly — preference modeling is implicit & uncontrollable
- Itemic tokens are not aligned with language space → general LLM cannot reason over SIDs
- Full CoT reasoning is too slow for real-time serving

• Contribution:

- First** work to Unified autoregressive process: textual rationale + itemic-token recommendation
- Target-aware reasoning scaffolding: rationale from relevant history, SFT on full noisy history



Figure 1: Examples of OneRec-Think's Unified Dialogue, Reasoning and Recommendation Framework.

OneRec-Think

Method:

- Itemic Alignment: Token warm-up $\rightarrow$ multi-task: persona grounding, sequential preference, dense captioning, general LM

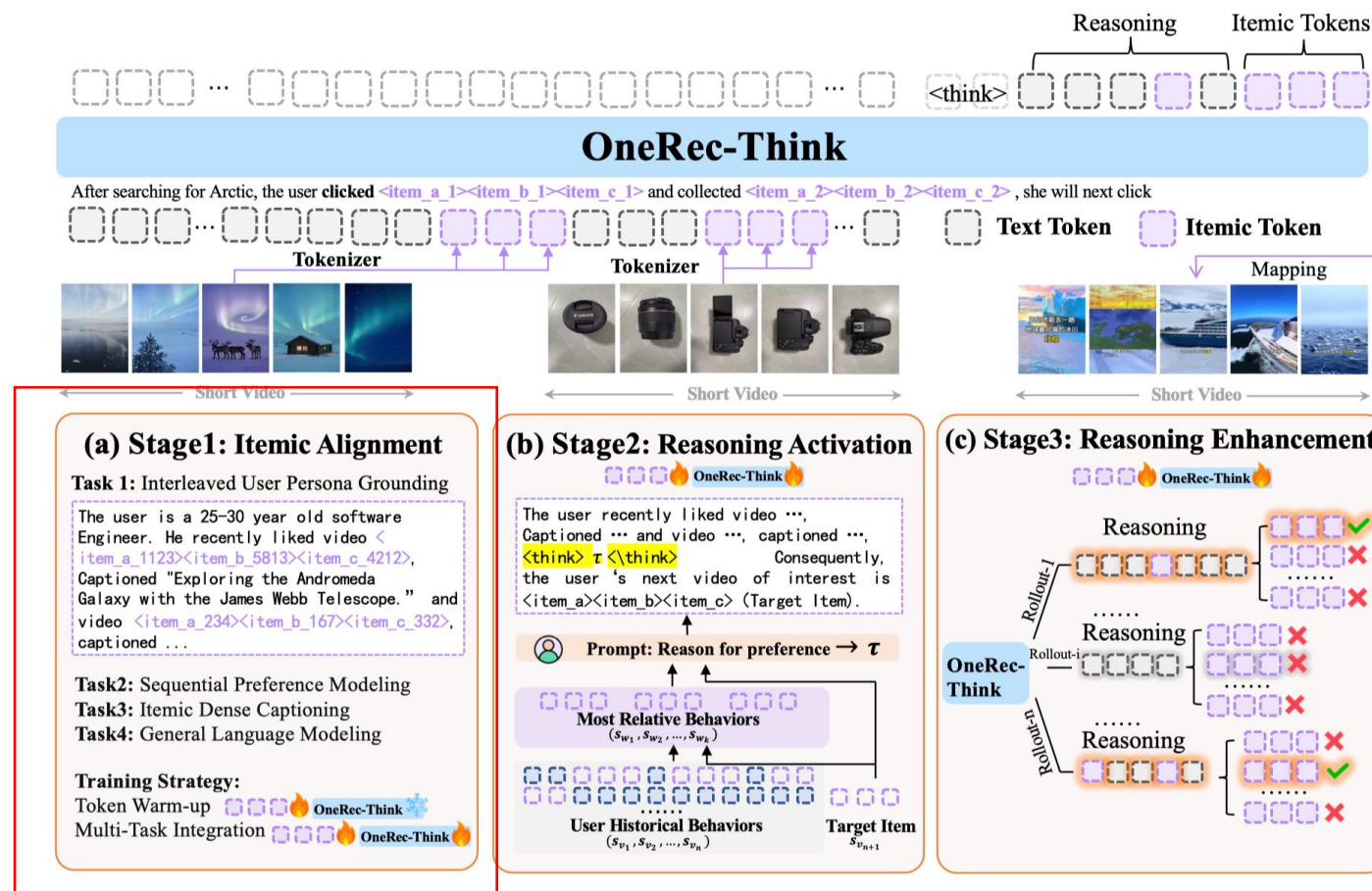


Figure 2: The framework of the OneRec-Think. In the first stage, we achieve item-level semantic alignment through multi-task pre-training. In the second stage, we activate explicit reasoning by prompting the model to generate preference rationales. In the third stage, we refine the reasoning paths through RL based on a reward tailored for recommendations.

OneRec-Think

Method:

- Itemic Alignment: Token warm-up $\rightarrow$ multi-task: persona grounding, sequential preference, dense captioning, general LM
- Reasoning SFT: Generate rationales from top-10 target-relevant behaviors; distill to training on full noisy history

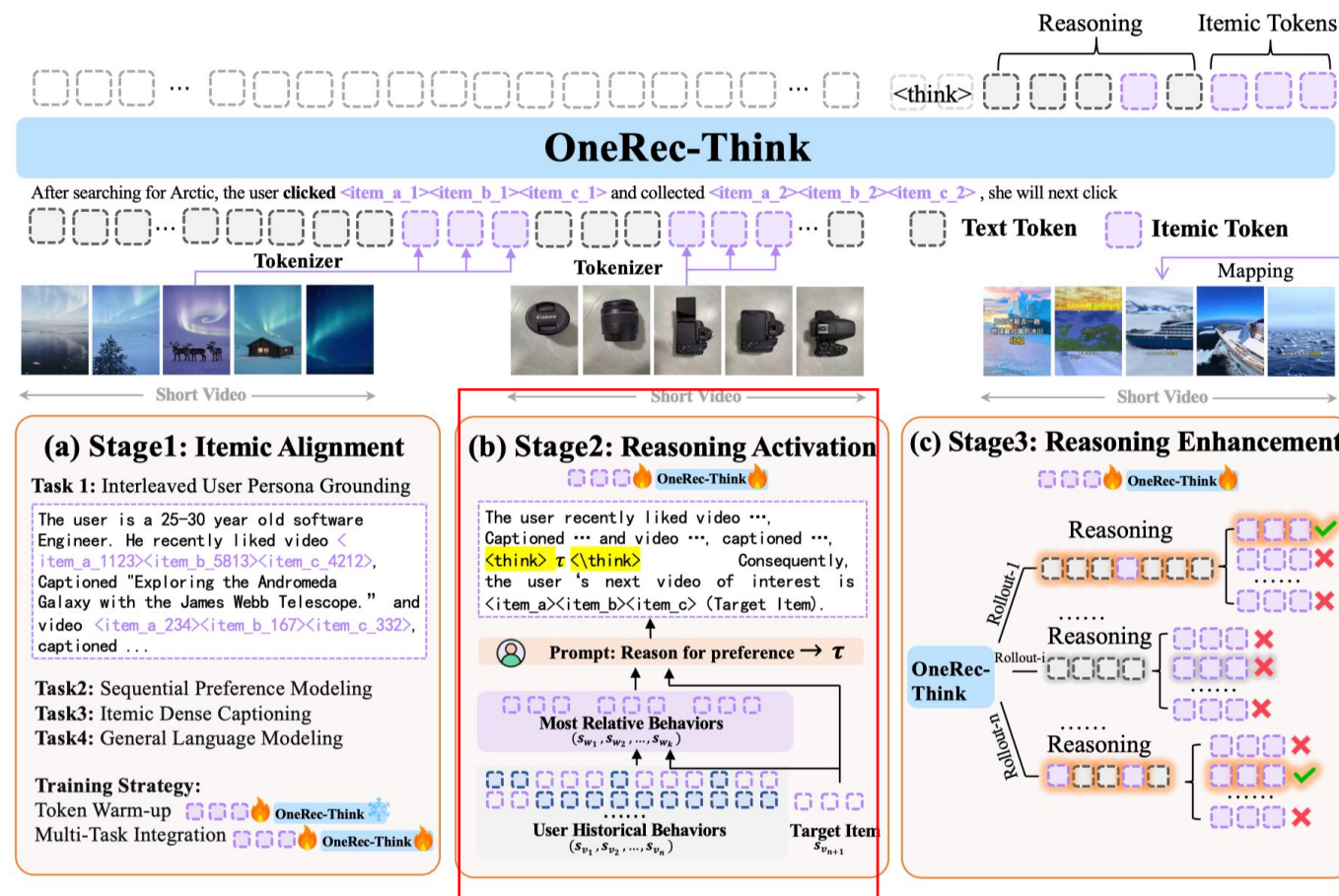


Figure 2: The framework of the OneRec-Think. In the first stage, we achieve item-level semantic alignment through multi-task pre-training. In the second stage, we activate explicit reasoning by prompting the model to generate preference rationales. In the third stage, we refine the reasoning paths through RL based on a reward tailored for recommendations.

OneRec-Think

Method:

- Itemic Alignment: Token warm-up $\rightarrow$ multi-task: persona grounding, sequential preference, dense captioning, general LM
- Reasoning SFT: Generate rationales from top-10 target-relevant behaviors; distill to training on full noisy history
- GRPO: Each reasoning rollout $\rightarrow$ beam search; best SID-token match count = reward

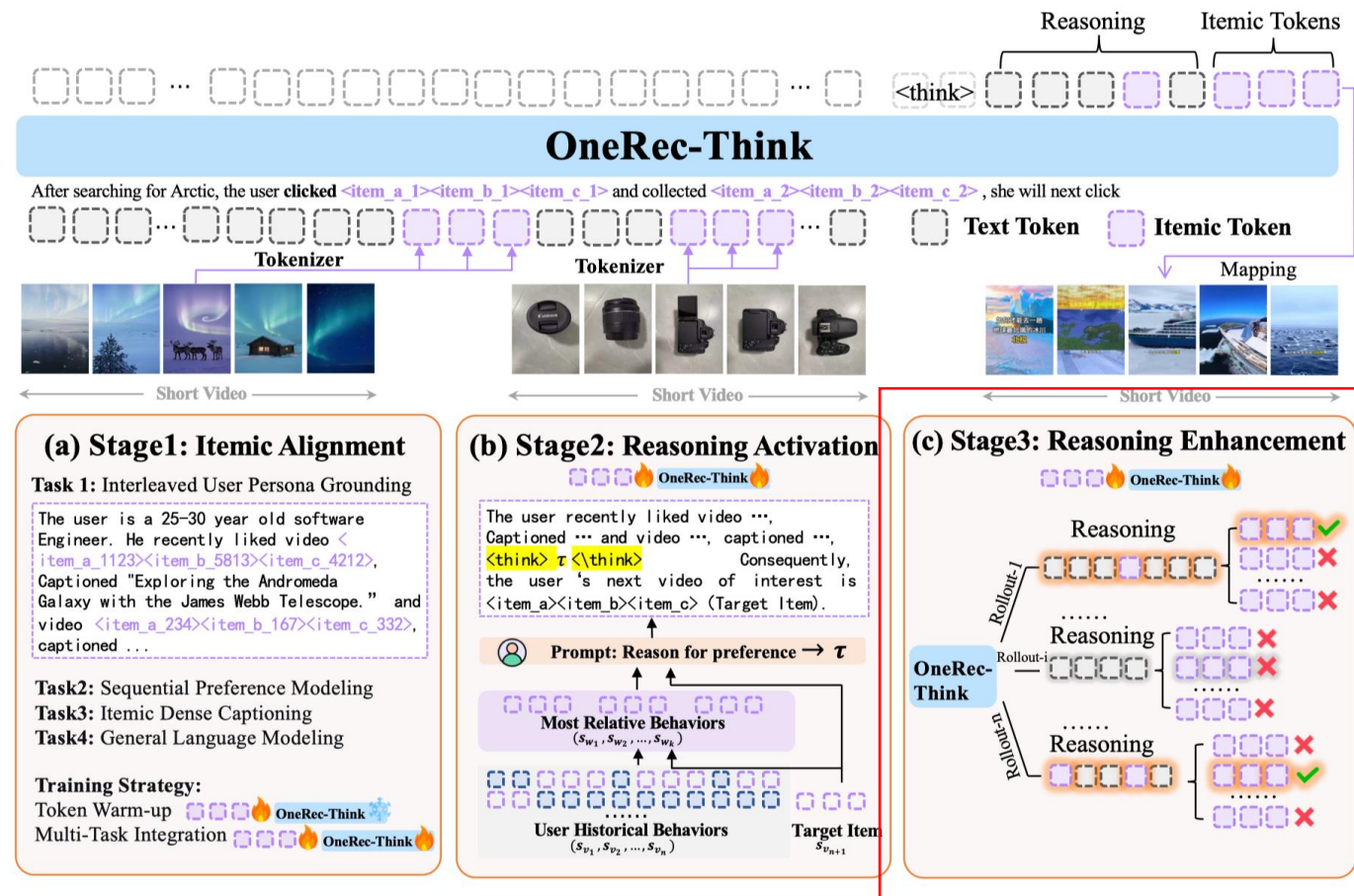


Figure 2: The framework of the OneRec-Think. In the first stage, we achieve item-level semantic alignment through multi-task pre-training. In the second stage, we activate explicit reasoning by prompting the model to generate preference rationales. In the third stage, we refine the reasoning paths through RL based on a reward tailored for recommendations.

QARM

• Motivation:

- **Representation Unmatching:** Pretrained MLLMs (CLIP, BLIP2, etc.) are trained on NLP/CV self-supervised objectives, misaligned with recommendation-specific user behavior signals
- **Representation Unlearning:** Multimodal features are stored as fixed cached inputs to the recommendation model, blocking gradient updates and end-to-end learning

• Contribution:

- **Item Alignment:** Fine-tunes the MLLM via contrastive learning on business-derived item-item pairs to align representations with downstream recommendation distributions
- **Quantitative Code (VQ + RQ):** Discretizes continuous multimodal representations into learnable Semantic IDs, enabling end-to-end training within the recommendation model

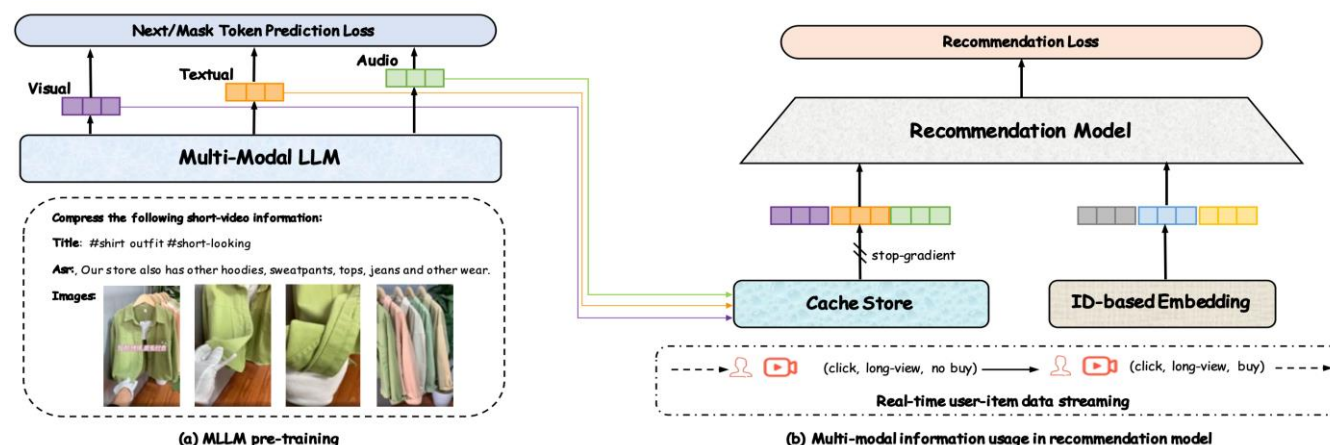


Figure 1: (a) Representation Unmatching: multi-modal features are obtained through down-streaming task-agnostic self-supervised tasks like image-text matching. (b) Representation Unlearning: generated multi-modal representation is always only served as additional fixed static input feature for recommendation model.

QARM

• Method:

- Item Alignment: Mines item-item pairs from U2I/I2I retrieval models; fine-tunes MLLM with batch-contrastive loss using multimodal (text + image + audio) inputs only

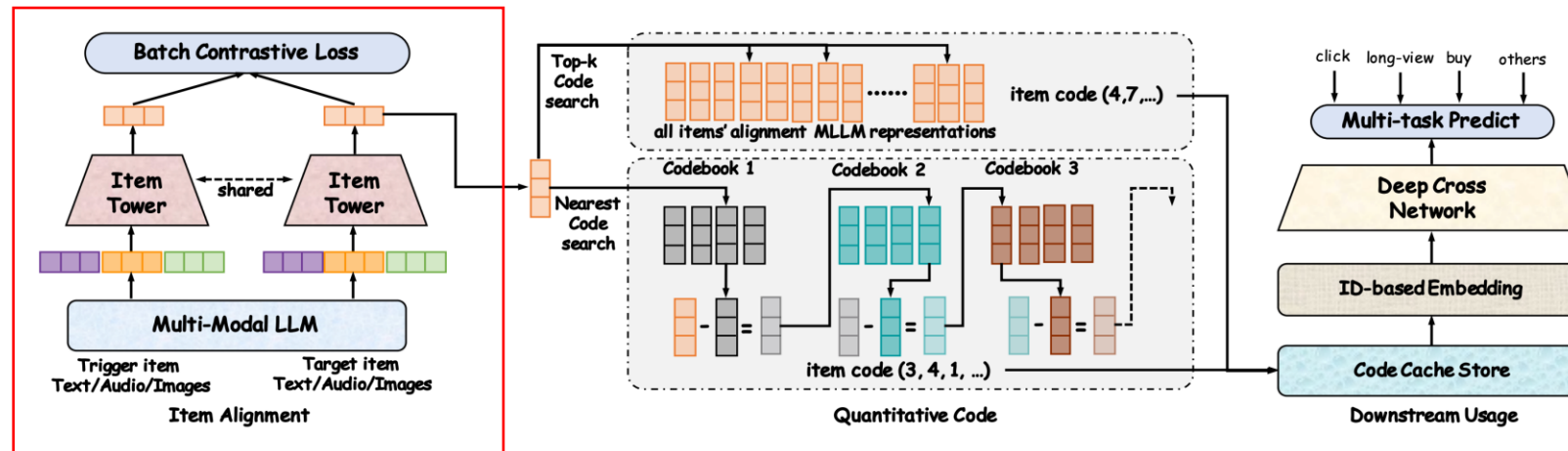


Figure 3: Sketches of QARM workflow, the left sub-figure gives the training framework of our item alignment mechanism, the middle sub-figure expresses our quantitative mechanism, and the right sub-figure indicates the learnable ID-based training.

QARM

• Method:

- Item Alignment: Mines item-item pairs from U2I/I2I retrieval models; fine-tunes MLLM with batch-contrastive loss using multimodal (text + image + audio) inputs only
- VQ Code: Uses all item representations as a codebook; assigns each item its Top-K (K=25) nearest neighbors as discrete VQ codes

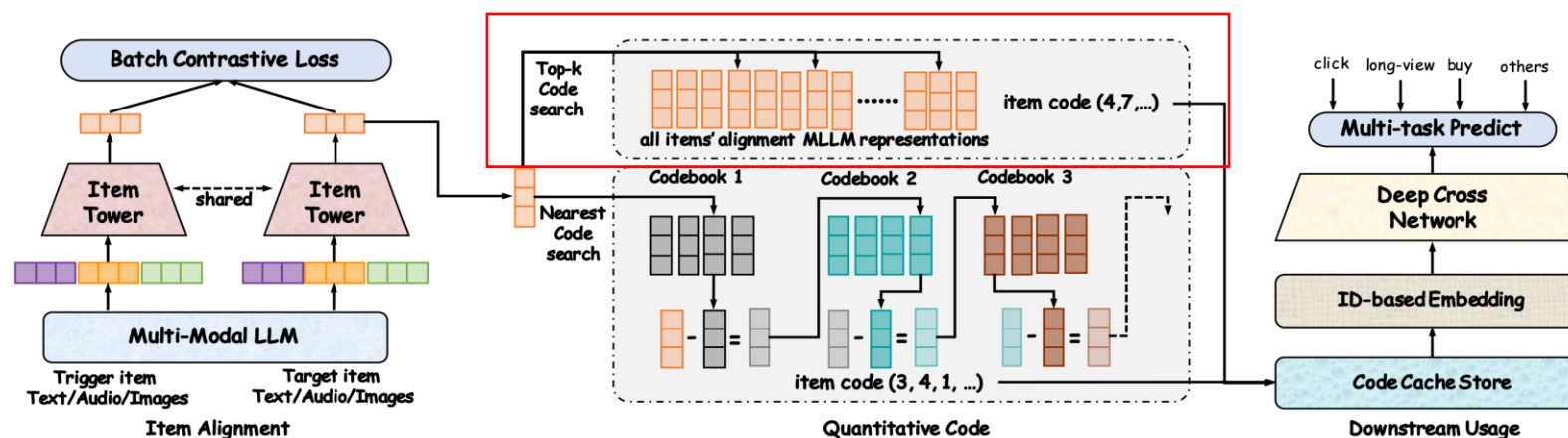


Figure 3: Sketches of QARM workflow, the left sub-figure gives the training framework of our item alignment mechanism, the middle sub-figure expresses our quantitative mechanism, and the right sub-figure indicates the learnable ID-based training.

QARM

• Method:

- Item Alignment: Mines item-item pairs from U2I/I2I retrieval models; fine-tunes MLLM with batch-contrastive loss using multimodal (text + image + audio) inputs only
- VQ Code: Uses all item representations as a codebook; assigns each item its Top-K (K=25) nearest neighbors as discrete VQ codes
- RQ Code: L-level (L=6) residual K-means — each level quantizes the residual from the previous level, yielding hierarchical Semantic IDs

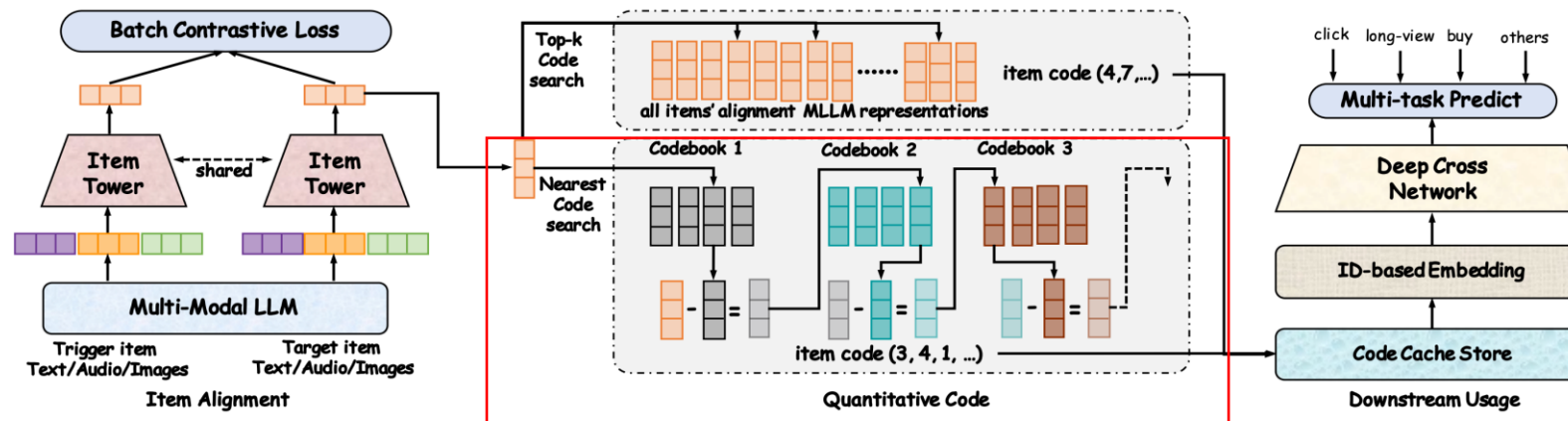


Figure 3: Sketches of QARM workflow, the left sub-figure gives the training framework of our item alignment mechanism, the middle sub-figure expresses our quantitative mechanism, and the right sub-figure indicates the learnable ID-based training.

QARM

• Method:

- Item Alignment: Mines item-item pairs from U2I/I2I retrieval models; fine-tunes MLLM with batch-contrastive loss using multimodal (text + image + audio) inputs only
- VQ Code: Uses all item representations as a codebook; assigns each item its Top-K (K=25) nearest neighbors as discrete VQ codes
- RQ Code: L-level (L=6) residual K-means — each level quantizes the residual from the previous level, yielding hierarchical Semantic IDs

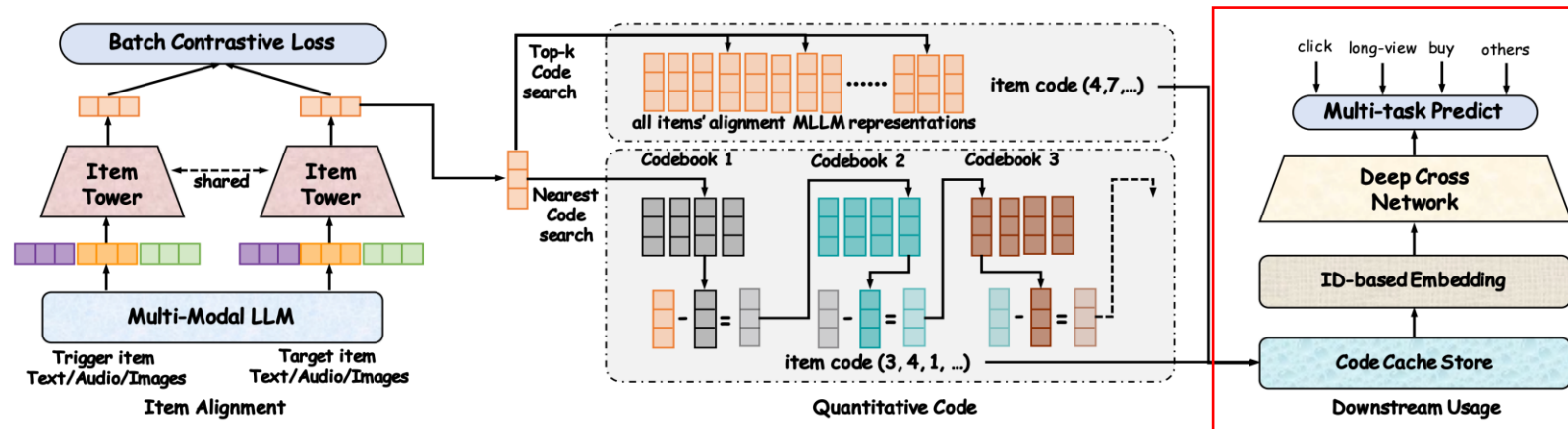


Figure 3: Sketches of QARM workflow, the left sub-figure gives the training framework of our item alignment mechanism, the middle sub-figure expresses our quantitative mechanism, and the right sub-figure indicates the learnable ID-based training.

➤ Summary:

- **Sparse ID** lacks semantics and generalizability, but offers accurate grounding
- **Text-Based** is semantically rich and generalizable, but grounding is difficult
- **SID-Based** is the most promising direction, balancing compact vocabulary, semantics, and grounding

➤ Future:

- **Adaptive tokenization:** Dynamically adjusts to handle collision rate changes, new item influx, and cross-scenario generalization
- **Tokenizer evaluation framework:** Current metrics (collision rate, information entropy, codebook utilization) are all surface-level proxies that cannot directly guide the design of tokenizers truly optimized for downstream recommendation performance

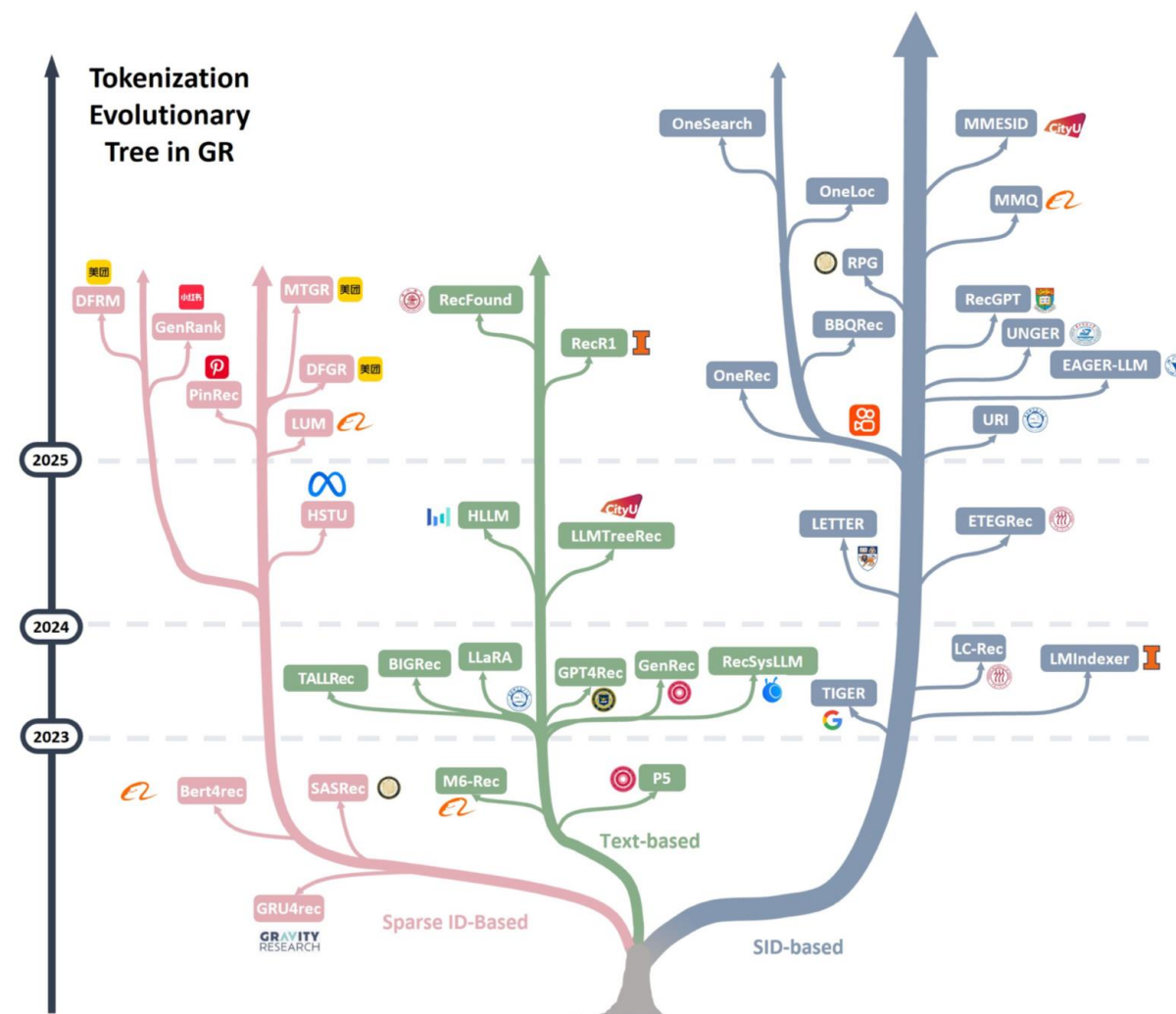


Figure 3. The evolution of tokenizer paradigms, showing the transition from sparse ID-based and text-based models toward semantic ID approaches, with affiliations noted beside each work.

1 Introduction (9:00 - 9:15)



Xiaopeng Li



2 Tokenization (9:15 - 9:45)



Xiaopeng Li

3 Model Structure (9:45 - 10:30)



Ziwei Liu



15mins

4 Optimization (10:45 - 11:15)



Wenlin Zhang Kaifeng Guo



5 Application (11:15 - 11:30)



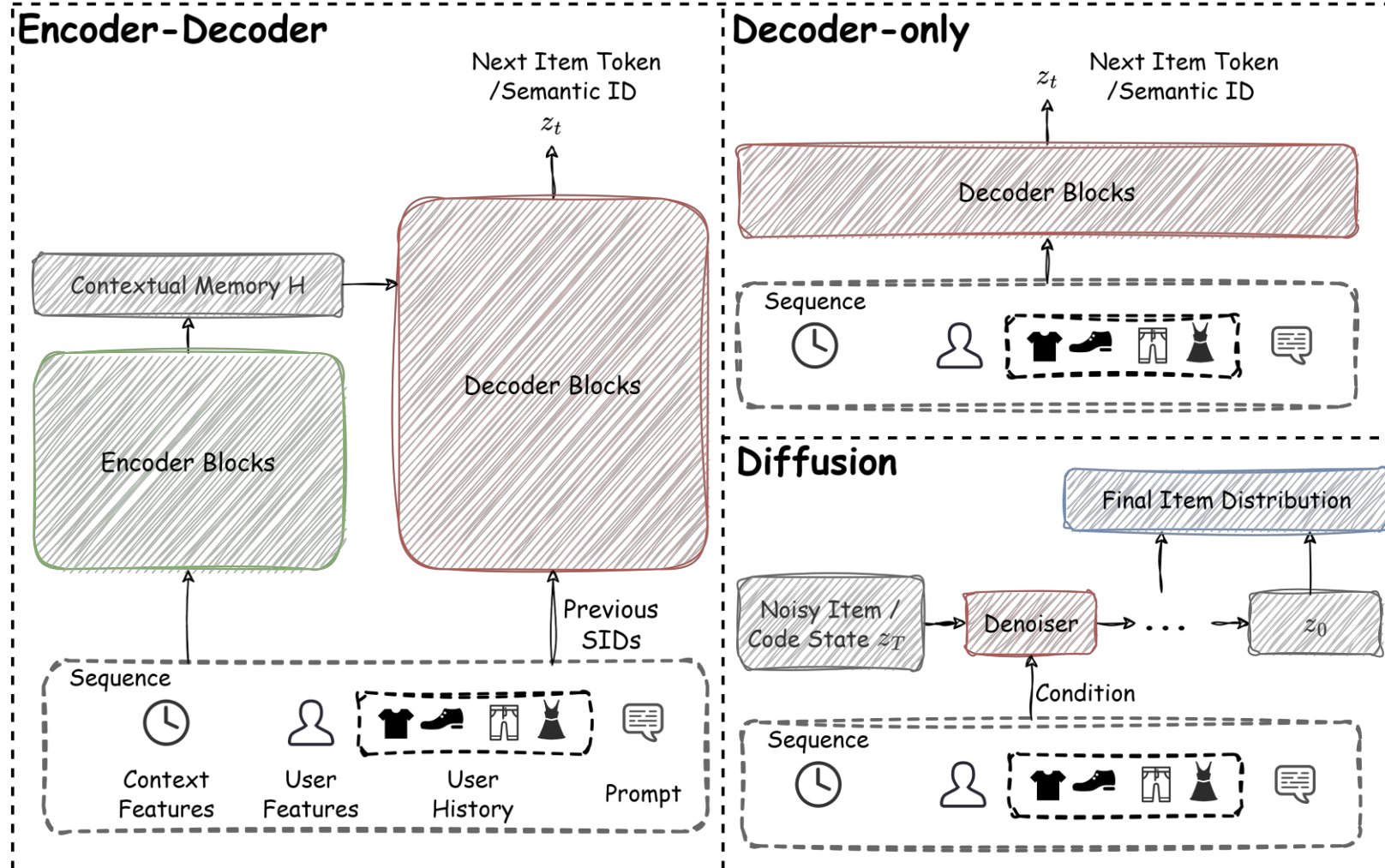
Yejing Wang



6 Challenge & Conclusion (11:30 - 11:45)



Bo Chen



- **Encoder-Decoder**

- Encoding rich information.
- Decoding the SID of next item.

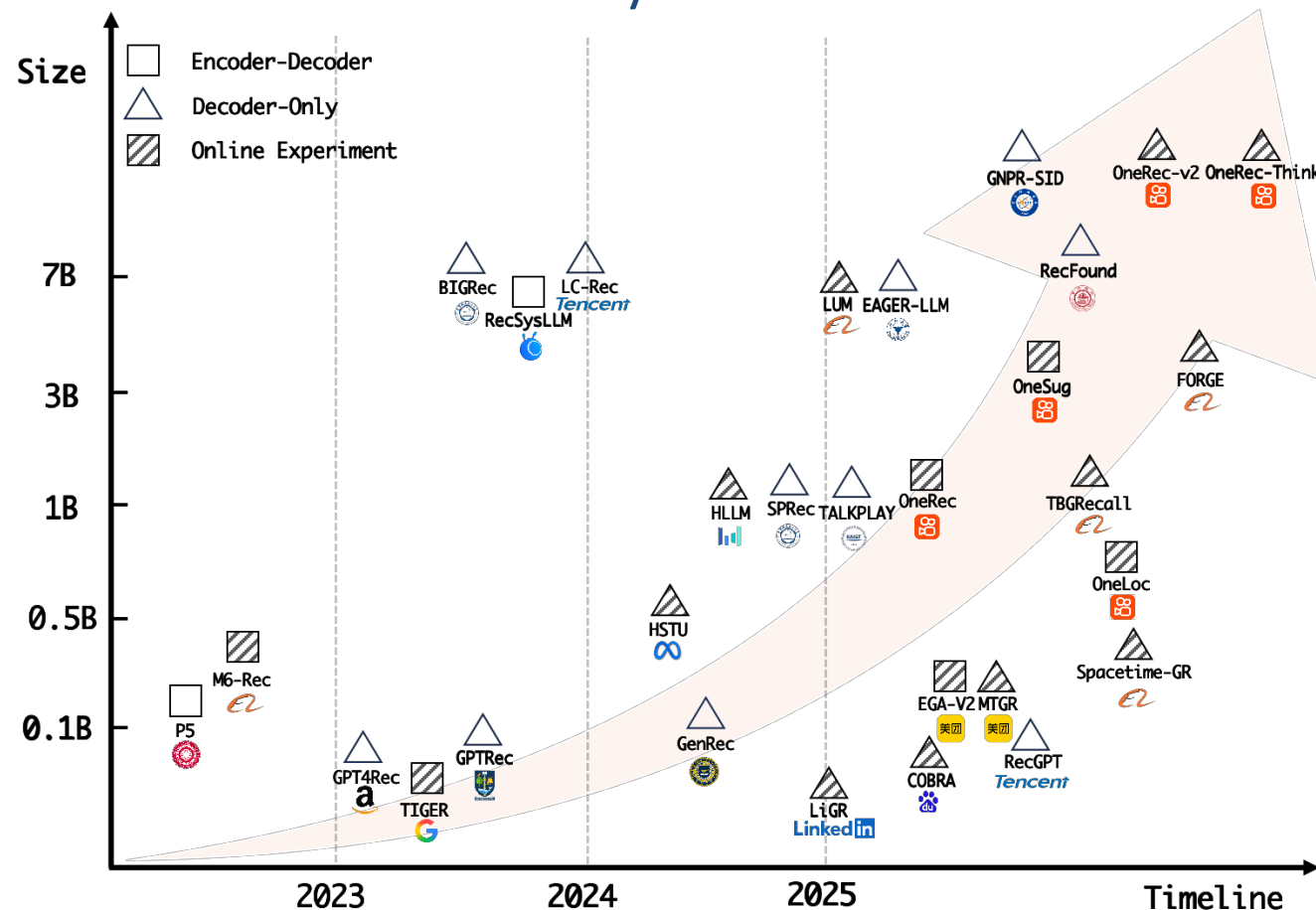
- **Decoder-only**

- Putting all the information in causal sequence.

- **Diffusion-based**

- Denoising the state to the final stage to derive the possible SID.

Model structure Analysis:



Emerging Trend: Decoder-Only + Scaling

- One causal sequence for behavior, context, and candidates.
- Scale long histories with efficient, serving-ready generation.
- Examples: HSTU, COBRA, MTGR, RecGPT.

- **One-page map:**
 - Compare conditioning, generated object, and serving trade-off.

Architecture family	Context / conditioning	Generated object	Serving bias / examples
Encoder-Decoder Architecture	Encoder conditions prompts, instructions, modalities, and local context.	Text, item IDs, semantic IDs, or local-service targets.	Rich but heavier conditioning; P5, InstructRec/VIP5, OneRec-v1, OneLoc.
Decoder-Only Architecture	Causal stream over user actions, candidates, and context tokens.	Next actions/items or sparse-dense retrieval representations.	Throughput-oriented scaling; BIGRec, Llama4Rec, COBRA, RecGPT
Diffusion-based Architecture	Iterative denoising from noisy candidate or code states.	Denoised item/code distributions or candidate sets.	Parallel or non-AR generation; diffusion-based GR.

Axis	P5	InstructRec / VIP5	M6-Rec	OneRec-v1	OneLoc
Input	Prompt text + item IDs	Instructions + visual tokens	Segmented user / candidate / feature fields	Multi-scale user behaviors	Geo-aware SIDs + location
Encoder	T5 encoder	Instruction / visual conditioning	M6 foundation backbone	Multi-path behavior encoder	Location-aware attention
Decoder / output	Text-to-text outputs	Richer-context outputs	Task outputs + late interaction	Causal semantic-ID decoder	Local semantic-ID decoder + RL
Lesson	Backbone reuse	Richer conditioning	Foundation model + segments	End-to-end generator	Serving-specialized context

- **Shared question:**

- What context should the encoder preserve, what target should the decoder generate, and how does serving reshape the architecture?

- **Encoder:** represent user history, task prompt, modality, and context as conditional states.
- **Decoder:** emit text, item tokens, semantic IDs, explanations, or local-service targets.
- **Key question:** what enters the encoder, and what object should the decoder generate?

- **Backbone reuse:** keep a T5-style text-to-text architecture.
- **Conditioning:** add instructions, visual tokens, prompts, or geo context on the encoder side.
- **Decoder structure:** move from one output stream to behavior/semantic or geo-aware code streams.
- **Optimization:** attach auxiliary or deployment-aware objectives around the backbone.

Background

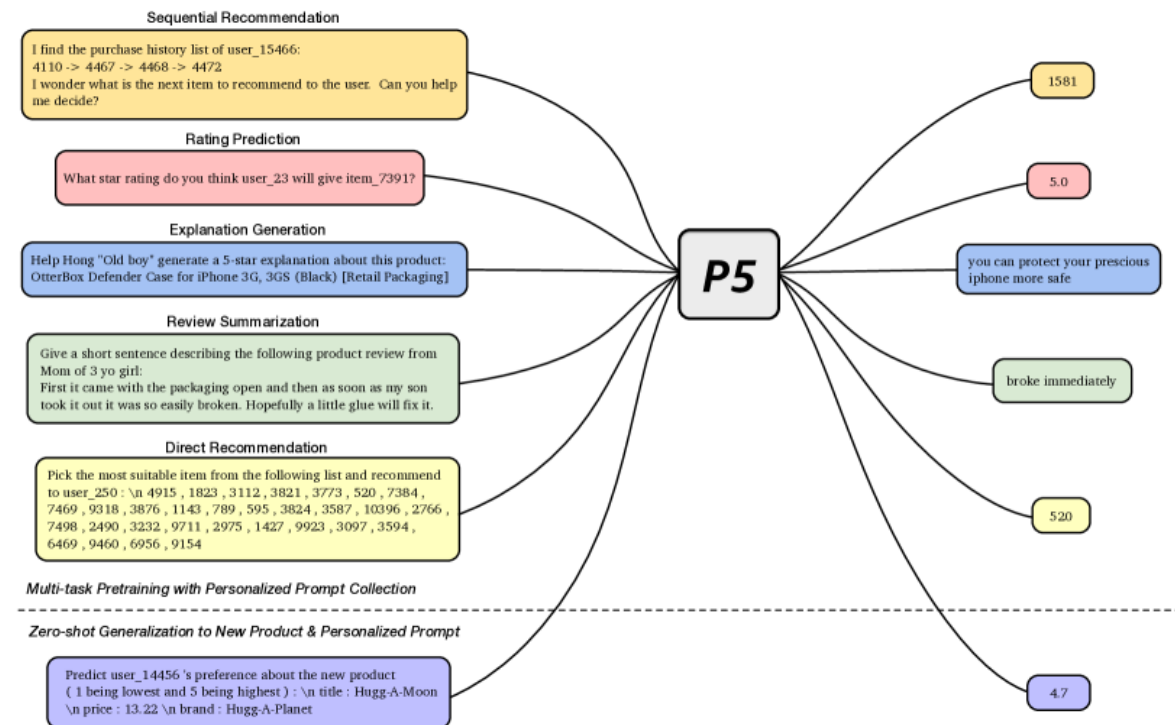
- P5 starts from a standard T5 text-to-text encoder-decoder.

Motivation

- Multiple recommendation tasks need one prompt interface instead of task-specific heads.

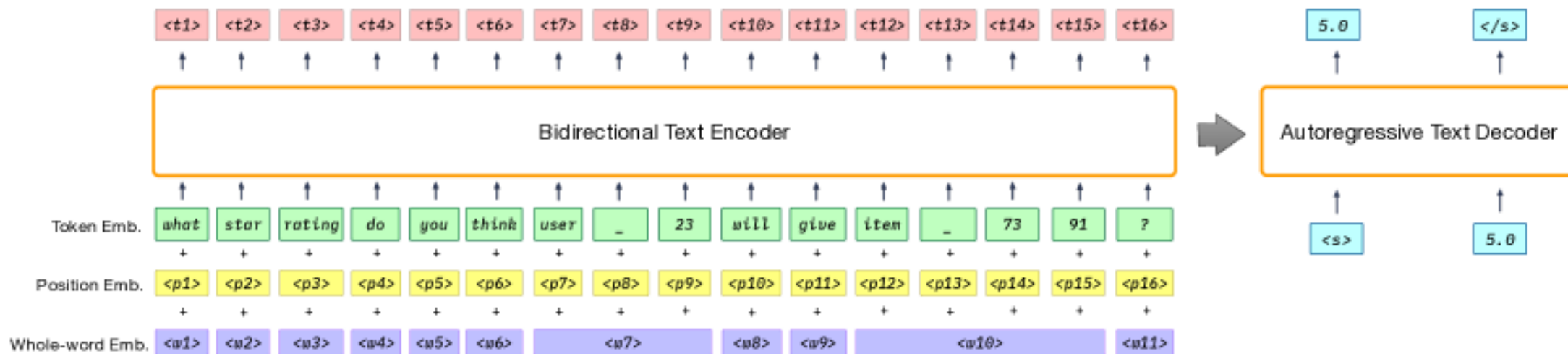
Contributions

- Serialize users, items, histories, and tasks into prompts, then decode task outputs.



