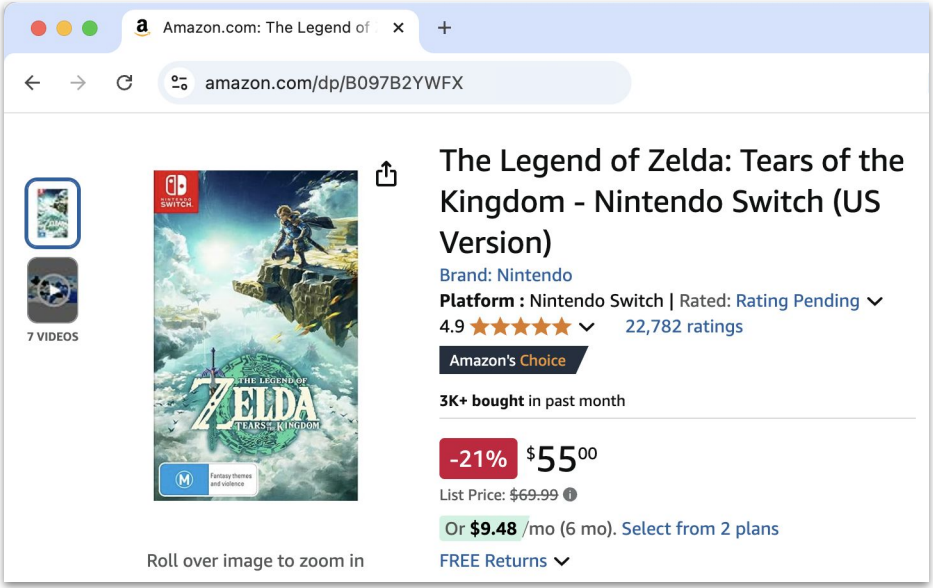
03

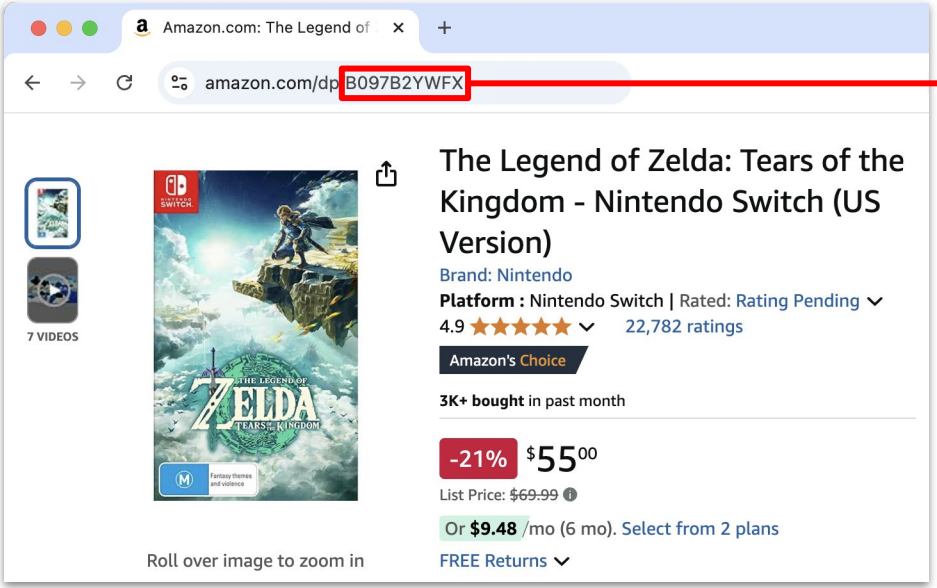
Semantic ID

-based Generative Recommendation

How to Index an Item in RecSys?



How to Index an Item in RecSys?

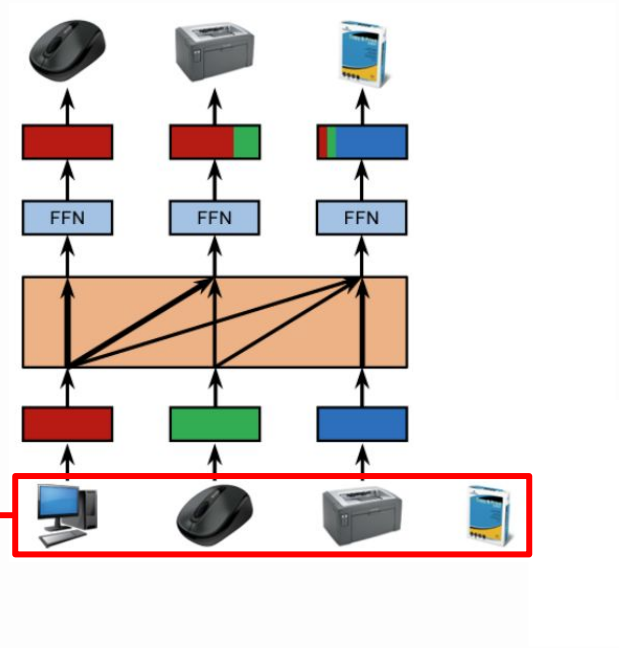


Item ID:

B097B2YWFX

How to Index an Item in RecSys?

Example: SASRec [*ICDM'18*]

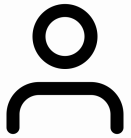


Each item is indexed by a unique **item ID**

Kang and McAuley. Self-Attentive Sequential Recommendation. ICDM 2018.

How to Index an Item in LLMs

I wanna some popular Nintendo games



How to Index an Item in LLMs

I wanna some popular Nintendo games

LLMs

How about  or  ?

How to Index an Item in LLMs

I wanna some popular Nintendo games

LLMs

How about  or  ?

How to index items in LLMs? **Item ID**?

How to Index an Item in LLMs

How many tokens in LLMs?

Meta	Llama 3	~128,000
OpenAI	GPT-4o	~200,000
Google DeepMind	Gemma 2	~256,000

How to Index an Item in LLMs

How many tokens in LLMs?

Meta	Llama 3	~128,000
OpenAI	GPT-4o	~200,000
Google DeepMind	Gemma 2	~256,000

How many item IDs?

Amazon-Reviews-2023	~48,200,000

How to Index an Item in LLMs

How many tokens/item IDs in LLMs/RecSys?

Difficult to align these vocabularies given so many tokens

~128,000

~200,000

~256,000

~48,200,000

How to Index an Item in LLMs

Is there a way to
index a large volume of items
using a compact vocabulary?

