GRAM: Generative Recommendation via Semantic-aware Multi-granular Late Fusion

Sunkyung Lee[1], Minjin Choi[2], Eunseong Choi[1], Hye-young Kim[1], Jongwuk Lee[1]

Sungkyunkwan University (SKKU), Republic of Korea[1], Samsung Research, Republic of Korea[2]

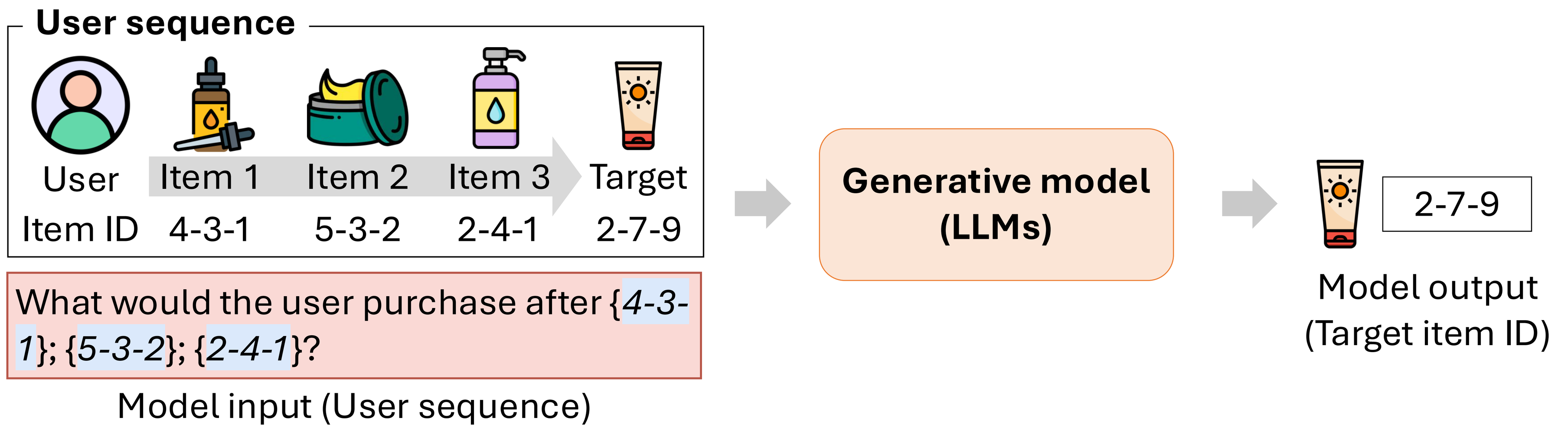
sk1027@skku.edu

Paper Code

ACL 2025 VIENNA JULY 27 - AUGUST 1

Generative Recommendation

It aims to directly generate a **target item identifier (ID)** from user history.
- It can directly leverage the extensive knowledge of LLMs by formulating recommendations into a text-to-text generation task.
- Typically, users are represented by concatenating item IDs into a sequence.