- **Input arrangement:** template user, history, item, and task fields as encoder text.
- **Generation:** use the T5 decoder for ratings, item tokens, explanations, and reviews.
- **Takeaway:** architecture novelty is mainly the prompt interface, not new layers.

$$x = \text{template}(u, \mathcal{H}_u, i, \tau), \quad p_{\theta}(y \mid x) = \prod_t p_{\theta}(y_t \mid y_{<t}, x)$$



▪ Background

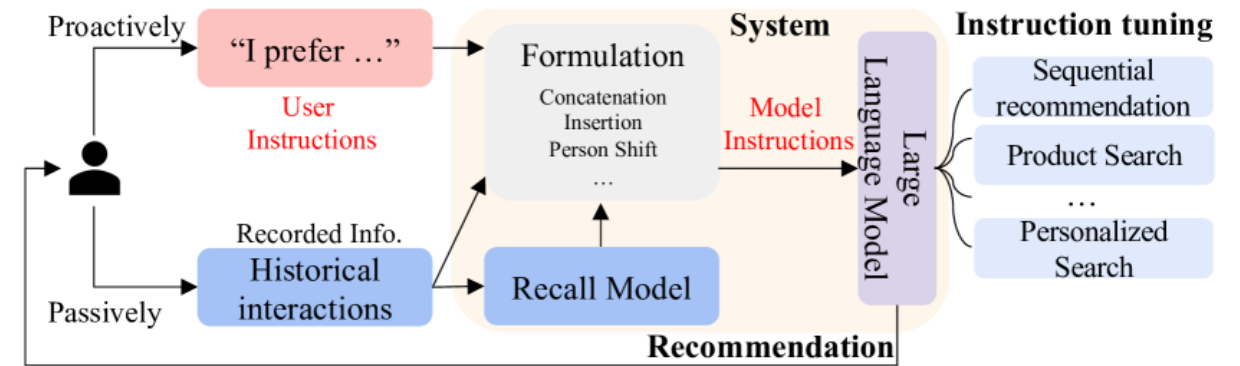
- P5 leaves intent and multimodal item evidence mostly to the prompt text.

▪ Motivation

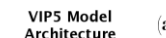
- A stronger encoder side should carry preferences, instructions, and visual signals.

▪ Contributions

- InstructRec injects preference instructions; VIP5 adds visual tokens while preserving T5.



- $$x = [I_u; P_u; V_i; \tau], \quad p_\theta(y | x) = \prod_t p_\theta(y_t | y_{<t}, x)$$



■ Motivation

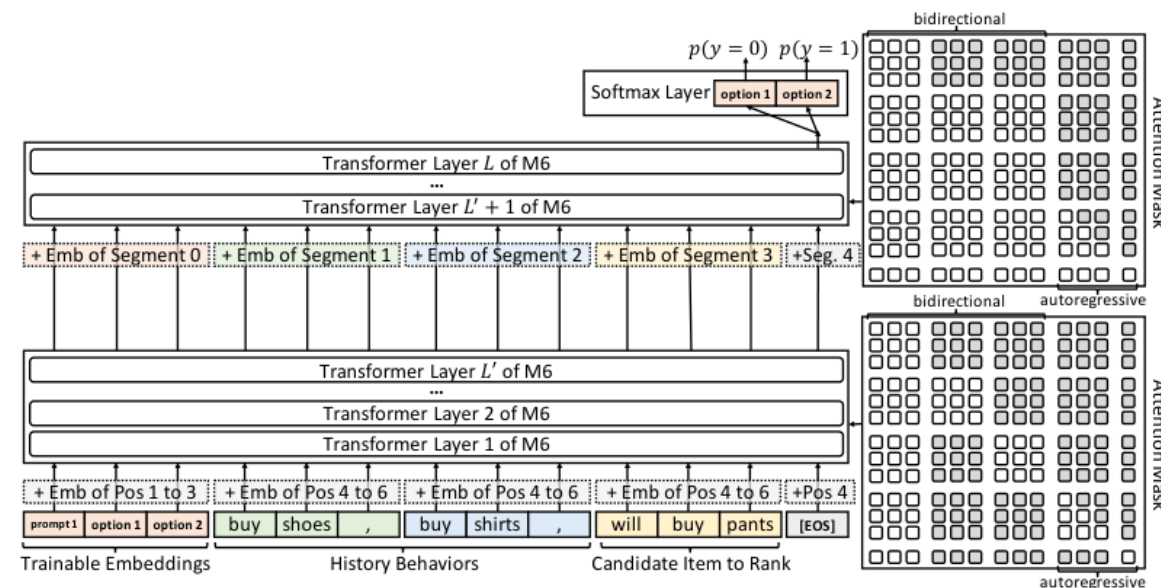
- Foundation models offer one generative interface for heterogeneous recommendation tasks.

■ Challenge

- User, candidate, and feature fields must remain interaction-aware and efficient to adapt.

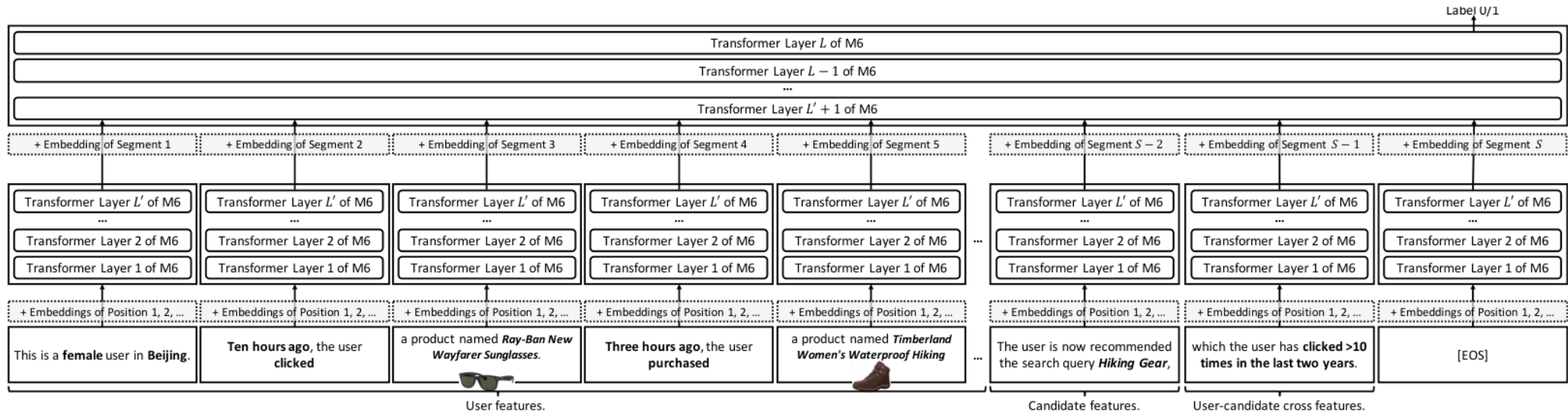
■ Contribution

- M6-Rec combines an M6 backbone with multi-segment forward passes and late interaction.



- A shared M6 backbone reads user, candidate, behavior, and feature segments in one model.
- Late interaction keeps user and candidate features separable until the scoring stage.

$$p_{\theta}(y | x) = \prod_t p_{\theta}(y_t | y_{<t}, x), \quad x = [x_{\text{user}}; x_{\text{candidate}}; x_{\text{feature}}]$$



- M6-Rec reports gains in CTR prediction and all-item retrieval, while Figure 7 shows zero-shot transfer.

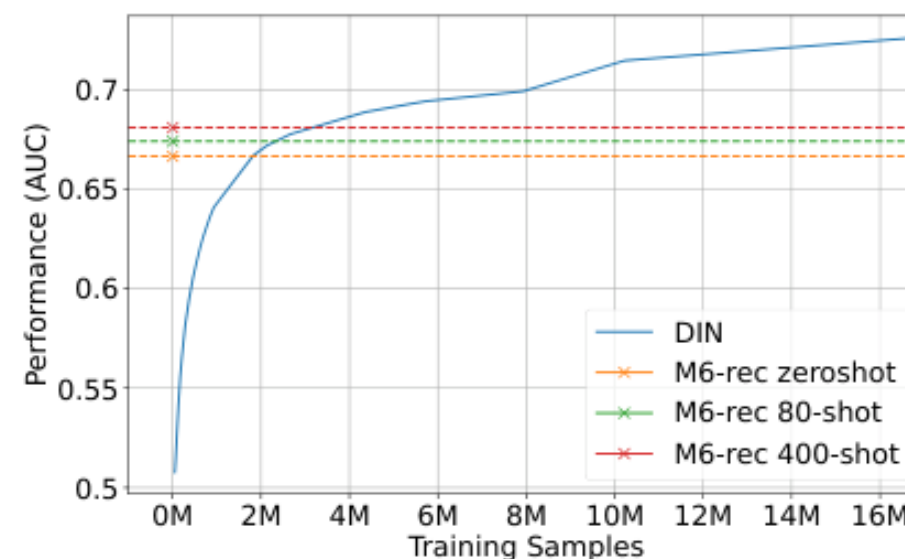
Main task results

Method	Method Type	Datasets	
		AlipayQuery↑	TaoProduct↑
DIN	ID embeddings	0.7332	0.7611
M6-Rec	Text semantics	0.7508	0.7995

All-item retrieval (HitRate@100)

Method	Method Type	Test Sets	
		All Items↑	Unseen Items↑
YouTubeDNN	ID embeddings	54.4%	<i>fail</i>
TwinBERT	Text semantics	69.6%	49.6%
M6-Rec	Text semantics	74.1%	57.0%

Zero-shot transfer (AlipayQuery)



Main signal: M6-Rec reaches 0.7508/0.7995 AUC in CTR and 74.1% HitRate@100 in retrieval.

▪ Motivation

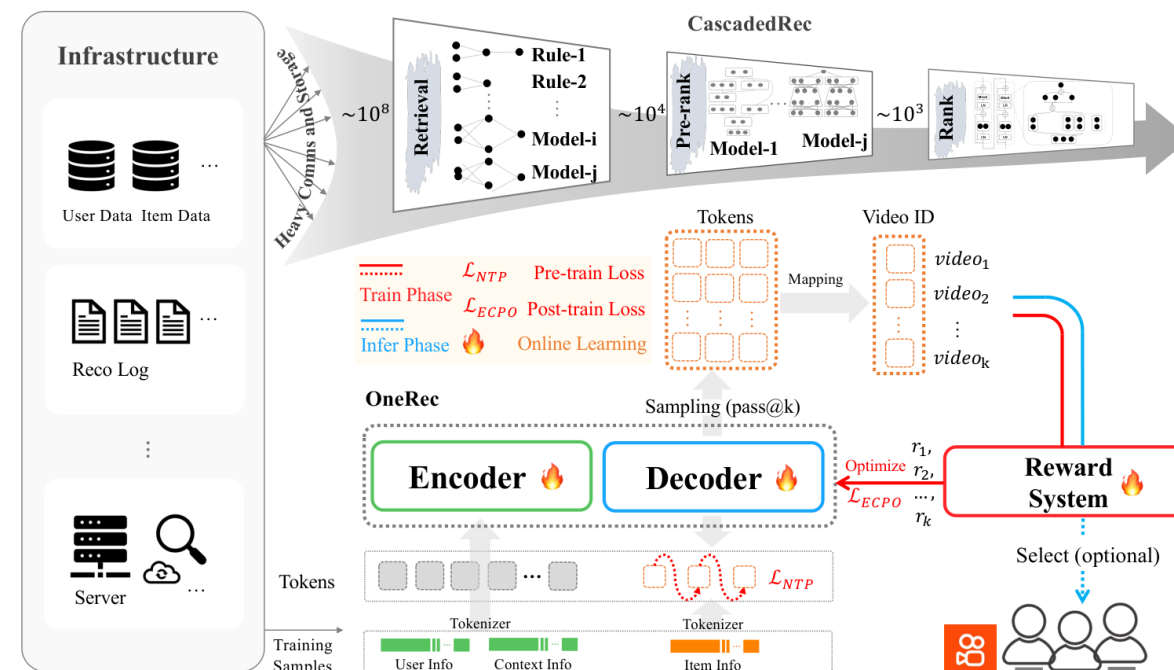
- Cascaded retrieval and ranking pipelines fragment optimization and waste modern accelerator capacity.

▪ Challenge

- A generator must use rich user behavior while directly optimizing the final recommendation outcome.

▪ Contribution

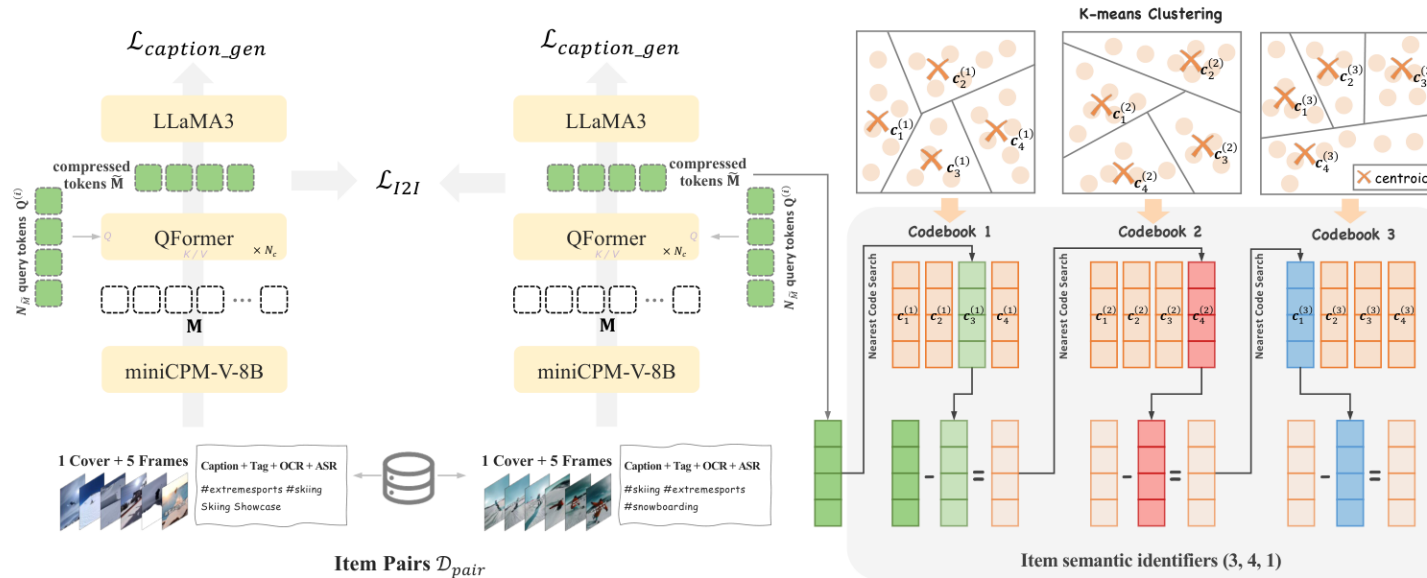
- OneRec replaces the cascade with an end-to-end encoder-decoder generator guided by rewards.



Encoder-Decoder: OneRec-v1

- The tokenizer aligns multimodal item content with collaborative item-pair signals before quantization.
- RQ-Kmeans turns each item into coarse-to-fine semantic IDs, giving the decoder a compact fixed target vocabulary.

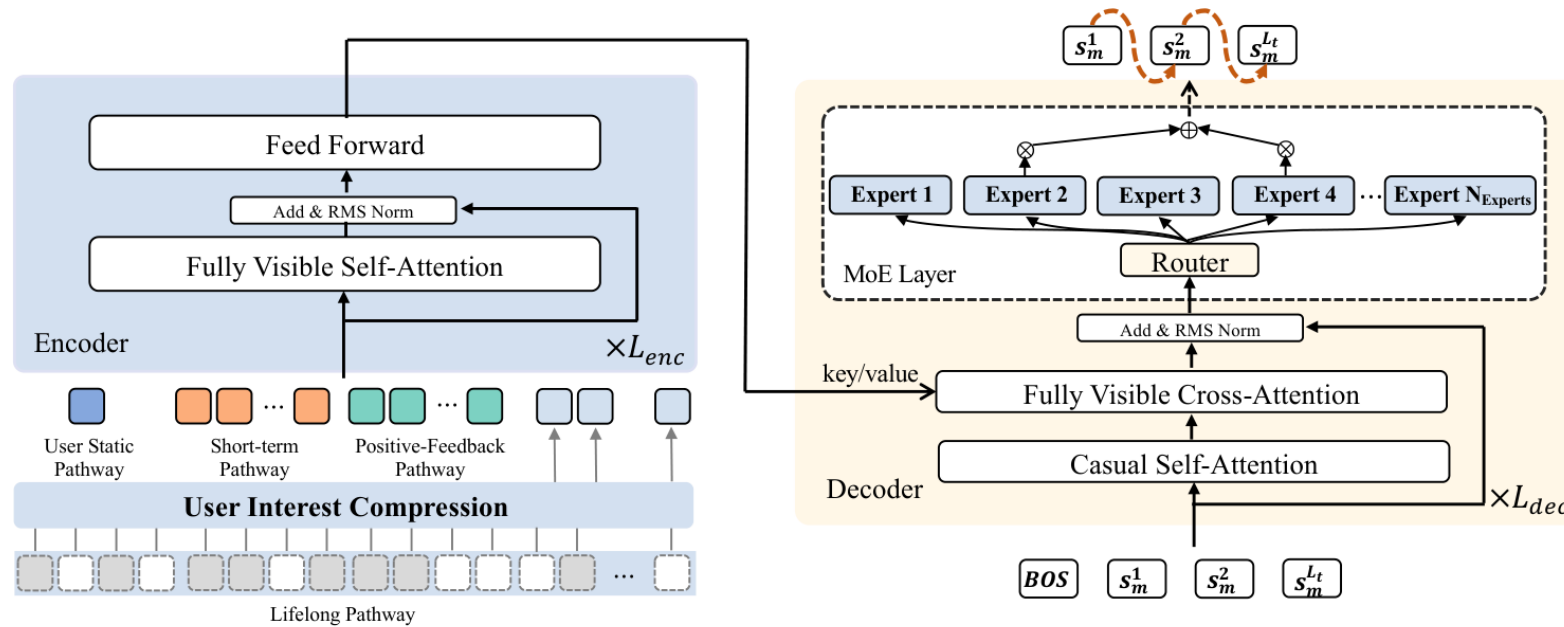
$$s_i^{(\ell)} = \arg \min_k \|R_i^{(\ell)} - c_k^{(\ell)}\|, \quad \mathbf{s}_i = [s_i^{(1)}, \dots, s_i^{(L_t)}]$$



Encoder-Decoder: OneRec-v1

- The encoder-decoder uses static, short-term, positive-feedback, and lifecycle behavior pathways.
- A causal decoder generates coarse-to-fine semantic IDs through cross-attention to encoder states.

$$p_{\theta}(z_{1:L} | x) = \prod_{l=1}^L p_{\theta}(z_l | z_{<l}, \text{Enc}_{\theta}(x))$$



- We pair offline feature ablation with online RL results under different Pass@K settings.

Offline: feature ablation (Table 2)

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%

Online: RL impact under Pass@K (Table 6)

	Method	vtr	Watch time	App Stay Time	Video View ¹
Pass@32	OneRec w/o RL	0.1978	+1.62%	-0.10%	-4.18%
	OneRec w/ RL	0.2138	+3.17%	+0.39%	-9.87%
	Relative Impr.	+8.08%	+1.55%	+0.49%↑↑↑	-3.69%
Pass@128	OneRec w/o RL	0.2239	+4.61%	+1.11%	-12.75%
	OneRec w/ RL	0.2387	+5.22%	+1.49%	-15.06%
	Relative Impr.	+6.61%	+1.53%	+0.38%↑↑	-2.65%
Pass@512	OneRec w/o RL	0.2444	+6.32%	+1.66%	-15.54%
	OneRec w/ RL	0.2494	+5.88%	+1.75%	-13.88%
	Relative Impr.	+2.05%	-0.41%	+0.09%↑	+1.97%

- **Main signal:**

- Offline features improve P-Score by 28.88%; online RL at Pass@32 raises App Stay Time by +0.49%.

▪ Background

- Local-life recommendation is conditioned by history, current location, and nearby POIs.

▪ Motivation

- Plain SID generation misses geography, neighborhood structure, and business signals.

▪ Contributions

- OneLoc injects geography into encoder attention, decoder prompts, and RL rewards.

Encoder-Decoder: OneLoc

Geo-aware attention

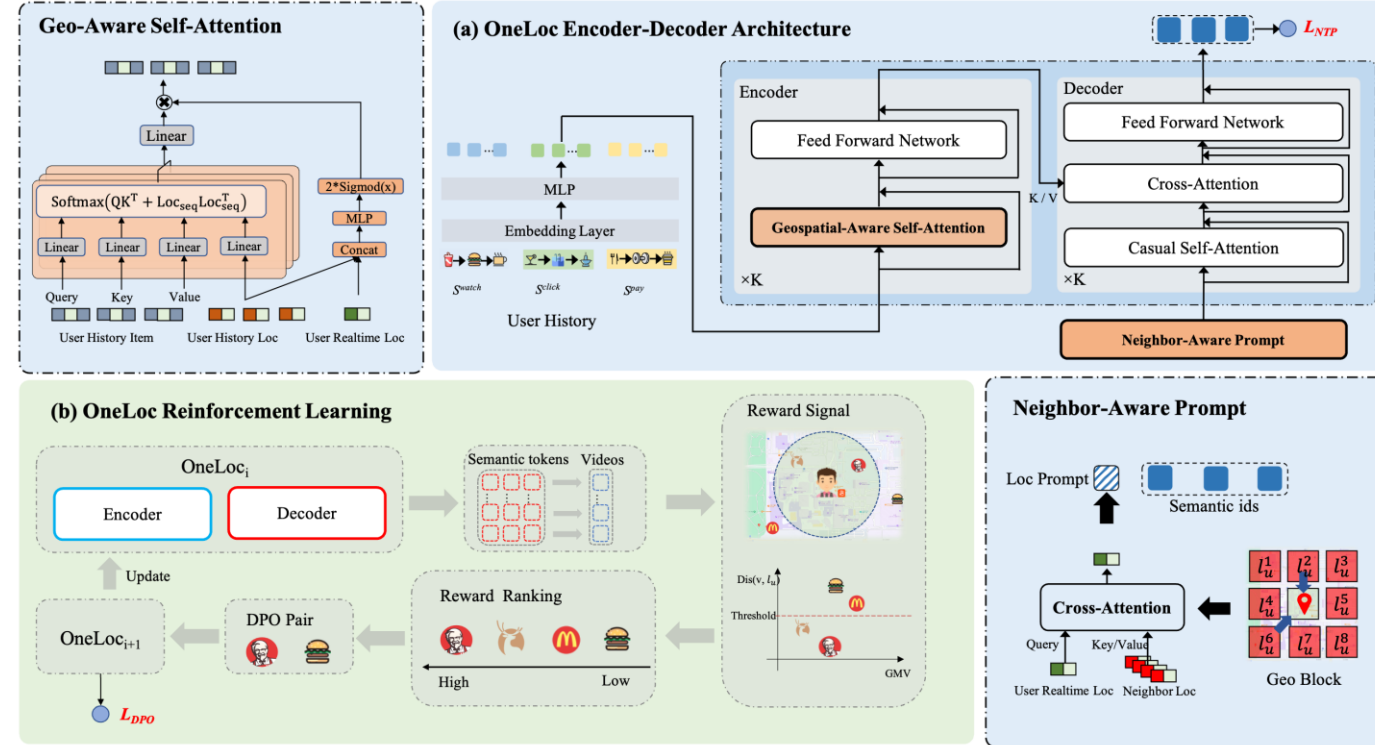
- Adding location similarity to encoder self-attention.

Location gate

- Modulating encoder outputs before decoder cross-attention.

Neighbor prompt

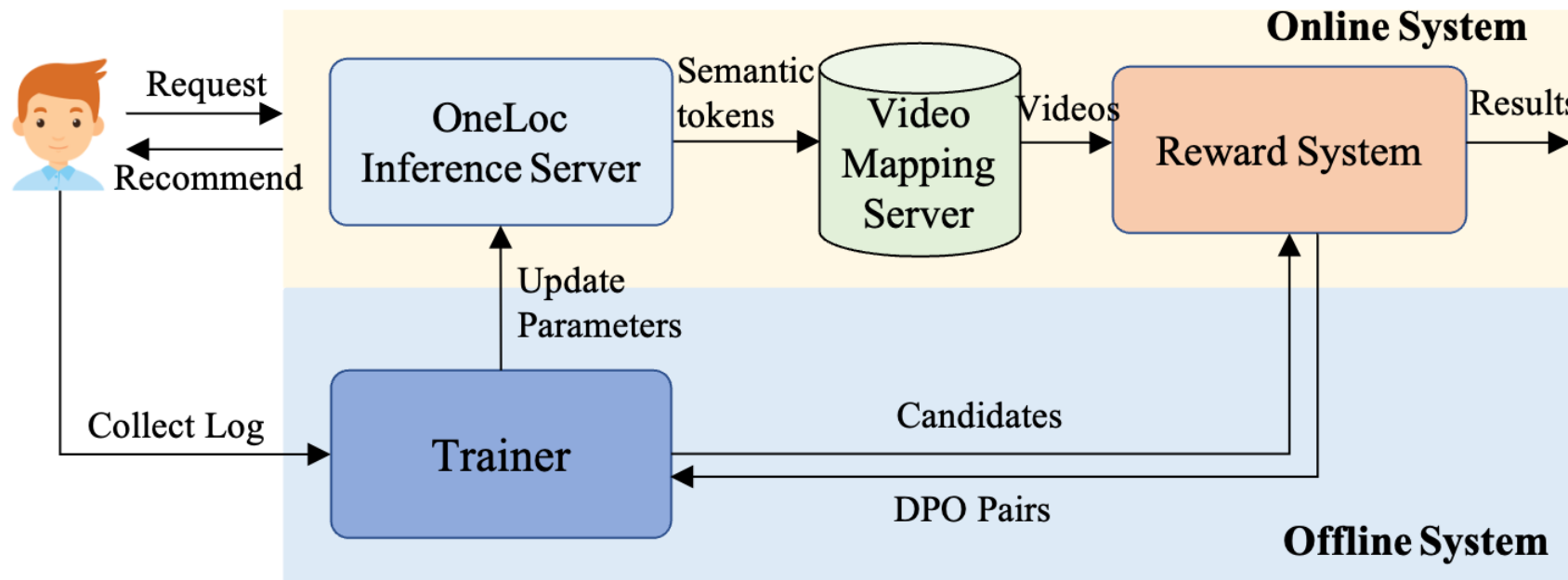
- Guiding the decoder with surrounding POI context.



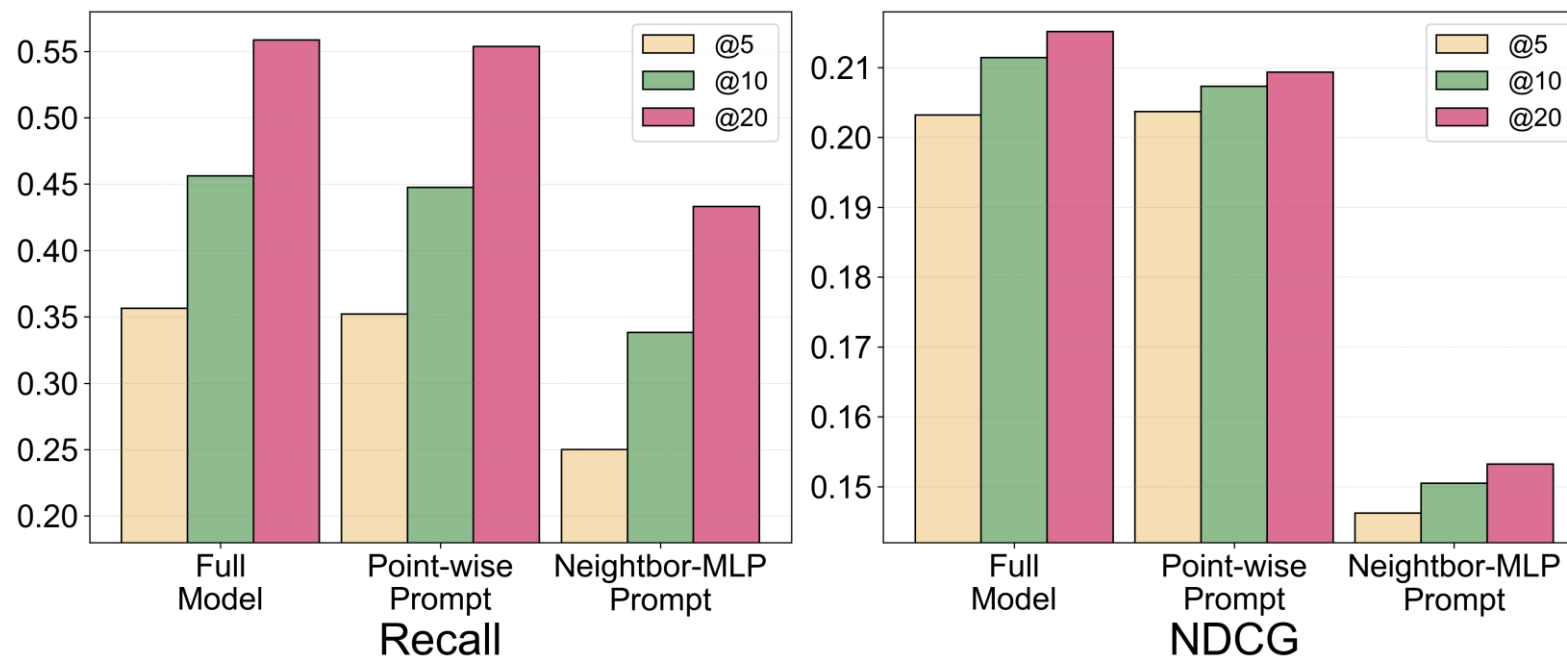
$$A = \text{softmax}(QK^\top / \sqrt{d} + \lambda S_{\text{geo}}), \quad p(z_t | z_{<t}, h, p_{\text{nbr}})$$

Encoder-Decoder: OneLoc

- **Offline system:** learn geo-aware SIDs and train the encoder-decoder generator.
- **Online system:** decode local candidates, then pass them to retrieval/ranking service.
- **Takeaway:** architecture changes are tied to serving, not only model accuracy.

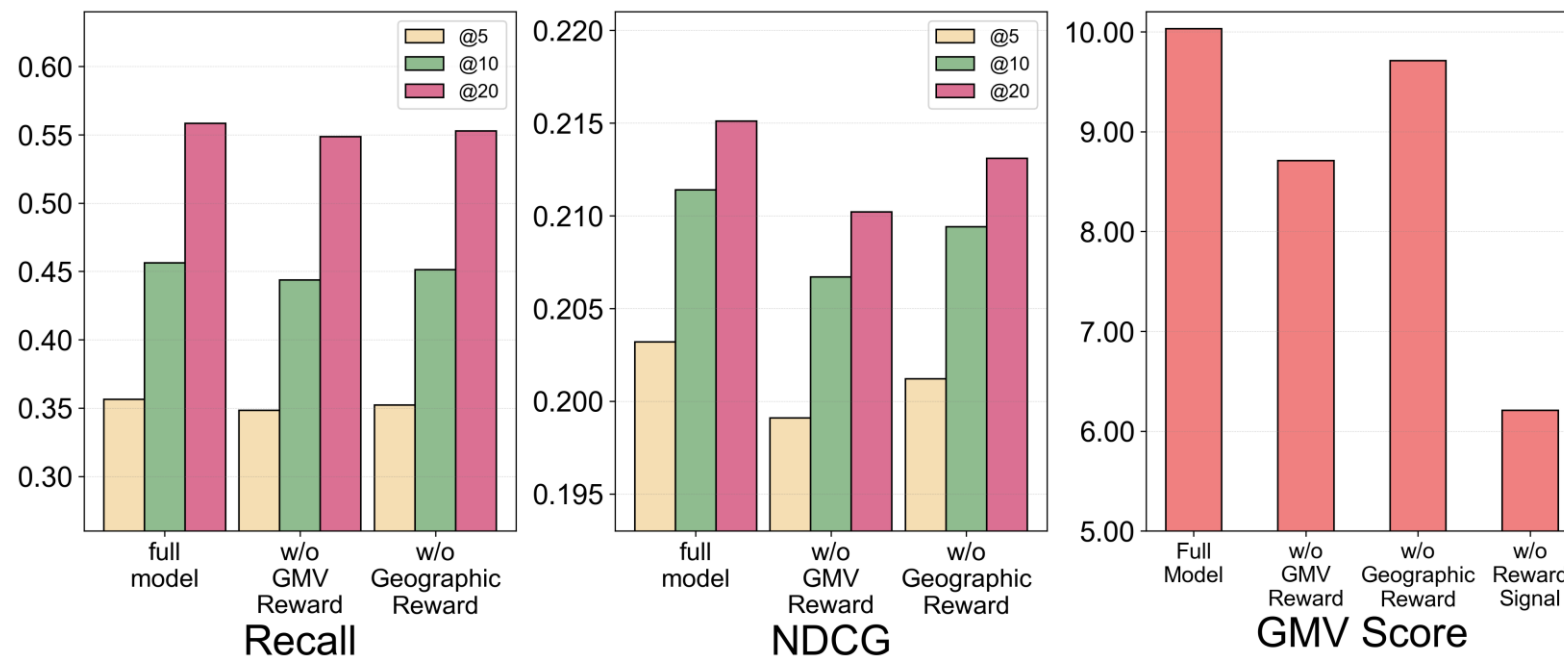


- **Prompt variants:** compare point-wise, neighbor-MLP, and neighbor-aware prompt designs.



- **Main signal:**
 - Neighbor-aware prompts improve local sensitivity in the decoder.

- **Reward variants:** combine Recall, NDCG, distance, and GMV signals.



- **Main signal:**
 - RL rewards move generated candidates toward deployment objectives.

Encoder-Decoder Work Summary



August 9–13, 2026
Jeju, Korea

Axis	P5	InstructRec / VIP5	M6-Rec	OneRec-v1	OneLoc
Input	Prompt text + item IDs	Instructions + visual tokens	Segmented user / candidate / feature fields	Multi-scale user behaviors	Geo-aware SIDs + location
Encoder	T5 encoder	Instruction / visual conditioning	M6 foundation backbone	Multi-path behavior encoder	Location-aware attention
Decoder / output	Text-to-text outputs	Richer-context outputs	Task outputs + late interaction	Causal semantic-ID decoder	Local semantic-ID decoder + RL
Lesson	Backbone reuse	Richer conditioning	Foundation model + segments	End-to-end generator	Serving-specialized context

- **Bridge:**

- As more context and target design move into one generative interface, can a scaled causal decoder absorb the encoder?

- **Background:** early encoder-decoder GR mainly reuses a strong text-to-text backbone.
- **Motivation:** gains come from richer conditioning before generation.
- **Takeaway:** later systems specialize output streams, semantic codes, and serving objectives.

- Encoder-decoder is strong when context is rich and heterogeneous.
- Industrial serving exposes a cost problem: multi-encoder or multi-stream designs are heavier.
- Decoder-only models move the story toward one scalable sequence and high-throughput generation.

Why Decoder-Only Scales



KDD2026
August 9–13, 2026
Jeju, Korea

Architecture	Computation	Scaling	Recommendation Interface
Encoder-Decoder	Separate encoding + cross-attention	Two stacks to scale	Encode history, then decode an item
Decoder-Only	One causal Transformer	One reusable backbone	Predict the next token or item
Why the shift?	Simpler training path	Easier model, data, and context scaling	Direct fit for generation

- **Takeaway:**
 - Decoder-only turns recommendation into a standard autoregressive scaling problem.

Two Paths of Decoder-Only



August 9–13, 2026
Jeju, Korea

Paradigm	Item Representation	Prediction Target	Selected Papers
LLM-Based Recommendation	Natural-language semantics	Item text or relevance	BIGRec; Llama4Rec
SID-Based Generative Rec	Discrete semantic item IDs	Next SID sequence	COBRA; RecGPT
Shared Foundation	Decoder-only Transformer	Autoregressive generation	Same backbone, different output space

- **Background:**

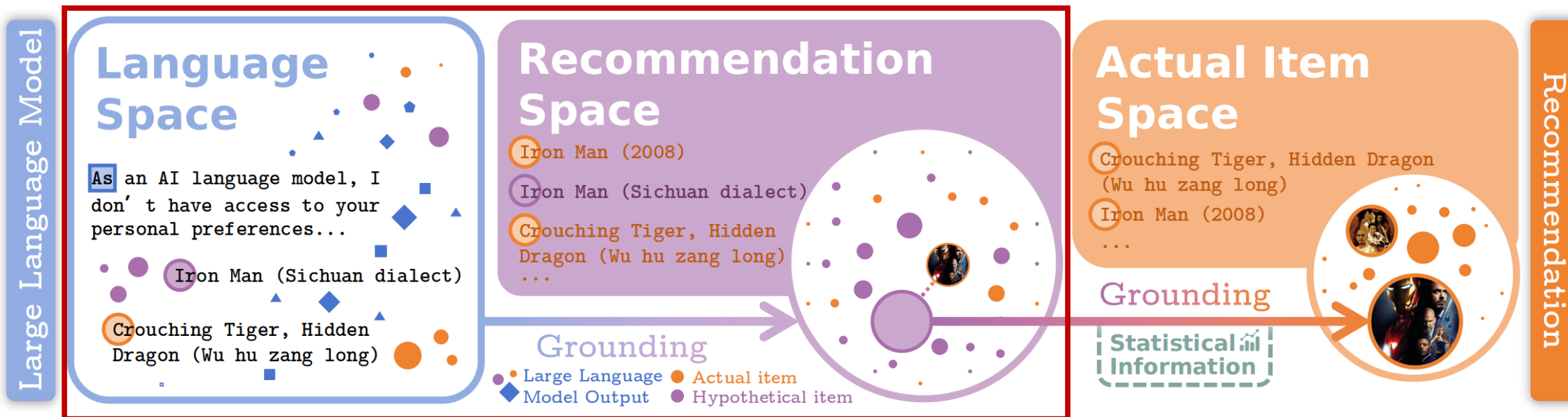
- LLMs provide strong semantic priors for recommendation.
- Existing LLM4Rec methods mainly use sampled candidate sets.

- **Motivation:**

- Preserve meaningful and flexible item generation.
- Ground free-form outputs to valid catalog items.
- Integrate popularity and collaborative signals.

- **Contributions:**

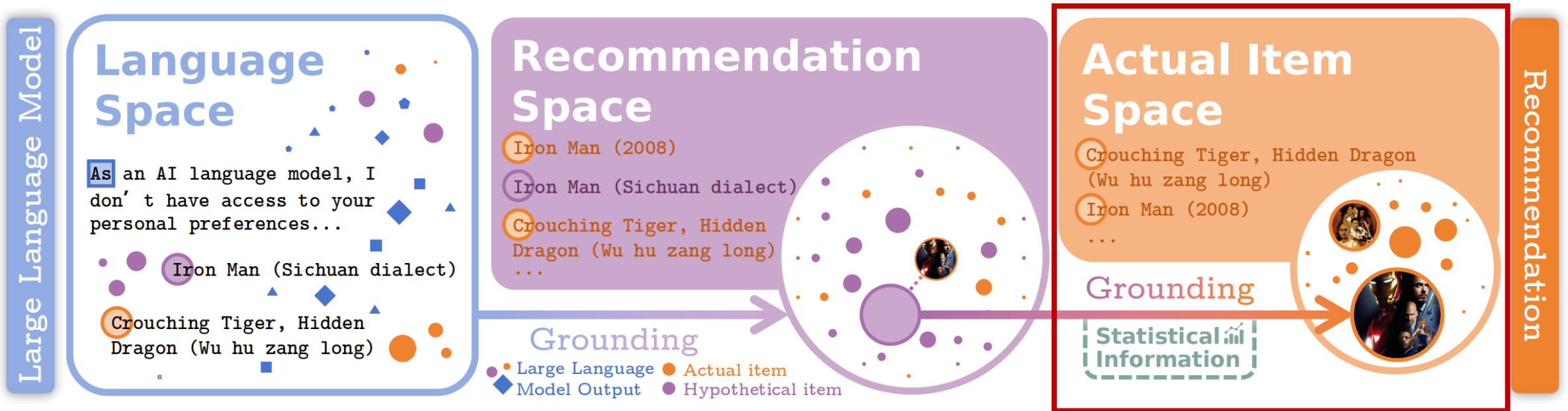
- Bi-step grounding: language space $\rightarrow$ recommendation space $\rightarrow$ actual item space.
- Flexible integration of semantic and behavioral information.



- **Language-to-Recommendation Grounding:** instruction-tune on user histories and target item text.

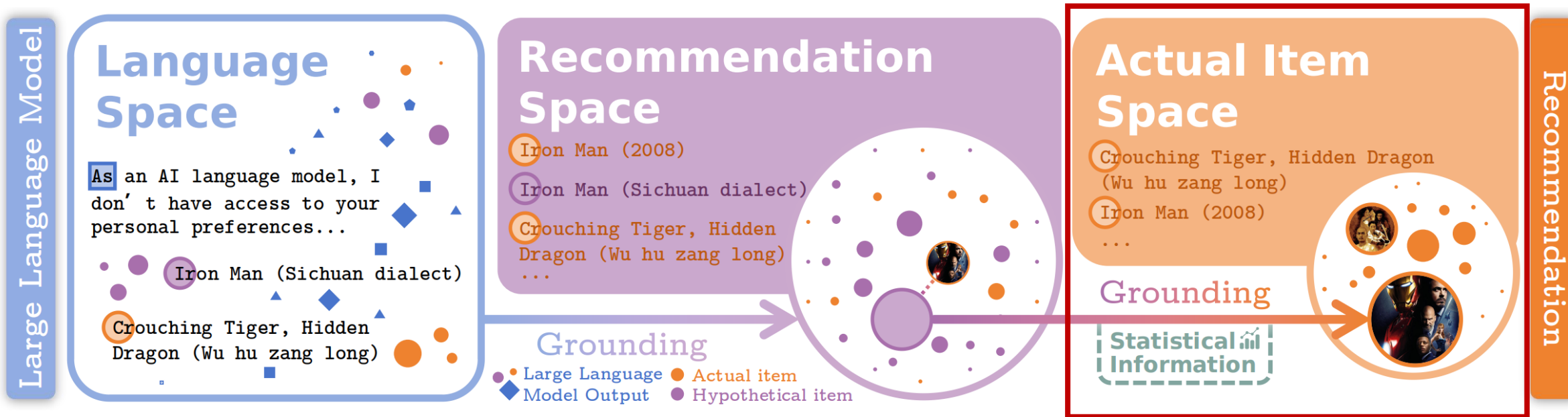
- Constrain open-ended generation to preference-aligned item concepts.

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)"



- **Catalog grounding:** rank real items by L2 distance to the generated concept.

$$D_i = ||\mathbf{emb}_i - \mathbf{oracle}||_2,$$



- **Catalog grounding:** retrieve the closest real items by semantic distance.

- **Statistical reweighting:** inject popularity or collaborative scores during reranking.

$$D_i = ||\text{emb}_i - \text{oracle}||_2,$$



$$\begin{cases} C_i = \frac{N^i}{\sum_{j \in I} N^j}, \\ P_i = \frac{C_i - \min_{j \in I} \{C_j\}}{\max_{j \in I} \{C_j\} - \min_{j \in I} \{C_j\}}, \end{cases} \quad \begin{cases} \hat{D}_i = \frac{D_i - \min_{j \in I} \{D_j\}}{\max_{j \in I} \{D_j\} - \min_{j \in I} \{D_j\}}, \\ \tilde{D}_i = \frac{\hat{D}_i}{(1 + P_i)^\gamma}, \end{cases}$$

Decoder-only: BIGRec



August 9-13, 2026
Jeju, Korea

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%

- **Main signal:**

- With only 1,024 samples, BIGRec substantially outperforms data-matched baselines.

Decoder-only: BIGRec



August 9–13, 2026
Jeju, Korea

Injected	Model	NG@1	NG@3	NG@5	NG@10	NG@20	HR@1	HR@3	HR@5	HR@10	HR@20
Movie											
Single Model (+None)	Most-Pop	0.0032	0.0076	0.0088	0.0121	0.0170	0.0032	0.0108	0.0138	0.0244	0.0438
	GRU4Rec	0.0047	0.0108	0.0151	0.0237	0.0351	0.0047	0.0155	0.0259	0.0527	0.0985
	Caser	0.0045	0.0113	0.0161	0.0242	0.0354	0.0045	0.0165	0.0281	0.0537	0.0986
	SASRec	0.0045	0.0119	0.0171	0.0268	0.0389	0.0045	0.0175	0.0302	0.0606	0.1088
	DROS	0.0087	0.0186	0.0245	0.0359	0.0493	0.0087	0.0261	0.0406	0.0761	0.1292
	BIGRec (0)	0.0020	0.0034	0.0046	0.0060	0.0095	0.0020	0.0044	0.0074	0.0120	0.0258
+ DROS	BIGRec (1024)	0.0176	0.0214	0.0230	0.0257	0.0283	0.0176	0.0241	0.0281	0.0366	0.0471
	Caser	0.0087	0.0183	0.0247	0.0354	0.0494	0.0087	0.0258	0.0404	0.0736	0.1296
	SASRec	0.0089	0.0184	0.0245	0.0357	0.0493	0.0089	0.0256	0.0409	0.0754	0.1307
	BIGRec (1024)	0.0176	0.0250	0.0315	0.0427	0.0562	0.0176	0.0308	0.0464	0.0813	0.1353
Game											
Single Model (+None)	Most-Pop	0.0000	0.0000	0.0000	0.0004	0.0018	0.0000	0.0000	0.0000	0.0014	0.0068
	GRU4Rec	0.0051	0.0080	0.0094	0.0109	0.0129	0.0051	0.0101	0.0135	0.0184	0.0263
	Caser	0.0059	0.0094	0.0111	0.0141	0.0177	0.0059	0.0119	0.0161	0.0256	0.0401
	SASRec	0.0113	0.0151	0.0164	0.0185	0.0204	0.0113	0.0179	0.0209	0.0275	0.0353
	P5	0.0094	0.0116	0.0131	0.0145	0.0167	0.0094	0.0134	0.0172	0.0216	0.0300
	DROS	0.0156	0.0194	0.0213	0.0244	0.0278	0.0156	0.0221	0.0269	0.0365	0.0500
+ DROS	BIGRec (0)	0.0020	0.0032	0.0036	0.0054	0.0073	0.002	0.004	0.005	0.0106	0.0184
	BIGRec (1024)	0.0133	0.0169	0.0189	0.0216	0.0248	0.0133	0.0195	0.0243	0.0329	0.0457
	BIGRec (full)	0.0221	0.0250	0.0272	0.0297	0.0319	0.0221	0.0270	0.0326	0.0401	0.0490
	Caser	0.0159	0.0199	0.0217	0.0249	0.0288	0.0159	0.0227	0.0279	0.0375	0.0529
	SASRec	0.0151	0.0197	0.0216	0.0247	0.0277	0.0151	0.0226	0.0277	0.0373	0.0488
	BIGRec (1024)	0.0133	0.0243	0.0268	0.0302	0.0338	0.0133	0.0320	0.0377	0.0474	0.0619
+ DROS	BIGRec (full)	0.0221	0.0292	0.0312	0.0338	0.0375	0.0221	0.0340	0.0387	0.0478	0.0611

- **Main signal:**

- BIGRec remains competitive with full-data recommenders; collaborative signals further improve grounding.