Semantic IDs

(also called: SemID or SID)

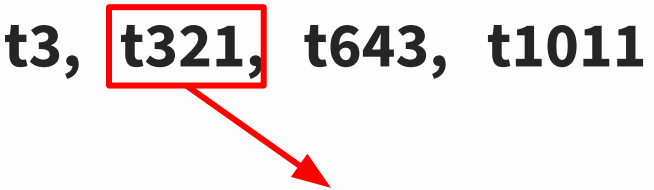
A few tokens that jointly index one item.

t3, t321, t643, t1011

Semantic IDs

(also called: SemID or SID)

A few tokens that jointly index one item.



Each token from a vocabulary shared by all items

Semantic IDs

(also called: SemID or SID)

A few tokens that jointly index one item.

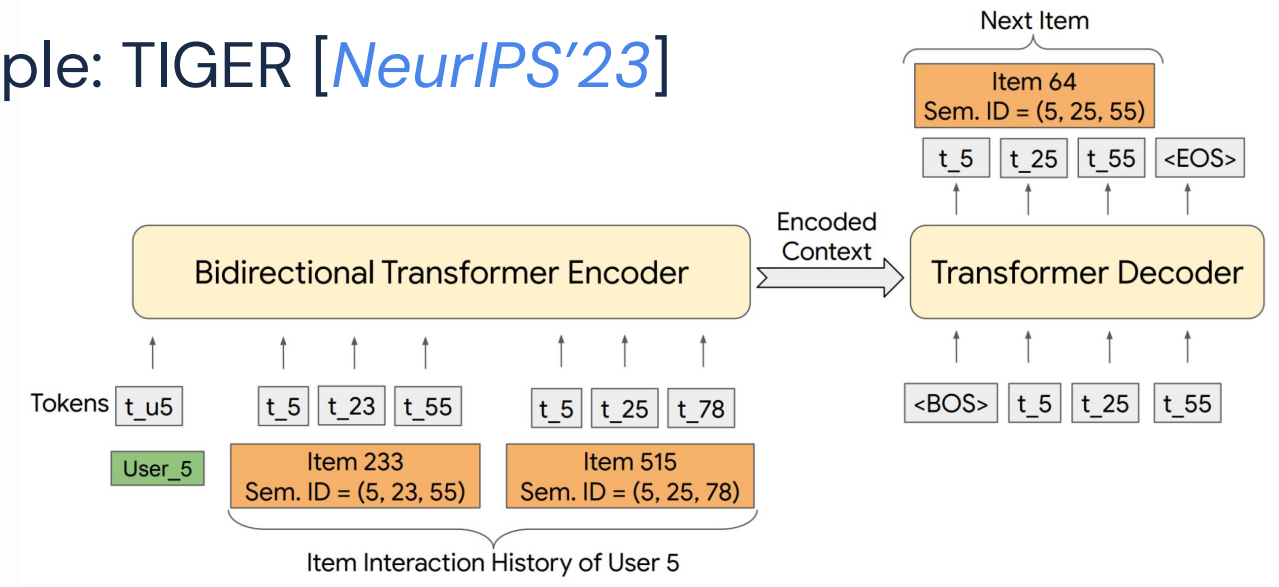
t3, t321, t643, t1011

Can index maximally $256^4 \approx 4.3 \times 10^9$ items with 1024 tokens

(4 tokens per item, each from a vocabulary of 256)

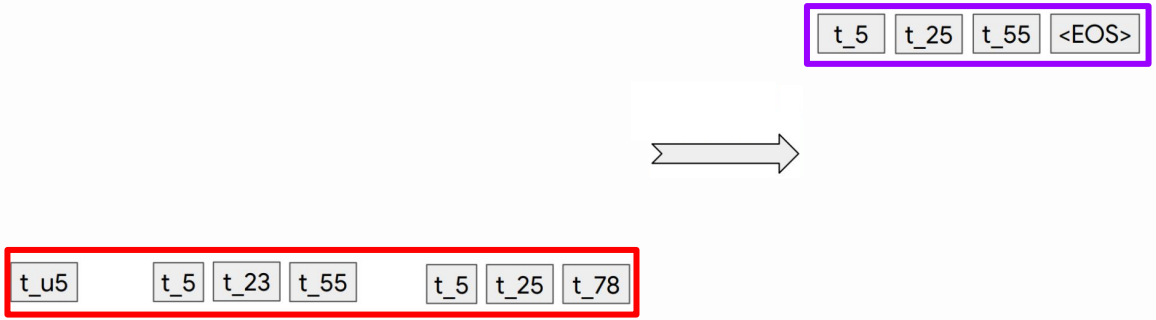
Generative Models based on Semantic IDs

Example: TIGER [*NeurIPS'23*]



Generative Models based on Semantic IDs

Example: TIGER [*NeurIPS'23*]



Recommendation as a **seq**-to-**seq** generation problem

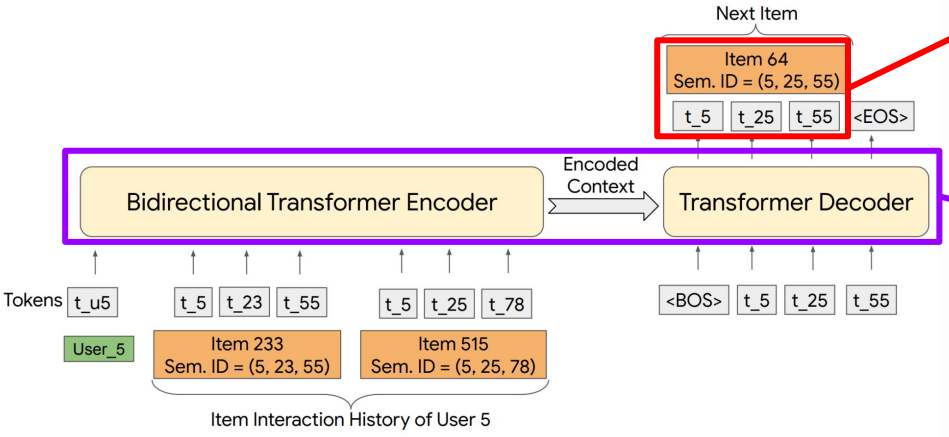
Generative Models based on Semantic IDs

Recommendation as a **seq**-to-**seq** generation problem

Input: user interacted items $\{c_{11}, c_{12}, c_{13}, c_{14}, c_{21}, c_{22}, \ldots\}$

Output: next item $\{c_{t1}, c_{t2}, c_{t3}, c_{t4}\}$

SemID-based Generative Recommendation



Part 1:

How to construct SIDs

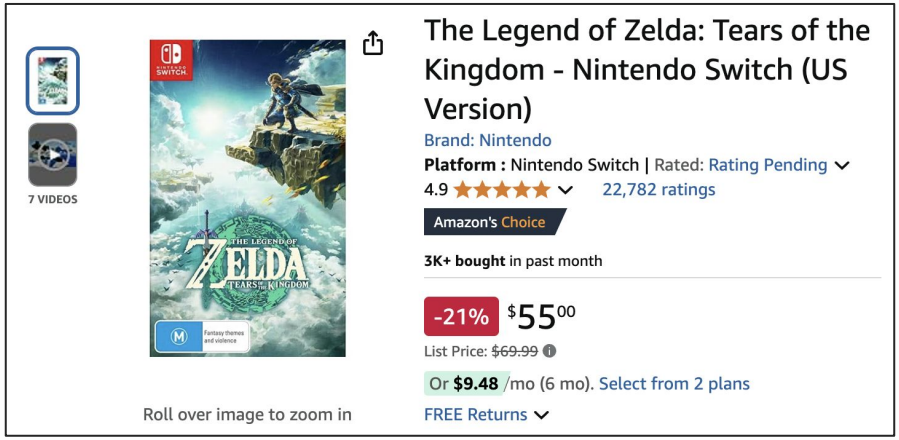
Part 2:

How to build SID-based Generative Rec Models

Part 1: Semantic ID Construction

Semantic ID Construction

Input: all data associated with the item (description, title, interactions, features, …)



Output: mapping between **items** ⇔ **Semantic IDs**

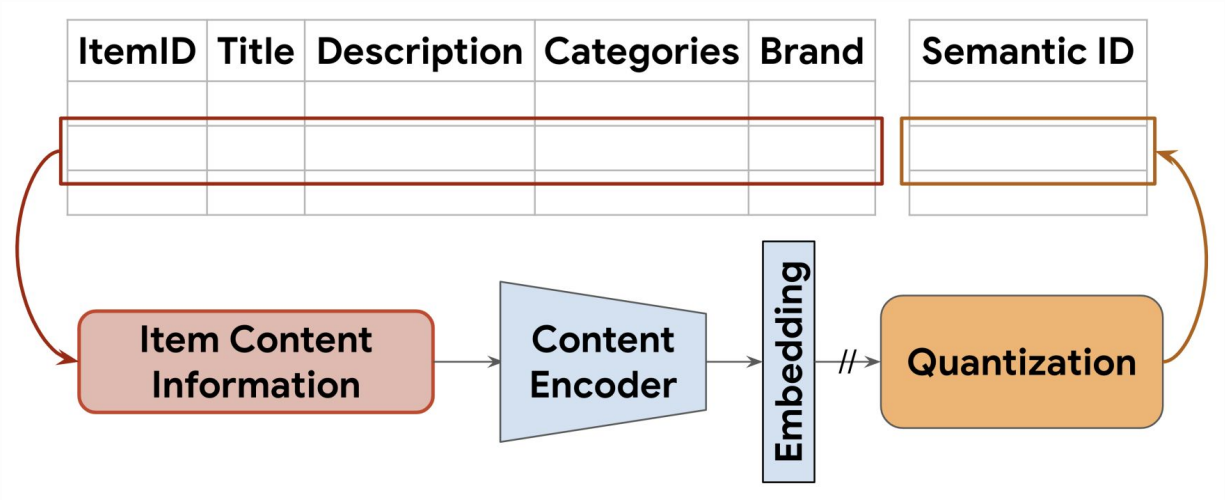
B097B2YWFX ⇔ {t3, t321, t643, t1011}

Part 1: Semantic ID Construction

(i) **First example**: TIGER and RQ-VAE-based SemIDs;

SemID Construction – First Example: TIGER

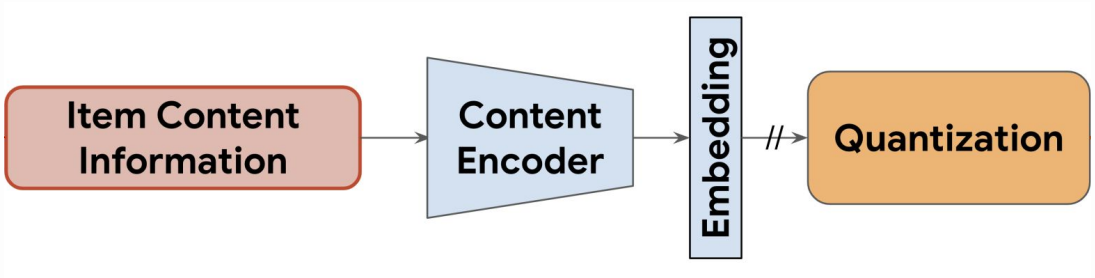
Input: concatenated text features



Output: mapping between **items** ⇔ **Semantic IDs**

SemID Construction – First Example: TIGER

Text ➤ Vector ➤ IDs



Rajput et al. Recommender Systems with Generative Retrieval. NeurIPS 2023.

SemID Construction – First Example: TIGER

1. Item Content Information (Text)

ItemID **Title** **Description**

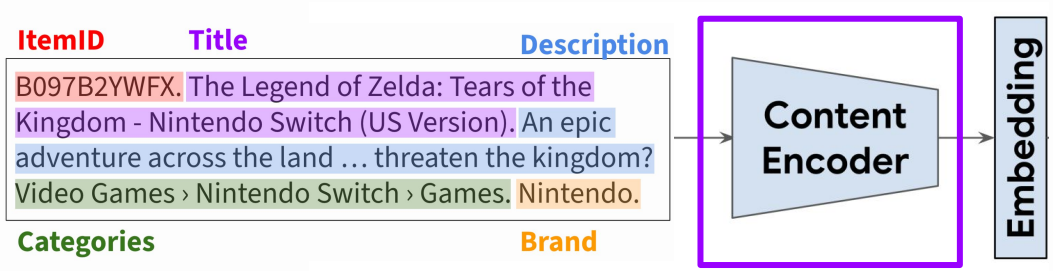
B097B2YWFX. The Legend of Zelda: Tears of the Kingdom - Nintendo Switch (US Version). An epic adventure across the land … threaten the kingdom? Video Games › Nintendo Switch › Games. Nintendo.