Takeaways

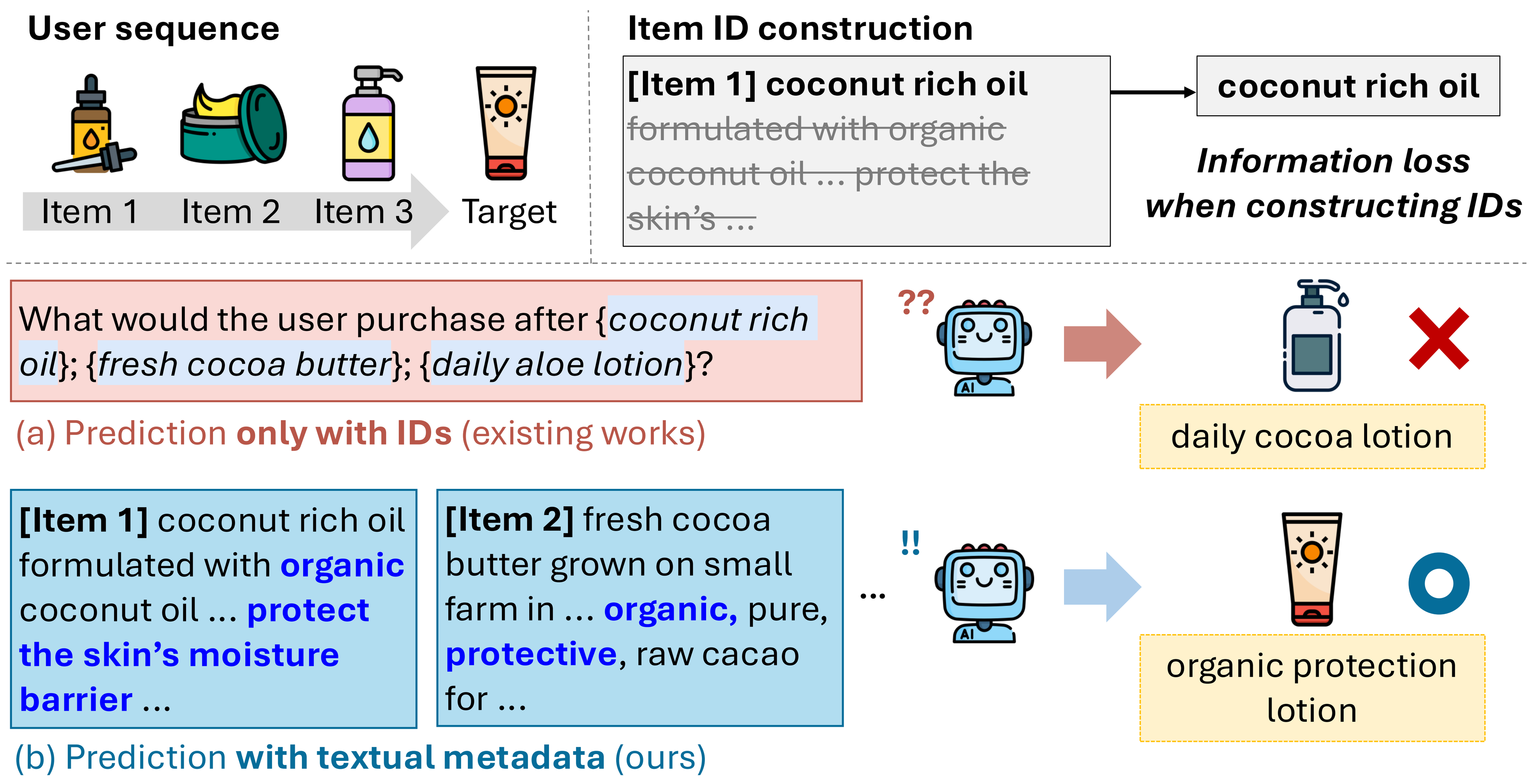
- ✓ A novel generative recommender for translating item relationships into LLM's vocabulary and processing rich metadata efficiently
- ✓ **Semantic-to-lexical translation** for encoding implicit item relationships into LLM vocabulary
- ✓ **Multi-granular late fusion** for efficiently processing rich item information without quadratic complexity

Research Question

How can LLMs effectively **understand** and **utilize rich item information** for recommendation?

Limitations of Existing Methods

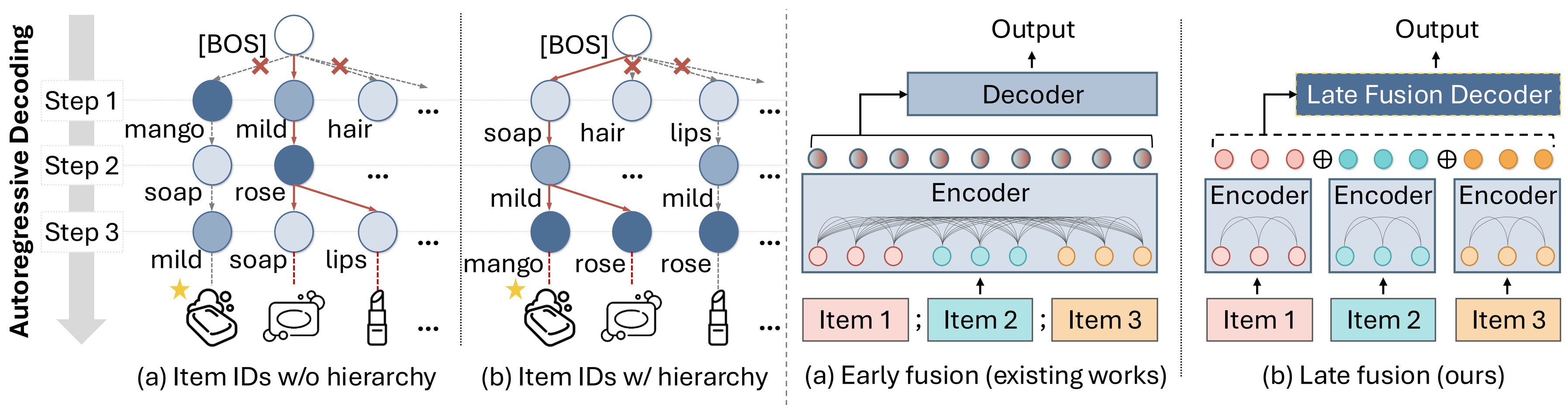
Existing works use **rich item metadata** only for **constructing short item IDs**.
This causes valuable **item information to be lost** during prediction.