- **Background:**

While **Conventional methods** excel at mining collaborative information and modeling sequential behavior, they struggle with **data sparsity** and **long-tail problem**.

↑
Significant Gap
↓

LLM-based methods are proficient at utilizing rich textual contexts, but face challenges in utilizing **collaborative or sequential information**.

- **Research Questions:**

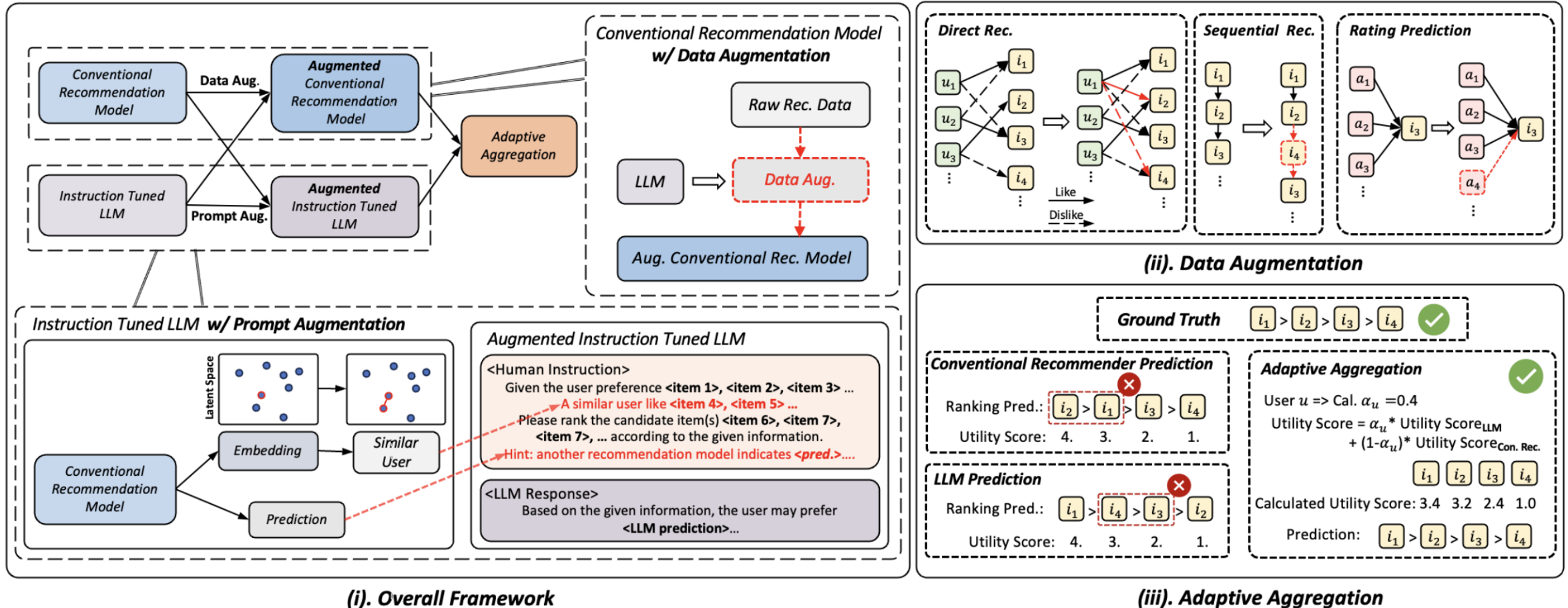
How to leverage **LLM's reasoning ability** in conventional methods?

How to introduce **collaborative/sequential information** in LLM-based methods?

How to **aggregate** the LLM-based method and conventional methods properly?

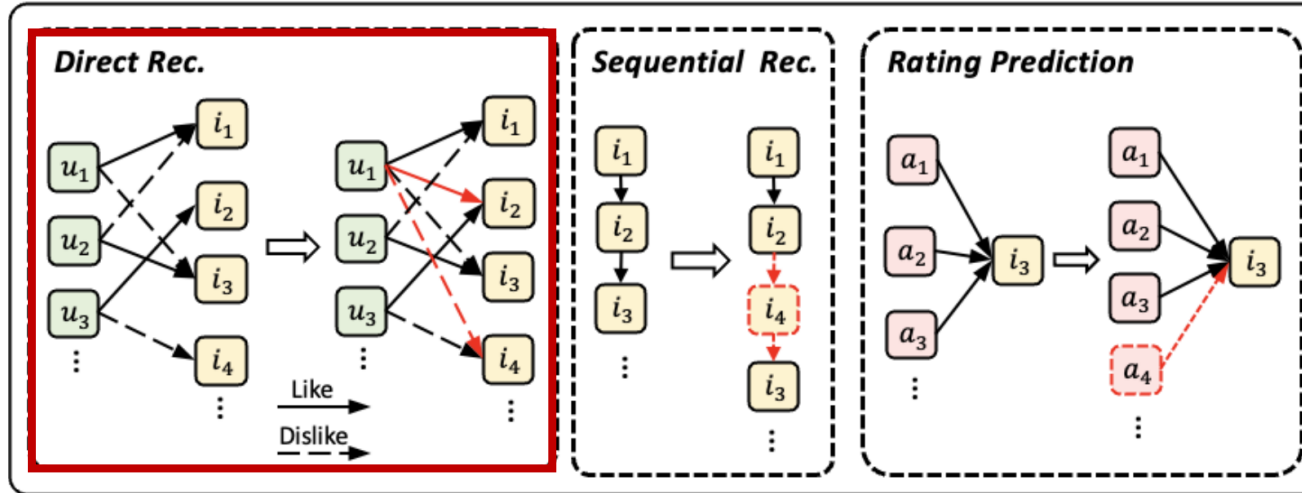
Decoder-only: LlamaRec

- To overcome the above problems, the authors propose LlamaRec, which allows the conventional & LLM-based augmentation to **mutually augment each other** and **adaptively aggregates the results**.



Decoder-only: LlamaRec

- Specifically, for the first problem ‘**Q1**: How to leverage LLM’s reasoning ability in conventional methods?’, The authors designed the **Data Augmentation** module.

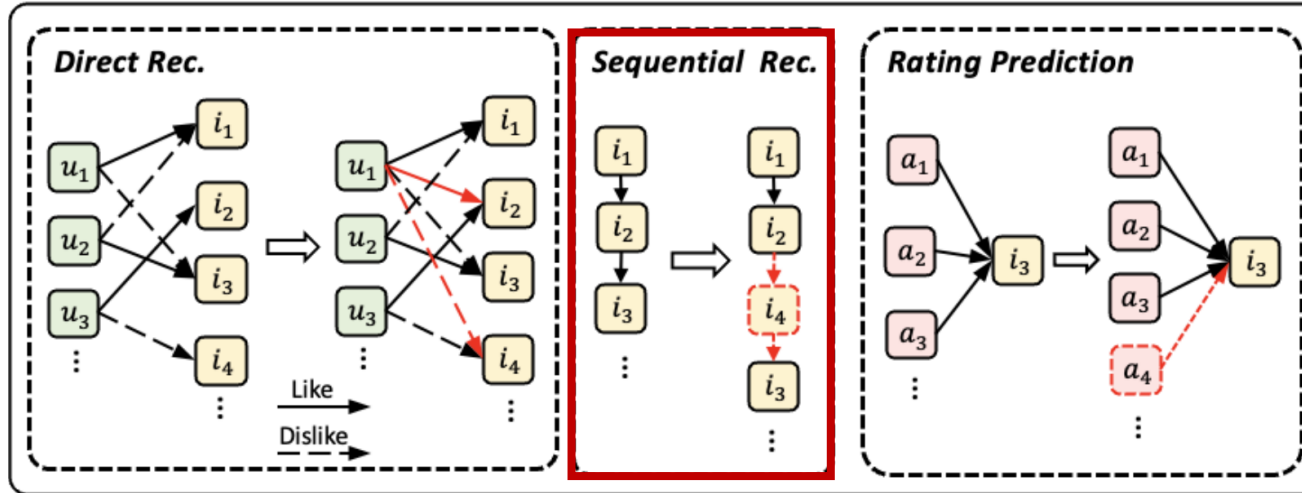


- **Data Augmentation:**
- **Direct Recommendation:**
 - Leveraging LLM to rank each item pair and form the corresponding BPR loss:

$$\mathcal{L}'_{BPR} = - \sum_{(u,i,j) \in \mathcal{D}'} \log \sigma(\hat{y}_{ui} - \hat{y}_{uj})$$

Decoder-only: LlamaRec

- Specifically, for the first problem ‘**Q1**: How to leverage LLM’s reasoning ability in conventional methods?’, The authors designed the **Data Augmentation** module.



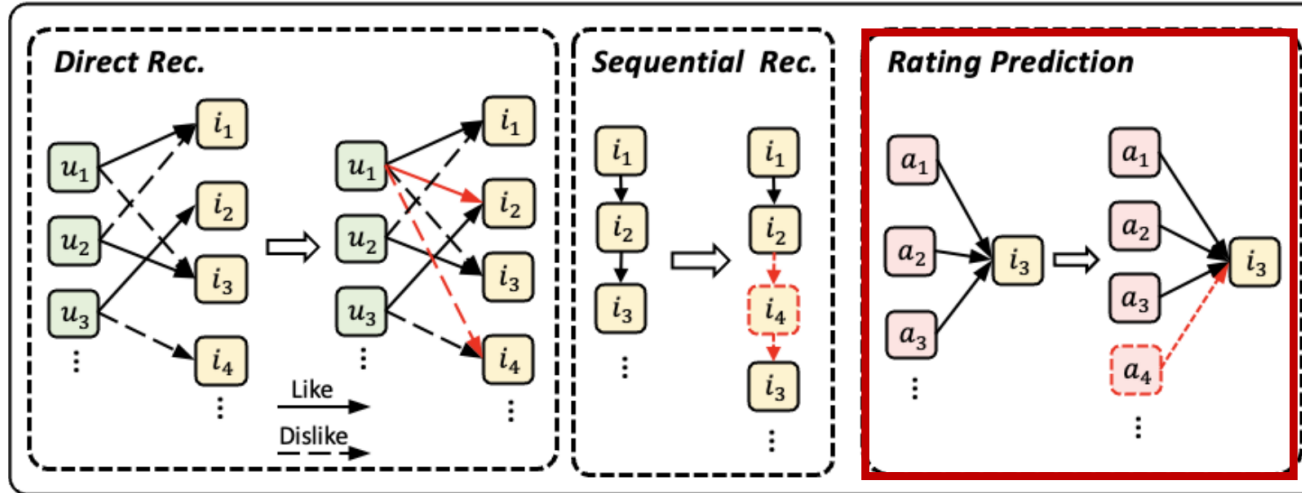
- Data Augmentation:
- Direct Recommendation:
 - Leveraging LLM to rank each item pair and form the corresponding BPR loss:

$$\mathcal{L}'_{BPR} = - \sum_{(u,i,j) \in \mathcal{D}'} \log \sigma(\hat{y}_{ui} - \hat{y}_{uj})$$

- Sequential Recommendation:**
 - Leveraging LLM to predict the user preferred items in the un-interacted list, and then inserting them in the interaction sequence.

Decoder-only: LlamaRec

- Specifically, for the first problem ‘**Q1**: How to leverage LLM’s reasoning ability in conventional methods?’, The authors designed the **Data Augmentation** module.



- Data Augmentation:
- Direct Recommendation:
 - Leveraging LLM to rank each item pair and form the corresponding BPR loss:

$$\mathcal{L}'_{BPR} = - \sum_{(u,i,j) \in \mathcal{D}'} \log \sigma(\hat{y}_{ui} - \hat{y}_{uj})$$

- Sequential Recommendation:
 - Leveraging LLM to predict the user preferred items in the un-interacted list, and then inserting them in the interaction sequence.
- Rating Prediction:
 - Leveraging LLM’s world knowledge to provide side information of item features.

Decoder-only: LlamaRec

- For the second problem ‘**Q2**: How to introduce collaborative/sequential information in LLM-based methods?’, The authors design the two **Prompt Augmentation** strategies.

Top-*k* Recommendation Prompt Example:

Instruction: Rank the candidate movies based on user historical interactions and make the top *k* recommendations.

Interaction History: Beyond Rangoon (1995); Alien (1979); Hollow Reed (1996); Primary Colors (1998); ...; Birds, The (1963)

Candidate Items: Last Dance (1996); Remains of the Day, The (1993); Assassins (1995); ...; Fatal Instinct (1993)

Similar User Interaction History: L.A. Confidential (1997); Apt Pupil (1998); Kolya (1996); ...; Star Wars (1977)

Conventional Model Prediction: Remains of the Day, The (1993); Addiction, The (1995); ...; Fugitive, The (1993)

Output: Fugitive, The (1993); Angel Baby (1995); ...; Remains of the Day, The (1993)

Rating Prediction Prompt Example:

Instruction: Predict the rating of a target movie based on the user’s historical movie ratings.

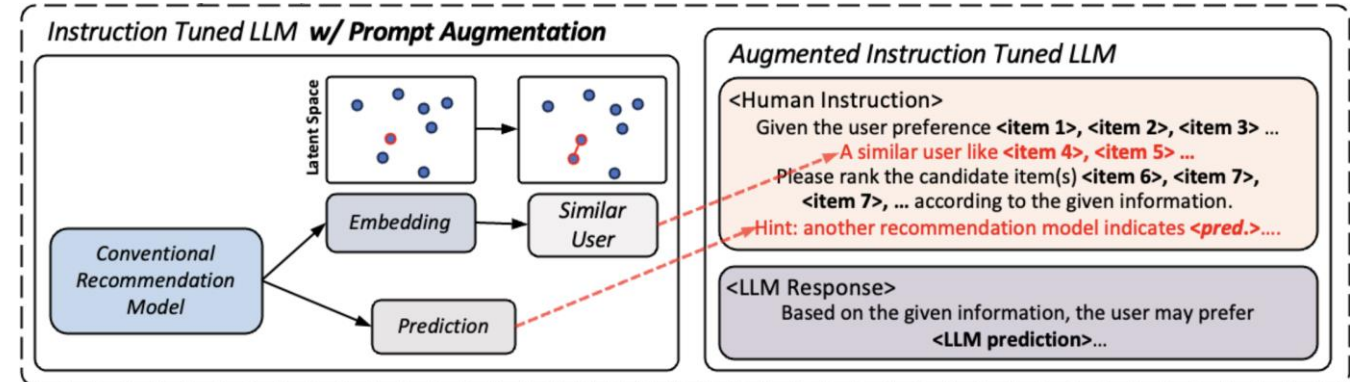
Rating History: Independence Day (1996): 3; Grosse Fatigue (1994): 3; Face/Off (1997): 4; ...; Shall We Dance? (1996): 3

Candidate Item: Pink Floyd - The Wall (1982)

Similar User Rating History: L.A. Confidential (1997): 3; Apt Pupil (1998): 4; ...; English Patient, The (1996): 3

Conventional Model Prediction: 3.2

Output: 3



- Information from Similar User:**
 - Regarding the interacted items of similar users as collaborative information to enrich the prompt for LLM Instruction-tuning.
- Knowledge from Conventional Rec Model:**
 - Combining the prediction results from conventional models with the prompt.

Decoder-only: LlamaRec

- Moreover, for 'Q3: How to aggregate the LLM-based method and conventional methods properly?', LlamaRec introduce the **Adaptive Aggregation**.

- Long-tail coefficient:

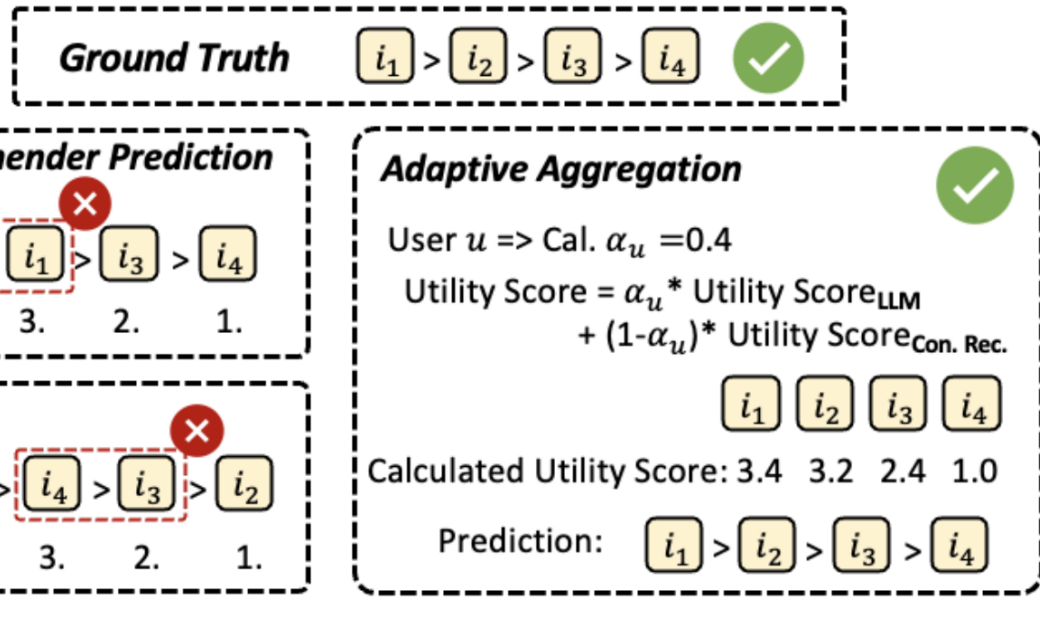
$$\ell_u = \log(N(u) + 1)$$

- The Utility Score of **Rating Prediction** task can be formulize as:

$$\alpha_u = \max\left(\frac{\ell_{max} - \ell_u}{\ell_{max} - \ell_{min}}, \alpha_2\right) \cdot \alpha_1$$

$$U_u = \alpha_u U_{LLM} + (1 - \alpha_u) U_{Rec}$$

For the **top- k recommendation** task, the LLM is employed to rerank the item list and the **Utility Score** can be calculated as the same.



Decoder-only: LlamaRec



August 9-13, 2026
Jeju, Korea

- For the experimental results, LlamaRec achieves optimal results under **three public datasets** and **three recommendation tasks**.

Backbone	Method	ML-100K				ML-1M				BookCrossing			
		H@3 ↑	N@3 ↑	H@5 ↑	N@5 ↑	H@3 ↑	N@3 ↑	H@5 ↑	N@5 ↑	H@3 ↑	N@3 ↑	H@5 ↑	N@5 ↑
MF	Base	0.0455	0.0325	0.0690	0.0420	0.0255	0.0187	0.0403	0.0248	0.0294	0.0227	0.0394	0.0269
	IFT	0.0546	0.0388	0.0790	0.0488	0.0242	0.0175	0.0410	0.0244	0.0247	0.0177	0.0377	0.0230
	Llama4Rec	0.0645*	0.0474*	0.0919*	0.0588*	0.0281*	0.0203*	0.0433*	0.0265*	0.0365*	0.0284*	0.0462*	0.0324*
	Impro.	18.13%	22.16%	16.33%	20.49%	10.20%	8.56%	5.61%	6.85%	24.15%	25.55%	17.26%	20.45%
LightGCN	Base	0.0492	0.0343	0.0744	0.0447	0.0283	0.0203	0.0432	0.0264	0.0358	0.0272	0.0480	0.0322
	IFT	0.0537	0.0381	0.0846	0.0507	0.0268	0.0193	0.0441	0.0263	0.0287	0.0202	0.0448	0.0268
	Llama4Rec	0.0647*	0.0476*	0.0967*	0.0608*	0.0304*	0.0222*	0.0461*	0.0286*	0.0434*	0.0338*	0.057*	0.0394*
	Impro.	20.48%	24.93%	14.30%	19.92%	7.42%	9.36%	4.54%	8.33%	21.23%	24.26%	18.75%	22.36%
MixGCF	Base	0.0526	0.0401	0.0757	0.0496	0.0159	0.0115	0.0238	0.0147	0.0426	0.0330	0.0556	0.0384
	IFT	0.0617	0.0452	0.0906	0.0570	0.0162	0.0114	0.0259	0.0154	0.0337	0.0243	0.0506	0.0312
	Llama4Rec	0.0690*	0.0515*	0.0949*	0.0621*	0.0174*	0.0128*	0.0259	0.0162*	0.0495*	0.0384*	0.0635*	0.0441*
	Impro.	11.83%	13.94%	4.75%	8.95%	7.41%	11.30%	0.00%	5.19%	16.20%	16.36%	14.21%	14.84%
SGL	Base	0.0505	0.0380	0.0729	0.0472	0.0284	0.0206	0.0434	0.0267	0.0419	0.0319	0.0566	0.0380
	IFT	0.0520	0.0392	0.0792	0.0503	0.0275	0.0202	0.0438	0.0269	0.0326	0.0237	0.0499	0.0307
	Llama4Rec	0.0632*	0.0479*	0.0917*	0.0596*	0.0308*	0.0224*	0.0480*	0.0294*	0.0501*	0.0393*	0.0634*	0.0448*
	Impro.	21.54%	22.19%	15.78%	18.49%	8.45%	8.74%	9.59%	9.29%	19.57%	23.20%	12.01%	17.89%

Overall Performance achieved by Direct Recommendation Models

Decoder-only: LlamaRec



August 9-13, 2026
Jeju, Korea

- For the experimental results, LlamaRec achieves optimal results under **three public datasets** and **three recommendation tasks**.

Backbone	Method	ML-100K				ML-1M				BookCrossing			
		H@3 ↑	N@3 ↑	H@5 ↑	N@5 ↑	H@3 ↑	N@3 ↑	H@5 ↑	N@5 ↑	H@3 ↑	N@3 ↑	H@5 ↑	N@5 ↑
SASRec	Base	0.0187	0.0125	0.0385	0.0205	0.0277	0.0165	0.0502	0.0257	0.0086	0.0049	0.0163	0.0081
	IFT	0.0204	0.0136	0.0379	0.0207	0.0241	0.0159	0.0473	0.0254	0.0124	0.0086	0.0185	0.0111
	Llama4Rec	0.0238*	0.0155*	0.0449*	0.0240*	0.0293*	0.0201*	0.0504	0.0287*	0.0142*	0.0098*	0.0227*	0.0131*
	Impro.	16.67%	13.97%	16.62%	15.94%	5.78%	21.82%	0.40%	11.67%	14.52%	13.95%	22.70%	18.02%
BERT4Rec	Base	0.0153	0.0104	0.0294	0.0161	0.0107	0.0069	0.0211	0.0112	0.0088	0.0058	0.0161	0.0088
	IFT	0.0174	0.0119	0.0326	0.0100	0.0106	0.0071	0.0188	0.0104	0.0127	0.0092	0.0180	0.0113
	Llama4Rec	0.0198*	0.0134*	0.0332	0.0189*	0.0115*	0.0078*	0.0206	0.0115*	0.0154*	0.0108*	0.023*	0.0139*
	Impro.	13.79%	12.61%	1.84%	17.39%	7.48%	9.86%	-2.37%	2.68%	21.26%	17.39%	27.78%	23.01%
CL4SRec	Base	0.0243	0.0143	0.0436	0.0222	0.0259	0.0153	0.0492	0.0248	0.0083	0.0048	0.0165	0.0082
	IFT	0.0230	0.0149	0.0428	0.0230	0.0234	0.0155	0.0447	0.0241	0.0102	0.0071	0.0177	0.0102
	Llama4Rec	0.0255*	0.0182*	0.0440	0.0255*	0.0278*	0.0185*	0.0482	0.0268*	0.0138*	0.0093*	0.0220*	0.0127*
	Impro.	4.94%	22.15%	0.92%	10.87%	7.34%	19.35%	-2.03%	8.06%	35.29%	30.99%	24.29%	24.51%

Overall Performance achieved by Sequential Recommendation Models

Decoder-only: LlamaRec



August 9–13, 2026
Jeju, Korea

For the experimental results, LlamaRec achieves optimal results under **three public datasets** and **three recommendation tasks**.

Backbone	Method	ML-100K		ML-1M		BookCrossing	
		RMSE ↓	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓	MAE ↓
LLaMA	IFT	1.2792	0.8940	1.2302	0.8770	2.0152	1.3782
DeepFM	Base	1.0487	0.8082	0.9455	0.7409	1.7738	1.3554
	Llama4Rec	1.0306*	0.7987*	0.9360*	0.7321*	1.6958*	1.2843*
	Impro.	1.73%	1.18%	1.00%	1.19%	4.40%	5.25%
NFM	Base	1.0284	0.8005	0.9438	0.7364	2.121	1.5984
	Llama4Rec	1.0189*	0.7961	0.9369*	0.7303*	1.9253*	1.4473*
	Impro.	0.92%	0.55%	0.73%	0.83%	9.23%	9.45%
DCN	Base	1.0478	0.8063	0.9426	0.7342	2.0216	1.4622
	Llama4Rec	1.0367*	0.8033	0.9345*	0.7272*	1.8518*	1.3566*
	Impro.	1.06%	0.37%	0.86%	0.95%	8.40%	7.22%
AFM	Base	1.0471	0.8035	0.9508	0.7464	1.6516	1.2614
	Llama4Rec	1.0340*	0.7996	0.9426*	0.7394*	1.6244*	1.2259*
	Impro.	1.25%	0.49%	0.86%	0.94%	1.65%	2.81%
xDeepFM	Base	1.1472	0.8836	0.9519	0.7428	2.1756	1.6461
	Llama4Rec	1.0947*	0.8483*	0.9401*	0.7336*	1.9610*	1.4833*
	Impro.	4.58%	4.00%	1.24%	1.24%	9.86%	9.89%
AutoInt	Base	1.0500	0.8120	0.9471	0.7404	1.9148	1.4501
	Llama4Rec	1.0369*	0.8059*	0.9382*	0.7326*	1.7917*	1.3492*
	Impro.	1.25%	0.75%	0.94%	1.05%	6.43%	6.96%

Overall Performance achieved by Rating Prediction Models

▪ LLMRec Bottleneck

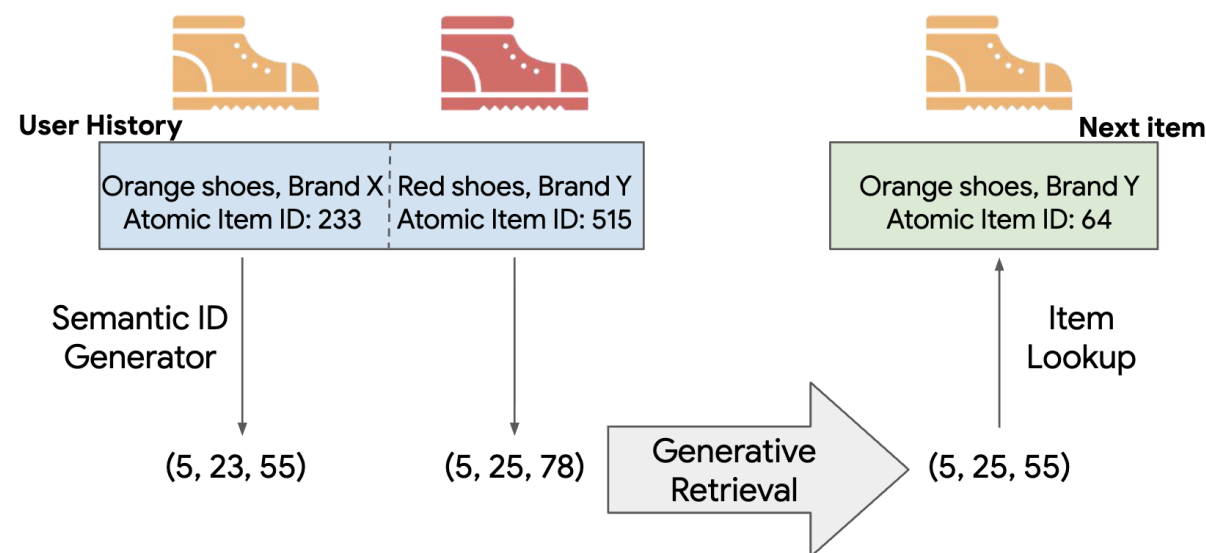
- Free-form outputs are semantically rich but require catalog grounding.

▪ SID Reformulation

- Represent each item as a discrete semantic token sequence.
- Predict the next item directly in a catalog-native output space.

▪ What changes

- Valid decoding and catalog-scale generation become easier.
- Representation quality becomes the new bottleneck.



Decoder-only: OneRec-v2

- **Background**

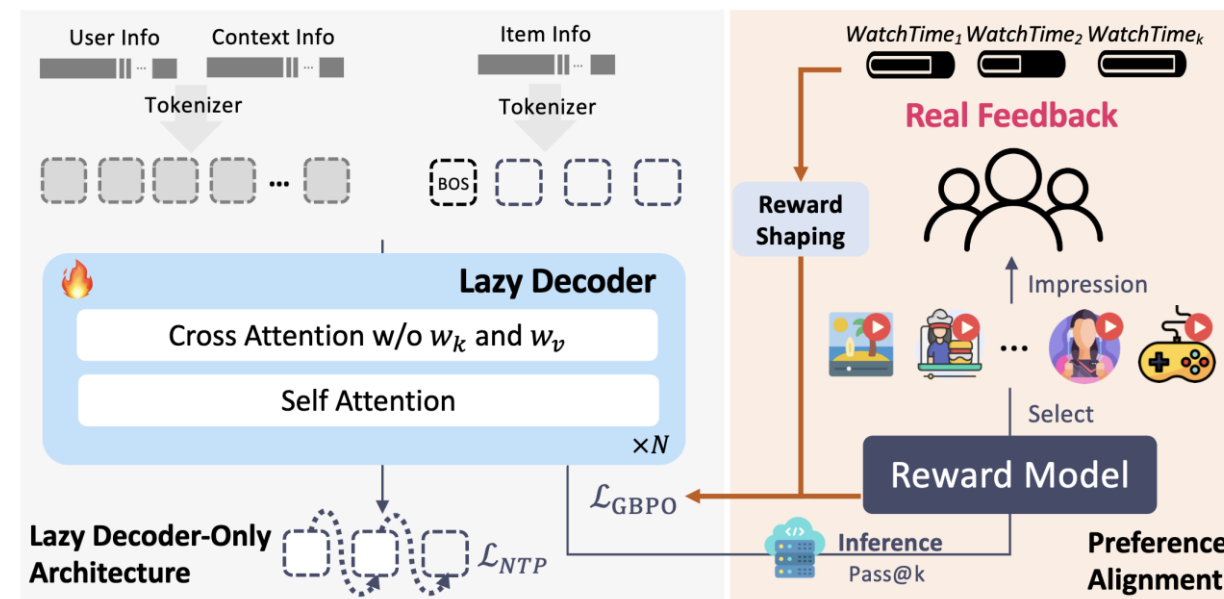
- OneRec introduces a GR framework
- Its encoder–decoder architecture faces scalability limitations
- Limited model capacity leads to suboptimal performance

- **Motivation**

- Move beyond OneRec’s encoder–decoder design
- Embrace a decoder-only architecture for scalable autoregressive generation

- **Contributions**

- Enable effective scaling with a lazy decoder-only architecture



Context Processor

- Integrating heterogeneous and multi-modal user behavioral signals.

$$d_{\text{context}} = S_{\text{kv}} \cdot L_{\text{kv}} \cdot G_{\text{kv}} \cdot d_{\text{head}},$$

$$\text{Context} = [\mathbf{C}_0, \mathbf{C}_1, \dots, \mathbf{C}_{S_{\text{kv}} \cdot L_{\text{kv}} - 1}],$$

Lazy Decoder Block

- Tokenizer: Utilizing the first 2 IDs and prepend a <BOS> to prepare the input

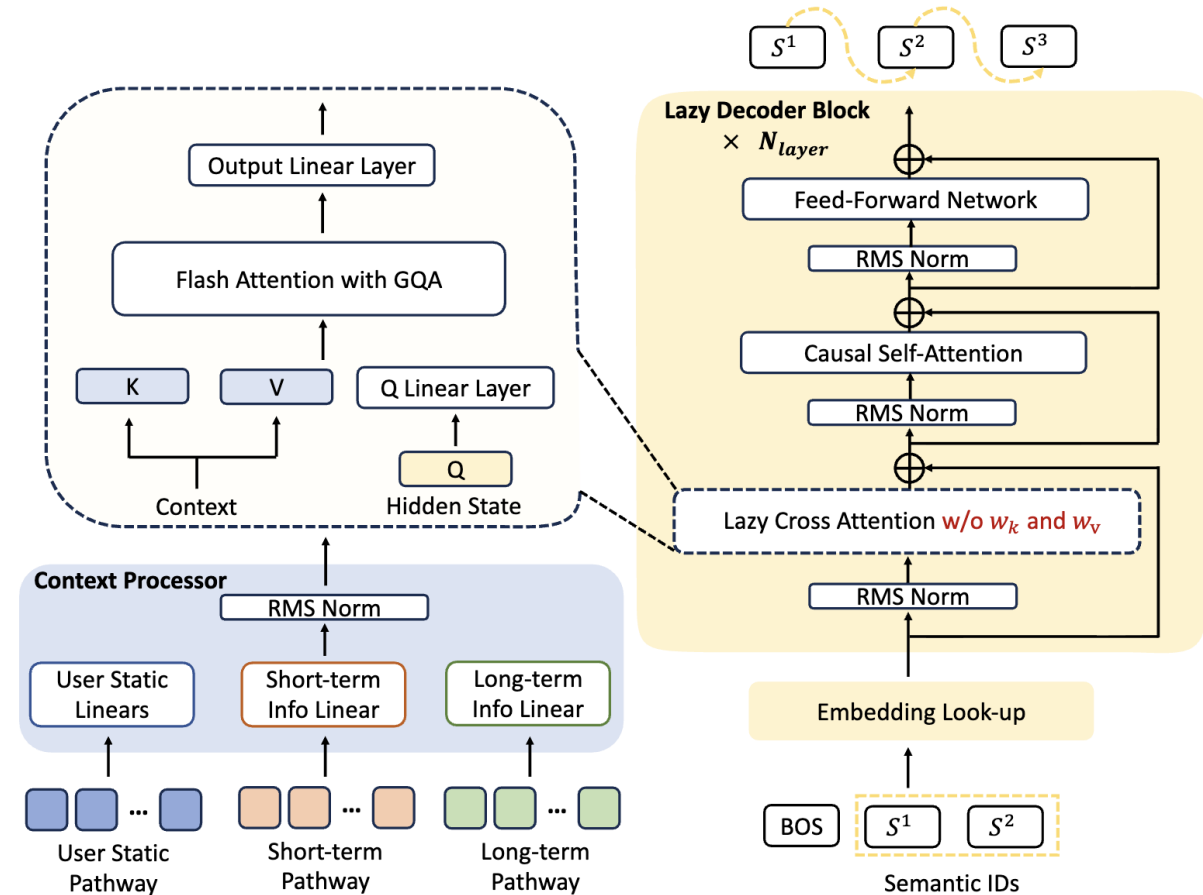
$$\mathbf{h}^{(0)} = \text{Embed}([\text{BOS}, s^1, s^2]) \in \mathbb{R}^{3 \times d_{\text{model}}}.$$

- Block Structure:

$$\mathbf{h}_{\text{cross}}^{(l)} = \mathbf{h}^{(l-1)} + \text{CrossAttn} \left(\text{RMSNorm}(\mathbf{h}^{(l-1)}), \mathbf{k}_{\text{kv}}, \mathbf{v}_{\text{kv}} \right),$$

$$\mathbf{h}_{\text{self}}^{(l)} = \mathbf{h}_{\text{cross}}^{(l)} + \text{SelfAttn} \left(\text{RMSNorm}(\mathbf{h}_{\text{cross}}^{(l)}) \right),$$

$$\mathbf{h}^{(l)} = \mathbf{h}_{\text{self}}^{(l)} + \text{FFN}^{(l)} \left(\text{RMSNorm}(\mathbf{h}_{\text{self}}^{(l)}) \right),$$



- **Background**

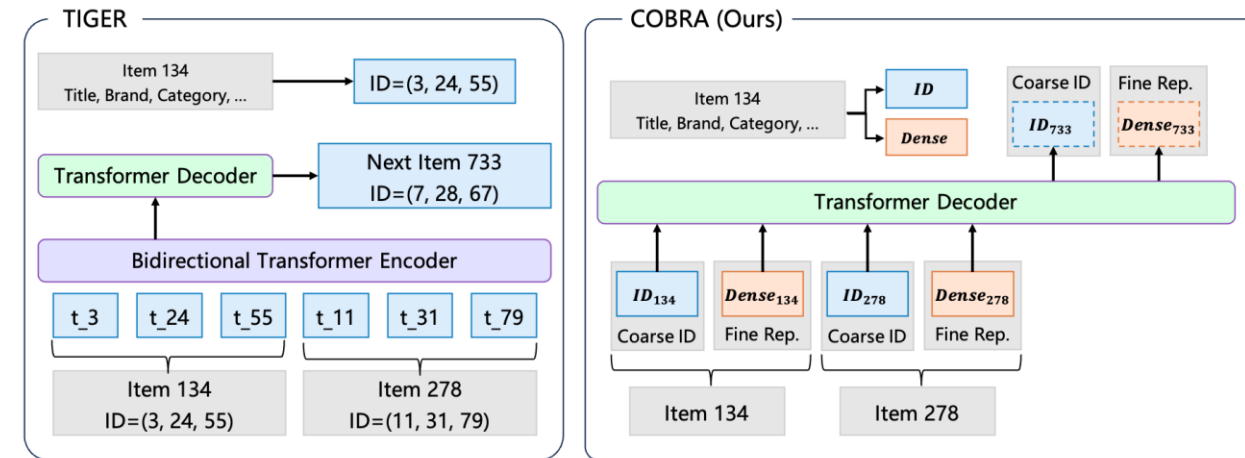
- Generative retrieval needs item representations that are both scalable and discriminative.

- **Motivation**

- Sparse SIDs retrieve efficiently but lose fine-grained matching signals.

- **Contributions**

- Couples sparse ID generation, dense-vector refinement, and BeamFusion in one coarse-to-fine pipeline.



Decoder-only: COBRA

- Cascaded representation**

- Pair every sparse ID with a dense item vector.

$$P(ID_{t+1}, \mathbf{v}_{t+1} | S_{1:t}) = P(ID_{t+1} | S_{1:t}) P(\mathbf{v}_{t+1} | ID_{t+1}, S_{1:t})$$

$$\mathbf{h}_t = [\mathbf{e}_t; \mathbf{v}_t]$$

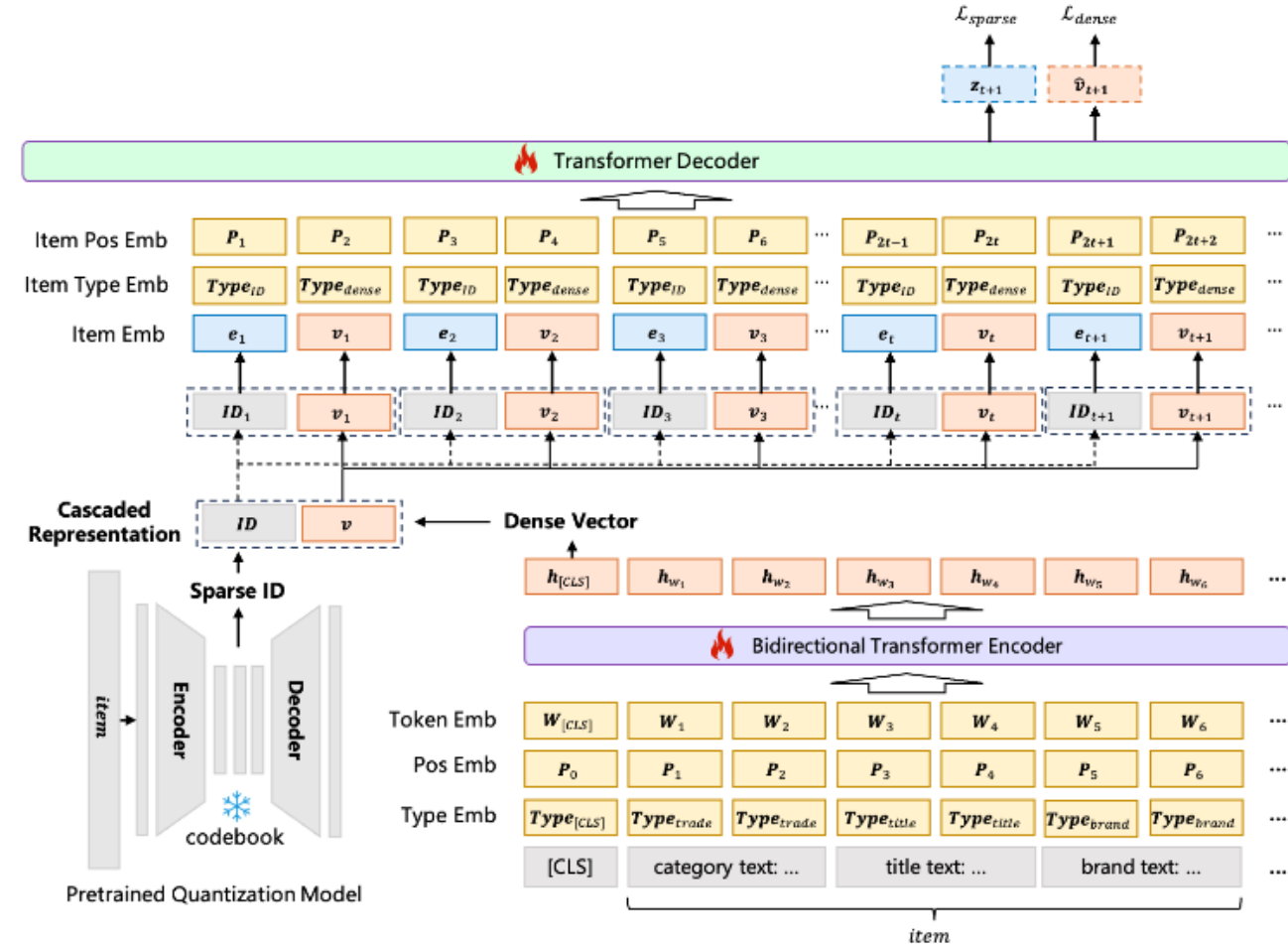
- Joint objective**

- Optimize sparse-token generation and dense-vector prediction together.

$$\mathcal{L}_{\text{sparse}} = - \sum_{t=1}^{T-1} \log \left(\frac{\exp(z_{t+1}^{ID_{t+1}})}{\sum_{j=1}^C \exp(z_{t+1}^j)} \right) \quad \mathcal{L}_{\text{dense}} = - \sum_{t=1}^{T-1} \log \frac{\exp(\cos(\hat{\mathbf{v}}_{t+1} \cdot \mathbf{v}_{t+1}))}{\sum_{item_j \in \text{Batch}} \exp(\cos(\hat{\mathbf{v}}_{t+1}, \mathbf{v}_{item_j}))}$$

- BeamFusion**

- Combine beam probability and ANN similarity at inference.



Decoder-only: COBRA

- **Sparse generation**

- Decode coarse semantic IDs to narrow the catalog.

$$\{\hat{\text{ID}}_{T+1}^k\}_{k=1}^M = \text{BeamSearch}(\text{TransformerDecoder}(S_{1:T}), M)$$

- **Dense refinement**

- Predict dense vectors and search within candidate partitions.