Categories **Brand**

SemID Construction – First Example: TIGER

2. Content Encoder + Embedding (Text ➤ Vector)

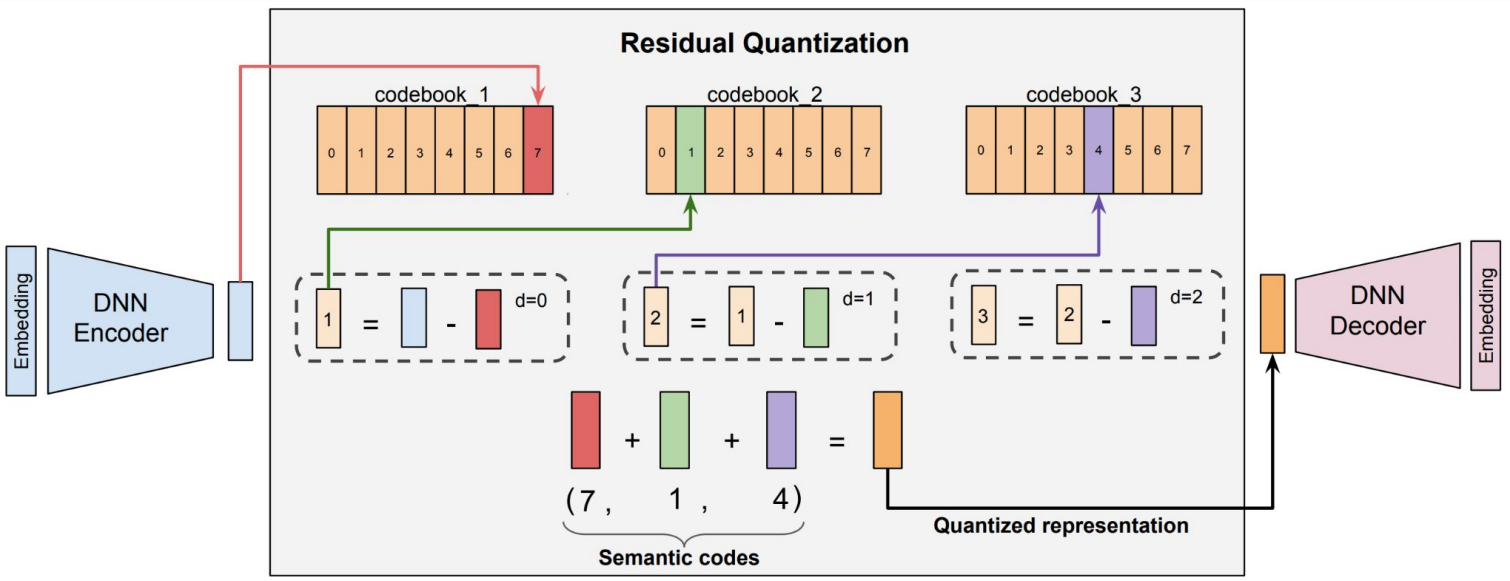
Pre-trained (fixed) sentence embedding model (**SentenceT5**)



Ni et al. Sentence-T5: Scalable Sentence Encoders from Pre-trained Text-to-Text Models. Findings of ACL 2022.
Rajput et al. Recommender Systems with Generative Retrieval. NeurIPS 2023.

SemID Construction – First Example: TIGER

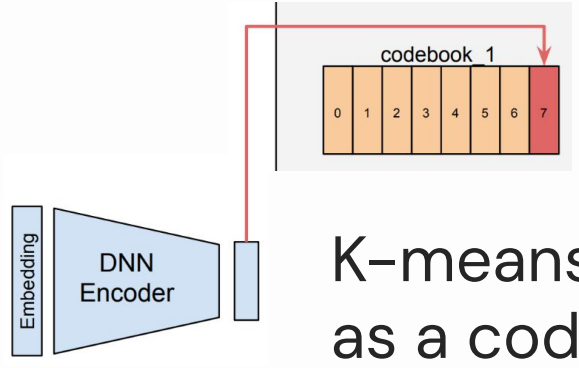
3. RQ-VAE Quantization (Vector ➤ IDs)



Zeghidour et al. SoundStream: An End-to-End Neural Audio Codec. TASLP 2022.
Rajput et al. Recommender Systems with Generative Retrieval. NeurIPS 2023.

SemID Construction – First Example: TIGER

3. RQ-VAE Quantization (Vector ➤ IDs)

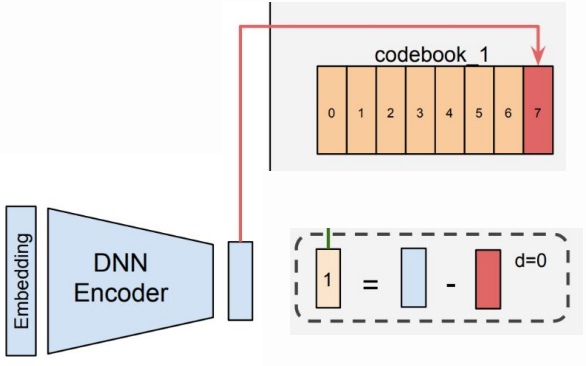


K-means: cluster center ID
as a code in the codebook

Zeghidour et al. SoundStream: An End-to-End Neural Audio Codec. TASLP 2022.
Rajput et al. Recommender Systems with Generative Retrieval. NeurIPS 2023.

SemID Construction – First Example: TIGER

3. RQ-VAE Quantization (Vector ➤ IDs)

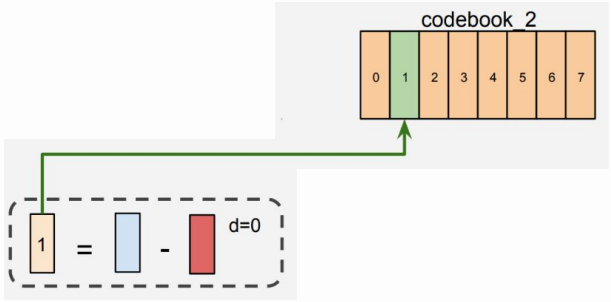


Residual of “input vector” and “clustering center vector”

Zeghidour et al. SoundStream: An End-to-End Neural Audio Codec. TASLP 2022.
Rajput et al. Recommender Systems with Generative Retrieval. NeurIPS 2023.