Key Challenges

Challenge 1: Capturing Item Relationships
- LLMs often struggle with **recommendation-specific semantics**.
  - **Hierarchical semantics**: "lipstick" and "mascara" belong to "cosmetics"
  - **Collaborative semantics**: users who bought item A also tend to buy item B

Challenge 2: Handling Rich Item Information
- Items contain rich yet lengthy metadata (titles, categories, descriptions).
- **Transformer's quadratic complexity** leads to computational bottleneck.



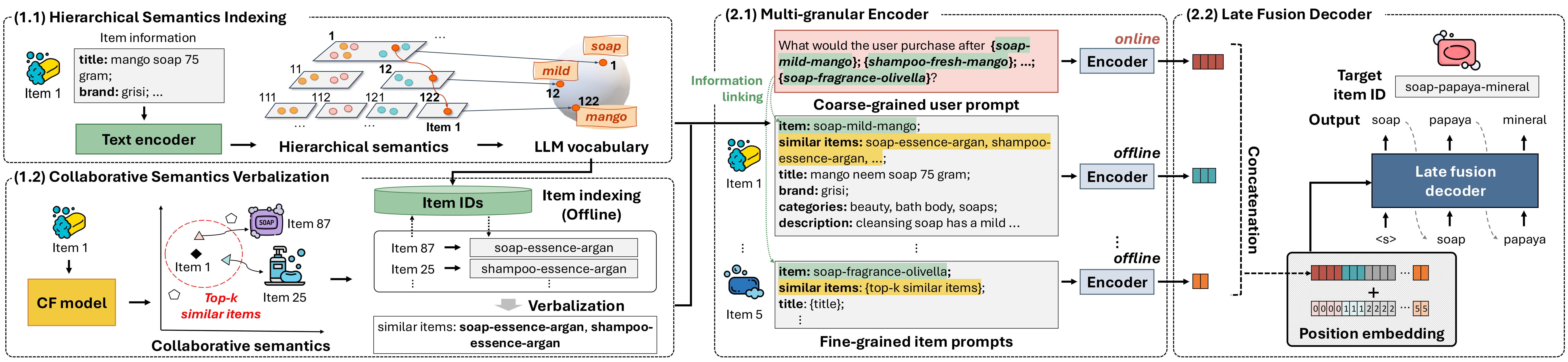
Challenge 1: Illustration of the hierarchy when autoregressively decoding IDs

Challenge 2: Schematic diagrams of early fusion and late fusion

GRAM: Generative Recommender via Semantic-Aware Multi-granular Late Fusion

(1) Semantic-to-Lexical Translation: Encoding item relationships in LLM vocabulary
- **(1.1) Hierarchical Semantics**: Hierarchically cluster item embeddings → Map them to LLM tokens → Create IDs that similar items share prefixes
- **(1.2) Collaborative Semantics**: Extract top-k similar items using CF model → Convert them into textual attributes

(2) Multi-granular Late Fusion: Efficiently processing rich metadata
- **(2.1) Multi-granular Encoder**: Separately encode **coarse-grained user prompts** for whole user preferences and **fine-grained item prompts** for detailed attributes
- **(2.2) Late Fusion Decoder**: Integrate prompts at decoding via cross-attention, generating target item IDs based on rich information