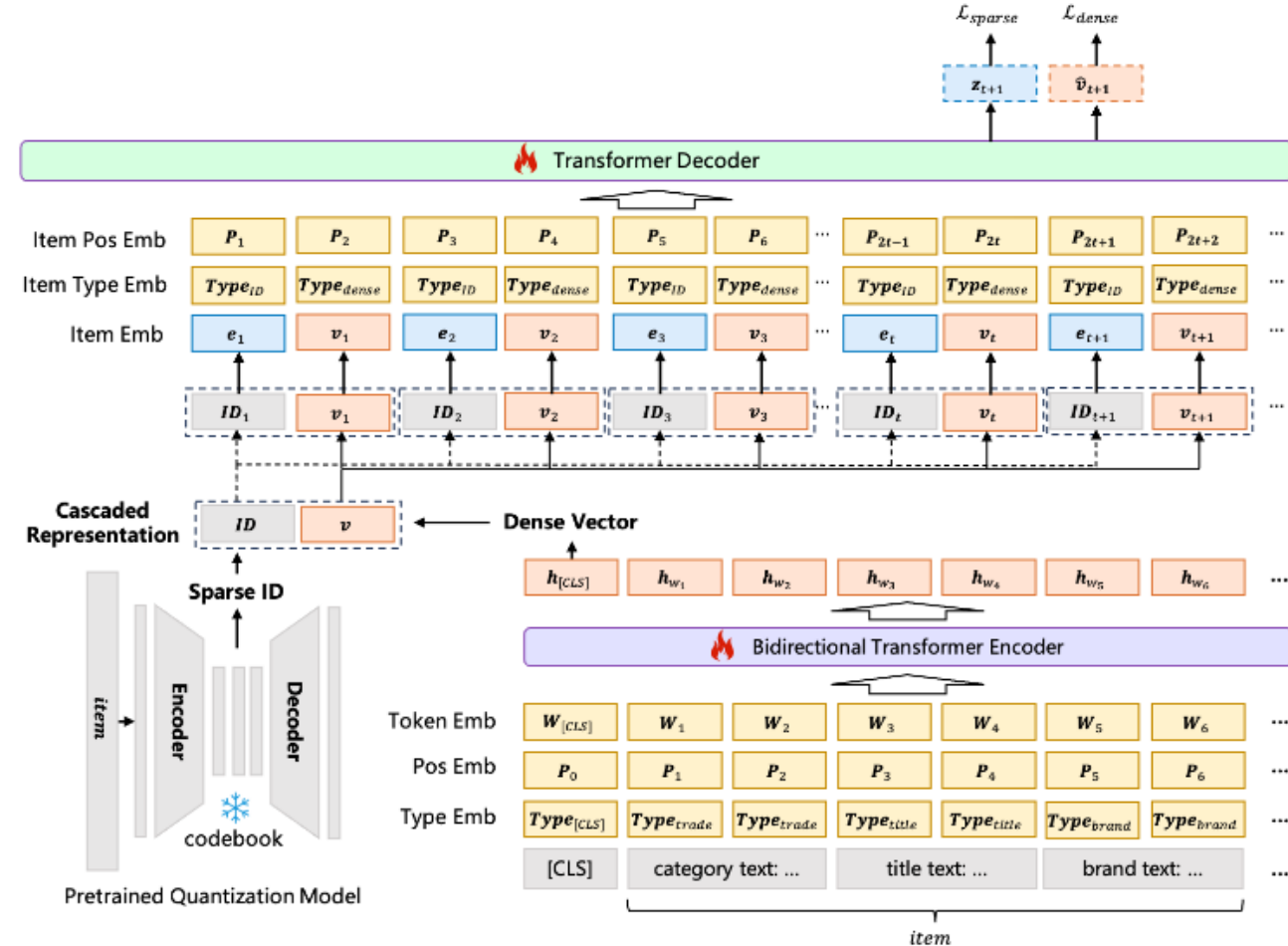
$$\hat{v}_{T+1}^k = \text{TransformerDecoder}([S_{1:T}, \text{Embed}(\hat{\text{ID}}_{T+1}^k)])$$

$$\mathcal{A}_k = \text{ANN}(\hat{\text{ID}}_{T+1}^k, C(\hat{\text{ID}}_{T+1}^k), N)$$

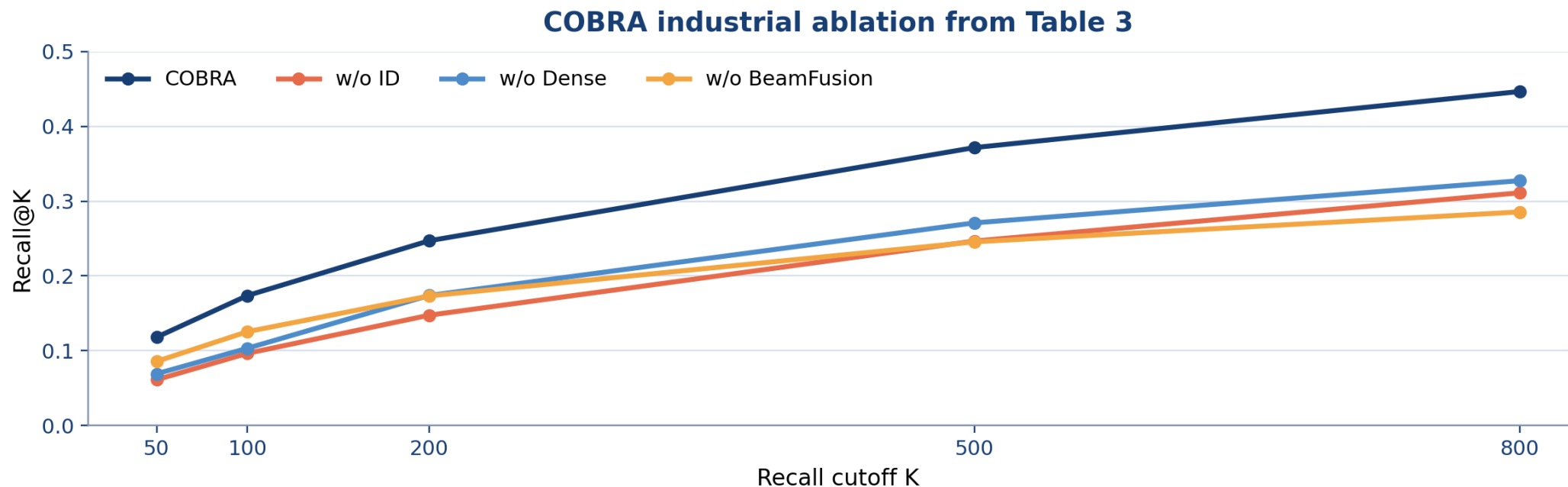
- **Output**

- Combine generative and nearest-neighbor evidence for final Top-K.

$$\mathcal{R} = \text{TopK}\left(\bigcup_{k=1}^M \mathcal{A}_k, \Phi, K\right)$$



- Industrial ablation removes sparse IDs, dense vectors, or BeamFusion.



- Main signal:**
 - All three components improve Recall; the gain adds ANN-search and score-fusion cost at serving time.

Decoder-only: RecGPT

• Background

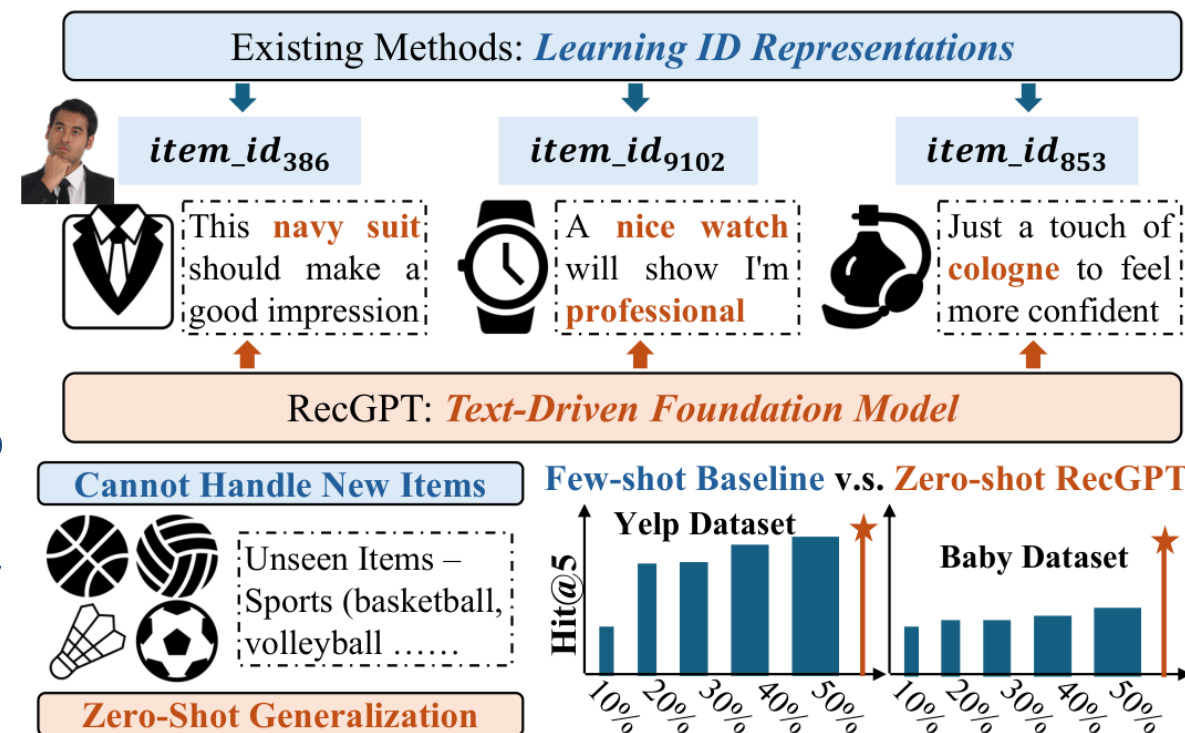
- Common RS struggles to perform cross-domain recommendation.
- A foundation RS needs a shared item representation space.

• Motivation

- Item textual semantics are hard to grounded back to valid catalog.
- Item representations need item-internal consistency and sequence-level causality.

• Contributions

- Text-driven item tokenization with Finite Scalar Quantization.
- Bidirectional-causal attention and Trie-constrained catalog-aware beam search.



Decoder-only: RecGPT

- Unified Item Tokenization:

- Representation:

- Objective for text encoder:

$$\mathbb{E}_{z \in \mathcal{Z}_n} \sum_{t=c+1}^n \log P(x_{z_t} | x_{z_{<t}}, \Phi_{z_{>c}}; \theta),$$

- Item embeddings:

$$e_i = \text{MPNet}(X_i), e_i \in \mathbb{R}^{d_L}$$

- Quantization:

- Finite Scalar Quantization:

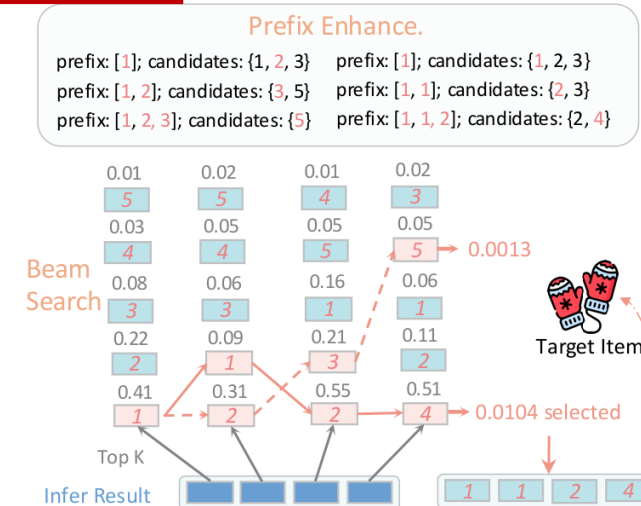
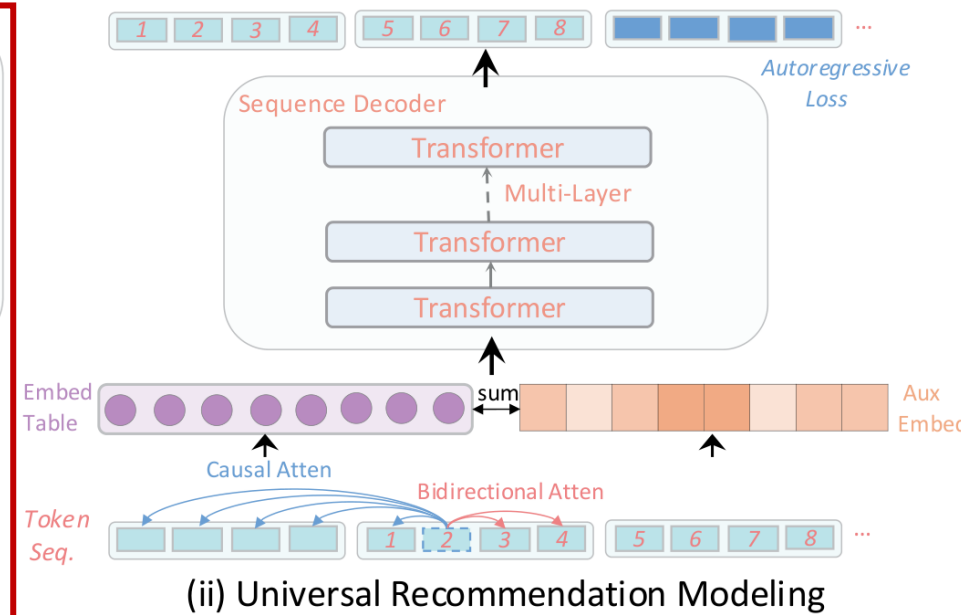
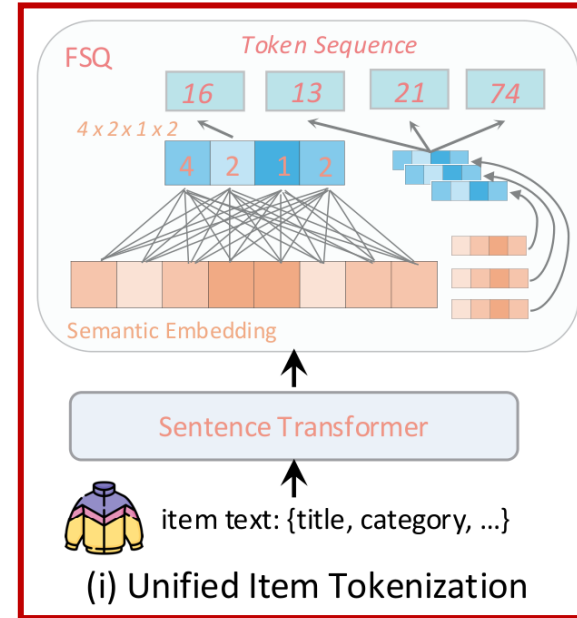
$$\text{FSQ}(e_i^k) = (L-1)\sigma(\mathcal{T}_{in}(e_i^k)) + \text{sg}[R[(L-1)\sigma(\mathcal{T}_{in}(e_i^k))]] - (L-1)\sigma(\mathcal{T}_{in}(e_i^k))$$

- Optimization:

- Reconstruction optimization:

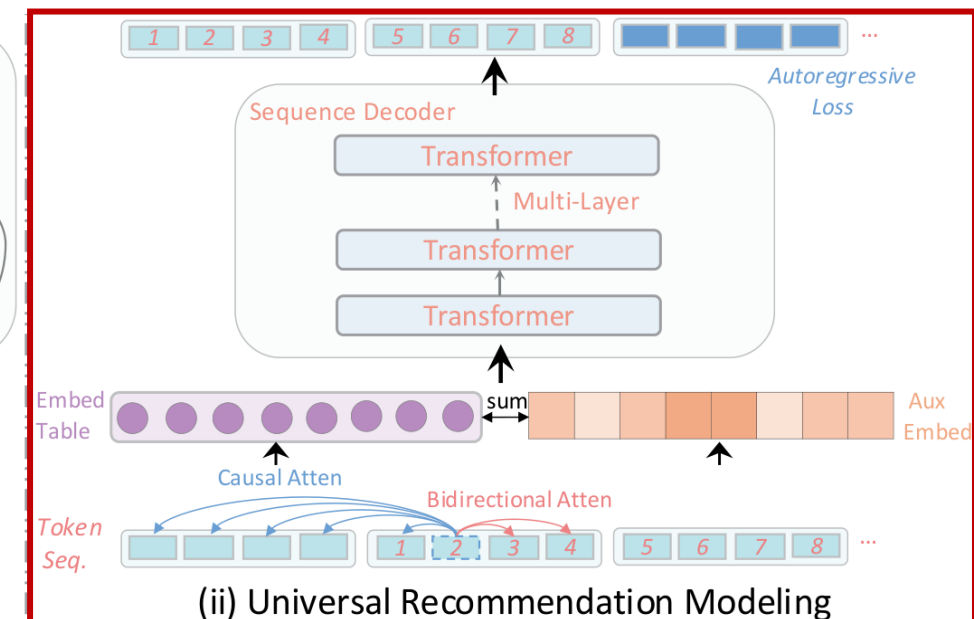
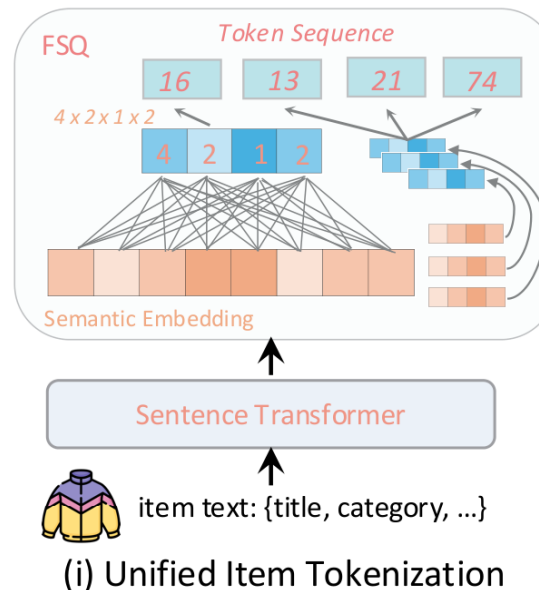
$$\mathcal{L}_{fsq} = \sum_{i=0}^{N-1} \|e_i - \text{Decoder}[\mathcal{T}_{out}(\hat{e}_i^0), \dots, \mathcal{T}_{out}(\hat{e}_i^{K-1})]\|_1$$

Jiang et al. "RecGPT: A Foundation Model for Sequential Recommendation." EMNLP 2025.

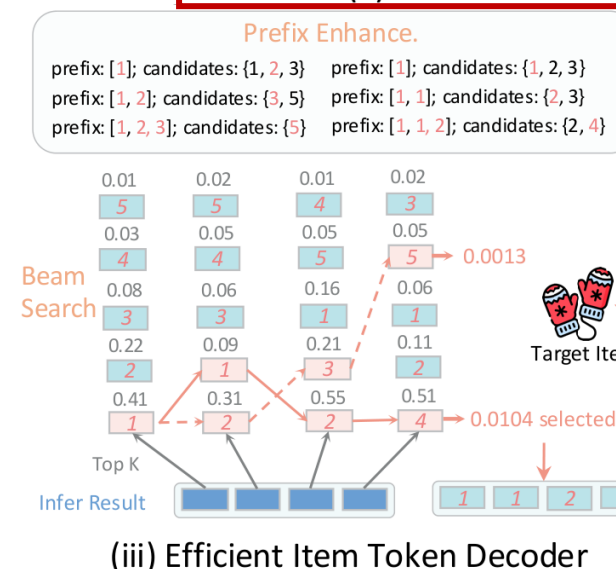


Decoder-only: RecGPT

- Unified Item Tokenization:
 - Mapping semantic embeddings to fixed-length semantic tokens.
- Universal Recommendation:
 - Bidirectional Attention for Item Tokens
 - Auxiliary Item Semantic Features:

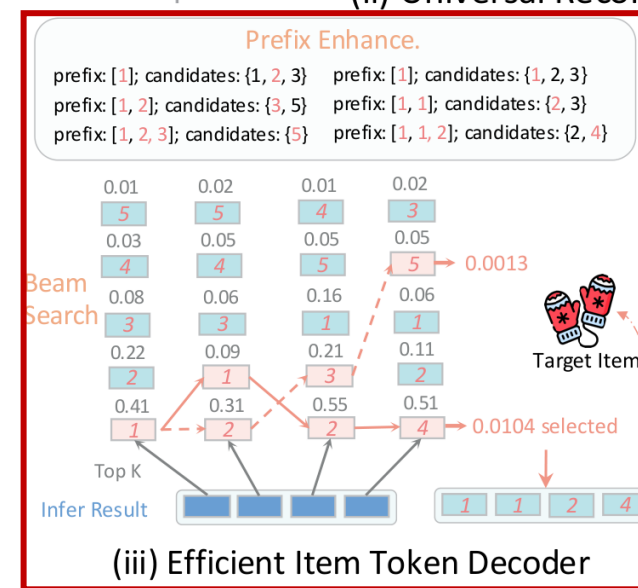
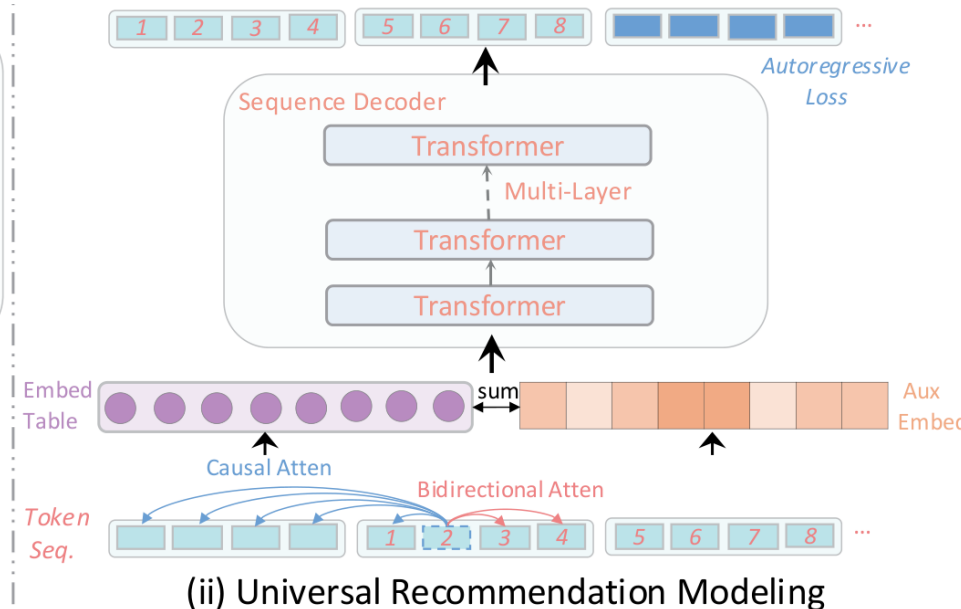
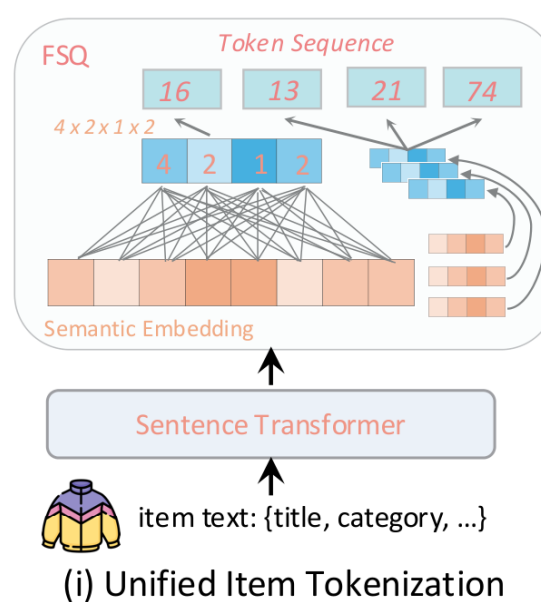


$$X = \text{LNorm}(E_{aux}) + \text{LNorm}(E_{wte}) + E_{wpe}$$



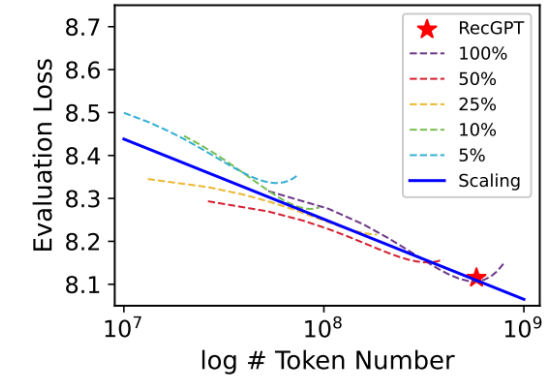
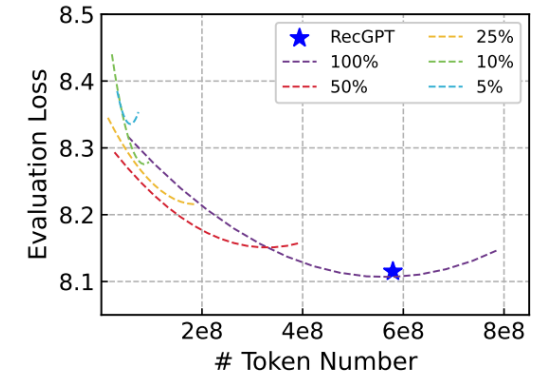
Decoder-only: RecGPT

- Unified Item Tokenization:
 - Mapping semantic embeddings to fixed-length semantic tokens.
- Universal Recommendation:
 - Giving the next item token sequence with a decoder-only backbone.
- Efficient Token Decoder:
 - Efficient Next-item Prediction
 - Prefix Enhancement



Decoder-only: RecGPT

Data	Metric	GRU4Rec		GRU4RecF		Caser		BERT4Rec		FDSA		CL4SRec		DuoRec		ICLRec		MAERec		RecGPT	
		@3	@5	@3	@5	@3	@5	@3	@5	@3	@5	@3	@5	@3	@5	@3	@5	@3	@5	@3	@5
Cross-Domain in Amazon Dataset																					
Baby	Hit@1	.0019		.0019		.0025		.0024		.0025		.0024		.0025		.0010		.0016		.0273*	
	Hit	.0051	.0090	.0060	.0091	.0062	.0099	.0061	.0096	.0064	.0099	.0060	.0093	.0061	.0103	.0046	.0082	.0055	.0090	.0281*	.0283*
	NDCG	.0036	.0052	.0043	.0056	.0046	.0061	.0045	.0060	.0047	.0061	.0045	.0059	.0046	.0063	.0031	.0045	.0038	.0052	.0277*	.0279*
Games	Hit@1	.0031		.0032		.0032		.0032		.0026		.0045		.0031		.0026		.0018		.0364*	
	Hit	.0079	.0133	.0093	.0133	.0093	.0139	.0082	.0135	.0084	.0135	.0112	.0163	.0078	.0117	.0060	.0107	.0078	.0119	.0374*	.0376*
	NDCG	.0058	.0080	.0066	.0083	.0066	.0084	.0061	.0083	.0059	.0080	.0082	.0103	.0058	.0074	.0046	.0065	.0052	.0069	.0370*	.0371*
Office	Hit@1	.0021		.0019		.0021		.0021		.0021		.0021		.0022		.0021		.0020		.0280*	
	Hit	.0042	.0059	.0043	.0053	.0040	.0059	.0031	.0049	.0042	.0061	.0043	.0058	.0040	.0056	.0038	.0049	.0036	.0047	.0293*	.0299*
	NDCG	.0033	.0040	.0033	.0037	.0032	.0040	.0027	.0034	.0033	.0041	.0034	.0040	.0032	.0039	.0031	.0035	.0029	.0034	.0287*	.0290*
Cross-Domain on Different Platforms																					
Yelp	Hit@1	.0011		.0008		.0007		.0013		.0009		.0021		.0019		.0003		.0029		.0161*	
	Hit	.0023	.0036	.0022	.0039	.0026	.0040	.0037	.0054	.0025	.0038	.0056	.0083	.0043	.0065	.0027	.0045	.0079	.0119	.0163*	.0166*
	NDCG	.0018	.0023	.0016	.0023	.0018	.0024	.0026	.0033	.0018	.0023	.0040	.0051	.0033	.0042	.0016	.0024	.0057	.0074	.0162*	.0163*
Washington	Hit@1	.0012		.0016		.0010		.0027		.0016		.0032		.0024		.0017		.0041		.0122*	
	Hit	.0030	.0043	.0037	.0056	.0033	.0046	.0073	.0114	.0038	.0053	.0085	.0127	.0067	.0105	.0043	.0069	.0103	.0154	.0128*	.0130
	NDCG	.0022	.0028	.0028	.0036	.0023	.0028	.0053	.0070	.0029	.0035	.0063	.0079	.0049	.0064	.0032	.0042	.0076	.0097	.0126*	.0127*
Steam	Hit@1	.0396		.0484		.0121		.0757		.1022		.0681		.1011		.0014		.0727		.1237*	
	Hit	.0675	.0830	.0737	.0881	.0350	.0538	.1002	.1161	.1231	.1366	.0986	.1177	.1223	.1351	.0433	.0678	.1018	.1172	.1246*	.1253
	NDCG	.0557	.0621	.0630	.0689	.0254	.0331	.0898	.0964	.1143	.1232	.0858	.0936	.1134	.1187	.0257	.0357	.0897	.0960	.1242*	.1245*



• Main signal:

- RecGPT is zero-shot, yet beats or matches few-shot baselines across several target domains; FSQ is the key ablation anchor.

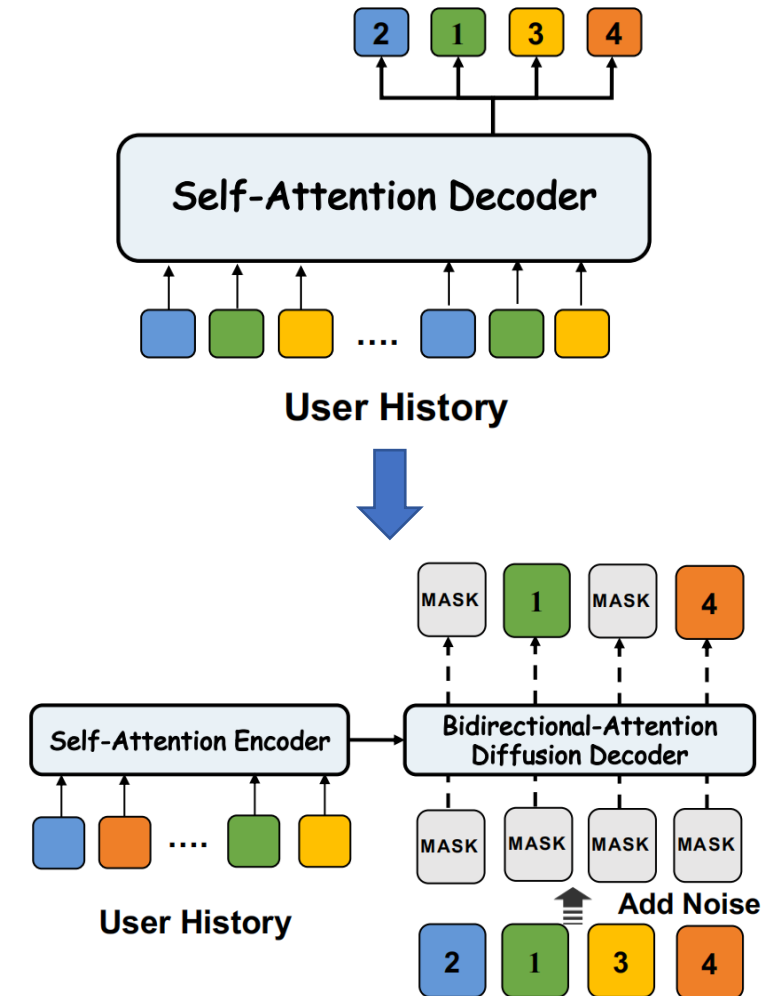
Axis	DiffGRM	MDGR
Tokenizer	PSE balanced digits	OPQ parallel codebook
Noising	On-policy coherent noising	Curriculum + history-aware masks
Denoising	Confidence-guided CPD	Warm-up anchors + parallel decoding
Lesson	Mask hard digits	Tune quality vs QPS

- **Shared question:**

- Which SID positions should be hidden, which should be restored first, and how do we keep Top-K generation valid?

Why Diffusion-Based Models?

- Semantic IDs are structured objects: **all digits jointly identify the same item.**
- Autoregressive decoding creates an **left-to-right information barrier.**
- Early SID **mistakes propagate** because later steps condition on an incorrect prefix.
- Masked denoising lets the model revisit uncertain positions and fill tokens in an adaptive order.



• Background

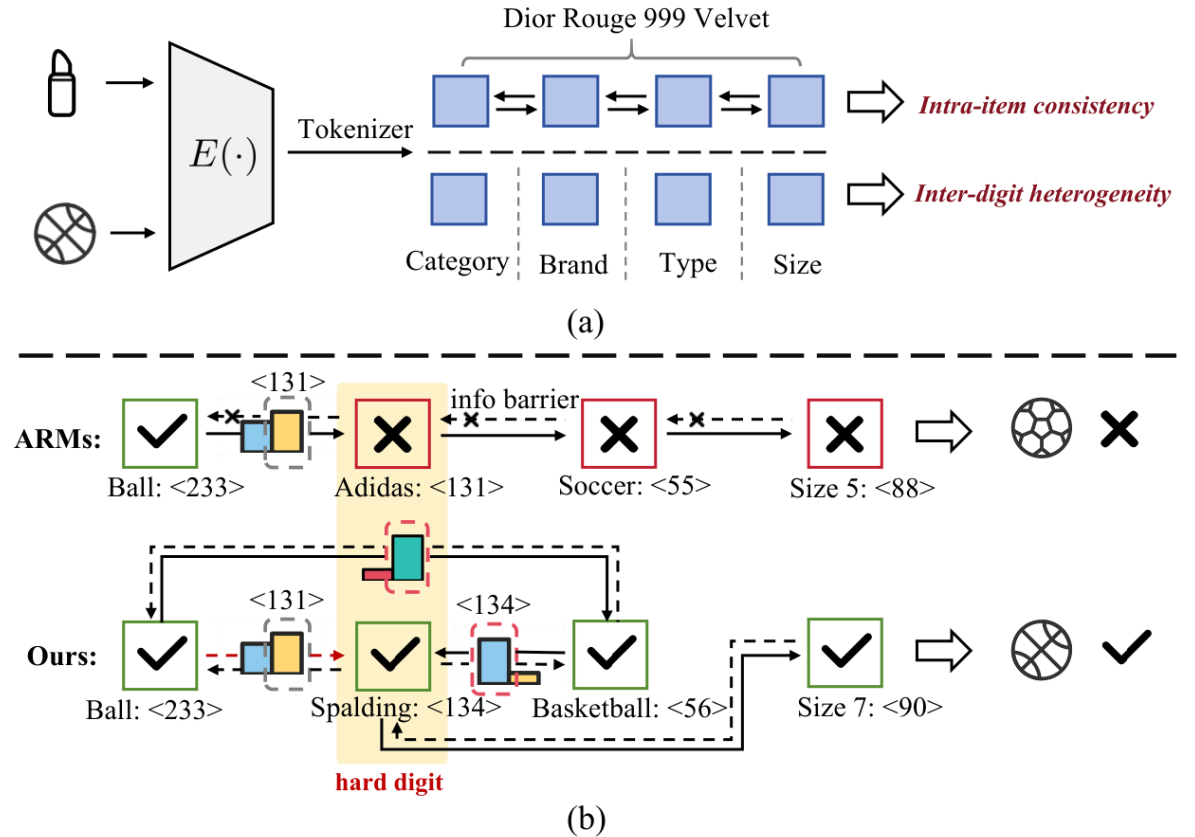
- GR represents each item as Semantic ID.
- SID is naturally structured with different semantic granularity.

• Motivation

- Autoregressive generation blocks later digits from helping earlier decisions.
- Uniform next-token loss ignores heterogeneous digit difficulty.

• Contributions

- Parallel Semantic Encoding and Confidence-guided Parallel Denoising.
- Masked-diffusion decoder for SID reconstruction.



- Parallel Semantic Encoding:**

- Leveraging OPQ-based partition-and-quantize scheme.

- On-policy Coherent Noising:**

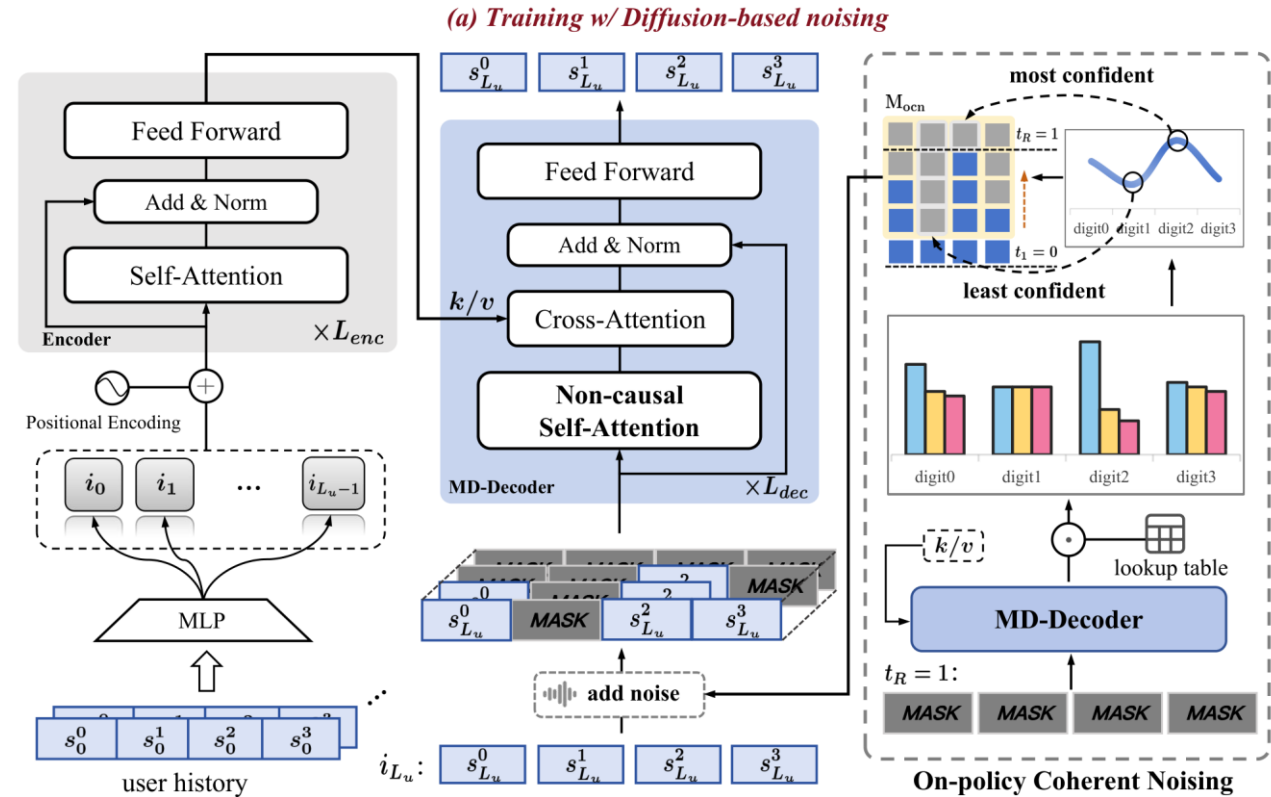
- Probe difficulty: Fully-masking and measuring each digit's uncertainty.

$$p_{\max}^{(k)} = \max_{v \in \{0, \dots, M-1\}} p_{\theta}^{(R,k)}(v), \quad \delta^{(k)} = 1 - p_{\max}^{(k)}$$

- Rank and coherently mask: Sorting digits from hard to easy, then masks the top hardest digits.

$$\mathbf{y}^{(r,k)} = \begin{cases} \mathbf{E}_{\text{mask}}[k], & \text{if } k \in \mathcal{M}^{(r)}, \\ \mathbf{E}_{\text{sid}}^{(k)}[s^k], & \text{if } k \notin \mathcal{M}^{(r)}, \end{cases} \quad \mathcal{M}^{(r)} = \{\sigma(1), \dots, \sigma(m_r)\},$$

- Train:** $\mathcal{L}^{(r)} = \frac{1}{|\mathcal{M}^{(r)}|} \sum_{k \in \mathcal{M}^{(r)}} \left(- \sum_{v=0}^{M-1} \tilde{q}_v^{(k)} \log p_{\theta}^{(r,k)}(v | \mathbf{y}^{(r)}, \mathbf{H}_u) \right) \quad \mathcal{L} = \frac{1}{R} \sum_{r=1}^R \mathcal{L}^{(r)},$



Confidence-guided Denoising:

- Global Beams Initialization: Choosing the top digit-codeword pairs across all positions

$$\mathcal{B}_R = \text{Top}_{B_{\text{act}}} \left\{ \log p_{\theta}(y_k=c \mid \mathbf{y}^{(R)}, \mathbf{H}_u) \right\},$$

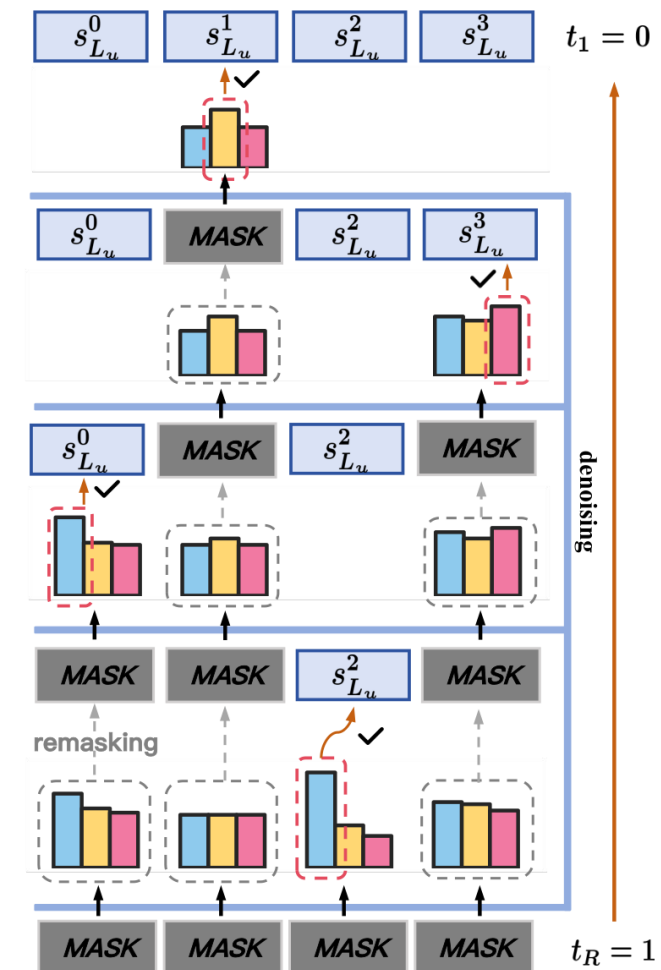
- Expansion: Filling one till-masked digit and accumulating the log-probability

$$s_{r-1}(b, k, c) = \text{score}_r(b) + \log p_{\theta}(y_k=c \mid \mathbf{y}_b^{(r)}, \mathbf{H}_u),$$

- Truncation: Retaining the best branches and deduplicating

$$\mathcal{B}_{r-1} = \text{Top}_{B_{\text{act}}} \left\{ s_{r-1}(b, k, c) \right\},$$

(b) Inference w/ Confidence-guided denoising



Diffusion-based: DiffGRM



August 9–13, 2026
Jeju, Korea

Methods	Sports and Outdoors				Beauty				Toys and Games			
	Recall @5	NDCG @5	Recall @10	NDCG @10	Recall @5	NDCG @5	Recall @10	NDCG @10	Recall @5	NDCG @5	Recall @10	NDCG @10
<i>Item ID-based (Discriminative)</i>												
GRU4Rec	0.0129	0.0086	0.0204	0.0110	0.0164	0.0099	0.0283	0.0137	0.0097	0.0059	0.0176	0.0084
HGN	0.0189	0.0120	0.0313	0.0159	0.0325	0.0206	0.0512	0.0266	0.0321	0.0221	0.0497	0.0277
SASRec	0.0233	0.0154	0.0350	0.0192	0.0387	0.0249	0.0605	0.0318	0.0463	0.0306	0.0675	0.0374
BERT4Rec	0.0115	0.0075	0.0191	0.0099	0.0203	0.0124	0.0347	0.0170	0.0116	0.0071	0.0203	0.0099
<i>Semantic-enhanced (Discriminative)</i>												
FDSA	0.0182	0.0122	0.0288	0.0156	0.0267	0.0163	0.0407	0.0208	0.0228	0.0140	0.0381	0.0189
S ³ -Rec	0.0251	0.0161	0.0385	0.0204	0.0387	0.0244	0.0647	0.0327	0.0443	0.0294	0.0700	0.0376
VQ-Rec	0.0208	0.0144	0.0300	0.0173	0.0457	0.0317	0.0664	0.0383	0.0497	0.0346	0.0737	0.0423
RecJPQ	0.0141	0.0076	0.0220	0.0102	0.0311	0.0167	0.0482	0.0222	0.0331	0.0182	0.0484	0.0231
<i>Semantic ID-based (Generative)</i>												
TIGER	0.0264	0.0181	0.0400	0.0225	0.0454	0.0321	0.0648	0.0384	0.0521	0.0371	0.0712	0.0432
HSTU	0.0258	0.0165	0.0414	0.0215	0.0469	0.0314	0.0704	0.0389	0.0433	0.0281	0.0669	0.0357
ActionPiece	<u>0.0316</u>	0.0205	<u>0.0500</u>	<u>0.0264</u>	0.0511	0.0340	0.0775	0.0424	—	—	—	—
RPG	0.0314	<u>0.0216</u>	0.0463	0.0263	<u>0.0550</u>	<u>0.0381</u>	<u>0.0809</u>	<u>0.0464</u>	<u>0.0592</u>	<u>0.0401</u>	0.0869	<u>0.0490</u>
DiffGRM	0.0363	0.0245	0.0550	0.0305	0.0603	0.0414	0.0876	0.0502	0.0618	0.0455	<u>0.0834</u>	0.0524
Improv.	+14.87%	+13.43%	+10.00%	+15.53%	+9.64%	+8.66%	+8.28%	+8.19%	+4.39%	+13.47%	-4.03%	6.94%

Variants	Sports	Beauty	Toys
<i>Semantic ID Setting</i>			
(1.1) PSE → RQ-Kmeans	0.0200	0.0343	0.0305
(1.2) PSE → Random	0.0138	0.0300	0.0206
<i>Training strategy</i>			
(2.1) w/o OCN	0.0250	0.0368	0.0385
(2.2) w/o On-policy	0.0263	0.0455	0.0430
<i>Inference strategy</i>			
(3.1) w/o CPD	0.0273	0.0496	0.0499
DiffGRM (ours)	0.0305	0.0502	0.0524

- **Main signal:**
 - DiffGRM improves Recall/NDCG across Amazon Reviews; PSE, OCN, and CPD each contribute to NDCG@10.

• Background

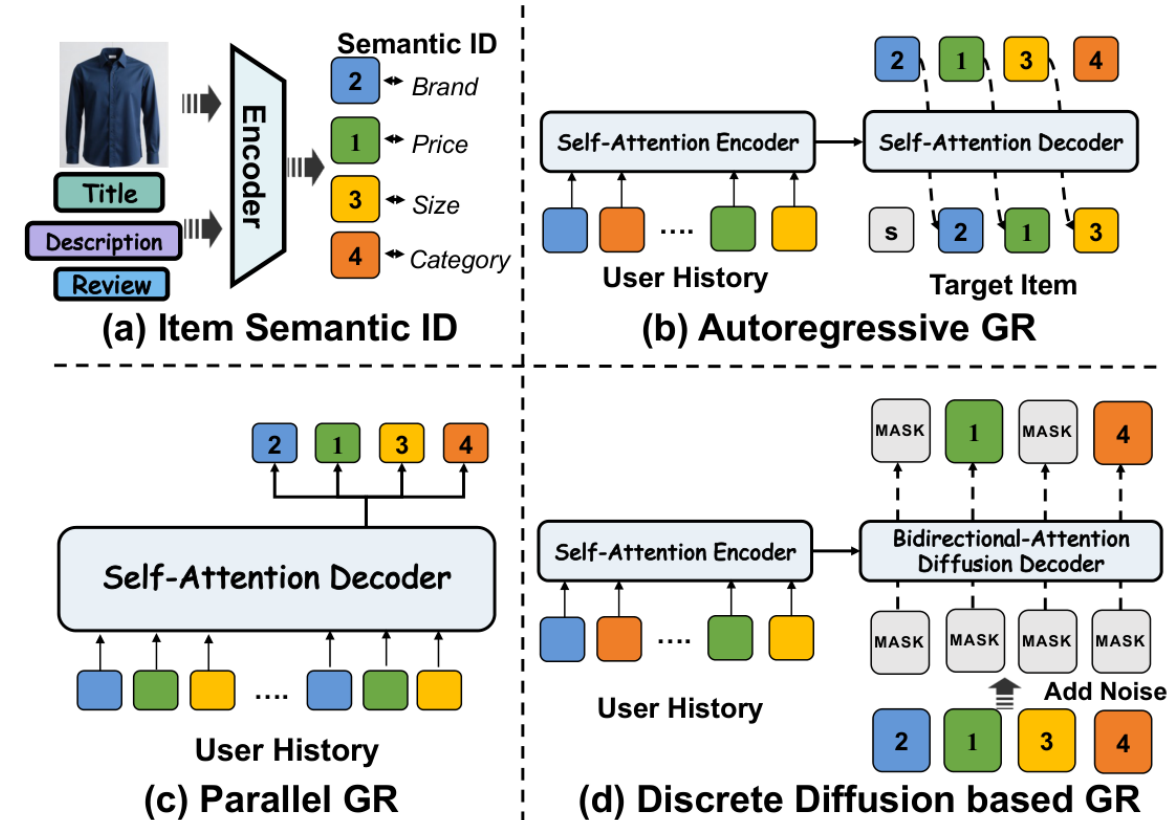
- Common GR generates SID in a fixed order.
- One-shot parallel GR is fast but lacks iterative refinement.