SemID Construction – First Example: TIGER

3. RQ-VAE Quantization (Vector ➤ IDs)

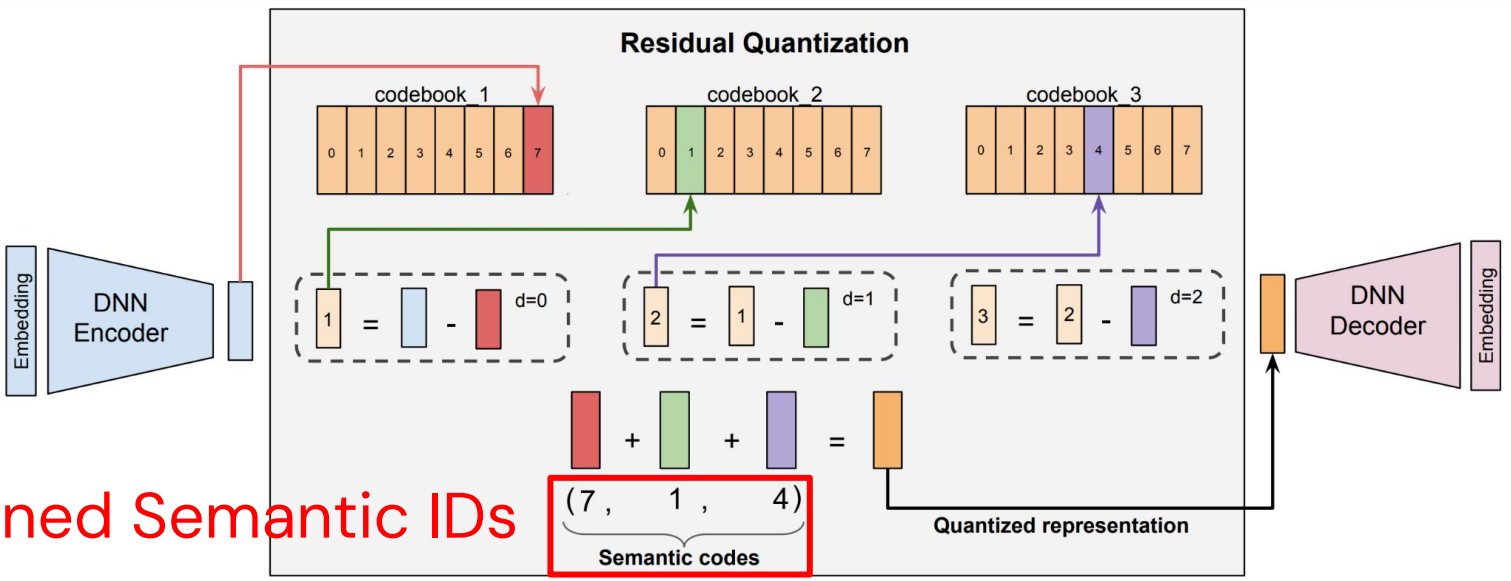


Residual as next level's input

Zeghidour et al. SoundStream: An End-to-End Neural Audio Codec. TASLP 2022.
Rajput et al. Recommender Systems with Generative Retrieval. NeurIPS 2023.

SemID Construction – First Example: TIGER

3. RQ-VAE Quantization (Vector ➤ IDs)



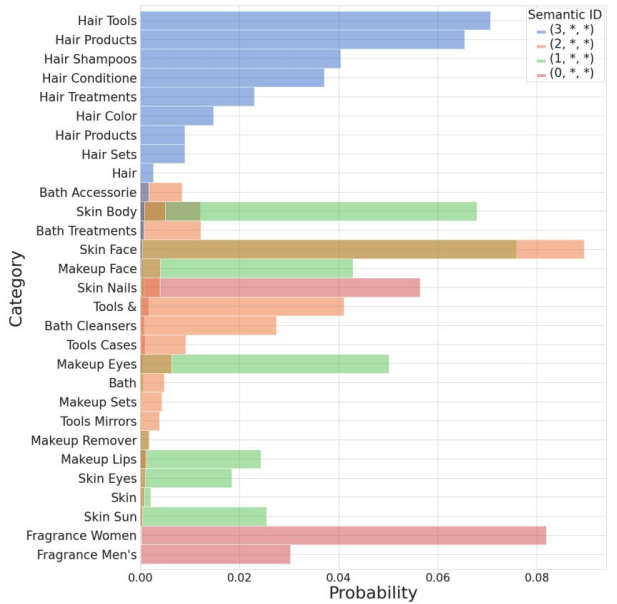
Learned Semantic IDs

Zeghidour et al. SoundStream: An End-to-End Neural Audio Codec. TASLP 2022.
Rajput et al. Recommender Systems with Generative Retrieval. NeurIPS 2023.

SemID Construction – First Example: TIGER

Properties of **RQ-VAE**-based SemIDs

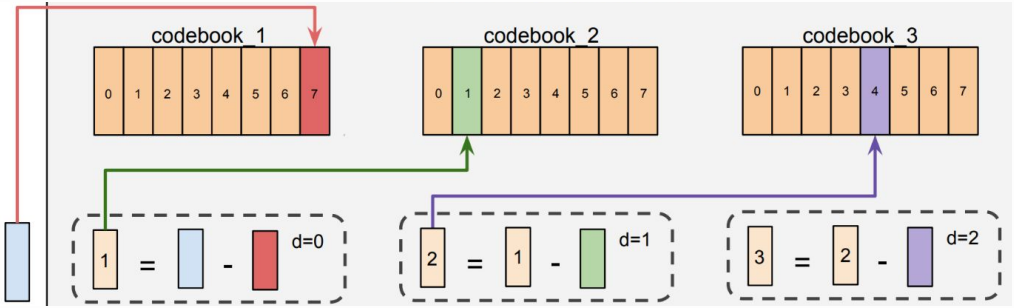
1. Semantic;



SemID Construction – First Example: TIGER

Properties of **RQ-VAE**-based SemIDs

1. Semantic;
2. Ordered / sequential dependent;



SemID Construction – First Example: TIGER

Collisions

(12, 24, 52)

(12, 24, 52)

SemID Construction – First Example: TIGER

Collisions

(12, 24, 52, 0)

(12, 24, 52, 1)