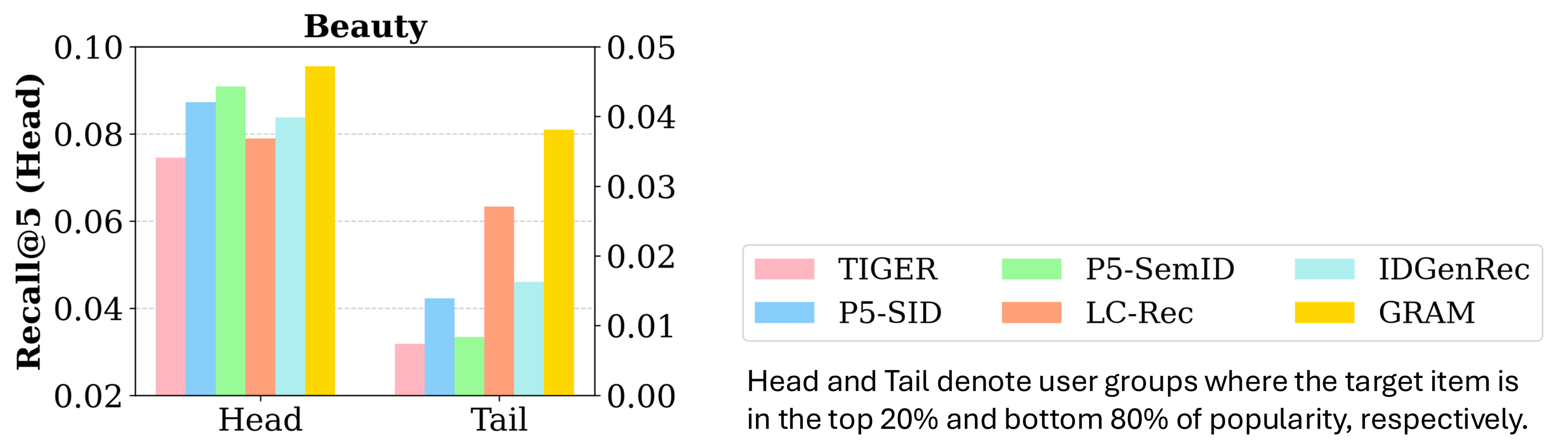
Experimental Results

- GRAM achieves **state-of-the-art performance over traditional and generative methods** in benchmark datasets. (Amazon Beauty, Toys, Sports, and Yelp datasets)
- All components of GRAM contribute to performance, with **collaborative semantics and item prompts** showing the most significant improvements.

Model	Beauty R@5	N@5	R@10	N@10
SASRec	0.0323	0.0200	0.0475	0.0249
FDSA	<u>0.0570</u>	<u>0.0412</u>	<u>0.0777</u>	<u>0.0478</u>
S^3Rec	0.0377	0.0235	0.0627	0.0315
P5-SID	0.0465	0.0329	0.0638	0.0384
TIGER	0.0352	0.0236	0.0533	0.0294
IDGenRec	0.0463	0.0328	0.0665	0.0393
LETTER	0.0364	0.0243	0.0560	0.0306
ELMRec	0.0372	0.0267	0.0506	0.0310
LC-Rec	0.0503	0.0352	0.0715	0.0420
GRAM	**0.0641**	**0.0451**	**0.0890**	**0.0531**
Gain (%)	12.4*	9.5*	14.5*	11.0*

Model	Beauty R@5	N@5
GRAM	**0.0641**	**0.0451**
w/o hierarchy	0.0605	0.0438
w/o CF (a_{CF})	0.0567	0.0396
w/o user prompt (T_u)	0.0634	0.0443
w/o item prompt (T_i)	0.0582	0.0404
w/o linking (a_{ID})	0.0628	0.0441
w/o position (P)	0.0563	0.0395

'*' indicates statistical significance ($p < 0.05$) by a paired t-test.
Please refer to the paper for the full results.



Head and Tail denote user groups where the target item is in the top 20% and bottom 80% of popularity, respectively.

- GRAM effectively handles **tail items through rich textual understanding**, showing up to 42.6% gain in R@5 compared to generative recommendation models.