• Motivation

- SIDs should model global attribute dependencies and user-specific token priorities.
- Industrial systems need a controllable quality-latency trade-off.

• Contributions

- OPQ-based parallel codebook, curriculum noising, and history-aware mask allocation.
- Warm-up based two-stage parallel decoding with beam search.

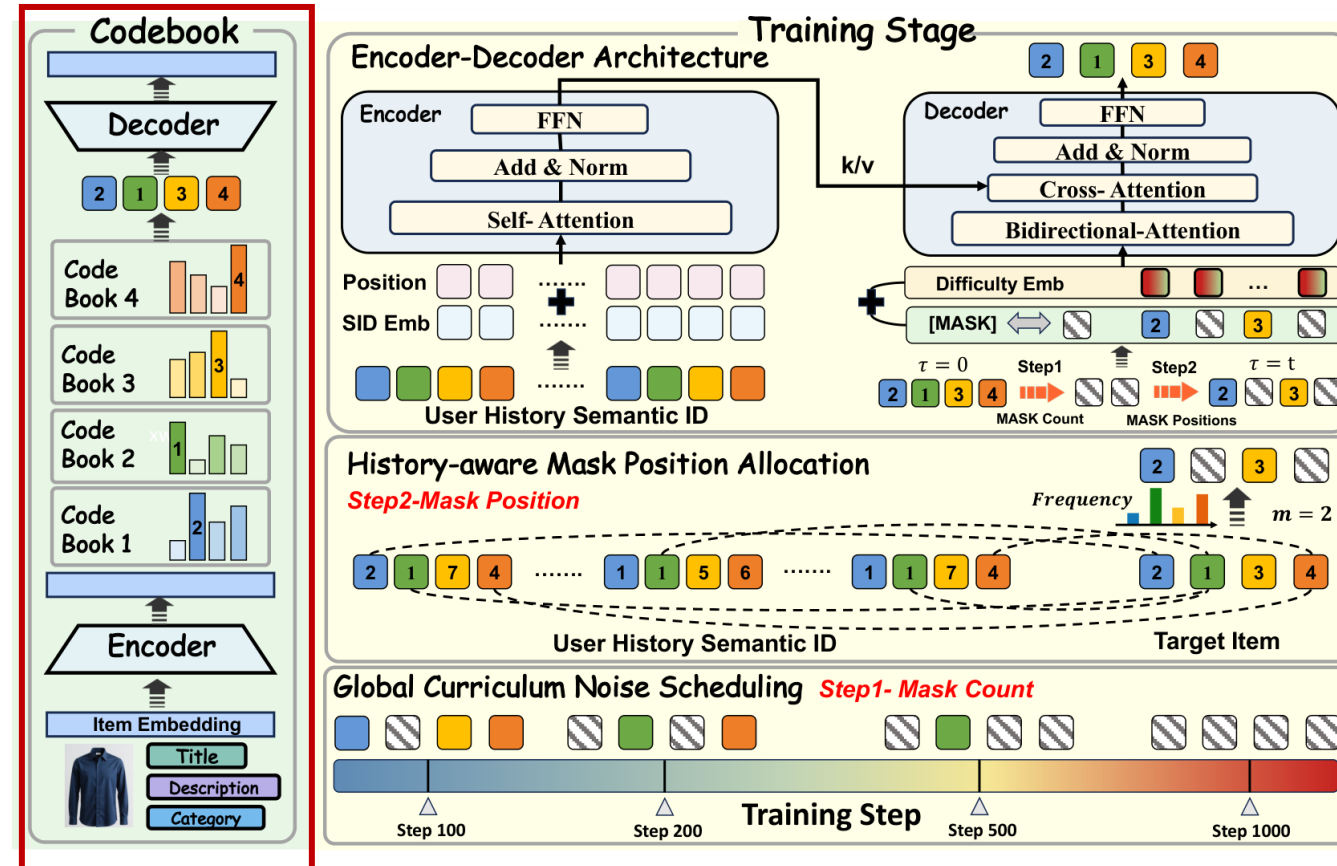


Parallel Codebook:

- Using OPQ-based parallel codebooks to build item SID

$$\mathbf{e}_i^\ell = f_\ell(\mathbf{e}_i) \in \mathbb{R}^{d_\ell}, \quad \ell = 1, \dots, L,$$

$$c_i^\ell = \arg \min_{j \in V_\ell} \|\mathbf{e}_i^\ell - C_\ell[j]\|_2^2,$$



• Training:

• Global Curriculum Noise Scheduling:

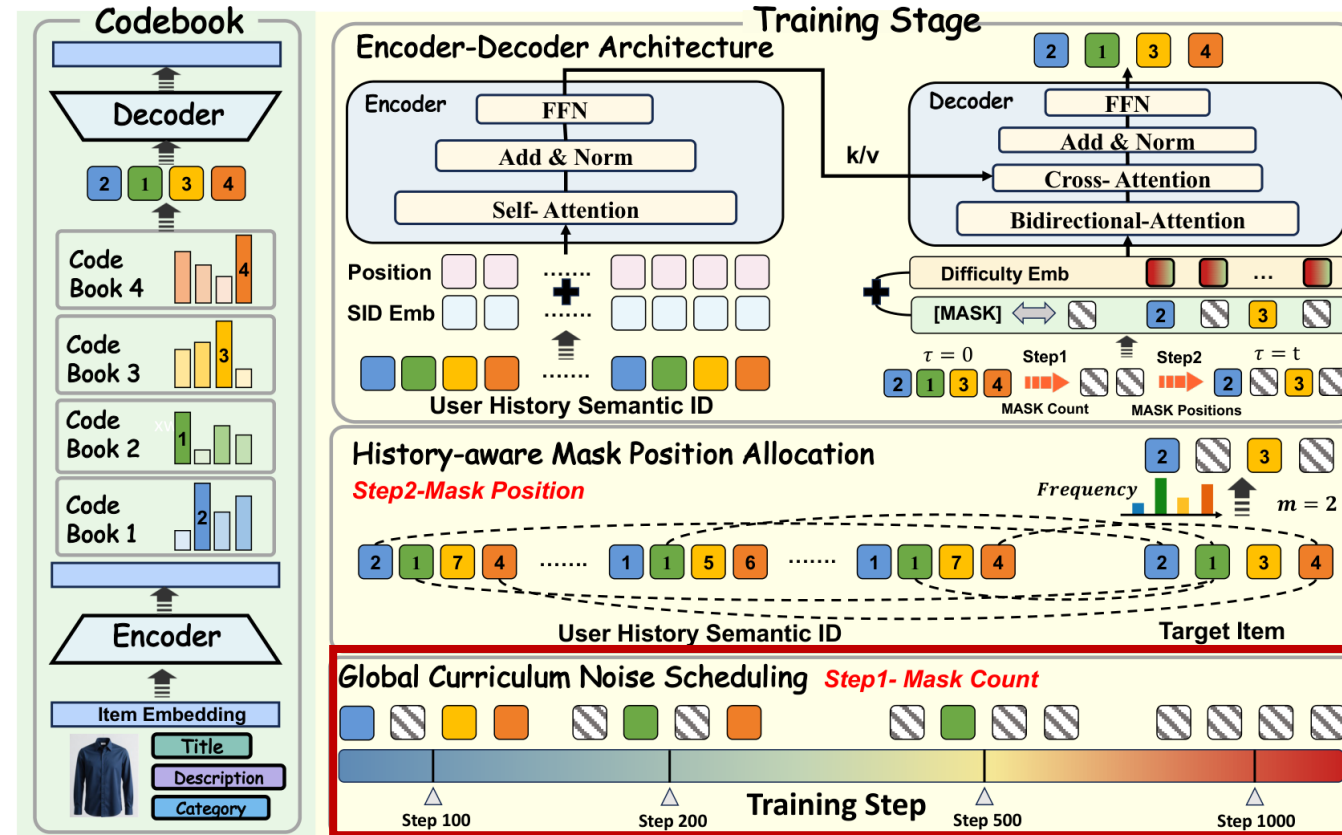
- Obtaining the stretched difficulty to interpolate the scores

$$\delta = \left(\sqrt{1 - \cos^2 \left(\frac{\pi}{2} (1 - \tau) \right)} \right)^Y,$$

$$s(k) = (1 - \delta) f_{\text{high}}(k) + \delta f_{\text{low}}(k), \quad k = 1, \dots, L.$$

- Deriving the probability distribution over the mask count

$$P_{\text{time}}(k) = \frac{s(k)}{\sum_{j=1}^L s(j)}, \quad k = 1, \dots, L.$$



• Training:

- Global curriculum noise scheduling:
- **History-aware Mask Position Allocation:**

- Calculating the Token Frequency:

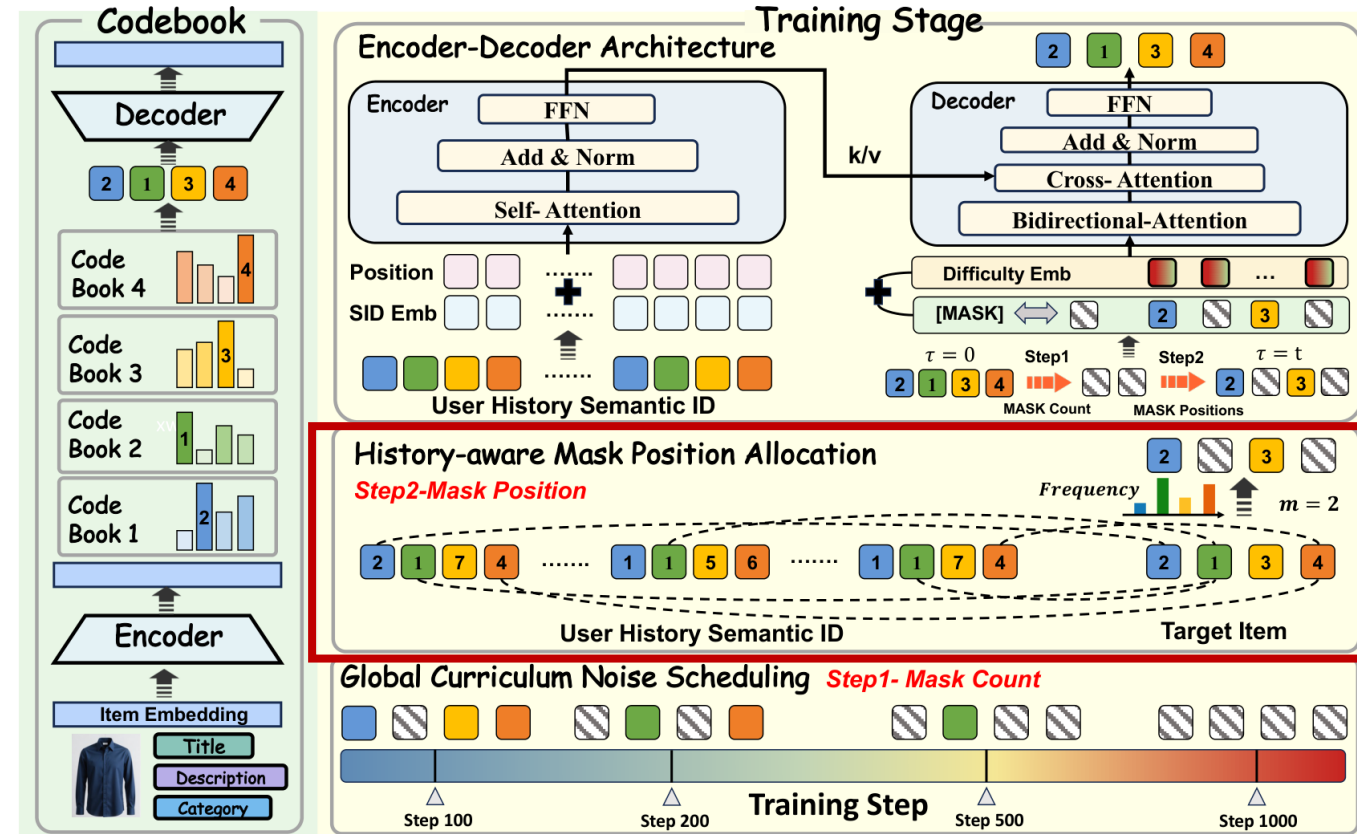
$$f^\ell = \#\{j \in s_u \mid c_j^\ell = c_{i^+}^\ell\}, \quad \ell = 1, \dots, L,$$

- Deriving the position-wise sampling distribution:

$$\mathbf{P}_{pos} = (p_{pos}^1, \dots, p_{pos}^L)$$

$$w^\ell = \frac{1}{f^\ell + \varepsilon}, \quad \mathbf{P}_{pos}^\ell = \frac{w^\ell}{\sum_{m=1}^L w^m}, \quad \ell = 1, \dots, L,$$

$$q(c_{i+1}^\ell \mid c_t^\ell, u, i^+) = \begin{cases} 1, & c_t^\ell = [\text{MASK}], c_{i+1}^\ell = [\text{MASK}], \\ \alpha_{t,\ell}(u, i^+), & c_t^\ell \neq [\text{MASK}], c_{i+1}^\ell = [\text{MASK}], \\ 1 - \alpha_{t,\ell}(u, i^+), & c_t^\ell \neq [\text{MASK}], c_{i+1}^\ell = c_t^\ell, \\ 0, & \text{otherwise,} \end{cases}$$



- Optimization: $\mathcal{L} = \mathbb{E}_{(u, i^+)} \mathbb{E}_{k \sim \mathbf{P}_{\text{time}}(\cdot)} \mathbb{E}_{\mathcal{M} \sim \mathbf{P}_{pos}(\cdot | k, u, i^+)} \left[- \sum_{\ell \in \mathcal{M}} \log p_{\theta}(c_{i^+}^\ell \mid \tilde{c}_{i^+}, s_u, \mathbf{d}_k) \right],$

- Inference:**

- Warm-up based Position Scheduling Strategy:**

$$m = \begin{cases} 1, & r < R_{\text{warm}}, \\ m_{\text{par}}, & r \geq R_{\text{warm}}, m_{\text{par}} > 1. \end{cases}$$

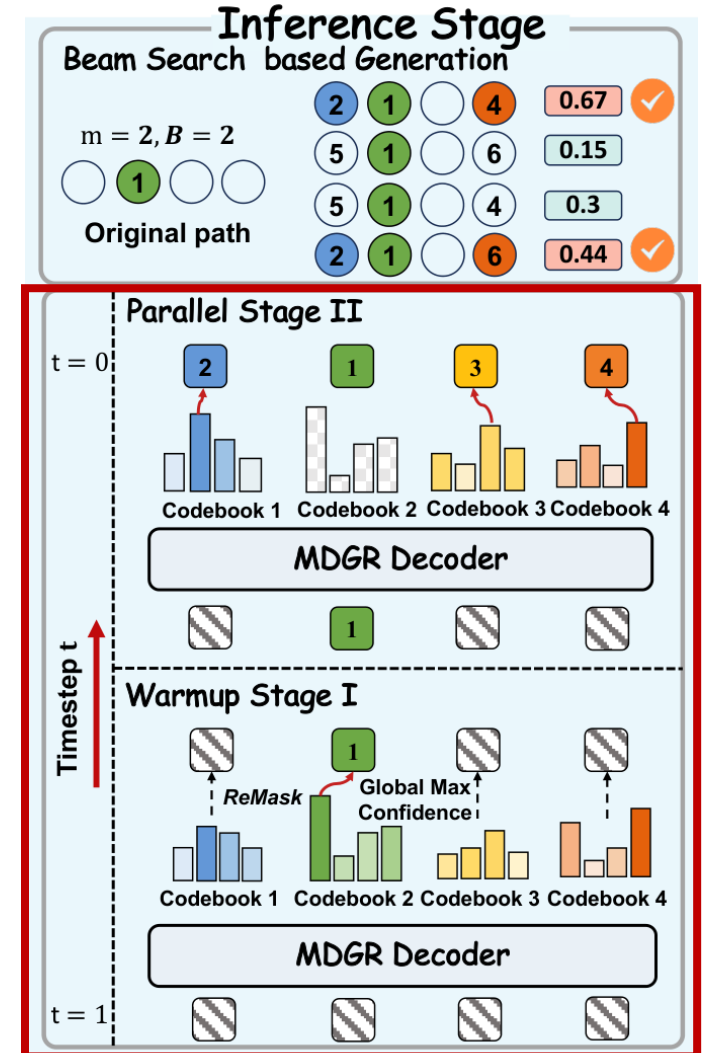
- Token Position Selection:**

- Calculating Confidence Score over Position:

$$\text{conf}(\ell) = \max_{c \in \mathcal{V}_\ell} p_\theta(\mathbf{x}_{(r)}^\ell = c \mid \mathbf{x}_{(r)}, \mathbf{s}_u, \mathbf{d}_{(r)}).$$

- Selecting the Highest for Decoding:

$$\mathcal{S}_{(r)} = \{\ell_1, \dots, \ell_m\} \subset \mathcal{M}_{(r)},$$



- Inference:

- Beam Search Based Generation:

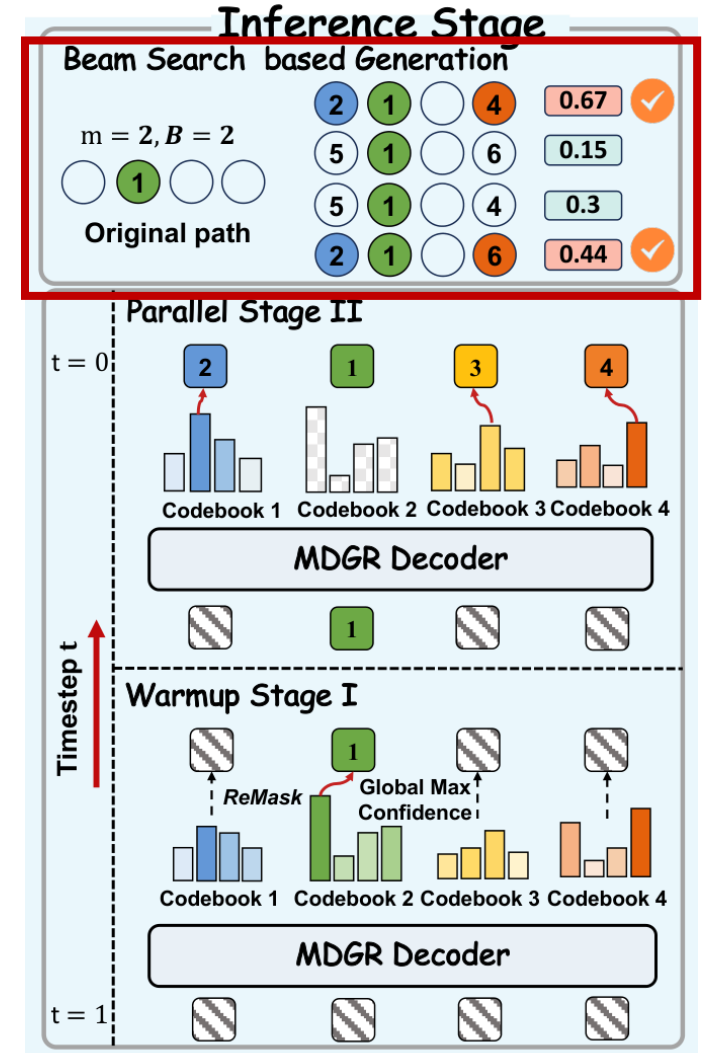
- Expanding the Position with Retained Token:

$$\mathbf{x}_{(r+1),(j_1,\dots,j_m)} = \begin{cases} \tilde{c}_{(r),b}^{\ell,j_k}, & \ell \in \mathcal{S}^{(r)}, k = 1, \dots, m, \\ c_{(r),b}^{\ell}, & \text{otherwise.} \end{cases}$$

$$v_{(r+1),(j_1,\dots,j_m)} = v_{(r),b} + \sum_{k=1}^m \log \tilde{p}_{(r),b}^{\ell,j_k}.$$

- Selecting the Top-B for Beam Search:

$$\mathbf{x}_{(r+1)} = \text{Top}_B \left\{ \left(\mathbf{x}_{(r+1),(j_1,\dots,j_m)}, v_{(r+1),(j_1,\dots,j_m)} \right) \right\}_{(r+1),(j_1,\dots,j_m)}.$$



Methods	Amazon Electronics				Amazon Books				Industry			
	Recall@5	NDCG@5	Recall@10	NDCG@10	Recall@5	NDCG@5	Recall@10	NDCG@10	Recall@5	NDCG@5	Recall@10	NDCG@10
<i>Item ID-based Discriminative Recommendation</i>												
SASRec	0.0321	0.0213	0.0490	0.0309	0.0506	0.0297	0.0697	0.0391	0.1413	0.0909	0.1695	0.1080
BERT4Rec	0.0324	0.0214	0.0495	0.0311	0.0509	0.0299	0.0699	0.0394	0.1421	0.0911	0.1693	0.1079
S ³ Rec	0.0320	0.0211	0.0489	0.0307	0.0514	0.0302	0.0704	0.0396	0.1456	0.0934	0.1722	0.1101
PinnerFormer	0.0330	0.0220	0.0502	0.0316	0.0525	0.0307	0.0716	0.0401	0.1483	0.0950	0.1766	0.1132
HeterRec	0.0336	0.0221	0.0510	0.0319	0.0522	0.0306	0.0712	0.0398	0.1592	0.1014	0.1890	0.1208
DiffuRec	0.0334	0.0220	0.0507	0.0317	0.0531	0.0311	0.0722	0.0403	0.1586	0.1013	0.1889	0.1207
VQ-Rec	0.0331	0.0217	0.0498	0.0312	0.0517	0.0301	0.0687	0.0384	0.1520	0.0975	0.1843	0.1177
<i>Semantic ID-based Generative Recommendation</i>												
TIGER	0.0348	0.0230	0.0524	0.0324	0.0552	0.0315	0.0731	0.0408	0.1620	0.1049	0.1943	0.1244
Cobra	<u>0.0359</u>	<u>0.0237</u>	<u>0.0544</u>	0.0338	0.0576	0.0330	<u>0.0763</u>	0.0426	0.1659	0.1059	0.1996	0.1285
RPG	0.0357	0.0236	0.0543	<u>0.0339</u>	<u>0.0581</u>	<u>0.0332</u>	0.0762	<u>0.0427</u>	<u>0.1679</u>	<u>0.1076</u>	<u>0.2018</u>	<u>0.1296</u>
<i>Ours</i>												
MDGR	0.0394	0.0258	0.0583	0.0367	0.0631	0.0361	0.0826	0.0463	0.1856	0.1192	0.2210	0.1422
Improv.	+9.75%	+8.86%	+7.17%	+8.26%	+8.61%	+8.73%	+8.26%	+8.43%	+10.45%	+10.78%	+9.54%	+9.72%

R_{warm}	m_{par}	Total step	Recall@10	QPS	QPS uplift
<i>Base Setting</i>					
0	1	8	0.2219	2.212	–
<i>Impact of R_{warm}</i>					
0	2	4	0.2127	3.015	+36.30%
2	2	5	0.2199	2.741	+23.92%
4	2	6	0.2210	2.527	+14.24%
6	2	7	0.2213	2.365	+6.92%
<i>Impact of m_{par}</i>					
4	3	6	0.2205	2.526	+14.19%
4	4	5	0.2204	2.739	+23.82%

• Main signal:

- MDGR improves up to 10.78% over the strongest baseline and shows deployment-aware efficiency trade-offs.

Work	Failure Mode Fixed	Training / Noising	Inference / Denoising
DiffGRM	SID digit consistency and heterogeneous difficulty	OCN masks uncertain digits	Confidence-guided parallel denoising
LLaDA-Rec	Fixed AR order and error accumulation	History and next-item masking	Adaptive remask plus beam search
MDGR	Generic diffusion ignores history and latency	Curriculum + history-aware allocation	Warm-up then parallel decoding

- **Takeaway:**

- Diffusion-based GR is a parallel SID generation architecture designed to avoid prefix-accumulated errors caused by autoregressive decoding.

Architecture Summary



KDD2026
August 9–13, 2026
Jeju, Korea

Family	Core Role	Strength	Open Issue
Encoder-Decoder	Encode user context, decode item tokens	Balances understanding and generation	Encoder-heavy compute and custom modules
Decoder-Only	Autoregressively model behavior/item streams	Simple scalable backbone	Needs task-aware masks and valid decoding
Diffusion-Based	Denoise the full target SID	Bidirectional target modeling and adaptive order	Still early for broad industrial integration

- Generative recommendation is moving toward unified, hardware-friendly architectures.
- Decoder-only models inherit scaling advantages but need recommendation-aware representation and masks.
- Diffusion-based models reframe item generation as adaptive reconstruction of structured semantic IDs.
- Across families, deployability depends on valid item retrieval, latency control, and task-aware information flow.

1 Introduction (9:00 - 9:15)



Xiaopeng Li



2 Tokenization (9:15 - 9:45)



Xiaopeng Li

3 Model Structure (9:45 - 10:30)



Ziwei Liu



15mins

4 Optimization (10:45 - 11:15)



Wenlin Zhang



Kaifeng Guo

5 Application (11:15 - 11:30)



Yejing Wang



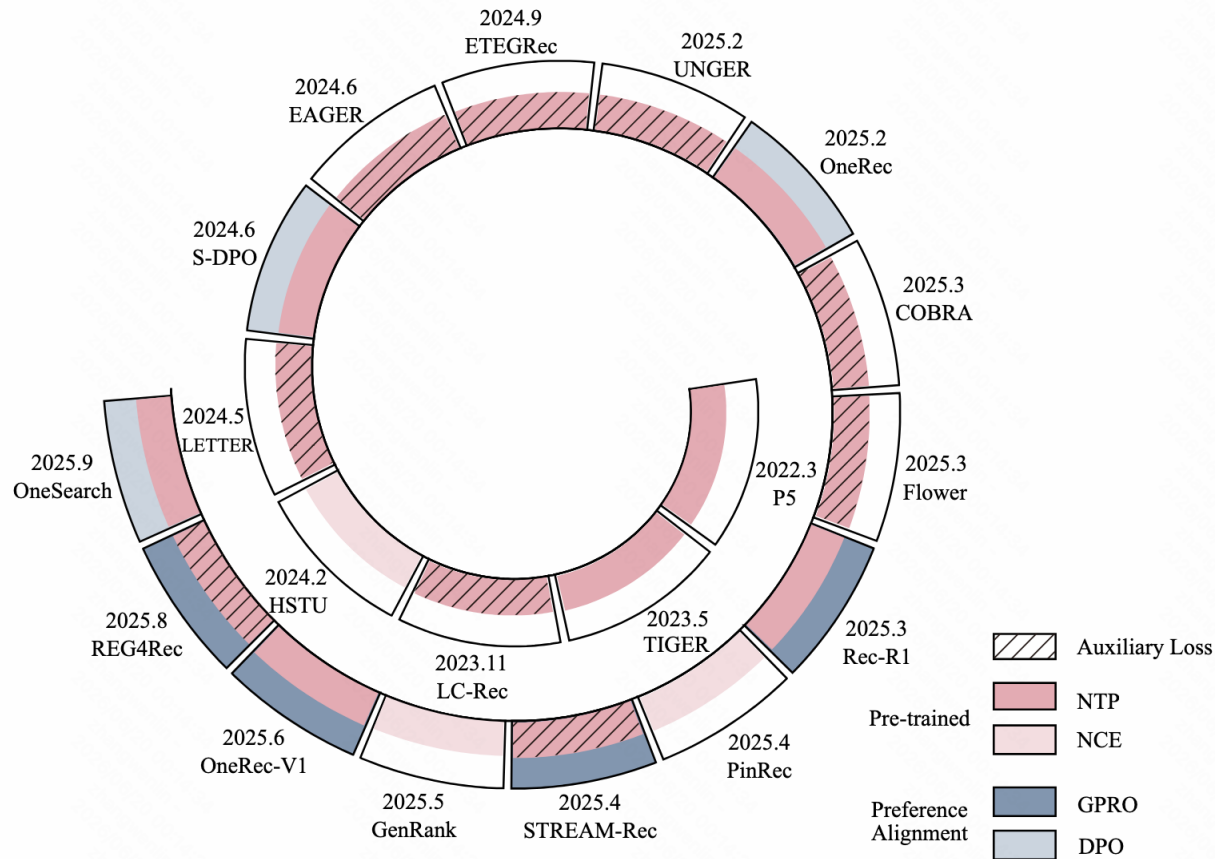
6 Challenge & Conclusion (11:30 - 11:45)



Bo Chen

- Traditional Recommendation
 - Traditional recommendation optimizes local scoring objectives, such as retrieval matching and ranking prediction.
- Generative Recommendation
 - Generative recommendation optimizes item generation, learning to produce item tokens or recommendation lists from user history.
- Optimization Shift
 - From **separating positive and negative items** to **modeling how suitable items are generated**, while further aligning generated results with ranking quality, user preference, and platform objectives.

We divide the categories of current optimization method in two types:



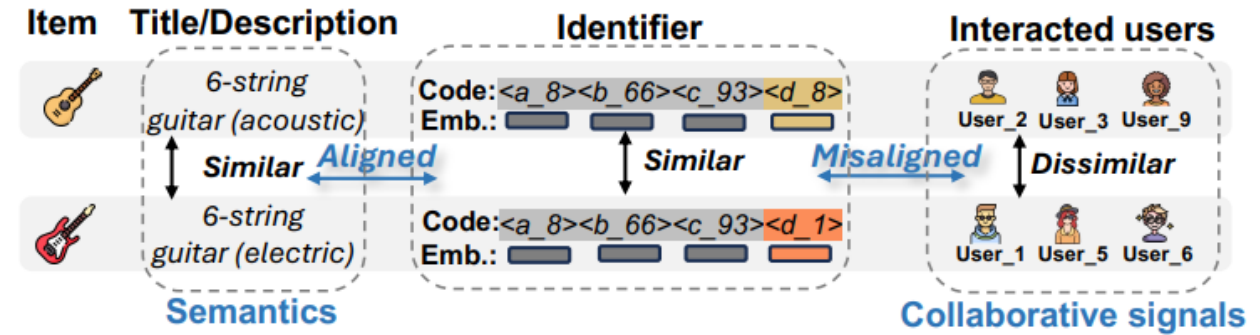
- Supervised learning:
 - Next-Token Prediction (NTP)
 - Noise-Contrastive Estimation (NCE)
 - Auxiliary Loss
- Preference Learning:
 - Direct Preference Optimization (DPO)
 - Group Relative Policy Optimization (GRPO)

	NTP			NCE	
Method	TIGER	LETTER	COBRA	PinRec	IntSR
Supervision target	next-item Semantic ID	better Semantic ID labels	sparse ID + dense vector	future item set / sampled softmax target	time-aligned candidate set
Core signal	vanilla NTP	ranking-aware NTP	composite supervision	sampled-softmax NCE	InfoNCE
What is improved	generative retrieval	label quality + ranking alignment	sparse-dense fusion	scalability to industrial catalog	avoid false negatives / temporal mismatch
Main takeaway	make retrieval generative	make NTP more ranking-aware	enrich supervision beyond sparse tokens	make contrastive training scalable	make contrastive training temporally correct

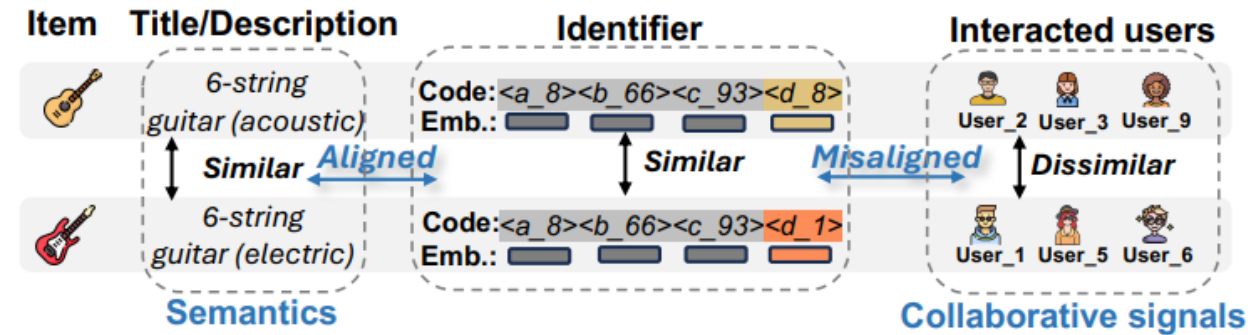
- Shared question: how can supervised GenRec optimize valid item generation and ranking quality from observed user feedback?

- Background
 - NTP task – Semantic ID.

- Background
 - NTP task – Semantic ID.
- Motivation
 - If the label sequence is biased or not aligned with ranking, the generative model may learn a good likelihood objective but a weak recommendation objective.



- Background
 - NTP task – Semantic ID.
- Motivation
 - If the label sequence is biased or not aligned with ranking, the generative model may learn a good likelihood objective but a weak recommendation objective.
- Contribution
 - LETTER improves supervised GenRec by learning **better item labels** with a tri-regularized tokenizer and by training with **ranking-guided next-token prediction**.



- Improving Label Quality
 - LETTER learns item labels with three regularizers: semantic regularization, collaborative regularization, and diversity regularization.

$$\mathcal{L}_{\text{LETTER}} = \mathcal{L}_{\text{Sem}} + \alpha \mathcal{L}_{\text{CF}} + \beta \mathcal{L}_{\text{Div}}$$

- Improving Label Quality
 - LETTER learns item labels with three regularizers: semantic regularization, collaborative regularization, and diversity regularization.

$$\mathcal{L}_{\text{LETTER}} = \mathcal{L}_{\text{Sem}} + \alpha \mathcal{L}_{\text{CF}} + \beta \mathcal{L}_{\text{Div}}$$

- Collaborative regularization
 - To inject collaborative signals into the code sequence.

$$\mathcal{L}_{\text{CF}} = -\frac{1}{B} \sum_{i=1}^B \frac{\exp(\langle \hat{\mathbf{z}}_i, \mathbf{h}_i \rangle)}{\sum_{j=1}^B \exp(\langle \hat{\mathbf{z}}_i, \mathbf{h}_j \rangle)}$$

- Diversity regularization
 - To address the code generation bias, it is promising to learn a more uniform distribution of the code embeddings

$$\mathcal{L}_{\text{Div}} = -\frac{1}{B} \sum_{i=1}^B \frac{\exp(\langle \mathbf{e}_{c_l}^i, \mathbf{e}_+ \rangle)}{\sum_{j=1}^{N-1} \exp(\langle \mathbf{e}_{c_l}^i, \mathbf{e}_j \rangle)}$$

- Ranking-guided generation loss
 - **Ranking Alignment Challenge:** Token likelihood does not fully reflect whether the true item ranks above hard negative items.
 - **Ranking-Guided Generation Loss:** LETTER modifies next-token prediction with a hard-negative-aware temperature to strengthen ranking supervision.

$$\begin{cases} \mathcal{L}_{\text{rank}} = - \sum_{t=1}^{|y|} \log P_{\theta}(y_t | y_{<t}, x), \\ P_{\theta}(y_t | y_{<t}, x) = \frac{\exp(p(y_t)/\tau)}{\sum_{v \in \mathcal{V}} \exp(p(v)/\tau)} \end{cases}$$

- LETTER improves both TIGER and LC-Rec, showing that its supervision design is not tied to one generative backbone.

Model	Instruments				Beauty				Yelp			
	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10
MF	0.0479	0.0735	0.0330	0.0412	0.0294	0.0474	0.0145	0.0191	0.0220	0.0381	0.0138	0.0190
Caser	0.0543	0.0710	0.0355	0.0409	0.0205	0.0347	0.0131	0.0176	0.0150	0.0263	0.0099	0.0134
HGN	0.0813	0.1048	0.0668	0.0774	0.0325	0.0512	0.0206	0.0266	0.0186	0.0326	0.0115	0.0159
Bert4Rec	0.0671	0.0822	0.0560	0.0608	0.0203	0.0347	0.0124	0.0170	0.0186	0.0291	0.0115	0.0159
LightGCN	0.0794	0.0100	0.0662	0.0728	0.0305	0.0511	0.0194	0.0260	0.0248	0.0407	0.0156	0.0207
SASRec	0.0751	0.0947	0.0627	0.0690	0.0380	0.0588	0.0246	0.0313	0.0183	0.0296	0.0116	0.0152
BigRec	0.0513	0.0576	0.0470	0.0491	0.0243	0.0299	0.0181	0.0198	0.0154	0.0169	0.0137	0.0142
P5-TID	0.0000	0.0001	0.0000	0.0000	0.0182	0.0432	0.0132	0.0254	0.0184	0.0263	0.0130	0.0155
P5-SemID	0.0775	0.0964	0.0669	0.0730	0.0393	0.0584	0.0273	0.0335	0.0202	0.0324	0.0131	0.0170
P5-CID	0.0809	0.0987	0.0695	0.0751	0.0404	0.0597	0.0284	0.0347	0.0219	0.0347	0.0140	0.0181
TIGER	0.0870	0.1058	0.0737	0.0797	0.0395	0.0610	0.0262	0.0331	0.0253	0.0407	0.0164	0.0213
LETTER-TIGER	0.0909	0.1122	0.0763	0.0831	0.0431	0.0672	0.0286	0.0364	0.0277	0.0426	0.0184	0.0231
LC-Rec	0.0824	0.1006	0.0712	0.0772	0.0443	0.0642	0.0311	0.0374	0.0230	0.0359	0.0158	0.0199
LETTER-LC-Rec	0.0913	0.1115	0.0789	0.0854	0.0505	0.0703	0.0355	0.0418	0.0255	0.0393	0.0168	0.0211

- Practical Values:** The consistent gains indicate that better item labels and ranking-guided next-token prediction both strengthen the supervised signal.

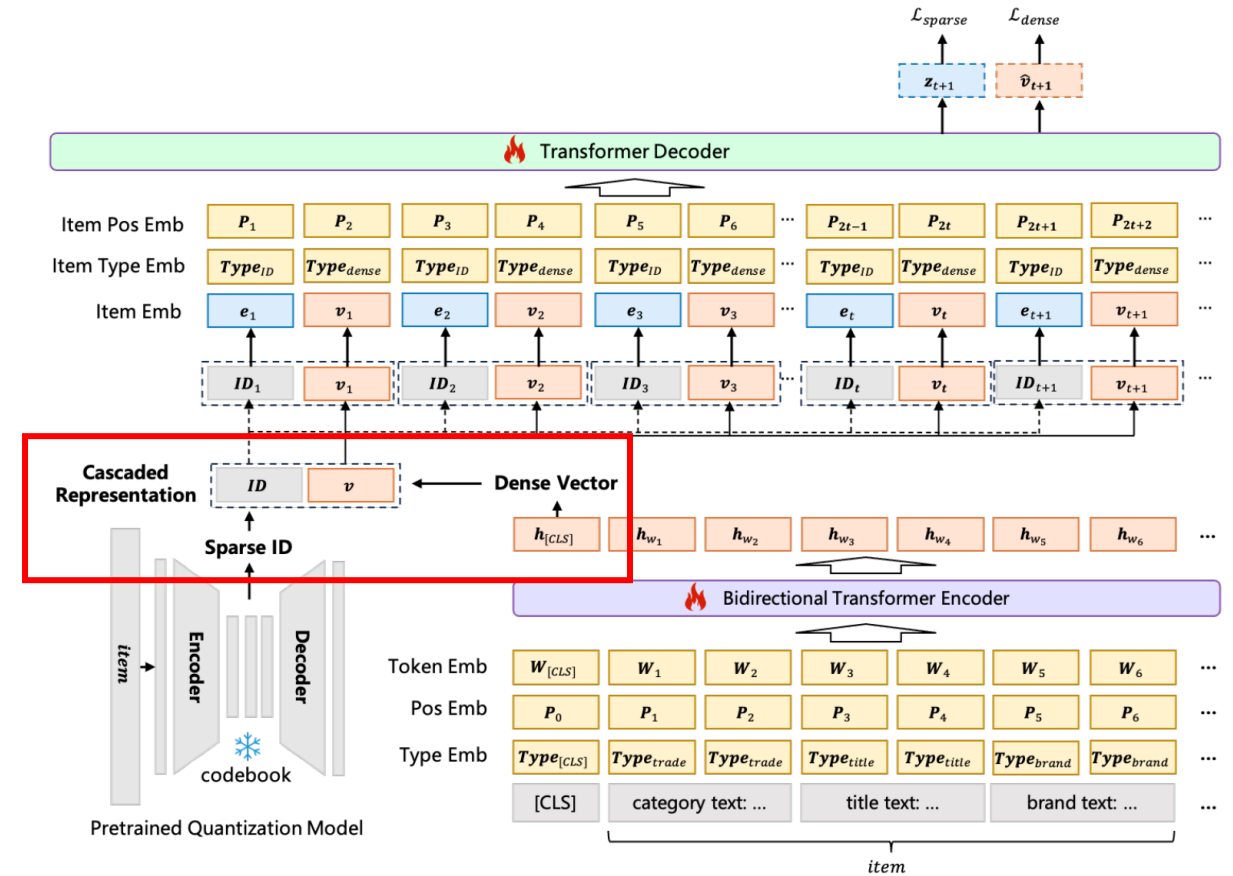
- Background
 - TIGER-style methods supervise the model mainly with sparse semantic-identifier tokens.

- Background
 - TIGER-style methods supervise the model mainly with sparse semantic-identifier tokens.
- Motivation
 - Sparse-token supervision is efficient, but it may lose fine-grained item similarity that is useful for ranking and retrieval.

- Background
 - TIGER-style methods supervise the model mainly with sparse semantic-identifier tokens.
- Motivation
 - Sparse-token supervision is efficient, but it may lose fine-grained item similarity that is useful for ranking and retrieval.
- Contribution
 - COBRA introduces **composite supervision**: the model learns both sparse identifier prediction and dense-vector prediction, then fuses them during inference.

- **Composite Supervision**

- **Sparse-Only Limitation:** A single sparse-token target may not provide enough supervision for fine-grained matching.
- **Composite Supervision:** COBRA trains the model with both sparse-identifier prediction and dense-vector prediction.

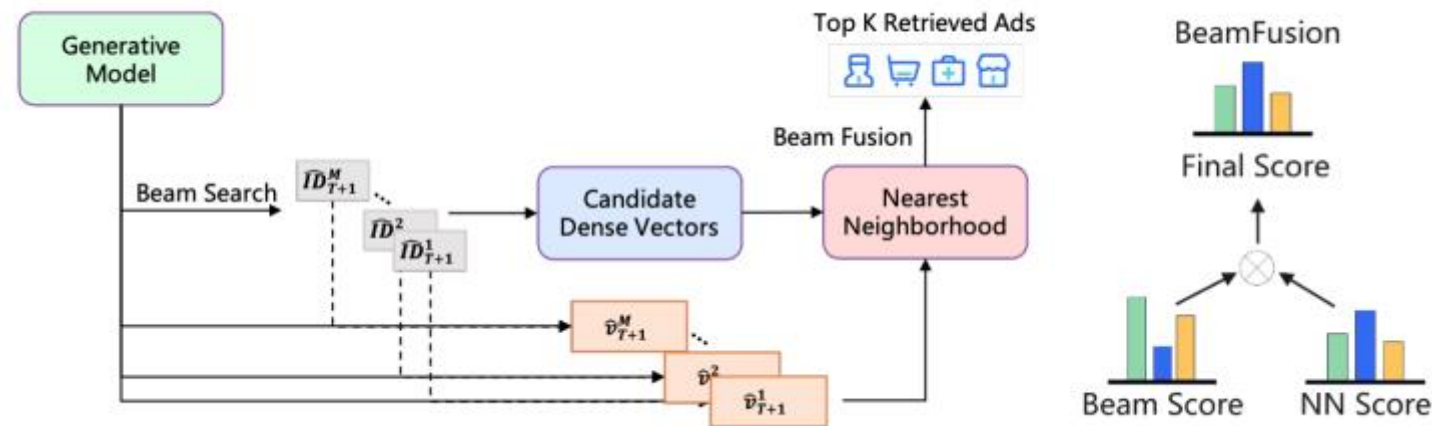


$$\mathcal{L} = \mathcal{L}_{sparse} + \mathcal{L}_{dense}$$

$$\mathcal{L}_{sparse} = - \sum_{t=1}^{T-1} \log \left(\frac{\exp(z_{t+1}^{ID_{t+1}})}{\sum_{j=1}^C \exp(z_{t+1}^j)} \right)$$

$$\mathcal{L}_{dense} = - \sum_{t=1}^{T-1} \log \frac{\exp(\cos(\hat{v}_{t+1} \cdot v_{t+1}))}{\sum_{item_j \in \text{Batch}} \exp(\cos(\hat{v}_{t+1}, v_{item_j}))}$$

- Supervision-Aware Inference with BeamFusion
 - Training-Inference Gap:** If the model learns both sparse and dense targets, inference should use both signals rather than only the generated identifier.
 - BeamFusion Ranking:** COBRA first uses sparse generation to produce candidates, then uses the predicted dense vector to refine the ranking.



- COBRA shows that composite supervision improves sparse-only generative retrieval.
- It improves over TIGER on Beauty, Sports, and Toys, while ablations show that sparse identifiers, dense prediction, and BeamFusion all contribute to performance.

	Method	R@5	N@5	R@10	N@10
Beauty	P5	0.0163	0.0107	0.0254	0.0136
	Caser	0.0205	0.0131	0.0347	0.0176
	HGN	0.0325	0.0206	0.0512	0.0266
	GRU4Rec	0.0164	0.0099	0.0283	0.0137
	BERT4Rec	0.0203	0.0124	0.0347	0.0170
	FDSA	0.0267	0.0163	0.0407	0.0208
	SASRec	0.0387	0.0249	0.0605	0.0318
	S ³ -Rec	0.0387	0.0244	0.0647	0.0327
	TIGER	0.0454	0.0321	0.0648	0.0384
	COBRA[Ours]	0.0537	0.0395	0.0725	0.0456

	Method	R@5	N@5	R@10	N@10
Sports	P5	0.0061	0.0041	0.0095	0.0052
	Caser	0.0116	0.0072	0.0194	0.0097
	HGN	0.0189	0.0120	0.0313	0.0159
	GRU4Rec	0.0129	0.0086	0.0204	0.0110
	BERT4Rec	0.0115	0.0075	0.0191	0.0099
	FDSA	0.0182	0.0122	0.0288	0.0156
	SASRec	0.0233	0.0154	0.0350	0.0192
	S ³ -Rec	0.0251	0.0161	0.0385	0.0204
	TIGER	0.0264	0.0181	0.0400	0.0225
	COBRA[Ours]	0.0305	0.0215	0.0434	0.0257

	Method	R@5	N@5	R@10	N@10
Toys	P5	0.0070	0.0050	0.0121	0.0066
	Caser	0.0166	0.0107	0.0270	0.0141
	HGN	0.0321	0.0221	0.0497	0.0277
	GRU4Rec	0.0097	0.0059	0.0176	0.0084
	BERT4Rec	0.0116	0.0071	0.0203	0.0099
	FDSA	0.0228	0.0140	0.0381	0.0189
	SASRec	0.0463	0.0306	0.0675	0.0374
	S ³ -Rec	0.0443	0.0294	0.0700	0.0376
	TIGER	0.0521	0.0371	0.0712	0.0432
	COBRA[Ours]	0.0619	0.0462	0.0781	0.0515

- **Practical Values:** COBRA shows that richer supervision is important for GenRec, because combining sparse identifier prediction with dense representation learning improves both retrieval and fine-grained matching.