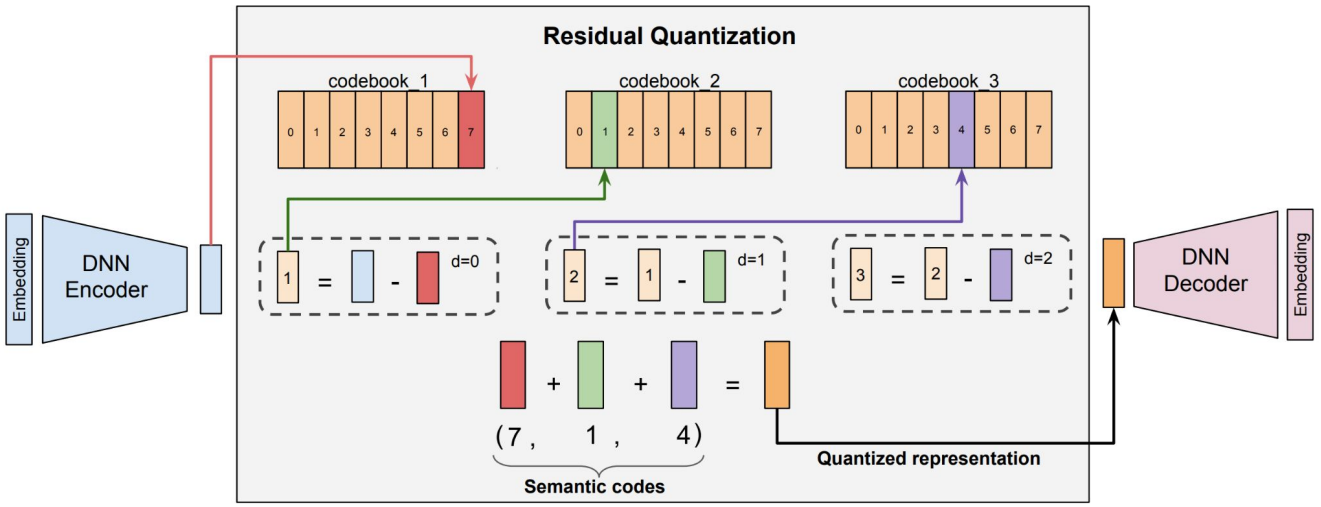
One extra token
to avoid conflicts

Part 1: Semantic ID Construction

(i) First example: TIGER and RQ-VAE-based SemIDs;
(ii) **Techniques** to construct SemIDs;