- Background
 - PinRec studies generative item retrieval in a large-scale industrial platform where the model directly generates item embeddings for retrieval.

- Background
 - PinRec studies generative item retrieval in a large-scale industrial platform where the model directly generates item embeddings for retrieval.
- Motivation
 - Strict next-token supervision is not fully aligned with feed recommendation, because users may engage with multiple future items in a session without a strict local order.

- Background
 - PinRec studies generative item retrieval in a large-scale industrial platform where the model directly generates item embeddings for retrieval.
- Motivation
 - Strict next-token supervision is not fully aligned with feed recommendation, because users may engage with multiple future items in a session without a strict local order.
- Contribution
 - PinRec keeps sampled softmax as the scalable supervision objective, but redesigns the positive target into relaxed future-item and multi-token prediction.

- **Sampled Softmax Loss:** PinRec contrasts the target item against sampled negatives to train over a large item vocabulary.

- Sampled softmax loss:

$$L_s(\hat{\mathbf{i}}_{u,t}, \mathbf{i}_{\text{target}}) = -\log \frac{\exp(s(\hat{\mathbf{i}}_{u,t}, \mathbf{i}_{\text{target}}))}{\exp(s(\hat{\mathbf{i}}_{u,t}, \mathbf{i}_{\text{target}})) + \sum_{\mathbf{i}_n \in \mathcal{N}} \exp(s(\hat{\mathbf{i}}_{u,t}, \mathbf{i}_n))}$$

- Standard next-token objective:

$$L_{\theta}^{\text{next}}(H(u, t_{\max})) = \frac{1}{|H(u, t_{\max})|} \sum_{t=1}^{t_{\max}-1} L_s(\hat{\mathbf{i}}_{u,t}, \mathbf{i}_{u,t+1})$$

- **Sampled Softmax Loss:** PinRec contrasts the target item against sampled negatives to train over a large item vocabulary.

- Sampled softmax loss:

$$L_s(\hat{\mathbf{i}}_{u,t}, \mathbf{i}_{\text{target}}) = -\log \frac{\exp(s(\hat{\mathbf{i}}_{u,t}, \mathbf{i}_{\text{target}}))}{\exp(s(\hat{\mathbf{i}}_{u,t}, \mathbf{i}_{\text{target}})) + \sum_{\mathbf{i}_n \in \mathcal{N}} \exp(s(\hat{\mathbf{i}}_{u,t}, \mathbf{i}_n))}$$

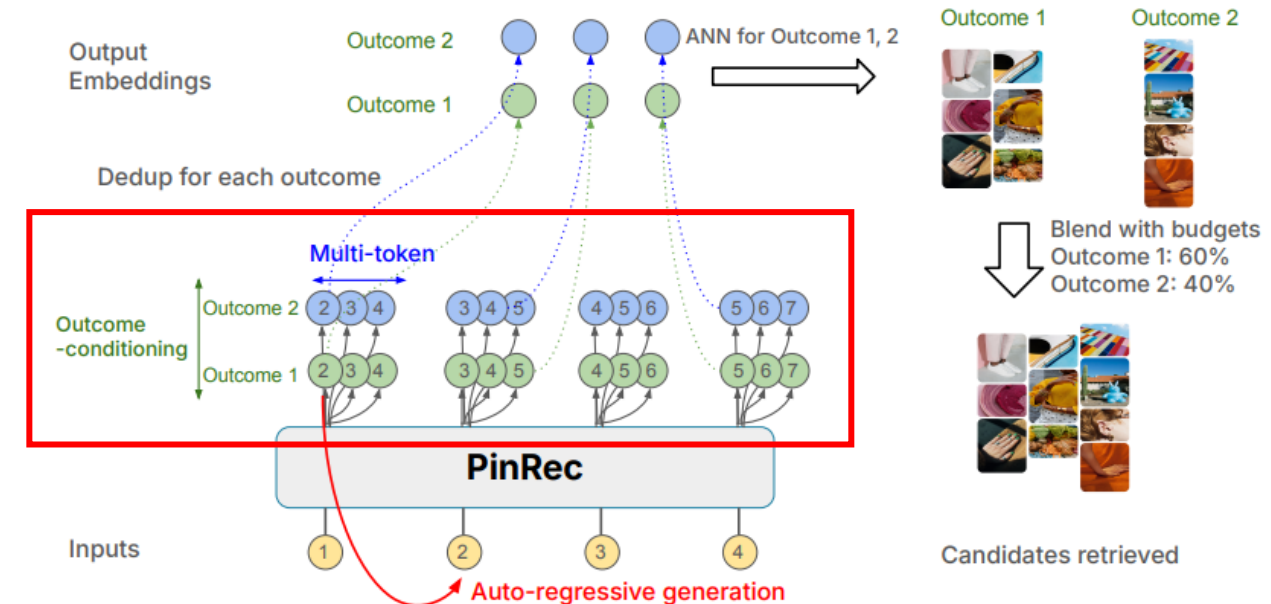
- Standard next-token objective:

$$L_{\theta}^{\text{next}}(H(u, t_{\max})) = \frac{1}{|H(u, t_{\max})|} \sum_{t=1}^{t_{\max}-1} L_s(\hat{\mathbf{i}}_{u,t}, \mathbf{i}_{u,t+1})$$

- **Intra-Feed Relaxation:** PinRec treats future items in the same feed session as valid positives instead of only supervising the immediate next item.

$$L_{\theta}^{\text{feed}}(H(u, t_{\max})) = \frac{1}{|H(u, t_{\max})|} \sum_{t_j} \min_{i \in T_{t_j}^{\text{feed}}} L_s(\hat{\mathbf{i}}_{u,t_j}, i)$$

- **Multi-Token Prediction:** PinRec samples multiple future positions and trains the model to generate several item embeddings at different temporal offsets.
- **Supervision Insight:** The model learns a richer future-item distribution while still using sampled negatives for scalable contrastive training.



- PinRec compares major baselines and model variants in Table, showing that relaxed multi-target supervision improves generative retrieval across Homefeed, Related Pins, and Search.

Table 1: Comparison of baselines and PinRec variants (unordered recall @ 10) across major surfaces at Pinterest.

Model	Homefeed	Related Pins	Search
SASRec [12]	0.382	0.426	0.142
PinnerFormer [18]	0.461	0.412	0.257
TIGER [22]	0.208	0.230	0.090
HTSU-OC [32]	0.596	0.539	0.179
PinRec-UC	0.608	0.521	0.350
PinRec-OC	0.625	0.537	0.352
PinRec-{MT, OC}	0.676	0.631	0.450

- Practical Values:** For supervised GenRec, NCE-style training should optimize the target set and the negative set together, not only approximate full softmax.

- Background
 - IntSR studies integrated search and recommendation, where the same generative sequence model must learn from user behavior under changing item availability.

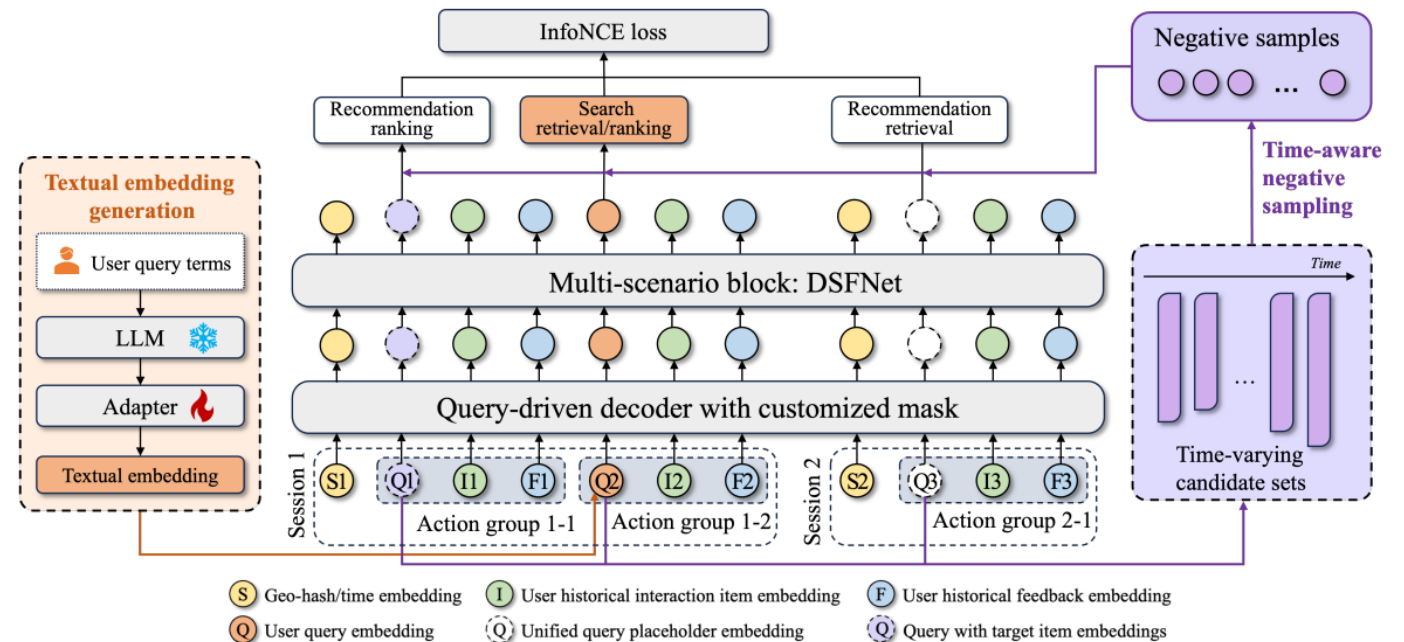
- Background
 - IntSR studies integrated search and recommendation, where the same generative sequence model must learn from user behavior under changing item availability.
- Motivation
 - Search and recommendation share users and items, but their supervised targets appear under different query contexts.
 - The item vocabulary changes over time, so negative samples must come from the candidate set available at the interaction time.

- Background
 - IntSR studies integrated search and recommendation, where the same generative sequence model must learn from user behavior under changing item availability.
- Motivation
 - Search and recommendation share users and items, but their supervised targets appear under different query contexts.
 - The item vocabulary changes over time, so negative samples must come from the candidate set available at the interaction time.
- Contribution
 - IntSR builds a query-driven integrated framework and trains it with temporally aligned InfoNCE supervision.

- IntSR Framework

- Query-Driven Integrated Sequence:** IntSR inserts query tokens into the behavior sequence so search, recommendation, and ranking targets can be predicted in one framework.

- Candidate-Aware Prediction Flow:** The model encodes historical behaviors and query context, then scores candidate items at query positions for supervised prediction.



- **Temporal Candidate Alignment**

- **Available-Candidate InfoNCE:** IntSR contrasts the positive item only against candidates available at the interaction timestamp.

$$L = -\frac{1}{|\mathcal{A}|} \sum_{u \in \mathcal{U}} \sum_{a \in \mathcal{A}_u} \delta_{u,a} \log \frac{\exp(z_{u,a,i^+})}{\sum_{i \in \mathcal{I}_{t_a}} \exp(z_{u,a,i})}$$

- **Temporal-Aligned Negative Sampling:** IntSR sets the sampling probability to zero for unavailable items before random, popularity-based, or hard negatives are selected.

$$\text{prob}_{i,t} = \begin{cases} \text{prob}_i, & \text{if } i \in \mathcal{I}_t, \\ 0, & \text{otherwise.} \end{cases}$$

- **Integrated Framework:** IntSR improves search performance by learning search and recommendation signals in one query-driven sequence model.
- **Temporal Candidate Alignment:** Aligning the negative pool with item availability consistently improves different negative sampling strategies.

Dataset	Model	HR@1	HR@5	HR@10	N@5	N@10
Amazon	HEM [†]	0.2497	0.6778	0.8267	0.4736	0.5221
	ZAM [†]	0.2954	0.7109	0.8468	0.5147	0.5590
	TEM [†]	0.4090	0.8185	0.9051	0.6303	0.6587
	CoPPS [†]	0.4052	0.8169	0.9051	0.6281	0.6570
	JSR [†]	0.3176	0.7038	0.8225	0.5173	0.5563
	USER [†]	0.4123	0.7631	0.8697	0.6000	0.6348
	UnifiedSSR [†]	0.3663	0.7744	0.8812	0.5847	0.6196
	UniSAR	0.5343	0.8190	0.8977	0.6875	0.7132
	IntSR	0.5678	0.8266	0.8920	0.7091	0.7305

Negative sampling strategy	HR@1	HR@5	HR@10	N@5	N@10
RNS	0.1426	0.3691	0.4991	0.2592	0.3012
RNS (aligned)	0.1810	0.4269	0.5573	0.3075	0.3497
PNS (best)	0.1430	0.3655	0.4914	0.2576	0.2983
PNS (best, aligned)	0.1760	0.3949	0.5327	0.2817	0.3264
HNS	0.1569	0.3880	0.5150	0.2763	0.3173
HNS (aligned)	0.1842	0.4305	0.5601	0.3112	0.3533

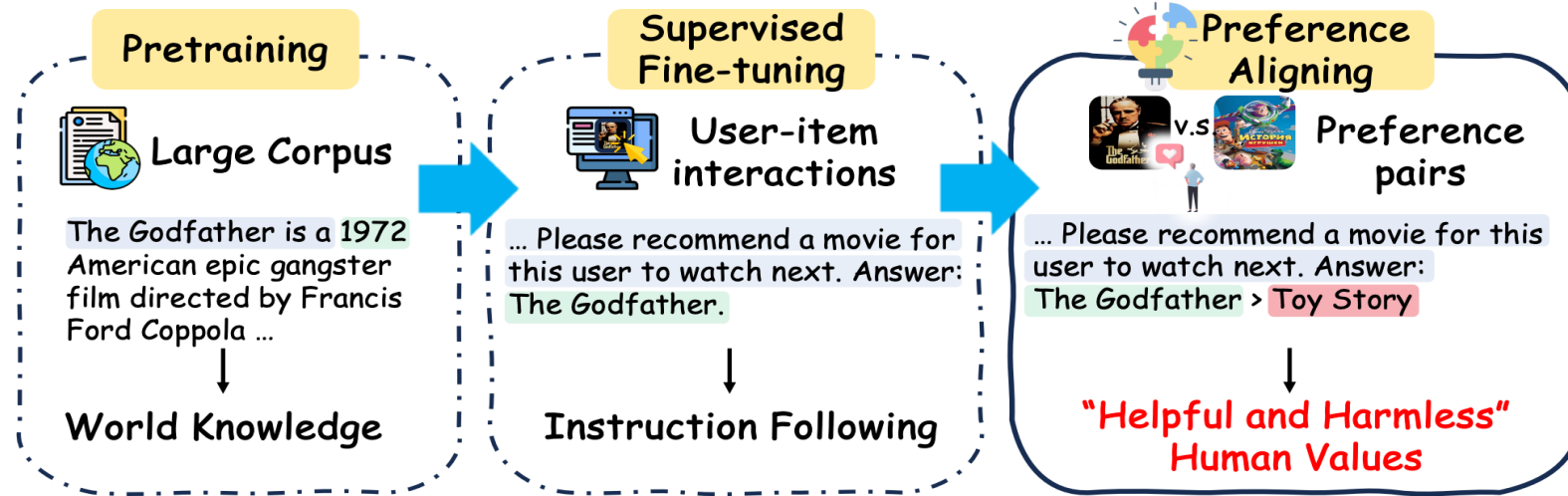
- **Practical Values:** Temporal candidate alignment is the key contribution, and aligned hard negative sampling gives the strongest result among the compared sampling strategies.

Supervised Learning Summary



August 9–13, 2026
Jeju, Korea

	NTP			NCE	
Axis	TIGER	LETTER	COBRA	PinRec	IntSR
Objective	NTP over Semantic IDs	Ranking-aware NTP	Composite supervision	Sampled-softmax NCE	Temporally aligned InfoNCE
Target	Next-item SID sequence	Learnable item tokens	Sparse ID + dense vector	Future item / outcome target	Query-time positive item
Alignment	Generation validity	Label & ranking alignment	Sparse-dense retrieval alignment	Industrial retrieval scalability	Candidate availability alignment
Lesson	Make retrieval generative	Improve token supervision	Enrich generated representation	Contrast positives with scalable negatives	Avoid false negatives from dynamic catalogs



Background

Preference alignment post-trains GenRec beyond behavior imitation.

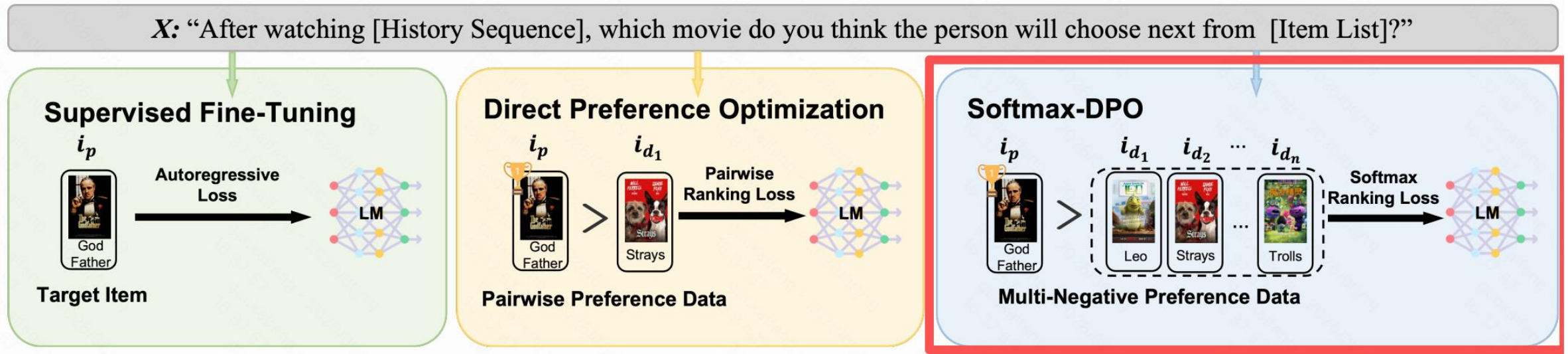
Motivation

NTP fits observed tokens, while platforms optimize delayed, list-level, and multidimensional preferences.

Taxonomy

DPO-based: chosen/rejected or multi-negative preference pairs.

GRPO-based: reward-scored lists with group-relative policy updates.



DPO category: pairwise preference becomes multi-negative ranking preference.

$$\mathcal{L}_{\text{S-DPO}}(\pi_{\theta}; \pi_{\text{ref}}) = -\mathbb{E}_{(x_u, e_p, \mathcal{E}_d) \sim \mathcal{D}} \left[\log \sigma \left(-\log \sum_{e_d \in \mathcal{E}_d} \exp \left(\beta \log \frac{\pi_{\theta}(e_d | x_u)}{\pi_{\text{ref}}(e_d | x_u)} - \beta \log \frac{\pi_{\theta}(e_p | x_u)}{\pi_{\text{ref}}(e_p | x_u)} \right) \right) \right]$$

Connection: the DPO reward can be written in a softmax-ranking form.

$$\mathcal{L}_{\text{softmax}} = -\mathbb{E}_{(u, i_p, \mathcal{I}_d)} \left[\log \sigma \left(-\log \sum_{i_d \in \mathcal{I}_d} \exp (f(u, i_d) - f(u, i_p)) \right) \right]$$

Result: S-DPO achieves the best HR@1 on Goodreads, LastFM, and MovieLens.

		Goodreads			LastFM			MovieLens		
		HR@1	ValidRatio	Rel.Ipv	HR@1	ValidRatio	Rel.Ipv	HR@1	ValidRatio	Rel.Ipv
Traditional	GRU4Rec	0.3867	1.0000	70.91%	0.2616	1.0000	153.36%	0.3750	1.0000	40.35%
	Caser	0.4174	1.0000	58.34%	0.2233	1.0000	196.82%	0.3861	1.0000	36.31%
	SASRec	0.3581	1.0000	84.56%	0.2233	1.0000	196.82%	0.3444	1.0000	52.82%
LM-based	LLaMA2	0.0233	0.3845	2736.48%	0.0246	0.3443	2594.31%	0.0421	0.4421	1150.12%
	ChatRec	0.3306	1.0000	99.91%	0.3770	1.0000	75.81%	0.2000	0.9895	163.15%
	MoRec	0.2877	1.0000	129.72%	0.1652	1.0000	301.21%	0.2822	1.0000	86.50%
	TALLRec	0.4983	0.9573	32.63%	0.4180	0.9836	58.56%	0.3895	0.9263	35.12%
	LLaRA	<u>0.5292</u>	0.9950	24.89%	<u>0.4508</u>	0.9918	47.03%	<u>0.4737</u>	0.9684	11.10%
Ours	S-DPO	0.6609	0.9900	-	0.6628	0.9992	-	0.5263	0.9895	-

Practical value: multi-negative DPO gives ranking-aware alignment beyond positive-only supervised fine-tuning.

Background

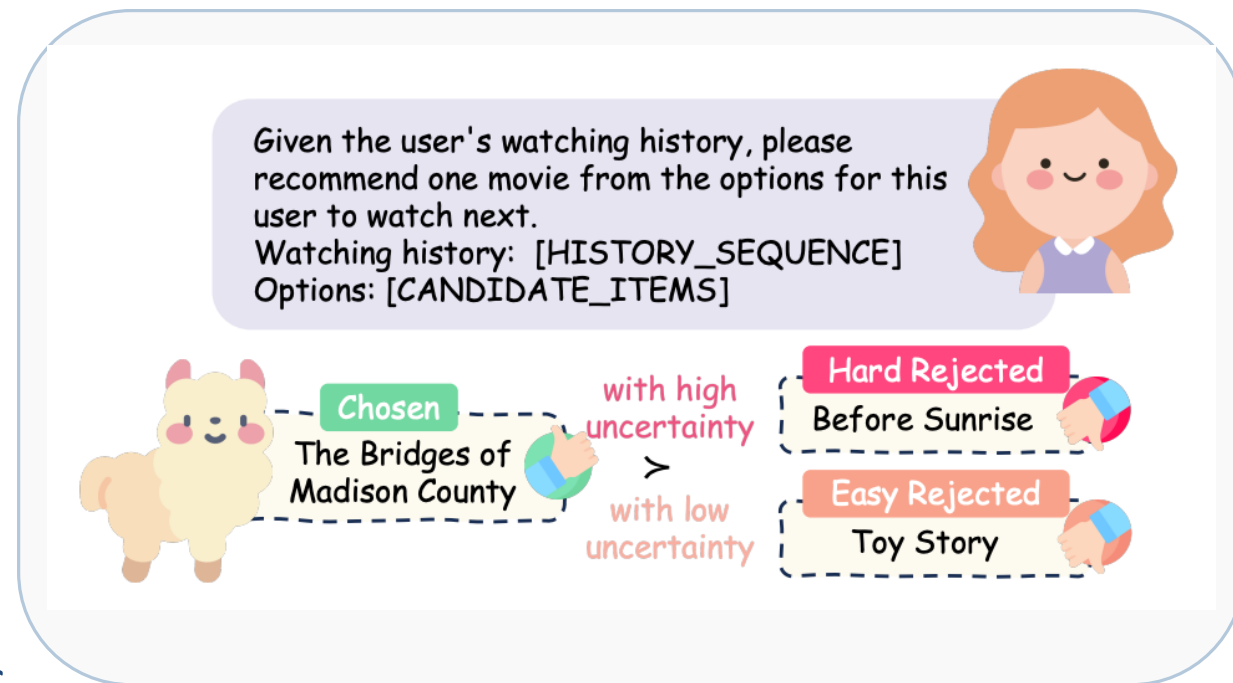
SFT fits only positive responses.

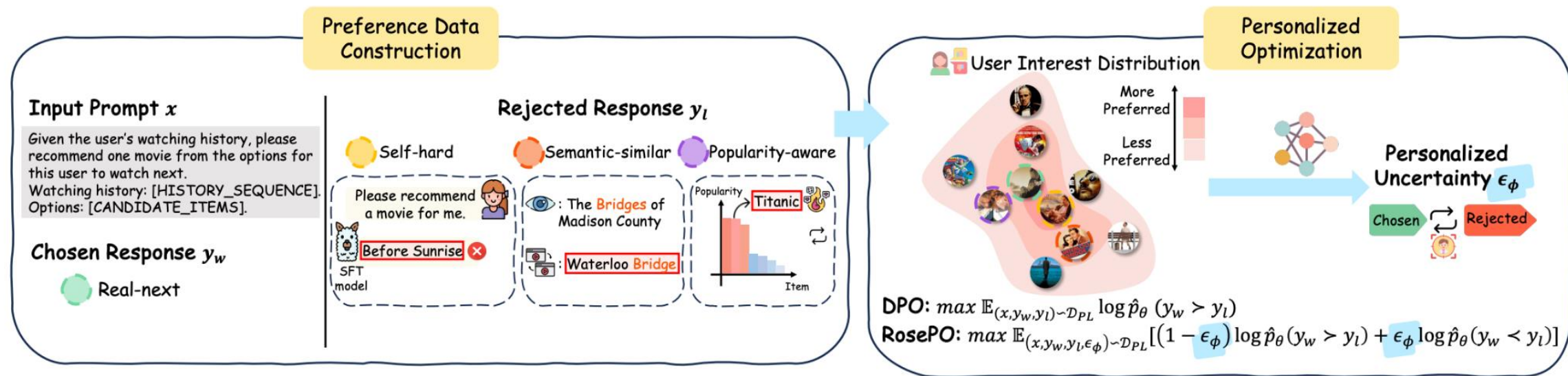
Motivation

DPO needs reliable chosen/rejected pairs;
weak negatives produce noisy preference signals.

Contribution

RosePO builds informative rejected samples
and smooths uncertain labels.





DPO objective

Learn to prefer the chosen response over the rejected response.

$$\begin{aligned} \hat{p}(y_w > y_l | x) &= \sigma(r(x, y_w) - r(x, y_l)) \\ &= \sigma \left(\beta \log \frac{\pi_\theta(y_w | x)}{\pi_{\text{ref}}(y_w | x)} - \beta \log \frac{\pi_\theta(y_l | x)}{\pi_{\text{ref}}(y_l | x)} \right), \end{aligned}$$

RosePO objective

Personalized smoothing avoids over-penalizing uncertain rejected samples.

$$\max_{\theta} \mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}_{PL}} \log \hat{p}(y_w > y_l | x).$$

Main results

RosePO-h consistently outperforms SFT and prior baselines.

Model	Movielens					Goodreads					Steam				
	HR@1	HR@5	N@5	HR@10	N@10	HR@1	HR@5	N@5	HR@10	N@10	HR@1	HR@5	N@5	HR@10	N@10
GRU4Rec	0.2767	0.6200	0.4614	0.7100	0.4902	0.4048	0.7852	0.6087	0.9187	0.6523	0.3610	0.7324	0.5542	0.8886	0.6050
Caser	0.3783	0.6383	0.5143	0.7267	0.5430	0.4161	0.8128	0.6284	0.9187	0.6630	0.3810	0.7587	0.5803	0.8879	0.6225
SASRec	0.2817	0.6183	0.4598	0.7184	0.4928	0.4102	0.7770	0.6067	0.9032	0.6475	0.3927	0.74856	0.5817	0.8699	0.6211
MoRec	0.2050	0.5083	0.3638	0.6716	0.4168	0.3152	0.7025	0.5147	0.8635	0.5670	0.3942	0.7617	0.5918	0.8857	0.6323
Chat-REC	0.2105	0.6842	0.4534	0.8211	0.4978	0.3694	0.7088	0.5412	0.8336	0.5814	0.4012	0.7306	0.5785	0.8370	0.6128
LLaRA	0.4379	0.7895	0.6285	0.8884	0.6602	0.5158	0.8060	0.6747	0.8859	0.7005	0.4927	0.8749	0.6988	0.9511	0.7236
PT	0.0968	0.3074	0.2032	0.5347	0.2753	0.1191	0.3418	0.2320	0.5820	0.3090	0.0868	0.3506	0.2214	0.5793	0.2947
SFT	0.4674	0.8800	0.6865	0.9537	0.7111	0.5165	0.8692	0.7053	0.9667	0.7373	0.5105	0.8788	0.7100	0.9548	0.7348
RosePO-h	0.5053	0.8926	0.7144	0.9537	0.7348	0.5398	0.8805	0.7221	0.9717	0.7520	0.5322	0.8836	0.7239	0.9619	0.7494

Sampling ablation

Self-hard negatives perform best, showing that informative rejected samples are crucial.

Model	Movielens					Goodreads					Steam				
	HR@1	HR@5	N@5	HR@10	N@10	HR@1	HR@5	N@5	HR@10	N@10	HR@1	HR@5	N@5	HR@10	N@10
uniform	0.4632	0.8695	0.6850	0.9305	0.7056	0.5285	0.8799	0.7171	0.9671	0.7458	0.5283	0.8811	0.7218	0.9592	0.7472
semantic	0.4400	0.8863	0.6843	0.9474	0.7042	0.5314	0.8819	0.7206	0.9661	0.7482	0.5121	0.8836	0.7143	0.9563	0.7379
popular	0.4758	0.8842	0.6981	0.9495	0.7196	0.5328	0.8812	0.7198	0.9657	0.7477	0.5261	0.8836	0.7219	0.9599	0.7467
self-hard	0.5053	0.8926	0.7144	0.9537	0.7348	0.5398	0.8805	0.7221	0.9717	0.7520	0.5322	0.8836	0.7239	0.9619	0.7494

Background

DPO-style alignment refines user preference beyond SFT.

Motivation

Vanilla DPO can become mode-seeking and over-recommend popular items.

Contribution

SPRec uses self-play negatives to suppress bias while keeping accuracy.

▪ **Token-level biases** in LLM-based recommendations



The Shawshank
Redemption



The
Godfather



The Silence
of the Lambs



The
Terminator



The
Prestige



Favoring movies titles with frequent tokens, e.g., “the”.

▪ **Item-level biases** in fine-tuned LLMs



The Dark
Knight



Batman



Batman
Begins



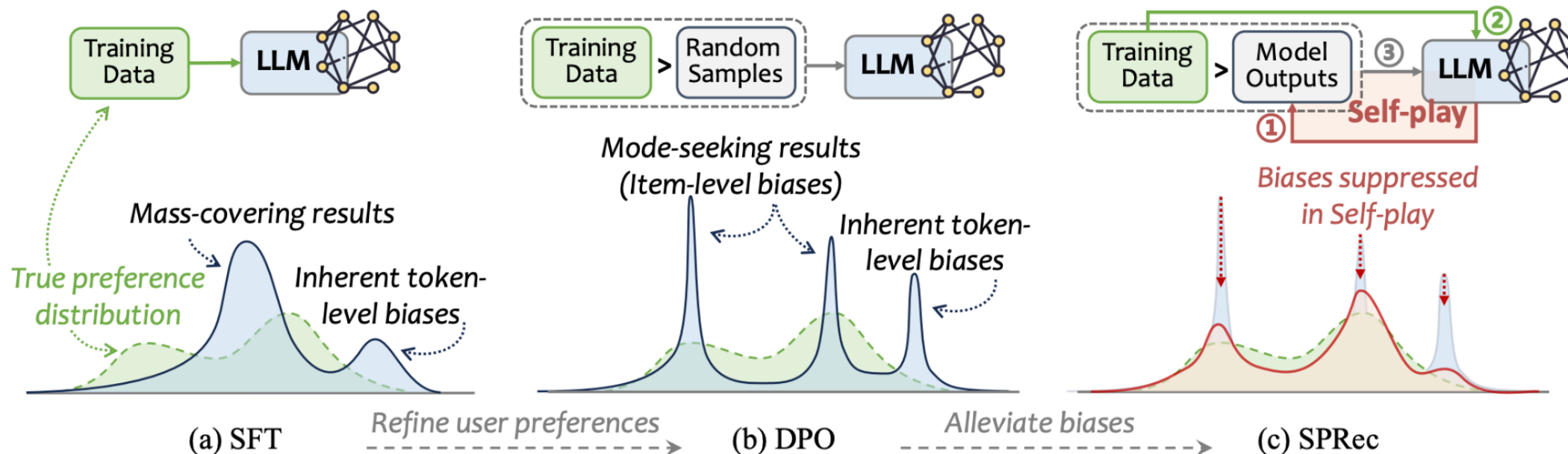
Batman
Returns



The Dark
Knight Rises



Focusing exclusively on the Batman film series.



Self-play update

The previous policy generates rejected samples; the next policy prefers real positives over biased outputs.

$$\pi_{\theta_{t+1}} \leftarrow \arg \max_{\pi_{\theta}} \mathbb{E}_{(x, y_w) \sim D, y_l \sim \pi_{\theta_t}(\cdot|x)} l(\pi_{\theta}; \pi_{\theta_t}; x, y_w, y_l).$$

Bias-aware DPO objective

High-probability biased outputs receive stronger correction.

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

Preference Alignment - DPO: SPRec



August 9-13, 2026
Jeju, Korea

Dataset	Model	DivRatio↑	ORRatio↓	MGU↓	HR↑	NDCG↑		SASRec	0.0021	1.0000	0.1205	0.0226	0.0170
MovieLens	SASRec	0.0031	1.0000	0.1209	0.0225	0.0136	CDs_and_Vinyl	BIGRec	0.3198	0.2597	0.0461	0.0132	0.0130
	BIGRec	0.1939	0.2561	0.0620	0.0347	0.0281		RW	0.2821	0.2648	0.0432	0.0112	0.0103
	RW	0.1918	0.2551	0.0577	0.0327	0.0276		D ³	0.3135	0.2672	0.0320	0.0103	0.0088
	D ³	0.1246	0.3238	0.0664	0.0266	0.0197		DMPO	0.3116	0.1740	0.0195	0.0090	0.0090
	DMPO	0.1827	0.2561	0.0529	0.0310	0.0264		SDPO	0.3218	0.2118	0.0380	0.0110	0.0106
	SDPO	0.1816	0.2449	0.0462	0.0310	0.0258		RosePO	0.3625	0.1853	0.0426	0.0090	0.0090
	RosePO	0.1857	0.2378	0.0538	0.0290	0.0244		SPRec	0.3859	0.1670	0.0242	0.0143	0.0140
	SPRec	0.2806	0.1510	0.0432	0.0388	0.0319		SASRec	0.0010	1.0000	0.1094	0.0660	0.0379
	SASRec	0.0030	1.0000	0.0458	0.0202	0.0139		BIGRec	0.1940	0.3910	0.0650	0.0780	0.0766
	BIGRec	0.1420	0.3170	0.0175	0.0310	0.0236		RW	0.2890	0.2710	0.0313	0.0760	0.0735
Goodreads	RW	0.1050	0.3730	0.0238	0.0380	0.0281	Steam	D ³	0.1580	0.4560	0.0511	0.0730	0.0718
	D ³	0.1199	0.2581	0.0260	0.0413	0.0324		DMPO	0.2270	0.2990	0.0443	0.0850	0.0834
	DMPO	0.1560	0.3310	0.0164	0.0410	0.0314		SDPO	0.2080	0.3510	0.0475	0.0820	0.0810
	SDPO	0.1580	0.3270	0.0161	0.0420	0.0315		RosePO	0.2310	0.3160	0.0499	0.0820	0.0805
	RosePO	0.1860	0.3230	0.0181	0.0300	0.0215		SPRec	0.2930	0.2560	0.0367	0.0910	0.0893
	SPRec	0.2090	0.2170	0.0099	0.0330	0.0250							

Key observations: best diversity, lowest over-recommendation, and competitive HR / NDCG.

Background

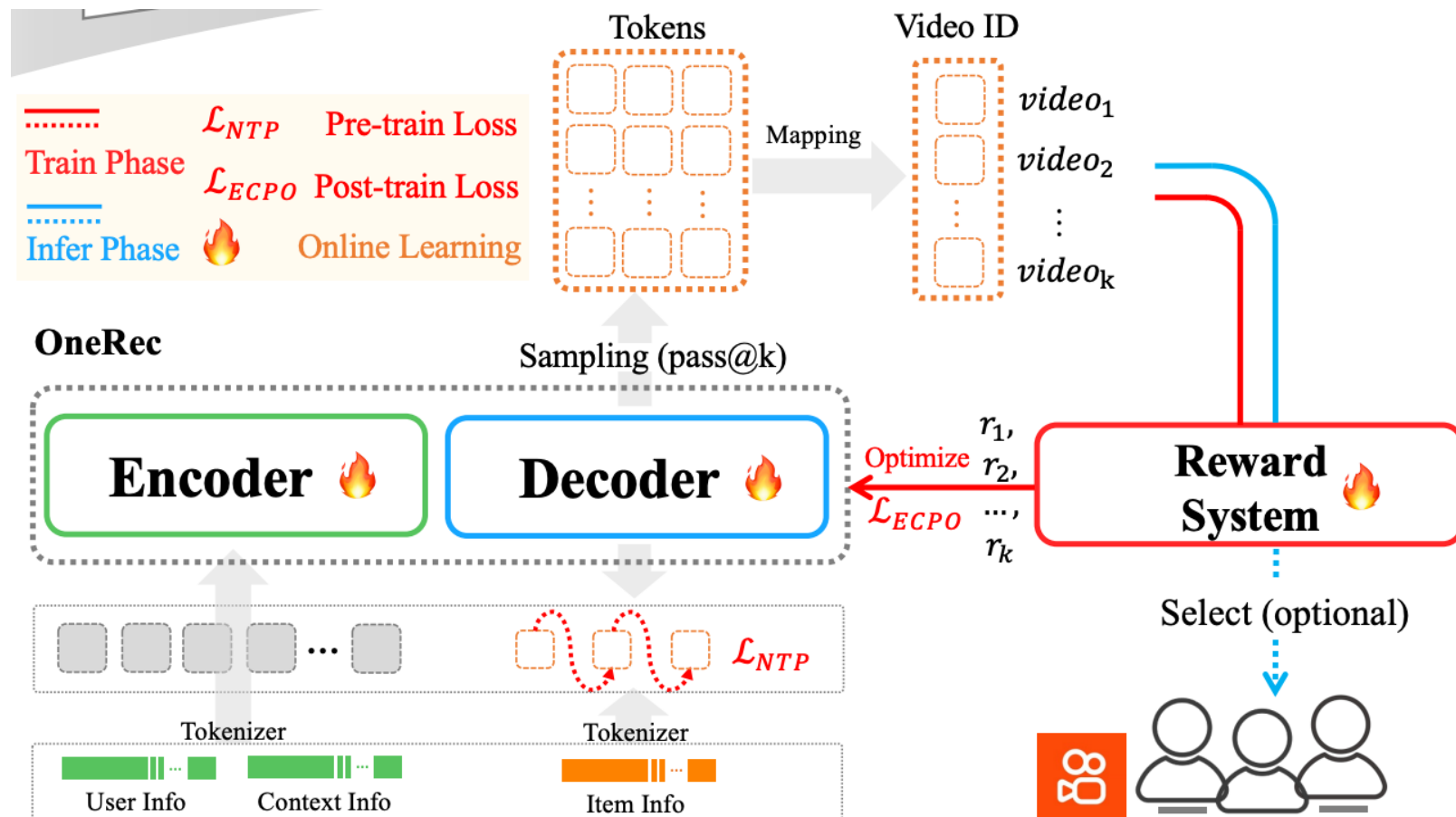
Industrial recommendation is a multi-stage cascade.

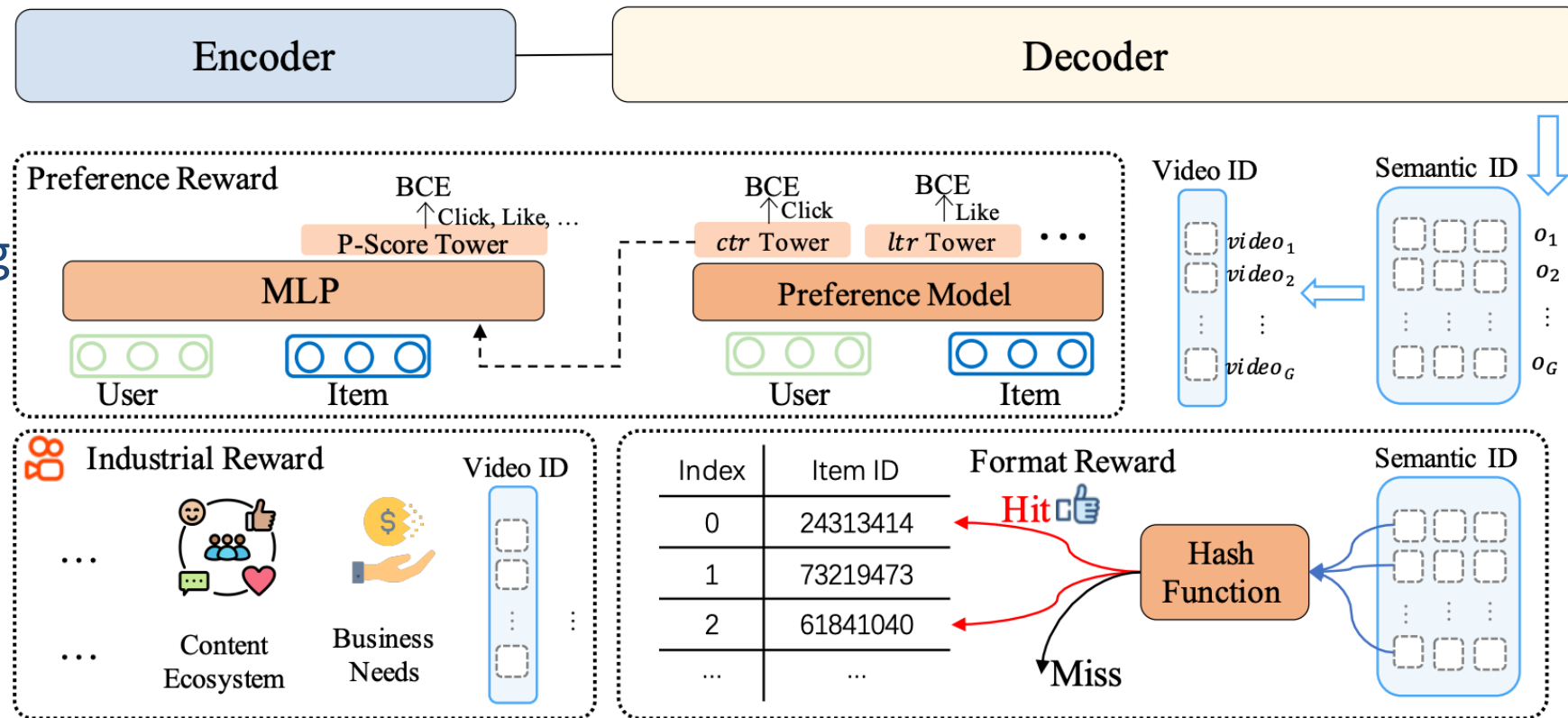
Motivation

Fragmented objectives make final user experience hard to optimize directly.

Contribution

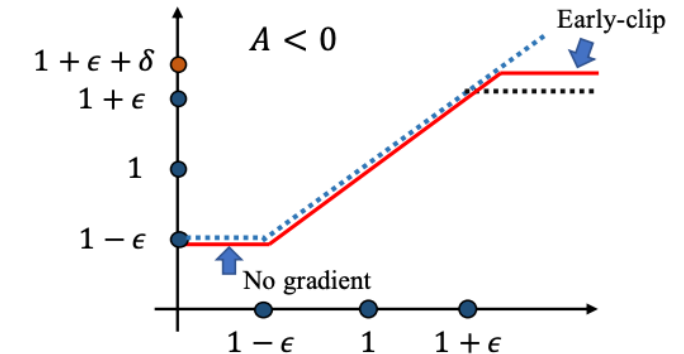
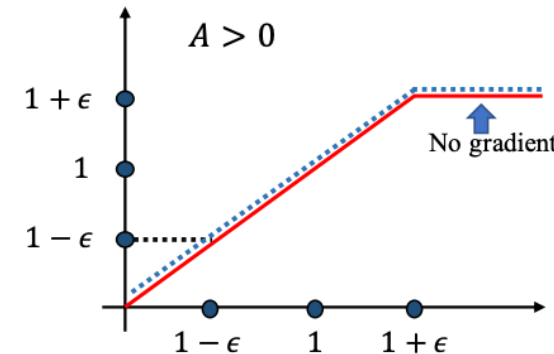
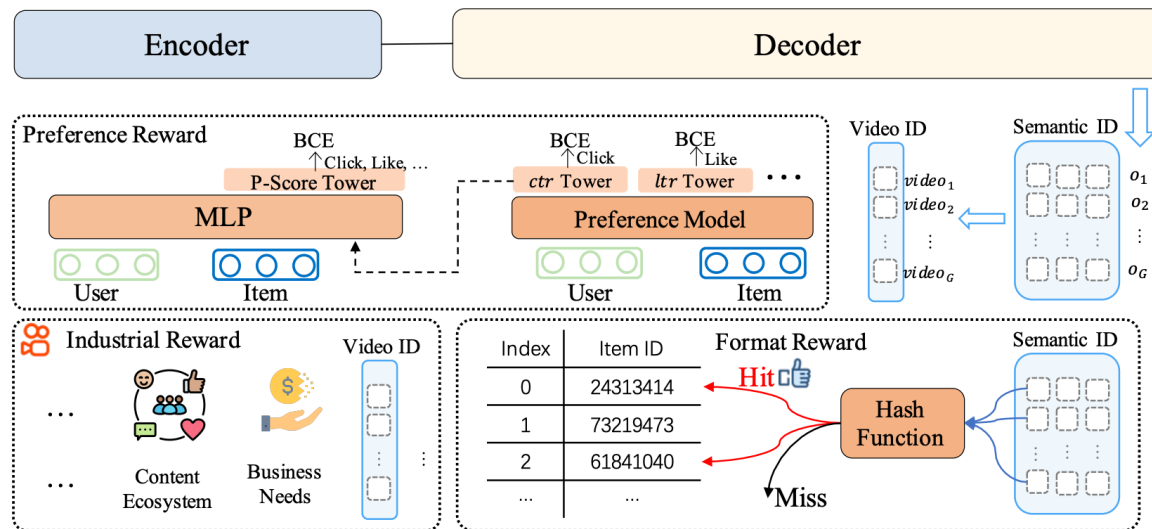
OneRec unifies retrieval/ranking and aligns generated lists with rewards.





$$\mathcal{J}_{ECPO}(\theta) = \mathbb{E}_{u \sim P(U), \{o_i\}_{i=1}^G \sim \pi_{\theta_{old}}} \left[\frac{1}{G} \sum_{i=1}^G \min \left(\frac{\pi_{\theta}(o_i|u)}{\pi'_{\theta_{old}}(o_i|u)} A_i, \text{clip} \left(\frac{\pi_{\theta}(o_i|u)}{\pi'_{\theta_{old}}(o_i|u)}, 1 - \epsilon, 1 + \epsilon \right) A_i \right) \right],$$

$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \dots, r_G\})}{\text{std}(\{r_1, r_2, \dots, r_G\})},$$



..... GRPO — ECPO

ECPO optimization

OneRec modifies GRPO with early clipping to stabilize RL and prevent risky probability increases for negative-advantage items.

$$\pi'_{\theta_{old}}(o_i|u) = \max\left(\frac{\text{sg}(\pi_{\theta}(o_i|u))}{1 + \epsilon + \delta}, \pi_{\theta_{old}}(o_i|u)\right), \quad \delta > 0,$$

	Method	vtr	Watch time	App Stay Time	Video View ¹
Pass@32	OneRec w/o RL	0.1978	+1.62%	-0.10%	-4.18%
	OneRec w/ RL	0.2138	+3.17%	+0.39%	-9.87%
	Relative Impr.	+8.08%	+1.55%	+0.49%↑↑↑	-3.69%
Pass@128	OneRec w/o RL	0.2239	+4.61%	+1.11%	-12.75%
	OneRec w/ RL	0.2387	+5.22%	+1.49%	-15.06%
	Relative Impr.	+6.61%	+1.53%	+0.38%↑↑	-2.65%
Pass@512	OneRec w/o RL	0.2444	+6.32%	+1.66%	-15.54%
	OneRec w/ RL	0.2494	+5.88%	+1.75%	-13.88%
	Relative Impr.	+2.05%	-0.41%	+0.09%↑	+1.97%

Key observations

RL improves VTR and stay time; beam search performs strongest.

	vtr	Watch time	App Stay Time	Video View ¹
Top- <i>k</i> +Top- <i>p</i>	0.2131	+4.45%	+1.16%	-13.61%
Beam Search	0.2162	+5.35%	+1.76%	-13.30%
Relative Impr.	+1.45%	+0.87%	+0.60%	+0.36%

Embarrassment: Thinking < Non-Thinking



奥迪汽车
pid: 176702595862



超跑
pid: 170269580415

模型思考过程 1

<think>视频观看历史显示用户对汽车保养、车型评测及优惠活动高度关注，近期观看的“<s_a_3313><s_b_4487>”相关视频可能涉及二手车交易、品牌对比或购车策略；视频内容中“短剧”标签与“运动”标签的高频出现，暗示用户对生活类内容和运动类视频有较强兴趣，但未明确关联到汽车领域。用户偏好通过“<s_a_456>”字标签指向汽车配件/装饰品，结合“<s_a_5184>”字标签可能与电子产品相关，形成跨品类兴趣链。用户兴趣呈现三重特征：1) 汽车领域（<s_a_3313><s_b_4487>、<s_a_456>、<s_a_1966>等标签）与生活类内容（短剧、运动）的交叉，反映对实用价值与娱乐性兼具的内容需求；2) 对汽车保养、优惠信息的关注，暗示潜在购车或维修需求；3) 跨品类兴趣（汽车配件与电子产品）表明用户可能具备多元消费能力，但需通过精准内容匹配提升转化率。推荐策略应兼顾汽车知识科普、优惠活动推送及生活类短视频，强化用户场景化兴趣联动。

</think>

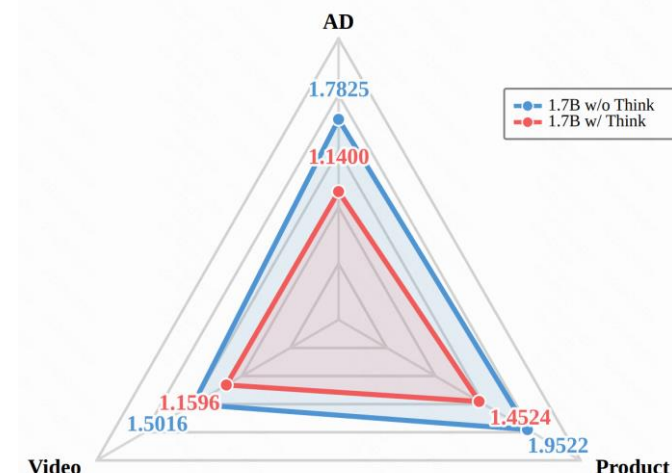
因此用户接下来可能喜欢的视频是<sid_begin><s_a_1966><s_b_2581><s_c_6305><sid_end>

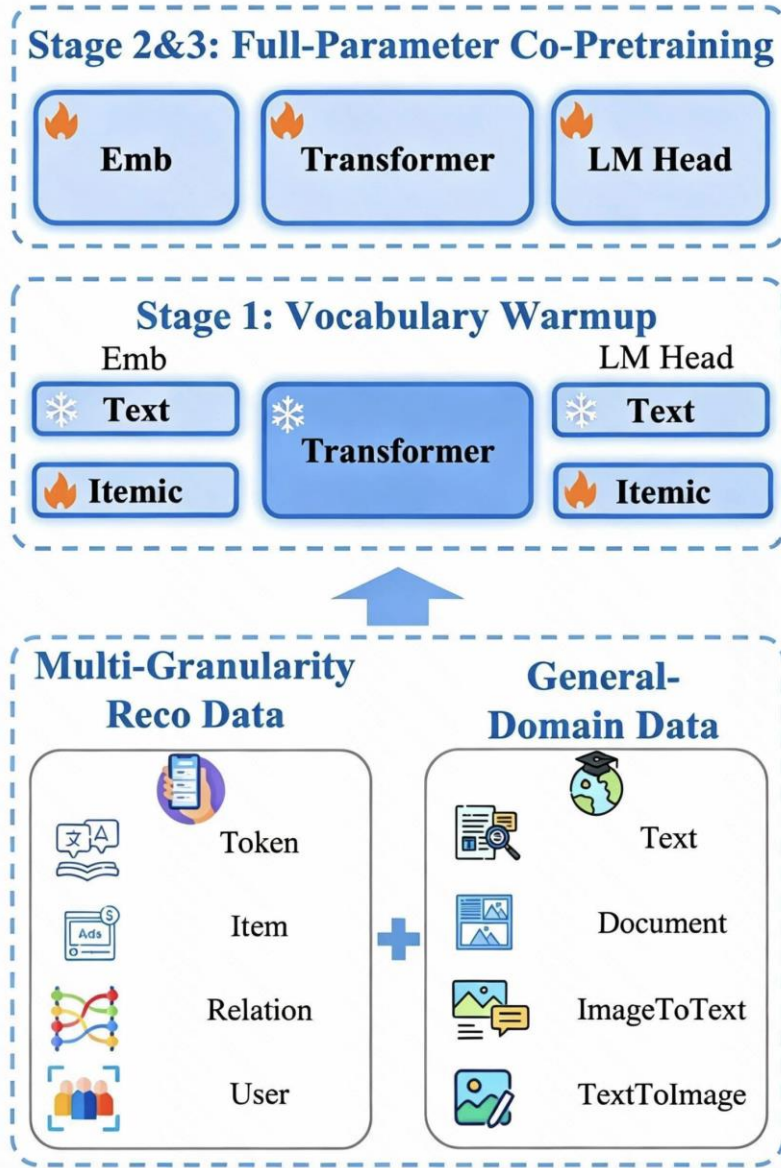


劳斯莱斯
pid: 176572517042

Embarrassing Phenomenon:

thinking is not significantly better than non-thinking on recommendation benchmarks.





Core Objective of Pre-training:

1. Build deep bidirectional alignment between recommendation items and natural language.
2. Adopt a three-stage progressive training schedule

Tokenizer for Unified Cross-Domain Representation

1. Quantization: 3-level RQ-Kmeans codebooks (8192 entries each) enable a 500-billion representation space.
2. Token Format
3. Unified multi-domain item representation for enhanced cross-domain generalization.

Preference Alignment - GRPO: OneReason

Pretrain: Perception Alignment

➤ Four-Granularity Alignment

➤ Three-Stage Recipe

SFT: Cognitive Structuring

➤ R0~R3: Data Curation&CoT training

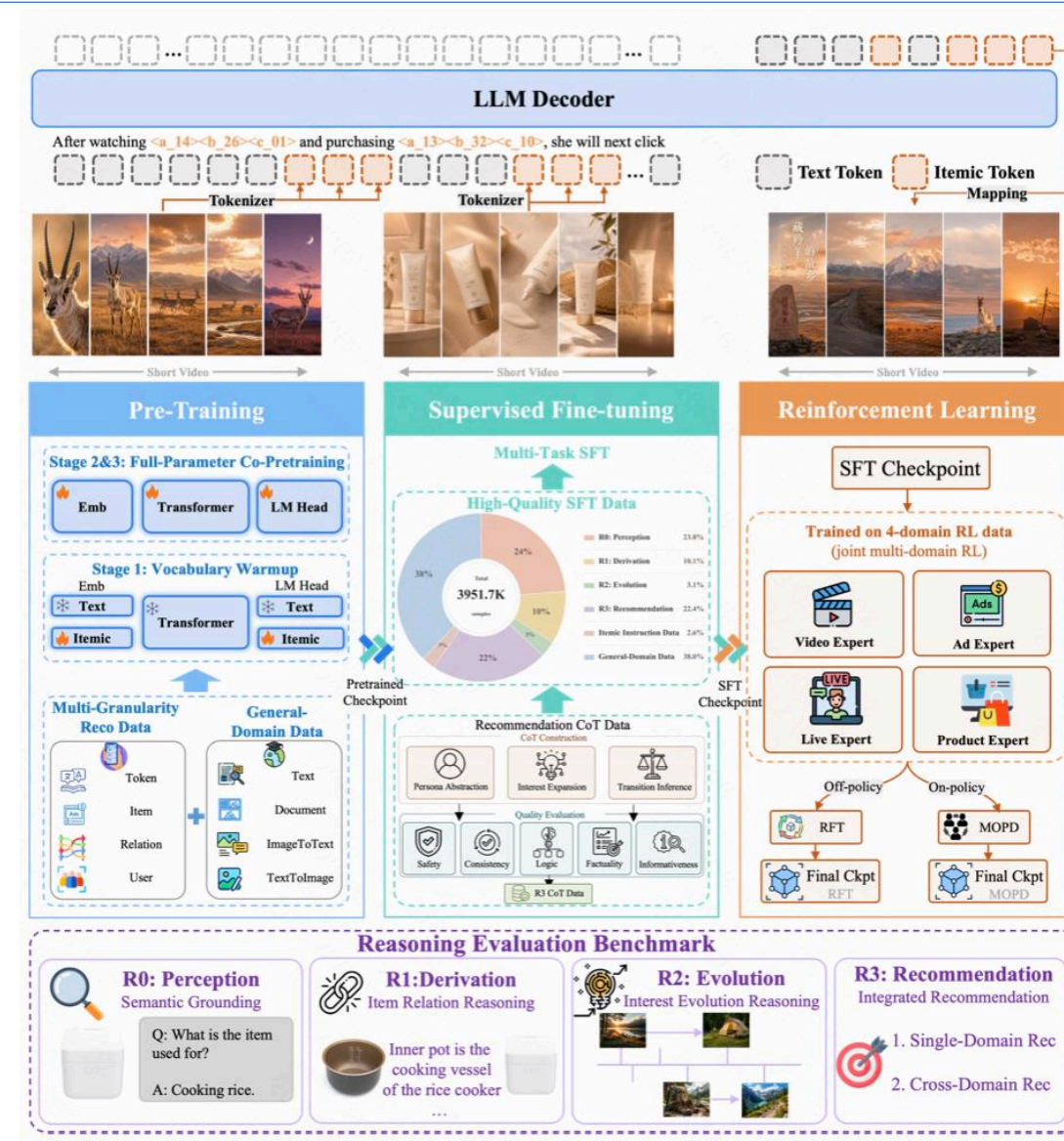
➤ Recommendation three-stage reasoning CoT

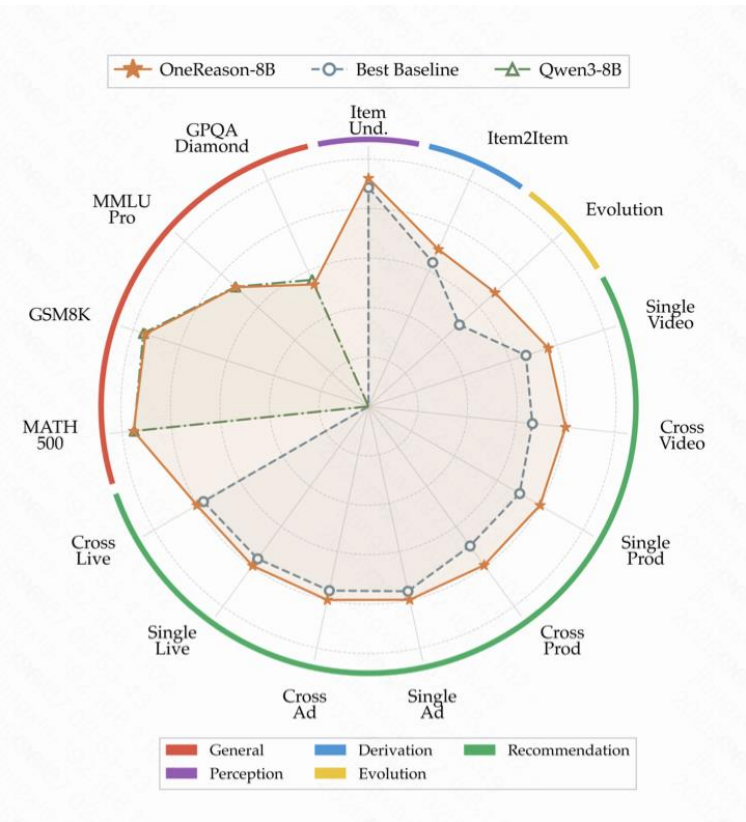
➤ Five-Dimension Evaluation

RL: unlock thinking capability

➤ Encourage the model to explore CoT

➤ GRPO Improvement: reward sparsity





Category	Model	Cross-Video		Cross-Product		Cross-Ad		Cross-Live	
		Pass@64	Recall@64*	Pass@64	Recall@64	Pass@64	Recall@64	Pass@64	Recall@64
ID-Based	SASRec	0.03	0.01	0.31	0.25	1.04	0.37	1.76	0.40
	HSTU	0.10	0.01	0.32	0.24	2.79	0.78	2.32	2.14
Text-Based	Qwen3-8B	0.05	0.01	0.15	0.12	0.48	0.09	2.10	1.85
	Qwen3-32B	0.33	0.03	0.84	0.63	1.21	0.30	5.64	5.10
	Qwen3-235B-A22B	0.24	0.02	0.64	0.49	0.77	0.19	5.10	4.66
	Deepseek-V3.2	0.11	0.01	0.38	0.31	0.62	0.13	3.46	3.12
	Claude-Opus-4.6	0.14	0.01	0.23	0.17	0.50	0.11	3.02	2.66
	Gemini-3-Preview	0.29	0.03	0.74	0.59	1.22	0.27	3.92	3.44
	GPT-4o-mini	0.19	0.02	0.73	0.55	1.21	0.28	4.01	3.57
	GPT-5.4	0.24	0.02	1.43	1.15	1.64	0.43	7.20	6.38
Itemic Token-Based	TIGER	0.88	0.07	0.21	0.17	7.65	2.39	2.32	1.78
	LC-Rec-SFT-Only-8B	0.22	0.02	0.06	0.05	2.83	0.67	0.89	0.71
	LC-Rec-SFT-Only-14B	0.20	0.01	1.03	0.73	5.99	1.94	3.76	3.09
	LC-Rec-PT-SFT-8B	1.49	0.13	3.95	3.00	15.85	6.55	19.32	16.70
	OneReason SFT non-thinking	1.33	0.11	3.94	2.96	15.73	6.49	18.05	15.52
	OneReason SFT thinking	0.71	0.06	2.18	1.65	9.16	3.41	16.43	14.32
	OneReason RFT non-thinking	2.08	0.19	5.20	3.96	17.56	7.26	21.01	18.17
	OneReason RFT thinking	2.41	0.24	5.47	4.19	17.78	7.50	21.10	18.35

* Video Recall@64 is notably low due to the largest target set (avg. 13.92 items) making full recall harder.

Compared with Qwen3-8B,
maintain general knowledge
capability

Recommendation-capability
advantage is significant

Two categories differ in supervision signal and optimization path.

DPO-based alignment

- Signal: chosen/rejected or multi-negative preferences
- Update: direct preference objective without online RL rollout
- Works: S-DPO, RosePO, SPRec

GRPO-based alignment

- Signal: reward-scored generated item lists
- Update: group-relative policy optimization with clipping and guardrails
- Work: OneRec, OneReason

DPO is pairwise/ranking preference alignment;
GRPO is reward-driven policy alignment.

Core insight: GenRec optimization should jointly connect token likelihood, preference categories, rewards, and serving constraints.

1 Objective-aware supervision

Align labels, negatives, and targets with ranking and list quality.

2 List / session-level credit

Assign rewards across positions, items, and SID tokens.

3 Reliable rewards and evaluation

Calibrate clicks, watch time, and GMV with uncertainty and off-policy checks.

4 Safe and efficient policy updates

Keep valid SIDs while controlling latency, QPS, and diversity.

1 Introduction (9:00 - 9:15)



Xiaopeng Li



2 Tokenization (9:15 - 9:45)



Xiaopeng Li

3 Model Structure (9:45 - 10:30)



Ziwei Liu



15mins

4 Optimization (10:45 - 11:15)



Wenlin Zhang



Kaifeng Guo

5 Application (11:15 - 11:30)



Yejing Wang



6 Challenge & Conclusion (11:30 - 11:45)



Bo Chen

Two Dimensions of Application

A. Cascaded Systems

Retrieval: TBGRecall

Ranking: HLLM

Re-ranking: KC-GenRec

End-to-End: (covered in Sec III-V)

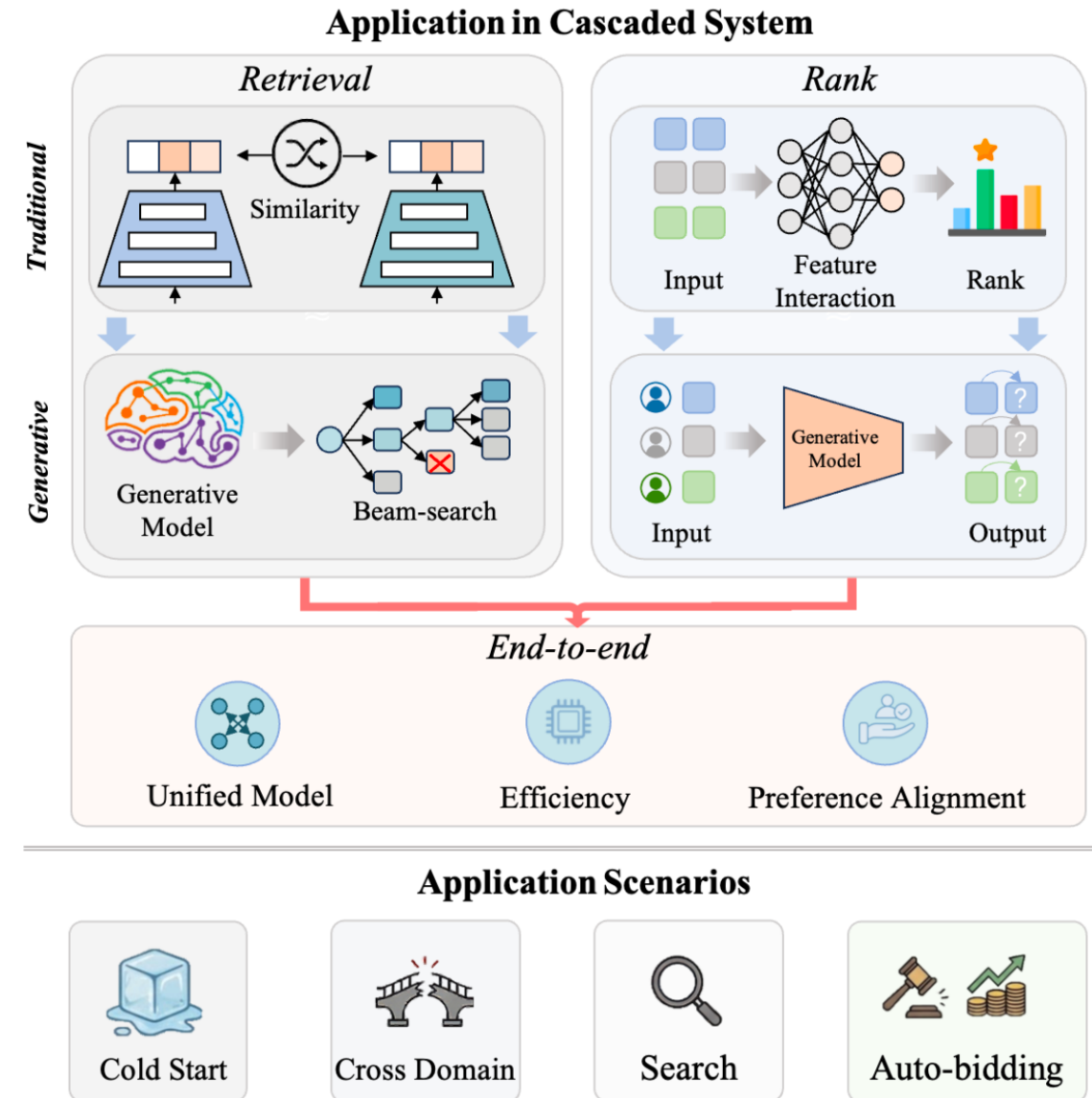
B. Various Scenarios

Cold Start: LC-REC

Cross Domain: GenCDR

Search: OneSearch, GenSAR

Auto-Bidding: GAVE



Cascaded Systems: Overview



KDD2026
August 9–13, 2026
Jeju, Korea

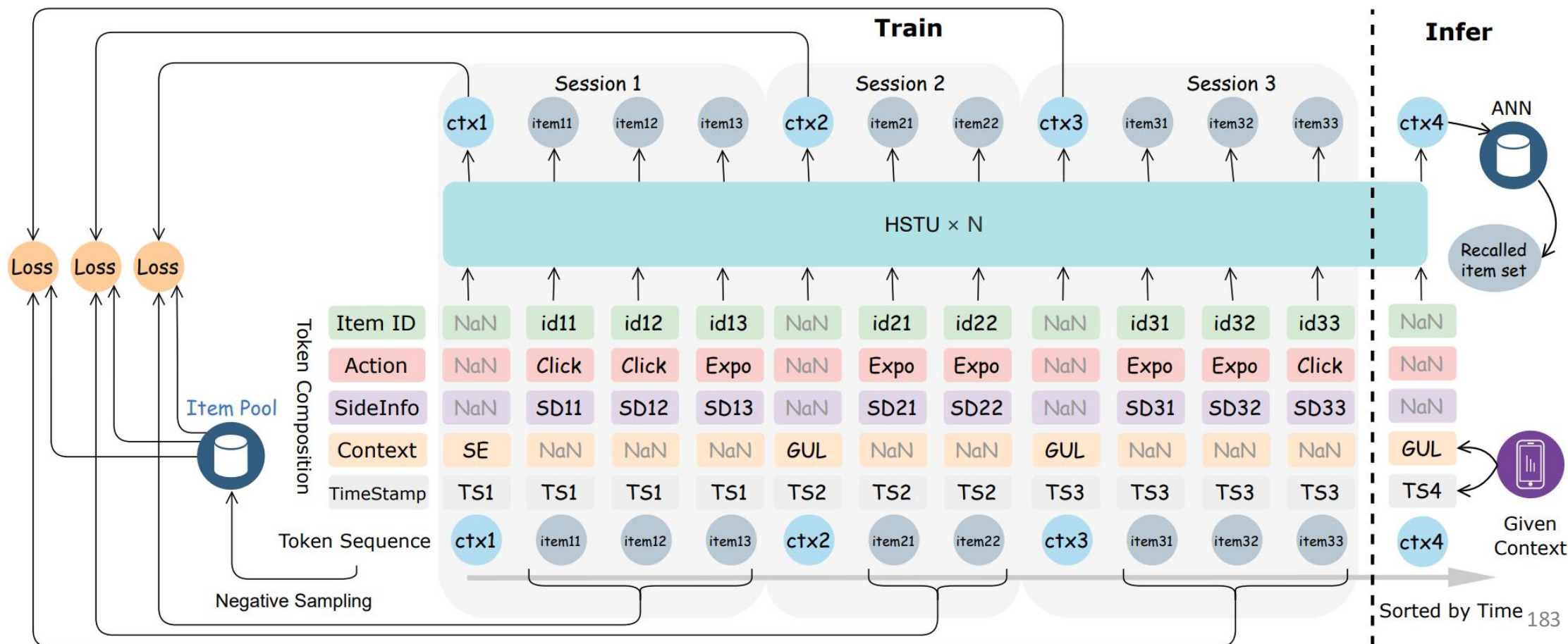
Topic	Work	Key Contribution
Retrieval	Kuaiformer	Multi-interest query tokens for diverse user interests
Retrieval	TBGRecall	Outcome-conditioned retrieval for e-commerce
Ranking	DFGR	Single/dual-stream generative ranking networks
Ranking	HLLM	Hierarchical LLMs for item & user modeling
Ranking	HSTU	Action token interleaving; unifies retrieval & ranking; Batched inference with KV cache; MTIA 2i chip co-design
Re-ranking	GoalRank	Generator-only ranking; group-relative optimization
Re-ranking	SORT-Gen	List-level multi-objective generative re-ranking
Re-ranking	KC-GenRec	Knowledge graph constrained generative re-ranker
End-to-End	OneRec	Unified decoder-only model; ECPO; 5× MFU improvement
End-to-End	OneSug	Unified search & recommendation; Prefix2Query
End-to-End	ETEGRec	End-to-end learnable item tokenization
End-to-End	COBRA	Bridging generative & dense retrieval paradigms

Topic	Work	Key Contribution
Retrieval	Kuaiformer	Multi-interest query tokens for diverse user interests
Retrieval	TBGRecall	Outcome-conditioned retrieval for e-commerce
Ranking	DFGR	Single/dual-stream generative ranking networks
Ranking	HLLM	Hierarchical LLMs for item & user modeling
Ranking	HSTU	Action token interleaving; unifies retrieval & ranking; Batched inference with KV cache; MTIA 2i chip co-design
Re-ranking	GoalRank	Generator-only ranking; group-relative optimization
Re-ranking	SORT-Gen	List-level multi-objective generative re-ranking
Re-ranking	KC-GenRec	Knowledge graph constrained generative re-ranker
End-to-End	OneRec	Unified decoder-only model; ECPO; 5× MFU improvement
End-to-End	OneSug	Unified search & recommendation; Prefix2Query
End-to-End	ETEGRec	End-to-end learnable item tokenization
End-to-End	COBRA	Bridging generative & dense retrieval paradigms

- Conventional autoregressive generative retrieval assumes ordered sequential dependencies (like text generation).
- But in e-commerce, items within a session have NO causal ordering — users interact with items after exposure, not in a fixed sequence.
- This creates a fundamental mismatch: next-token prediction is wrong for unordered item sets.
- TBGRRecall proposes **Next Session Prediction (NSP)** to replace next-token prediction.

TBGRRecall: Method

- NSP partitions input into multi-session sequences; each session contains one session token + item tokens, enabling parallel generation of unordered item sets.
- Session Mask + sw-RoPE: causal mask combined with session mask (intra-session items invisible to each other); session-wise rotary position embedding.



- Offline (Taobao, 1.2T interactions): HR@4000 = 29.45% vs online baseline (DT) 26.45%.
- Offline (RecFlow, 30M interactions): HR@4000 = 8.20% vs HSTU baseline 7.38%.
- Taobao 'Guess You Like' homepage A/B: Capturing 23.94% of total homepage exposures, +2.16% transaction amount, +0.60% transaction count.
- Key takeaway: Session-level autoregression solves the ordering mismatch for e-commerce retrieval.

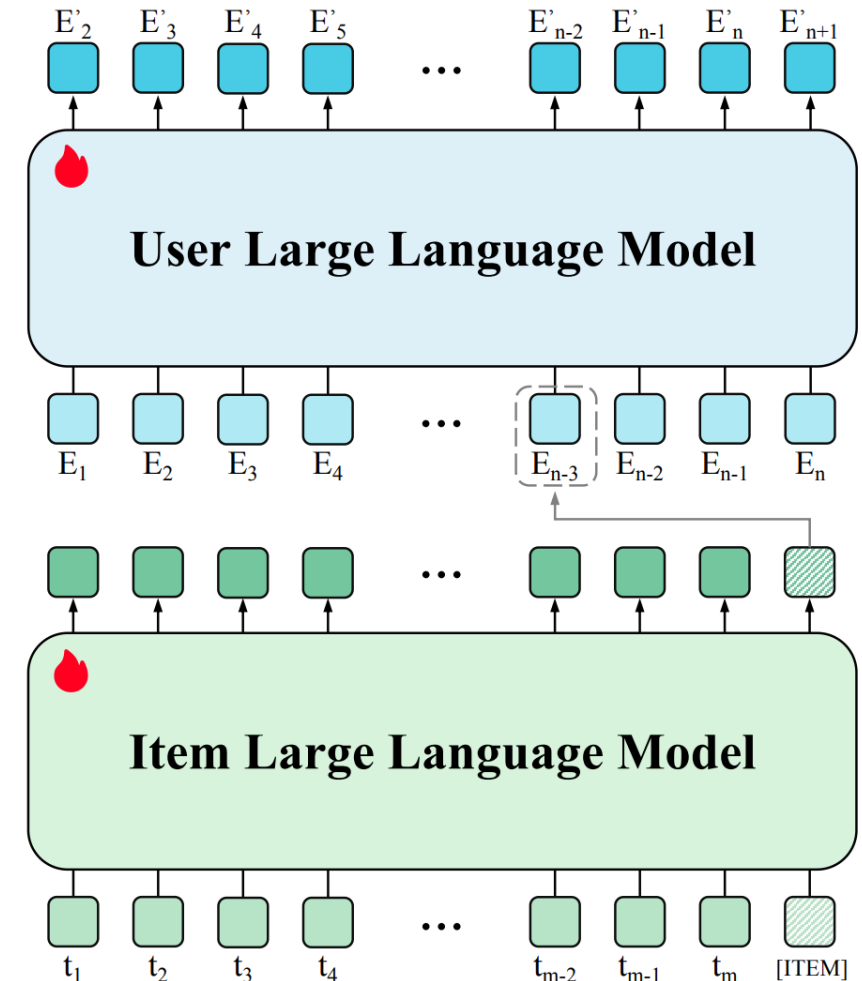
MODEL	RecFlow						TAOBao					
HR	@20	@100	@500	@1000	@2000	@4000	@20	@100	@500	@1000	@2000	@4000
YOUTUBEDNN	0.01%	0.08%	0.43%	1.02%	2.37%	3.28%	0.47%	1.29%	3.28%	4.75%	6.68%	9.22%
SASREC	0.04%	0.19%	1.00%	1.96%	3.66%	6.53%	0.50%	1.65%	4.47%	6.40%	8.79%	11.70%
BERT4REC	0.06%	0.24%	1.01%	1.87%	3.43%	6.09%	0.11%	0.44%	1.53%	2.50%	3.90%	5.96%
FDSA	0.02%	0.13%	0.54%	1.41%	3.90%	4.32%	0.52%	1.47%	3.70%	5.30%	7.44%	10.14%
HSTU	<u>0.06%</u>	<u>0.30%</u>	<u>1.21%</u>	<u>2.19%</u>	<u>4.06%</u>	<u>7.38%</u>	0.54%	1.78%	4.88%	7.01%	9.75%	13.11%
ONLINE	-	-	-	-	-	-	1.03%	3.53%	9.37%	13.36%	18.08%	23.57%
ONLINE(DT)	-	-	-	-	-	-	<u>1.26%</u>	<u>4.28%</u>	<u>11.03%</u>	<u>15.44%</u>	<u>20.62%</u>	<u>26.45%</u>
TBGRRECALL	0.26%	0.83%	2.45%	3.85%	5.61%	8.20%	1.53%	4.65%	11.81%	16.62%	22.57%	29.45%

PVR	TRANSACTION COUNT	TRANSACTION AMOUNT
23.94%	+0.60%	+2.16%

Topic	Work	Key Contribution
Retrieval	Kuaiformer	Multi-interest query tokens for diverse user interests
Retrieval	TBGRecall	Outcome-conditioned retrieval for e-commerce
Ranking	DFGR	Single/dual-stream generative ranking networks
Ranking	HLLM	Hierarchical LLMs for item & user modeling
Ranking	HSTU	Action token interleaving; unifies retrieval & ranking; Batched inference with KV cache; MTIA 2i chip co-design
Re-ranking	GoalRank	Generator-only ranking; group-relative optimization
Re-ranking	SORT-Gen	List-level multi-objective generative re-ranking
Re-ranking	KC-GenRec	Knowledge graph constrained generative re-ranker
End-to-End	OneRec	Unified decoder-only model; ECPO; 5× MFU improvement
End-to-End	OneSug	Unified search & recommendation; Prefix2Query
End-to-End	ETEGRec	End-to-end learnable item tokenization
End-to-End	COBRA	Bridging generative & dense retrieval paradigms

HLLM: Motivation & Context

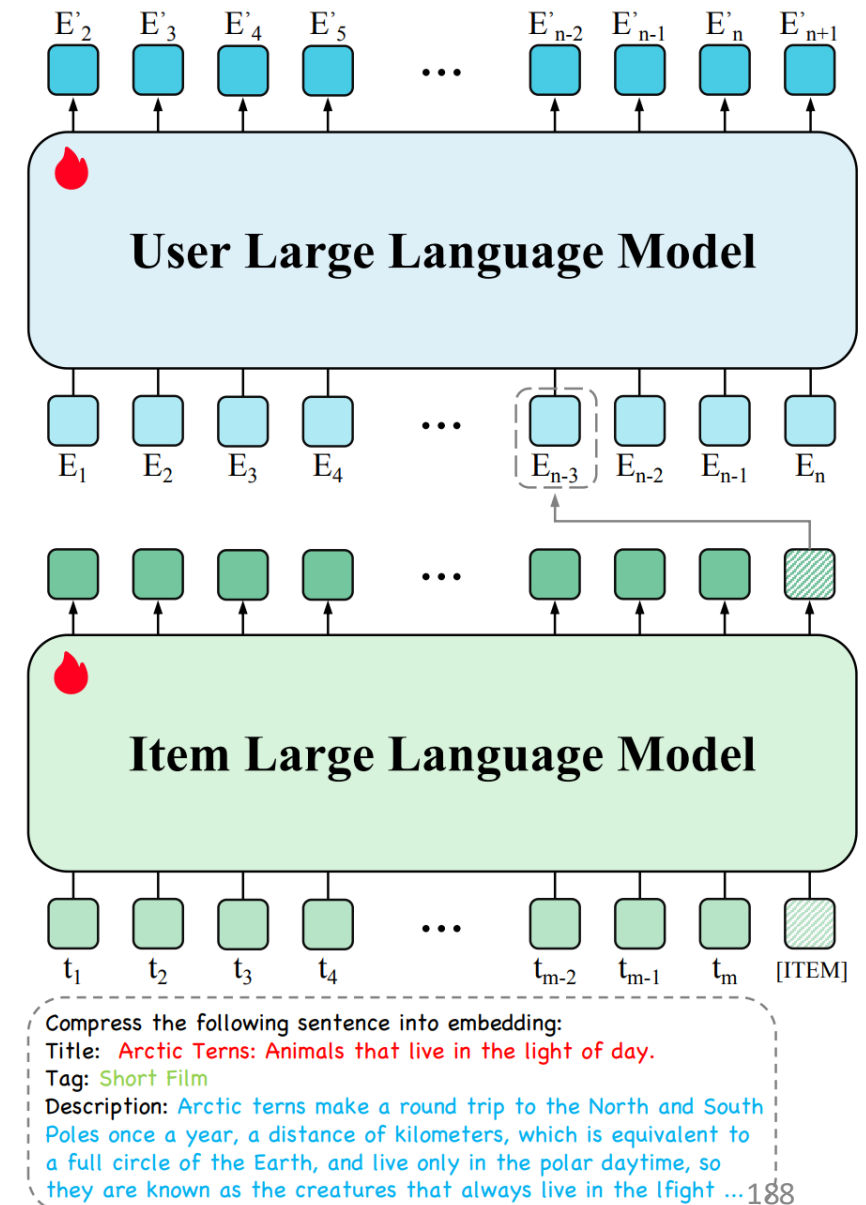
- Traditional ID-based sequential models (SASRec, BERT4Rec) struggle to capture rich semantic information from item content.
- LLMs have strong NLP capabilities but bridging text understanding and sequential behavior modeling is challenging.
- HLLM uses a hierarchical two-tier LLM architecture to leverage pre-trained knowledge for both items and users.



Compress the following sentence into embedding:
Title: Arctic Terns: Animals that live in the light of day.
Tag: Short Film
Description: Arctic terns make a round trip to the North and South Poles once a year, a distance of kilometers, which is equivalent to a full circle of the Earth, and live only in the polar daytime, so they are known as the creatures that always live in the light ...

HLLM: Method

- Hierarchical LLM architecture: one LLM models item features, another models user behavior sequences.
- **Item-level** LLM captures fine-grained semantic properties of each item: processing detailed item text (title, tags, description) and extracts content features via a [ITEM] token embedding; initialized from TinyLlama/Baichuan2.
- **User-level** LLM captures high-level preference patterns: taking sequences of item representations from Item LLM and modeling sequential behavior to predict the next item; also initialized from pre-trained LLMs.
- Performs top-K ranking directly through generative decoding rather than pointwise scoring.
- Both components are jointly trained end-to-end, allowing Item LLM to learn recommendation-specific representations.

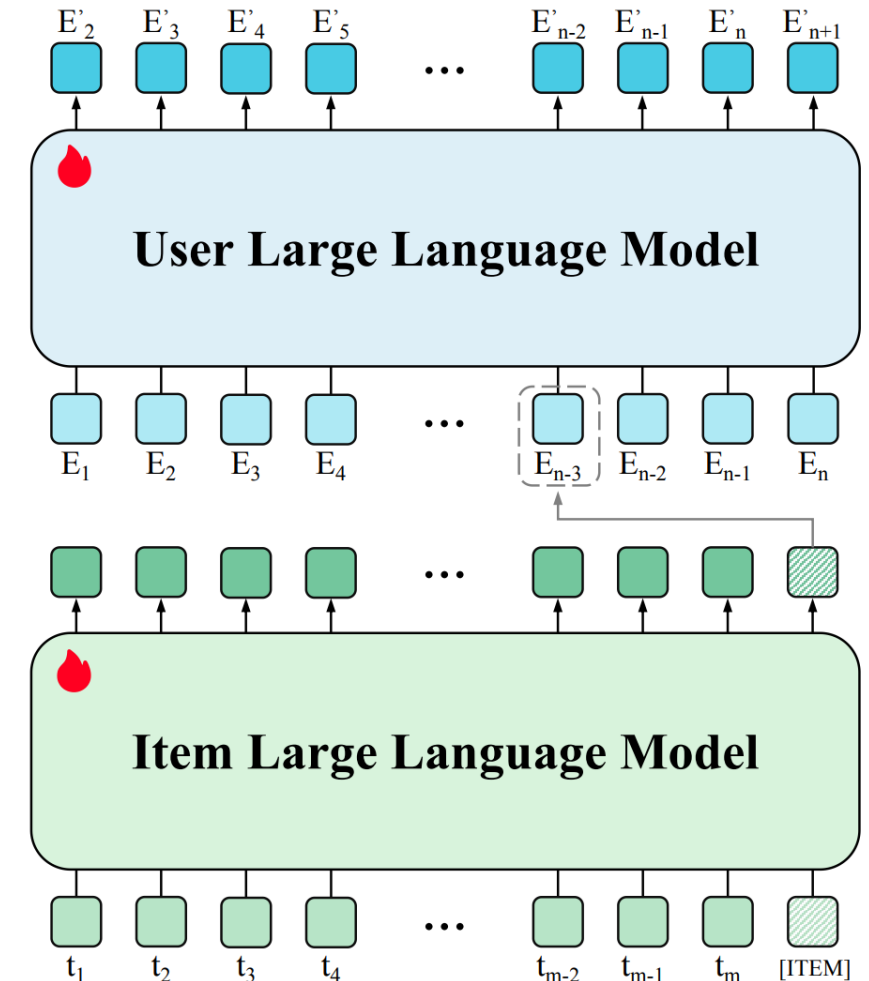


- Hierarchical LLM architecture: one LLM models item features, another models user behavior sequences.
- **Item-level** LLM captures fine-grained semantic properties of each item: processing detailed item text (title, tags, description) and extracts content features via a [ITEM] token embedding; initialized from TinyLlama/Baichuan2.
- **User-level** LLM captures high-level preference patterns: taking sequences of item representations from Item LLM and modeling sequential behavior to predict the next item; also initialized from pre-trained LLMs.
- Performs top-K ranking directly through generative decoding rather than pointwise scoring.
- Both components are jointly trained end-to-end, allowing Item LLM to learn recommendation-specific representations.

$$\mathcal{L}_{gen} = - \sum_{j=1}^b \sum_{i=2}^n \log \frac{e^{s(E'_{j,i}, E_{j,i})}}{e^{s(E'_{j,i}, E_{j,i})} + \sum_k^N e^{s(E'_{j,i}, E_{j,i,k})}}$$

$$\mathcal{L}_{cls} = -(y \cdot \log(x) + (1 - y) \cdot \log(1 - x))$$

$$\mathcal{L}_{dis} = \lambda \mathcal{L}_{gen} + \mathcal{L}_{cls}$$



Compress the following sentence into embedding:
 Title: Arctic Terns: Animals that live in the light of day.
 Tag: Short Film
 Description: Arctic terns make a round trip to the North and South Poles once a year, a distance of kilometers, which is equivalent to a full circle of the Earth, and live only in the polar daytime, so they are known as the creatures that always live in the light ...

HLLM: Key Results

- Amazon Books: HLLM-7B R@10=9.39, N@10=5.69 vs SASRec R@10=3.06 (~207% improvement).
- PixelRec (Pixel8M): R@10=6.129 vs SASRec 3.589 (~71% improvement).
- Online A/B test at ByteDance: **+0.7%** gain in production metrics.
- Key takeaway: Hierarchical LLMs transfer pre-trained knowledge to both item understanding and user behavior modeling.

Dataset	Method	R@10	R@50	R@200	N@10	N@50	N@200	Impv. (avg)
Pixel8M	SASRec _{vit} (2024)	3.589	-	-	1.941	-	-	-27.72%
	HSTU* (2024)	4.848	10.315	18.327	2.752	3.939	5.135	+0.0%
	SASRec*	5.083	10.667	18.754	2.911	4.123	5.331	+3.82%
	HSTU-1B*	5.120	11.010	19.393	2.879	4.159	5.411	+5.37%
	SASRec-1B*	5.142	10.899	19.044	2.915	4.166	5.383	+4.83%
	HLLM-1B (Ours)	6.129	12.475	21.179	3.539	4.919	6.221	+22.93%
Amazon Books	SASRec (2018)	3.06	7.54	14.31	1.64	2.60	3.62	+0.0%
	LEARN (2024)	4.07	9.79	18.74	2.24	3.71	4.83	+34.42%
	HSTU-large (2024)	4.78	10.82	19.08	2.62	3.93	5.17	+47.80%
	SASRec*	5.35	11.91	21.02	2.98	4.40	5.76	+64.96%
	SASRec-1B*	5.09	11.11	19.45	2.86	4.17	5.42	+55.68%
	HSTU-large*	5.00	11.29	20.13	2.78	4.14	5.47	+55.61%
	HSTU-1B*	5.25	12.03	21.60	2.89	4.36	5.80	+64.37%
	HLLM-1B (Ours)	6.97	14.61	24.78	3.98	5.64	7.16	+108.68%
	HSTU-large [†] *	6.49	12.22	19.81	3.99	5.24	6.38	+88.94%
	HLLM-1B-Scratch [†] (Ours)	6.85	13.95	23.19	4.02	5.56	6.95	+103.65%
	HLLM-1B [†] (Ours)	9.28	17.34	27.22	5.65	7.41	8.89	+166.42%
	HLLM-7B [†] (Ours)	9.39	17.65	27.59	5.69	7.50	8.99	+169.58%

Topic	Work	Key Contribution
Retrieval	Kuaiformer	Multi-interest query tokens for diverse user interests
Retrieval	TBGRecall	Outcome-conditioned retrieval for e-commerce
Ranking	DFGR	Single/dual-stream generative ranking networks
Ranking	HLLM	Hierarchical LLMs for item & user modeling
Ranking	HSTU	Action token interleaving; unifies retrieval & ranking; Batched inference with KV cache; MTIA 2i chip co-design
Re-ranking	GoalRank	Generator-only ranking; group-relative optimization
Re-ranking	SORT-Gen	List-level multi-objective generative re-ranking
Re-ranking	KC-GenRec	Knowledge graph constrained generative re-ranker
End-to-End	OneRec	Unified decoder-only model; ECPO; 5× MFU improvement
End-to-End	OneSug	Unified search & recommendation; Prefix2Query
End-to-End	ETEGRec	End-to-end learnable item tokenization
End-to-End	COBRA	Bridging generative & dense retrieval paradigms

- KC-GenRe targets Knowledge Graph Completion (KGC), not recommendation — it re-ranks candidate entities for KG triples.
- LLM-based KGC faces three issues: mismatch (generating entity names instead of IDs), misordering (wrong ranking), omission (missing valid answers).
- KC-GenRe formulates KGC re-ranking as candidate identifier sorting — entities labeled with option IDs ('A','B','C') and the LLM generates a sorted permutation.

(a) KGE
model
ranking
results

query ($e_h, r, ?$) : (Jackie Chan, played in movie, ?)

candidate entities : e_1 (King of Comedy), e_2 (Ip Man), e_3 (Police Story)

input construction

(b) LLM
re-ranking

Task instruction: ... output a ranking of these candidates.