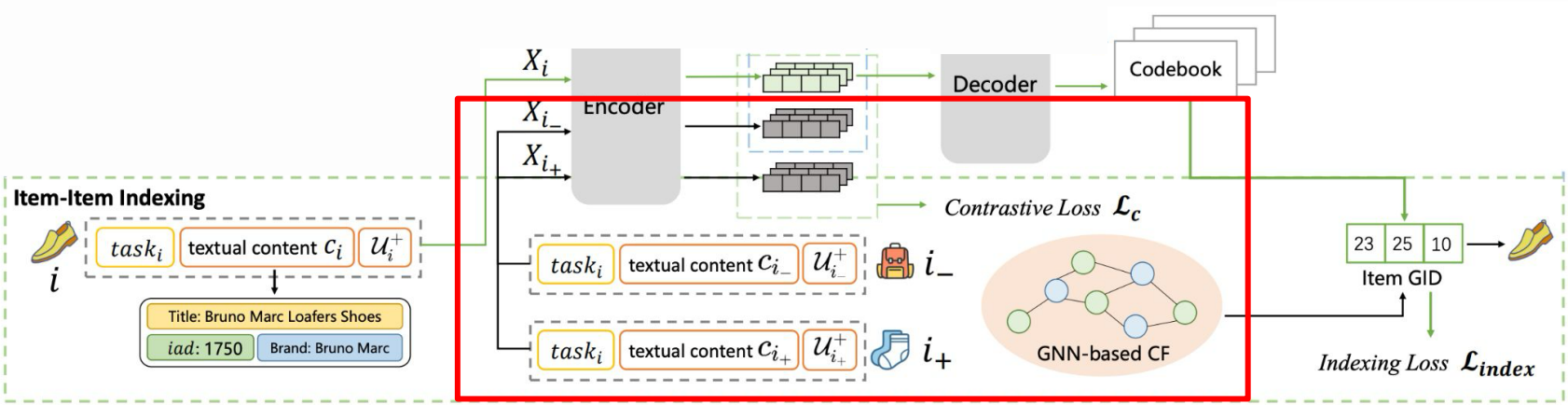
Techniques to Construct SemIDs

Residual Quantization



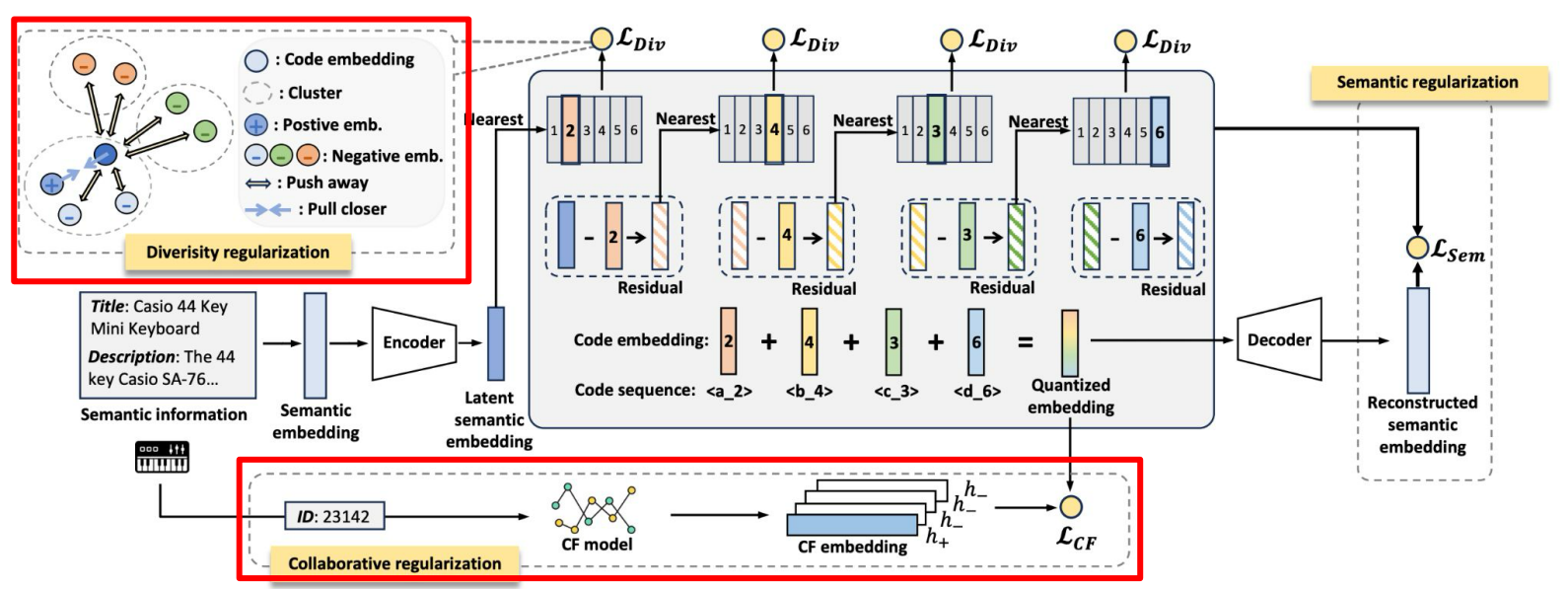
Techniques to Construct SemIDs

Residual Quantization + Item-level Regularization



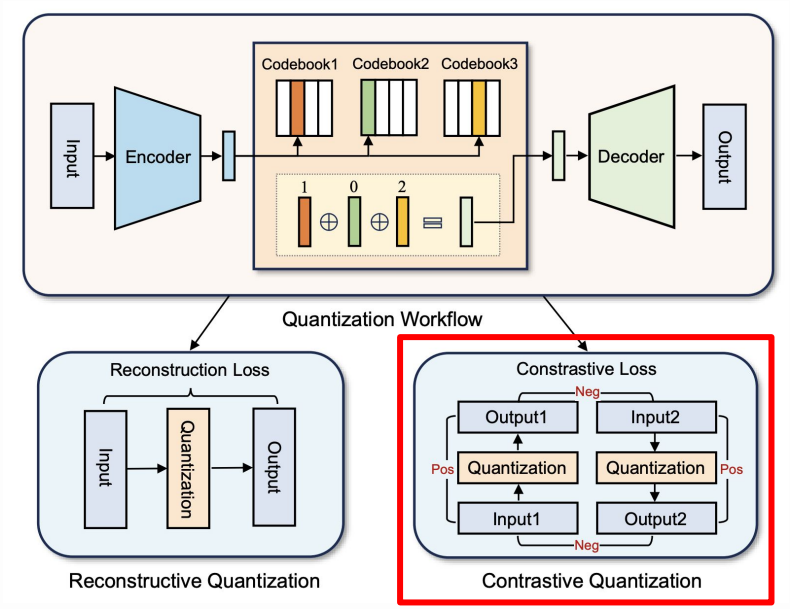
Techniques to Construct SemIDs

Residual Quantization + Item-level Regularization



Techniques to Construct SemIDs

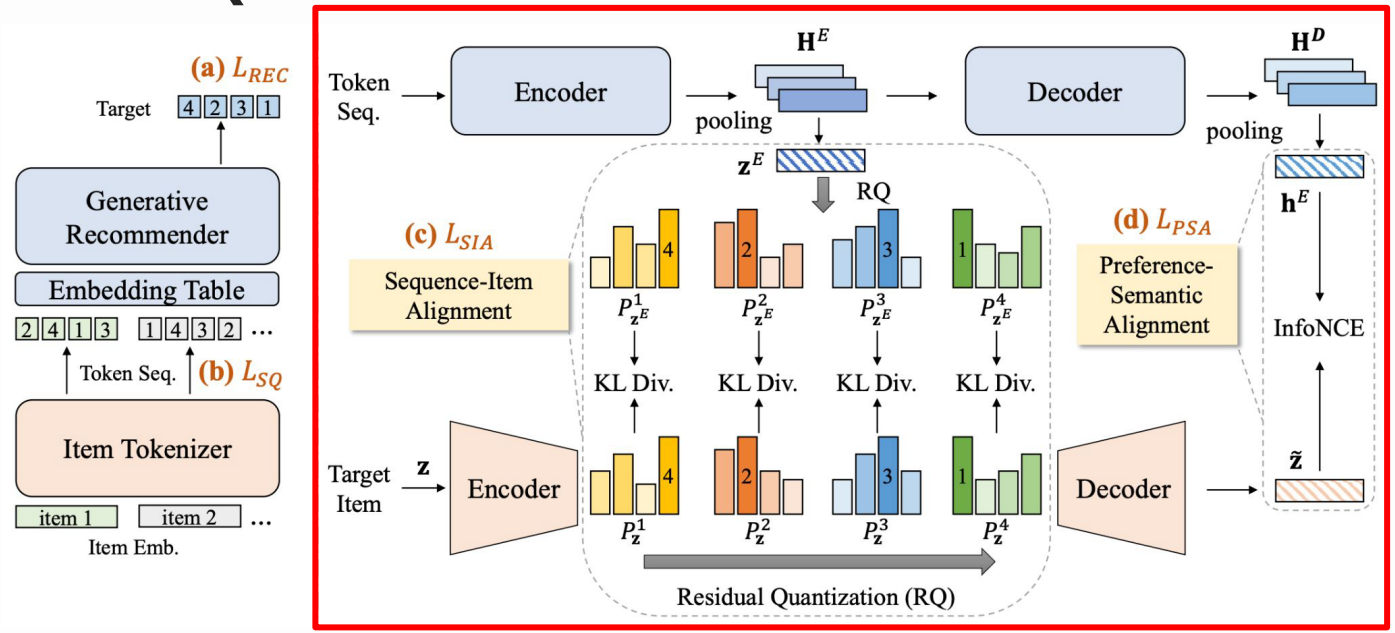
Residual Quantization + Item-level Regularization



Zhu et al. CoST: Contrastive Quantization based Semantic Tokenization for Generative Recommendation. RecSys 2024.

Techniques to Construct SemIDs

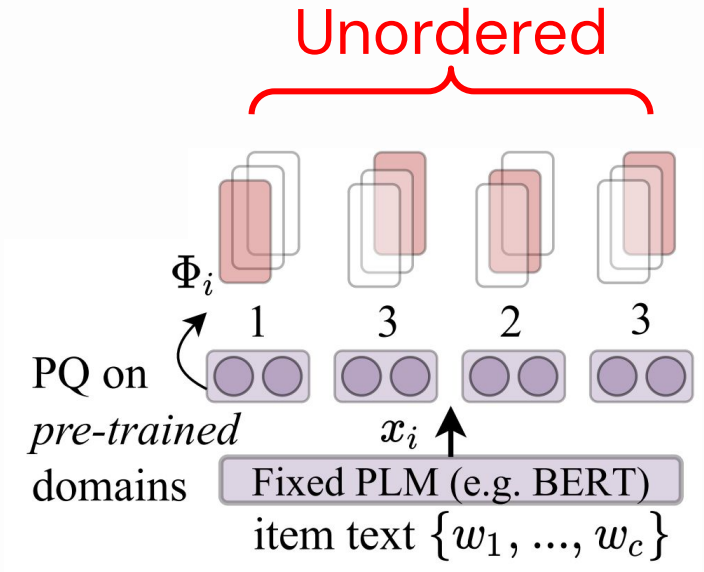
Residual Quantization + Recommendation Loss



Liu et al. End-to-End Learnable Item Tokenization for Generative Recommendation. arXiv:2409.05546.