Query: Jackie Chan played in movie ____ ?

Candidate answers: King of Comedy; Ip Man; Police Story

Generative LLMs e.g. GPT3, LLaMA

samples of generated text

Police Story; King of Comedy; Ip Man



Ranking: 1. Ip Man; 2. Police Story 2013; 3. King of Comedy.



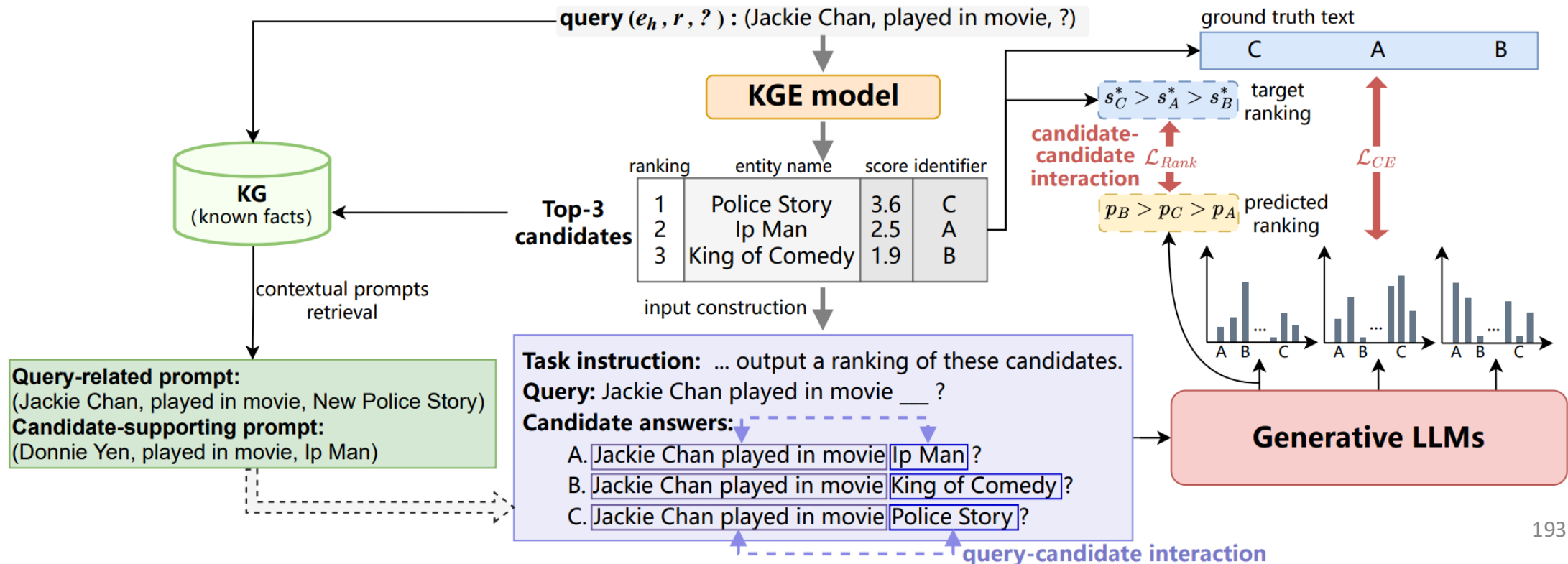
mismatch,
misordering
omission

The ranking would be : Ip Man; King of Comedy.



...

- Knowledge-guided interactive training: query-candidate interaction forms triples like 'Jackie Chan played in movie Ip Man?'; candidate-candidate interaction uses KGE scores as targets.
- Loss combines cross-entropy and ranking loss: $L_{Rank} = (C/K^2) \cdot \sum \max(0, p_i - p_j)$.
- Knowledge-augmented constrained inference: restricts output vocabulary to only unused option IDs at each decoding step, ensuring valid permutations.
- Retrieves query-related and candidate-supporting prompts via SBERT similarity.



KC-GenRe: Key Results



August 9–13, 2026
Jeju, Korea

- FB15K-237-N: MRR 0.399 (+6.7% vs PKGC), Hits@1 0.338 (+7.7%).
- Wiki27K: MRR 0.317 (+6.8% vs TuckER), Hits@1 0.274 (+8.9%).
- Ablation: removing constrained generation causes severe degradation, sometimes worse than no re-ranking.
- Key takeaway: Constrained inference is essential for valid generative re-ranking; the technique transfers to any domain with structured constraints.

Model	ReVerb20K					ReVerb45K				
	MRR	MR	Hits@1	Hits@3	Hits@10	MRR	MR	Hits@1	Hits@3	Hits@10
TransE (Bordes et al., 2013)	0.138	1150.5	0.034	0.201	0.316	0.202	1889.5	0.122	0.243	0.346
ComplEx (Trouillon et al., 2016)	0.038	4486.5	0.017	0.043	0.071	0.068	5659.8	0.054	0.071	0.093
R-GCN (Schlichtkrull et al., 2018)	0.122	1204.3	-	-	0.187	0.042	2866.8	-	-	0.046
ConvE (Dettmers et al., 2018)	0.262	1483.7	0.203	0.287	0.371	0.218	3306.8	0.166	0.243	0.314
KG-BERT (Yao et al., 2019)	0.047	420.4	0.014	0.039	0.105	0.123	1325.8	0.070	0.131	0.223
RotatE (Sun et al., 2019)	0.065	2861.5	0.043	0.069	0.108	0.141	3033.4	0.110	0.147	0.196
PairRE (Chao et al., 2021)	0.213	1366.2	0.166	0.229	0.296	0.205	2608.4	0.153	0.228	0.302
ResNet (Lovelace et al., 2021)	0.224	2258.4	0.188	0.240	0.292	0.181	3928.9	0.150	0.196	0.242
BertResNet-ReRank (Lovelace et al., 2021)	0.272	1245.6	0.225	0.294	0.347	0.208	2773.4	0.166	0.227	0.281
CaRe (Gupta et al., 2019)	0.318	973.2	-	-	0.439	0.324	1308.0	-	-	0.456
OKGIT (Chandrasekhar and Talukdar, 2021)	0.359	527.1	0.282	0.394	0.499	0.332	<u>773.9</u>	0.261	0.363	0.464
OKGSE (Xie et al., 2022a)	0.372	487.3	0.291	0.408	<u>0.524</u>	0.342	771.1	0.274	0.371	0.473
CEKFA (Wang et al., 2023b)	<u>0.387</u>	<u>416.7</u>	<u>0.310</u>	<u>0.427</u>	0.515	<u>0.369</u>	884.5	<u>0.294</u>	<u>0.409</u>	<u>0.502</u>
KC-GenRe	0.408	410.8	0.331	0.450	0.547	0.404	874.1	0.332	0.444	0.534

	QCI	CCI	QP	CP	CG	ReVerb20K		ReVerb45K	
						MRR	Hits@1	MRR	Hits@1
Base						0.367	0.288	0.352	0.274
1						0.350	0.263	0.334	0.246
2			✓	✓	✓	0.383	0.304	0.365	0.282
3	✓		✓	✓	✓	0.405	0.326	0.381	0.298
4	✓	✓		✓	✓	0.399	0.320	0.397	0.323
5	✓	✓	✓		✓	0.403	0.328	0.400	0.327
6	✓	✓			✓	0.370	0.284	0.360	0.271
7	✓	✓	✓	✓		0.367	0.289	0.352	0.273
KC-GenRe	✓	✓	✓	✓	✓	0.408	0.331	0.404	0.332

Model	QCI	CCI	DP	CG	Wiki27K		FB15K-237-N	
					MRR	Hits@1	MRR	Hits@1
Base					0.249	0.185	0.309	0.227
1					0.283	0.227	0.329	0.248
2			✓	✓	0.314	0.268	0.340	0.257
3	✓		✓	✓	0.311	0.266	0.361	0.284
4	✓	✓		✓	0.298	0.249	0.353	0.277
5	✓	✓	✓		0.245	0.178	0.303	0.218
KC-GenRe	✓	✓	✓	✓	0.317	0.274	0.399	0.338

All End-to-End works were covered in previous Sections:

- OneRec — unified decoder-only model; ECPO; 5× MFU
- OneSug — unified search & rec; Prefix2Query
- ETEGRec — end-to-end learnable tokenization
- COBRA — bridging generative & dense retrieval

Key insight: End-to-end models achieve 5× MFU improvement over cascaded systems and show clear scaling laws.

Summary: Cascaded Systems



KDD2026
August 9–13, 2026
Jeju, Korea

Topic	Work	Key Contribution
Retrieval	Kuaiformer	Multi-interest query tokens for diverse user interests
Retrieval	TBGRcall	Outcome-conditioned retrieval for e-commerce
Ranking	DFGR	Single/dual-stream generative ranking networks
Ranking	HLLM	Hierarchical LLMs for item & user modeling
Ranking	HSTU	Action token interleaving; unifies retrieval & ranking; Batched inference with KV cache; MTIA 2i chip co-design
Re-ranking	GoalRank	Generator-only ranking; group-relative optimization
Re-ranking	SORT-Gen	List-level multi-objective generative re-ranking
Re-ranking	KC-GenRe	Knowledge graph constrained generative re-ranker
End-to-End	OneRec	Unified decoder-only model; ECPO; 5× MFU improvement
End-to-End	OneSug	Unified search & recommendation; Prefix2Query
End-to-End	ETEGRec	End-to-end learnable item tokenization
End-to-End	COBRA	Bridging generative & dense retrieval paradigms

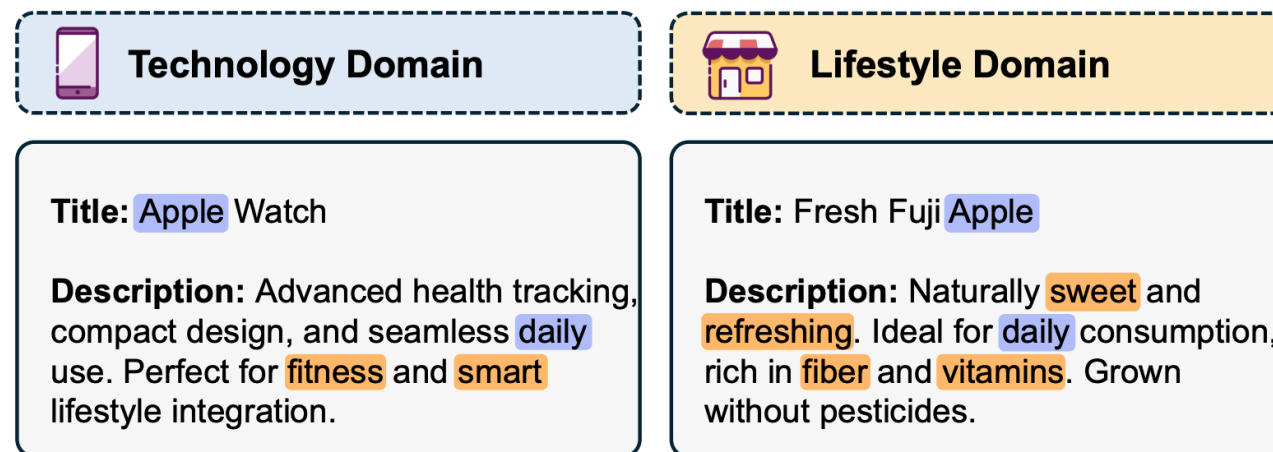
Application Scenarios: Overview



August 9–13, 2026
Jeju, Korea

Topic	Work	Key Contribution
Cold Start	LLMTreeRec	LLM-constructed hierarchical tree for item organization
Cold Start	LC-REC	RQ-VAE aligning collaborative & semantic token spaces
Cross Domain	RecGPT	FSQ-based tokenizer for cross-domain transfer
Cross Domain	GMC	Contrastive learning + domain-specific fine-tuning
Cross Domain	GenCDR	Domain-aware routing & prefix-tree decoding
Search	OneSearch	Single generative task unifying recall, rank, personalize
Search	GenSAR	Dual-purpose IDs for unified search & recommendation
Auto-Bidding	DiffBid	Diffusion models for bidding trajectory generation
Auto-Bidding	GAS	Transformer + MCTS for multi-objective bidding
Auto-Bidding	GAVE	Return-To-Go conditioned bidding generation

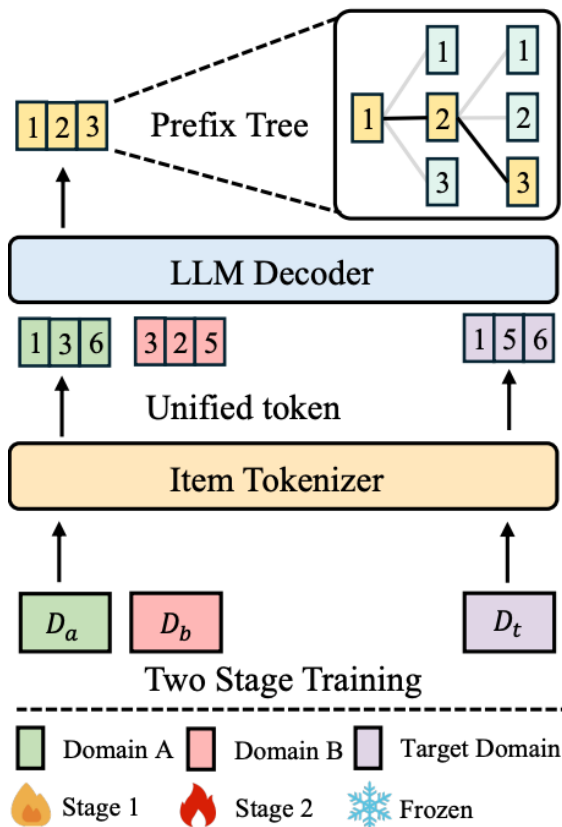
Topic	Work	Key Contribution
Cold Start	LLMTreeRec	LLM-constructed hierarchical tree for item organization
Cold Start	LC-REC	RQ-VAE aligning collaborative & semantic token spaces
Cross Domain	RecGPT	FSQ-based tokenizer for cross-domain transfer
Cross Domain	GMC	Contrastive learning + domain-specific fine-tuning
Cross Domain	GenCDR	Domain-aware routing & prefix-tree decoding
Search	OneSearch	Single generative task unifying recall, rank, personalize
Search	GenSAR	Dual-purpose IDs for unified search & recommendation
Auto-Bidding	DiffBid	Diffusion models for bidding trajectory generation
Auto-Bidding	GAS	Transformer + MCTS for multi-objective bidding
Auto-Bidding	GAVE	Return-To-Go conditioned bidding generation



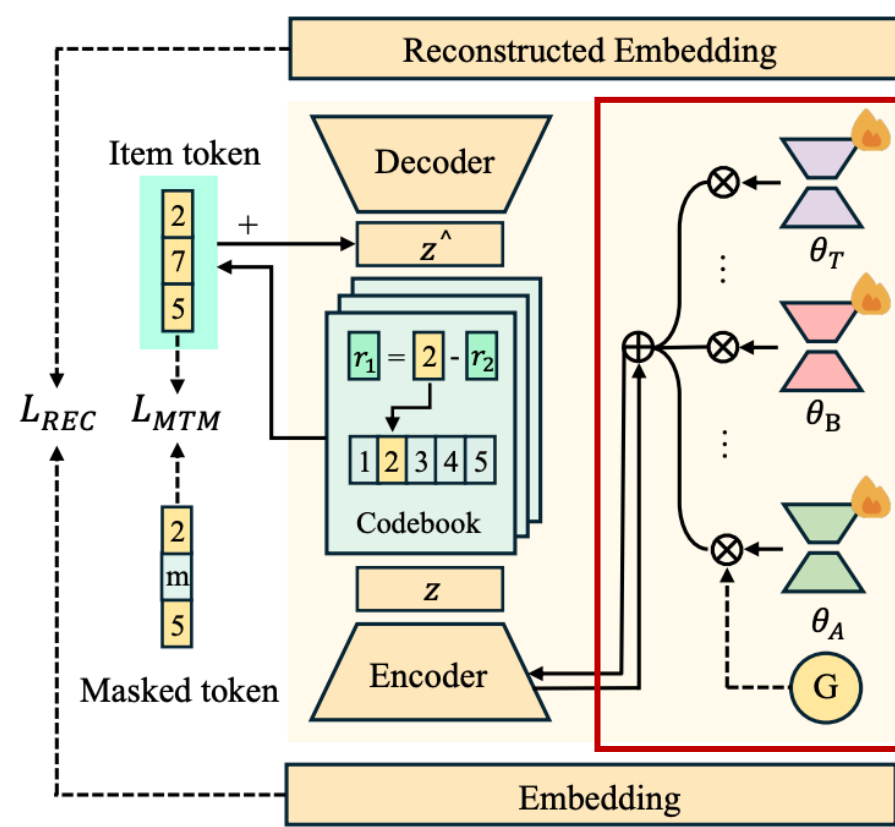
- **Item Tokenization Gap:** Prior cross-domain recommendation relies on ID-based transfer — but LLM tokenizers cannot represent item IDs across domains.
- **Domain Personalization Gap:** Shared model parameters cannot capture domain-specific patterns (domain personalization gap).
- GenCDR replaces ID-based transfer with semantic tokenization and autoregressive generation, introduces domain-aware decoding that constrains generation to valid items per domain.

- Domain-adaptive Tokenization: RQ-VAE with per-domain LoRA adapters and VIB (Variational Information Bottleneck) router — each domain gets adapted tokenization.

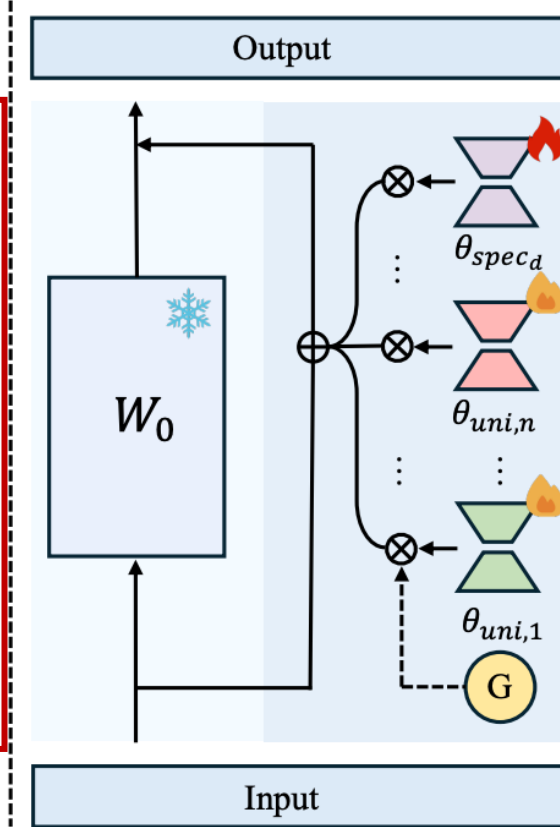
(a) Overall Pipeline



(b) Domain-adaptive Tokenization

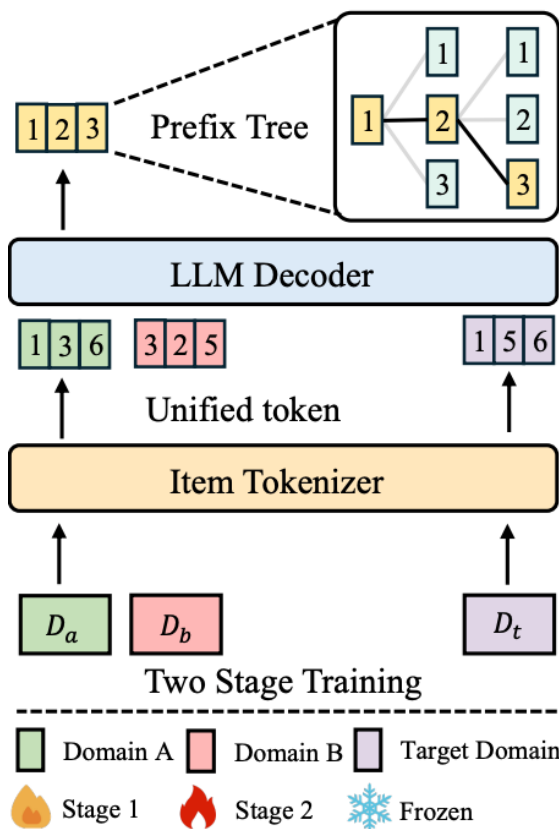


(c) Autoregressive Recommendation

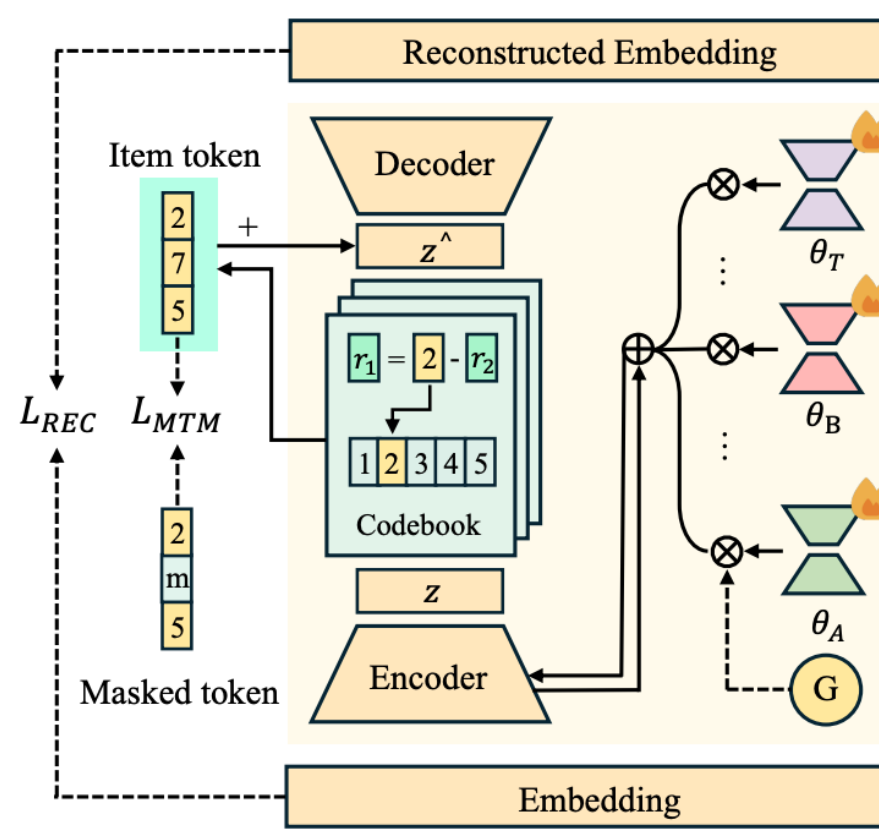


- Cross-domain Autoregressive Recommendation: Universal LoRA captures shared patterns + Specific LoRA captures domain-specific patterns, with dynamic routing between them.

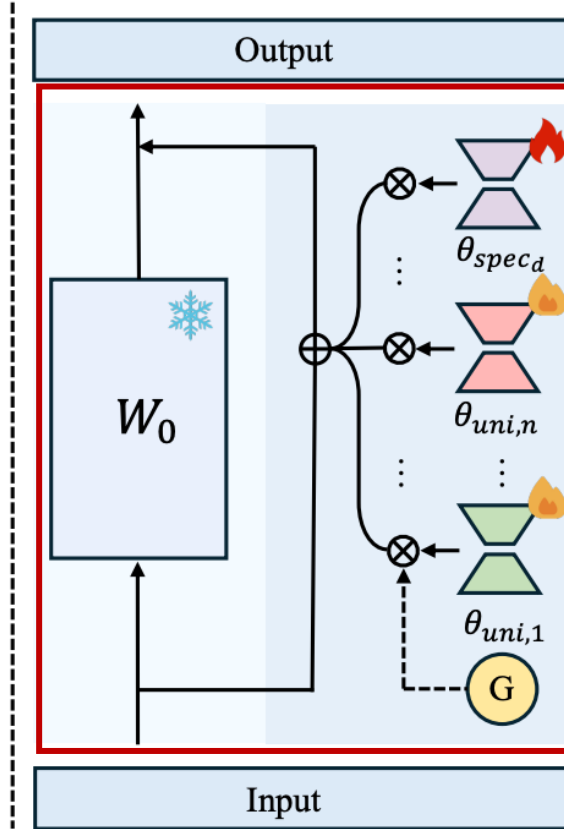
(a) Overall Pipeline



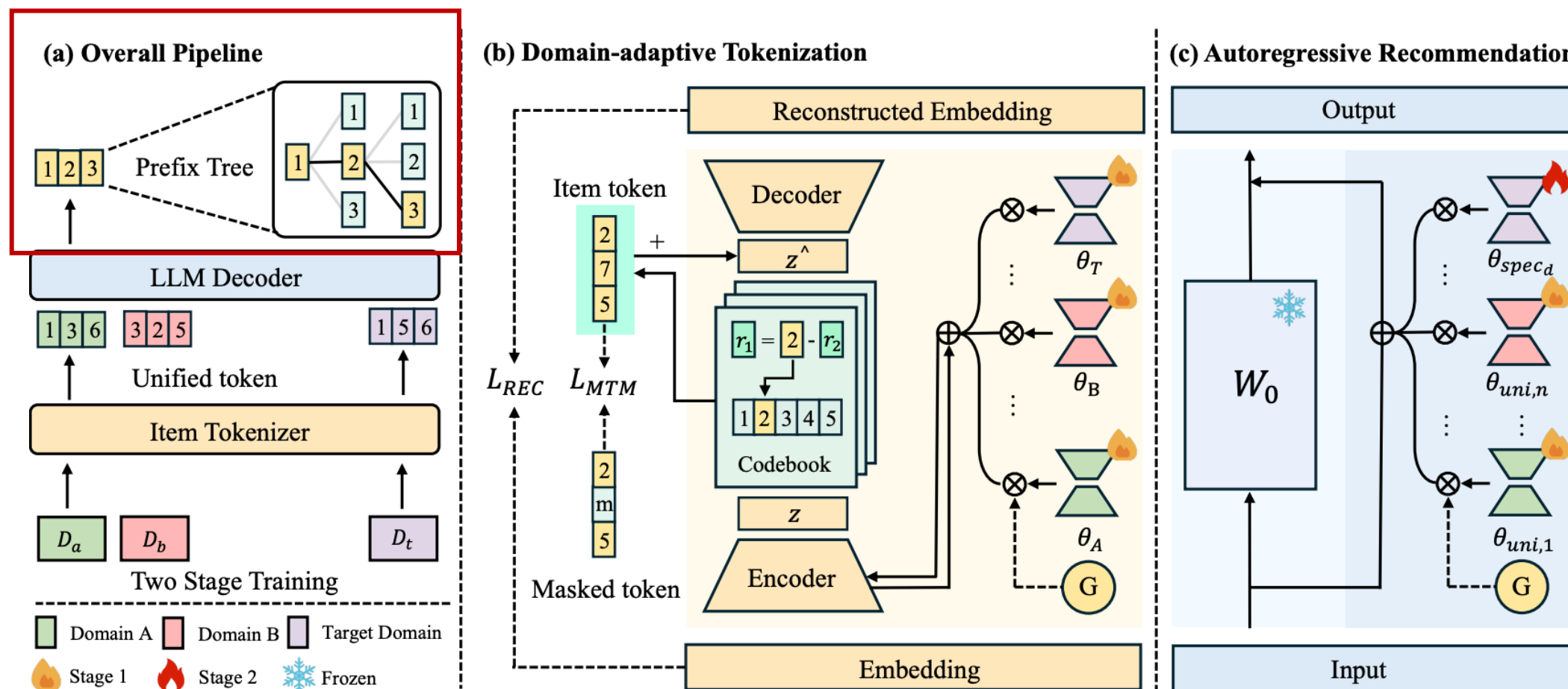
(b) Domain-adaptive Tokenization



(c) Autoregressive Recommendation



- Domain-aware Prefix-tree: constrains decoding to valid items per target domain, preventing cross-domain hallucination.



GenCDR: Key Results



August 9–13, 2026
Jeju, Korea

- NDCG@10 across six domains: Sports=0.0262, Clothing=0.0203, Phones=0.0512, Electronics=0.0283, Books=0.0212, Movies=0.1275.
- Per-domain LoRA adapters significantly outperform shared-parameter baselines.
- Domain-aware prefix-tree decoding eliminates invalid cross-domain item generation.
- Key takeaway: Semantic tokenization with domain-adaptive LoRA is more effective than ID-based transfer.

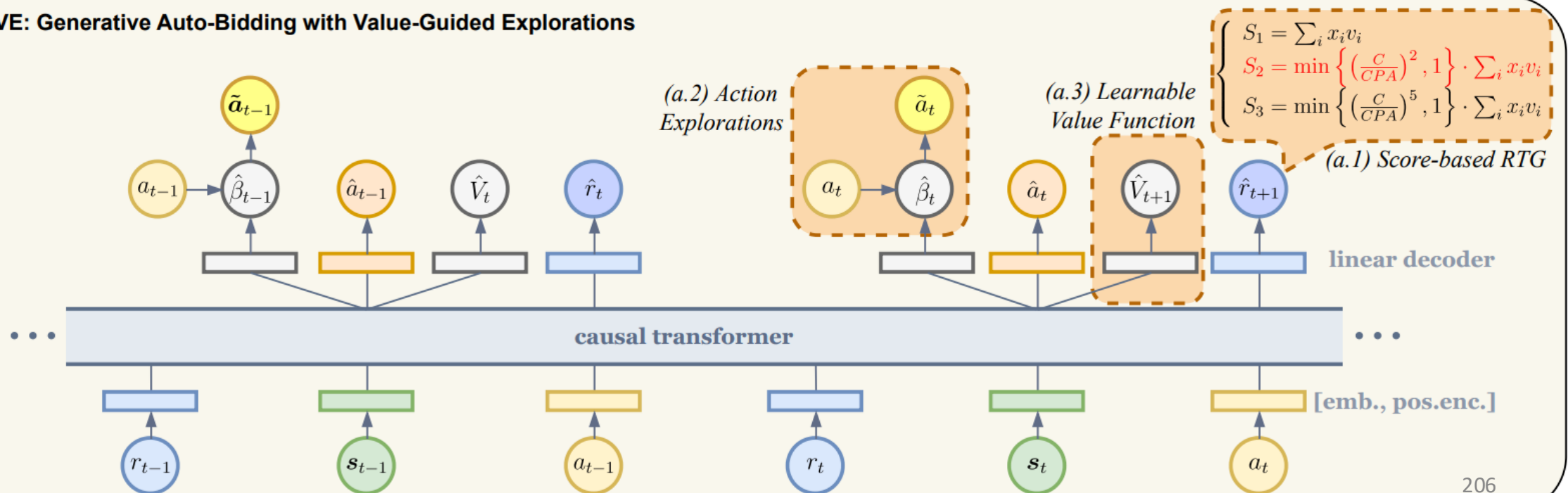
Dataset		Metric	SDSR			GenRec			CDSR			Ours
Scene	Domain		Bert4Rec	SASRec	STOSA	VQ-Rec	TIGER	HSTU	C2DSR	TriCDR	LLM4CDSR	GenCDR
Leisure	Sports	R@5	0.0188	0.0197	0.0236	0.0261	0.0267	0.0254	0.0265	0.0266	0.0263	0.0274
		N@5	0.0121	0.0126	0.0162	0.0238	0.0244	0.0241	0.0253	0.0255	0.0257	0.0261
		R@10	0.0325	0.0334	0.0346	0.0389	0.0397	0.0381	0.0395	0.0396	0.0398	0.0403
		N@10	0.0169	0.0173	0.0283	0.0281	0.0287	0.0277	0.0258	0.0259	0.0260	0.0262
	Clothing	R@5	0.0128	0.0132	0.0162	0.0171	0.0173	0.0175	0.0172	0.0174	0.0176	0.0181
		N@5	0.0078	0.0081	0.0119	0.0129	0.0125	0.0132	0.0158	0.0161	0.0163	0.0167
		R@10	0.0219	0.0227	0.0223	0.0248	0.0241	0.0253	0.0255	0.0258	0.0261	0.0265
		N@10	0.0105	0.0108	0.0135	0.0170	0.0167	0.0174	0.0191	0.0194	0.0196	0.0203
Technology	Phones	R@5	0.0331	0.0345	0.0415	0.0411	0.0423	0.0415	0.0428	0.0434	0.0431	0.0436
		N@5	0.0215	0.0224	0.0283	0.0308	0.0315	0.0327	0.0392	0.0396	0.0401	0.0411
		R@10	0.0524	0.0537	0.0618	0.0607	0.0613	0.0615	0.0589	0.0593	0.0614	0.0621
		N@10	0.0278	0.0287	0.0346	0.0399	0.0406	0.0425	0.0493	0.0505	0.0506	0.0512
	Electronics	R@5	0.0179	0.0186	0.0213	0.0219	0.0228	0.0232	0.0235	0.0238	0.0237	0.0241
		N@5	0.0118	0.0122	0.0148	0.0211	0.0214	0.0226	0.0229	0.0231	0.0230	0.0235
		R@10	0.0276	0.0285	0.0315	0.0318	0.0322	0.0328	0.0336	0.0339	0.0338	0.0342
		N@10	0.0149	0.0154	0.0172	0.0262	0.0269	0.0271	0.0278	0.0280	0.0279	0.0283
Entertainment	Books	R@5	0.0089	0.0093	0.0142	0.0175	0.0172	0.0181	0.0152	0.0155	0.0161	0.0192
		N@5	0.0071	0.0076	0.0117	0.0178	0.0177	0.0180	0.0143	0.0148	0.0153	0.0187
		R@10	0.0176	0.0182	0.0219	0.0224	0.0221	0.0230	0.0205	0.0211	0.0216	0.0237
		N@10	0.0158	0.0164	0.0165	0.0201	0.0198	0.0206	0.0182	0.0185	0.0189	0.0212
	Movies	R@5	0.1503	0.1542	0.1562	0.1680	0.1652	0.1682	0.1588	0.1601	0.1613	0.1713
		N@5	0.1015	0.1047	0.1063	0.1182	0.1156	0.1189	0.1092	0.1105	0.1149	0.1215
		R@10	0.1798	0.1825	0.1753	0.1922	0.1893	0.1931	0.1854	0.1865	0.1878	0.1971
		N@10	0.1211	0.1265	0.1223	0.1261	0.1255	0.1268	0.1203	0.1217	0.1225	0.1275

Topic	Work	Key Contribution
Cold Start	LLMTreeRec	LLM-constructed hierarchical tree for item organization
Cold Start	LC-REC	RQ-VAE aligning collaborative & semantic token spaces
Cross Domain	RecGPT	FSQ-based tokenizer for cross-domain transfer
Cross Domain	GMC	Contrastive learning + domain-specific fine-tuning
Cross Domain	GenCDR	Domain-aware routing & prefix-tree decoding
Search	OneSearch	Single generative task unifying recall, rank, personalize
Search	GenSAR	Dual-purpose IDs for unified search & recommendation
Auto-Bidding	DiffBid	Diffusion models for bidding trajectory generation
Auto-Bidding	GAS	Transformer + MCTS for multi-objective bidding
Auto-Bidding	GAVE	Return-To-Go conditioned bidding generation

- Rule-based methods (PID) lack flexibility for complex auction environments.
- RL/MDP methods assume state-independence, ignoring temporal dependencies across bidding steps.
- Offline RL training suffers from out-of-distribution (OOD) risks and behavioral collapse on fixed datasets.
- GAVE addresses all three with Return-To-Go conditioned generative bidding + value-guided exploration.

- a.1 Score-based Return-To-Go conditioner: encodes target metrics (desired conversions, CPA constraints) as differentiable return signals into the generation process.
- a.2 Value-guided action exploration: generates novel bidding actions with a learnable coefficient that ensures stability-preserving updates.
- a.3 Expectile-regression-based value function: estimates return upper bounds (not means) to guide exploration toward high-return trajectories and mitigate OOD risk.

(a) GAVE: Generative Auto-Bidding with Value-Guided Explorations



GAVE: Key Results



KDD2026
August 9–13, 2026
Jeju, Korea

- AuctionNet (100% budget): GAVE=376 vs best baseline GAS=359 (+4.74%).
- AuctionNet-Sparse (100%): GAVE=37.2 vs GAS=36.1 (+3.05%).
- Outperforms across ALL budget ratios (50%–150%).
- Won first place in NeurIPS 2024 AIGB Track auto-bidding competition; Online significance.
- Key takeaway: Return-conditioned generation + value-guided exploration bridges the offline-to-online gap in auto-bidding.

	Cost	Conversion	Target cost	CPA valid ratio
Nobid	+0.8%	+8.0%	+3.2%	/
Costcap	+2.0%	+3.6%	+2.2%	+1.9%

Dataset	Budget	DiffBid	USCB	CQL	IQL	BCQ	DT	CDT	GAS	GAVE	Improve
AuctionNet	50%	54	86	113	164	190	191	174	<u>193</u>	201*	4.15%
	75%	100	135	139	232	259	265	242	<u>287</u>	296*	3.14%
	100%	152	157	171	281	321	329	326	<u>359</u>	376*	4.74%
	125%	193	220	201	355	379	396	378	<u>409</u>	421*	2.93%
	150%	234	281	238	401	429	450	433	<u>461</u>	467*	1.30%
AuctionNet-Sparse	50%	9.9	11.5	12.8	16.5	17.7	14.8	11.2	<u>18.4</u>	19.6*	6.52%
	75%	15.4	14.9	16.7	22.1	24.6	22.9	18.0	<u>27.5</u>	28.3*	2.91%
	100%	19.5	17.5	22.2	30.0	31.1	29.6	31.2	<u>36.1</u>	37.2*	3.05%
	125%	25.3	26.7	28.6	37.1	34.2	34.3	31.7	<u>40.0</u>	42.7*	6.75%
	150%	30.8	31.3	35.8	43.1	37.9	44.5	39.1	<u>46.5</u>	47.4*	1.94%

Summary: Application Scenarios



August 9–13, 2026
Jeju, Korea

Topic	Work	Key Contribution
Cold Start	LLMTreeRec	LLM-constructed hierarchical tree for item organization
Cold Start	LC-REC	RQ-VAE aligning collaborative & semantic token spaces
Cross Domain	RecGPT	FSQ-based tokenizer for cross-domain transfer
Cross Domain	GMC	Contrastive learning + domain-specific fine-tuning
Cross Domain	GenCDR	Domain-aware routing & prefix-tree decoding
Search	OneSearch	Single generative task unifying recall, rank, personalize
Search	GenSAR	Dual-purpose IDs for unified search & recommendation
Auto-Bidding	DiffBid	Diffusion models for bidding trajectory generation
Auto-Bidding	GAS	Transformer + MCTS for multi-objective bidding
Auto-Bidding	GAVE	Return-To-Go conditioned bidding generation

22 works spanning the cascaded pipeline and 4 application scenarios:

Cascaded Systems:

Retrieval: similarity → generative next-token prediction

Ranking: feature tables → user-centric token sequences

Re-ranking: generator-only list-level multi-objective optimization

End-to-End: unified models with 5× MFU improvement

Scenario Solutions:

Cold Start: world knowledge + semantic identifiers → generalization

Cross Domain: unified tokenization + domain-aware decoding

Search: end-to-end generation unifying search & recommendation

Auto-Bidding: diffusion + MCTS + return-conditioned generation

Common thread: end-to-end optimization, unified objectives, scalable architectures.

Core insight: GenRec shifts from a *drop-in* module to a *generative nexus*—co-evolving with cascades on the integration side, creating new supply on the value side.

1 Cascaded Systems: Stage-aware hybrid orchestration

Evolving GenRec from a statically assigned pipeline role to a cascade-aware controller

2 Cascaded Systems: Feature & Token co-evolution

Aligning GenRec's semantic-ID/Token vocabulary with existing item feature stores

3 Application Scenarios: Interactive Recommendations

Suiting Genrec to agent-based and conversational recommendations, and shifting output from “select existing items” to “synthesize personalized content ”

4 Application Scenarios: Supply–demand generative loop

Constructing GenRec as a creator-facing tool that generates title/tag/thumbnaill variants aligned with the retrieval distribution

1 Introduction (9:00 - 9:15)



Xiaopeng Li



2 Tokenization (9:15 - 9:45)



Xiaopeng Li

3 Model Structure (9:45 - 10:30)



Ziwei Liu



15mins

4 Optimization (10:45 - 11:15)



Wenlin Zhang



Kaifeng Guo

5 Application (11:15 - 11:30)



Yejing Wang



6 Challenge & Conclusion (11:30 - 11:45)



Bo Chen

End-to-End and Reward Modeling

Scaling Models

Utilizing a **single unified model** to replace the cascade multi-stage system can save labor and engineering overheads, enabling more resources to the model scaling.

Future work may scale models toward **LLM-level capacity** while **maintaining millisecond latency** for balancing model size and efficiency.



Unified Reward Design

Rule-based or model-specific single-aspect reward signal is not enough. Designing **unified reward agents** powered by LLMs that automatically balance **user satisfaction, platform profit, fairness, and safety** without hand-crafted rules is a critical next step.



List-wise Rewards

Moving from **point-wise** to **list-wise** reward signals for recommendation quality is essential for improving the overall performance of end-to-end recommendation.

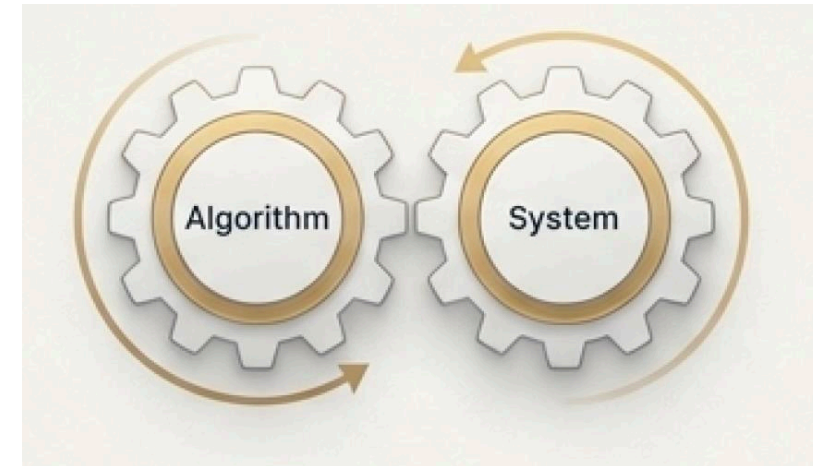
Designing a **holistic reward function** for the final recommendation list remains a considerable challenge.

Efficiency and Co-Design

Algorithm-System Co-design Framework

An integrated algorithm-system framework is needed for **streaming training** on hundreds of millions of daily samples and for inference serving under strict **low-latency, high-throughput requirements**.

- Efficient training and preference-alignment frameworks for keeping pace with streaming data
- Satisfy strict requirements on low latency and high concurrency



Efficiency Techniques

Ultra-long behavior modeling is both essential and a major efficiency bottleneck for generative recommendations. Research directions for efficient sequence modeling paradigms include **memory-augmented sequence compression**, **RAG-enhanced training** for enhancing overall efficiency.



Reasoning and Data

1

Reasoning Enhancement

Leveraging the reasoning capabilities to infer **users' short-/long-term interests** and analyze their behavior patterns.

- Constructing large-scale **personalized chain-of-thought data**
- Controlling **adaptive reasoning** depth



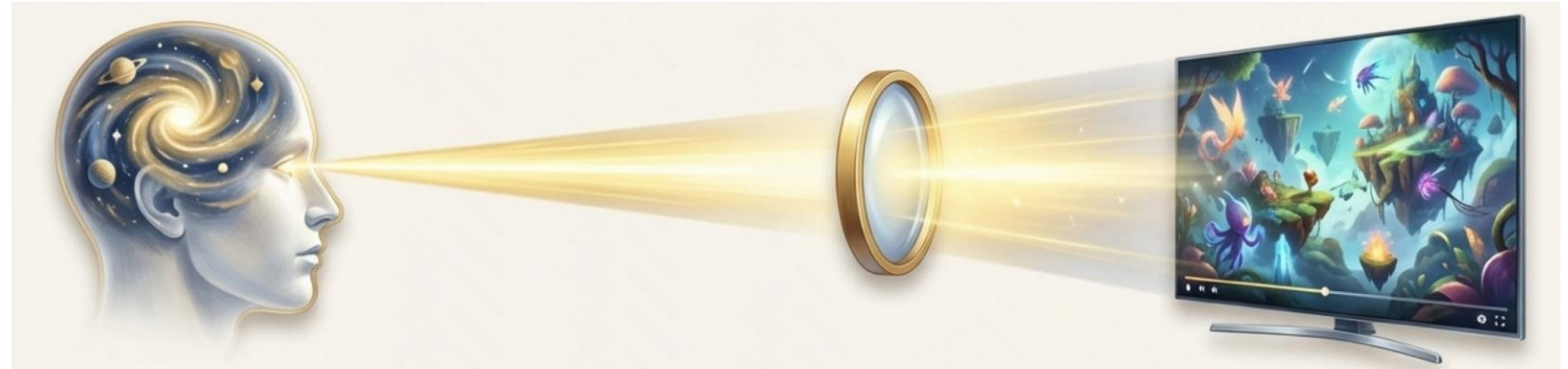
2

Data Optimization

Debiasing generative training from **exposure and position skew** is essential for improving the fairness and accuracy of recommendation systems.

The construction of **high-quality data** remains a fundamental bottleneck, such as CoT data, user and platform multi-aspect preference data, and explicit intent-level annotations.

Next Application



1

Conversational Agents

Building **conversational agents** that deliver **personalized dialogue** and maintain **user-centric memory** is crucial for enhancing user experience and engagement in next-generation recommendation systems.

2

From Recommendation to Generation

Multimodal generative models such as Kling and Sora foreshadow a **shift from retrieving existing content to synthesizing personalized video, audio, and ads in real time**. This paradigm introduces extremely **sparse feedback** and high **generation cost**.

1

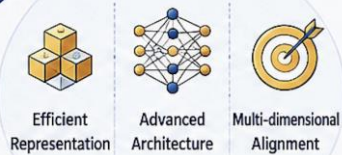


A unified framework

We organize generative recommender systems through a tri-dimensional decomposition: tokenization, architecture, and optimization, tracing the shift from discriminative ranking to generative recommendation.

Framework: tokenization · architectural design · optimization strategies

2



Key technical trends

The tutorial highlights semantic identifiers for compact and expressive representations, scalable architectures for efficient computation, and multi-dimensional preference alignment across users, platforms, and stakeholders.

Trends: efficient representation · scalable backbone · multi-objective alignment

3



Applications, challenges, and future directions

We discuss how generative recommendation applies across stages and scenarios, summarize current challenges, and outline future directions as a practical blueprint for academia and industry.

Scope: applications · challenges · future research directions

Thanks for Listening!



Survey Paper



Tutorial Slides

E-mail: xiaopli2-c@my.cityu.edu.hk