Techniques to Construct SemIDs

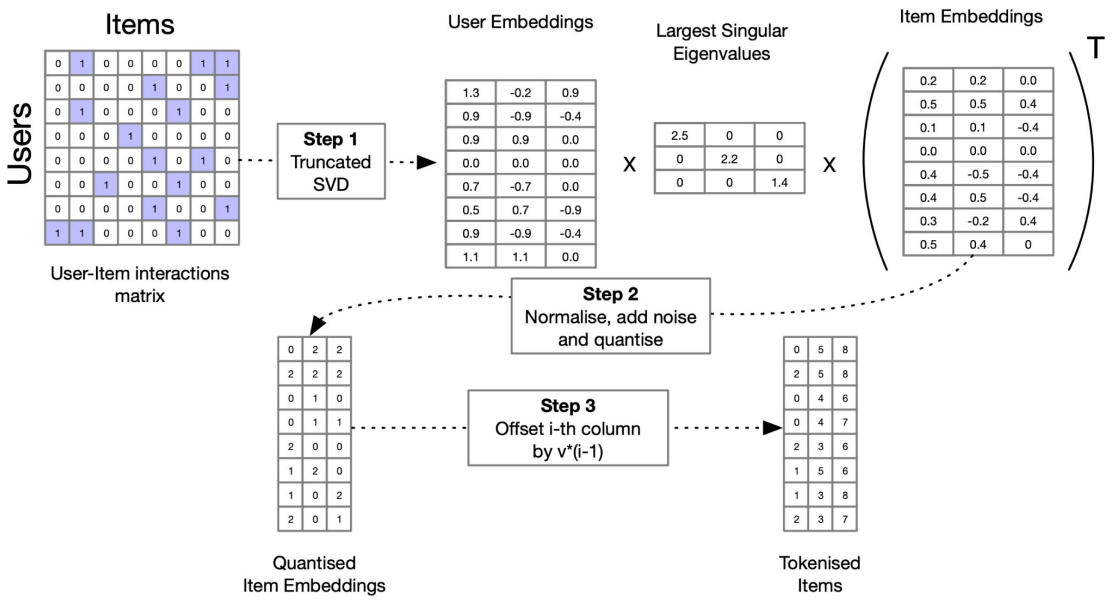
Product Quantization



Hou et al. Learning Vector-Quantized Item Representation for Transferable Sequential Recommenders. WWW 2023.

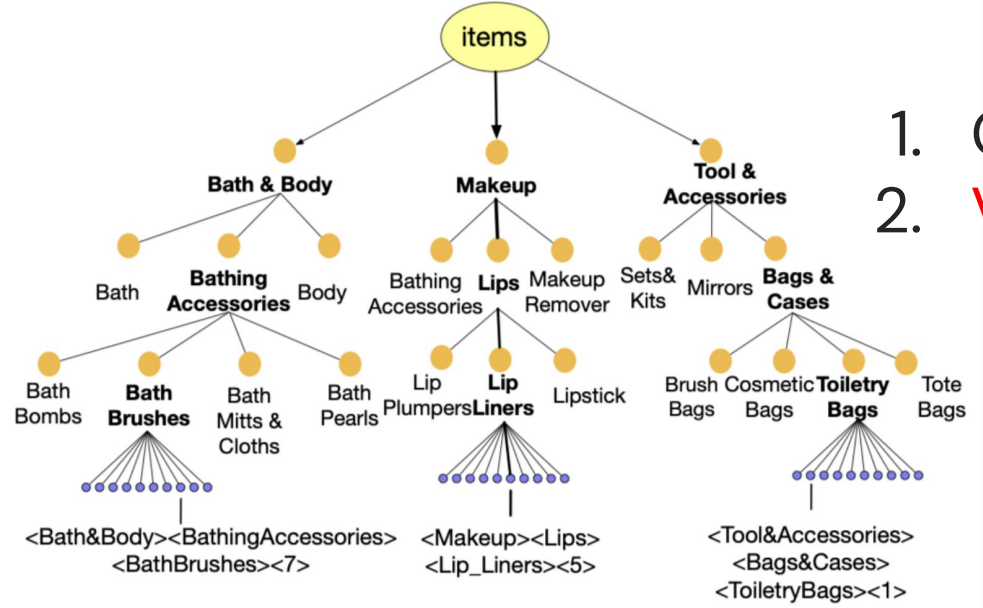
Techniques to Construct SemIDs

Product Quantization



Techniques to Construct SemIDs

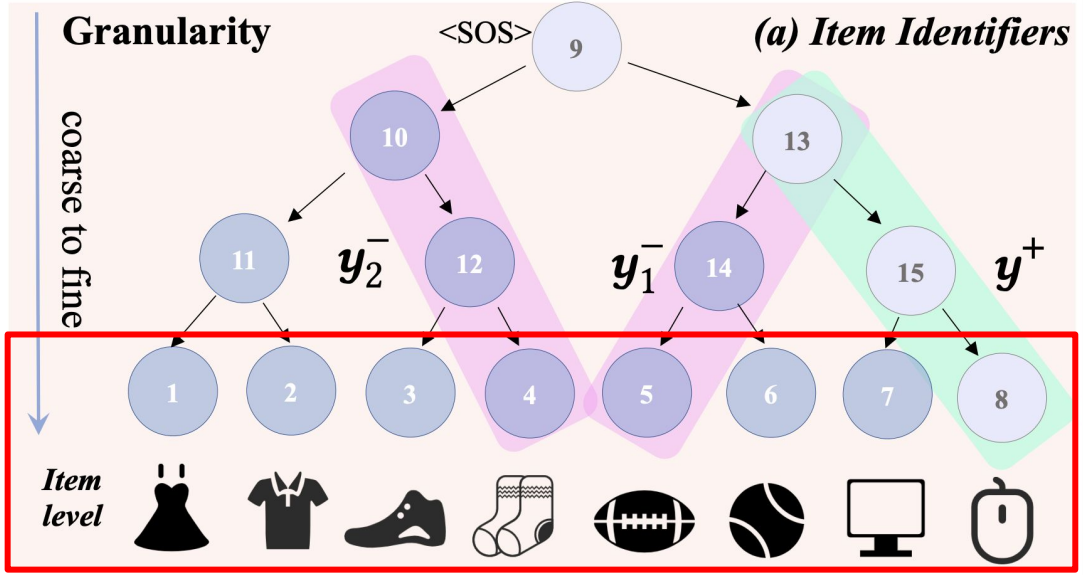
Hierarchical Clustering (Heuristics-based)



1. Ordered;
2. Variable-length SemIDs;

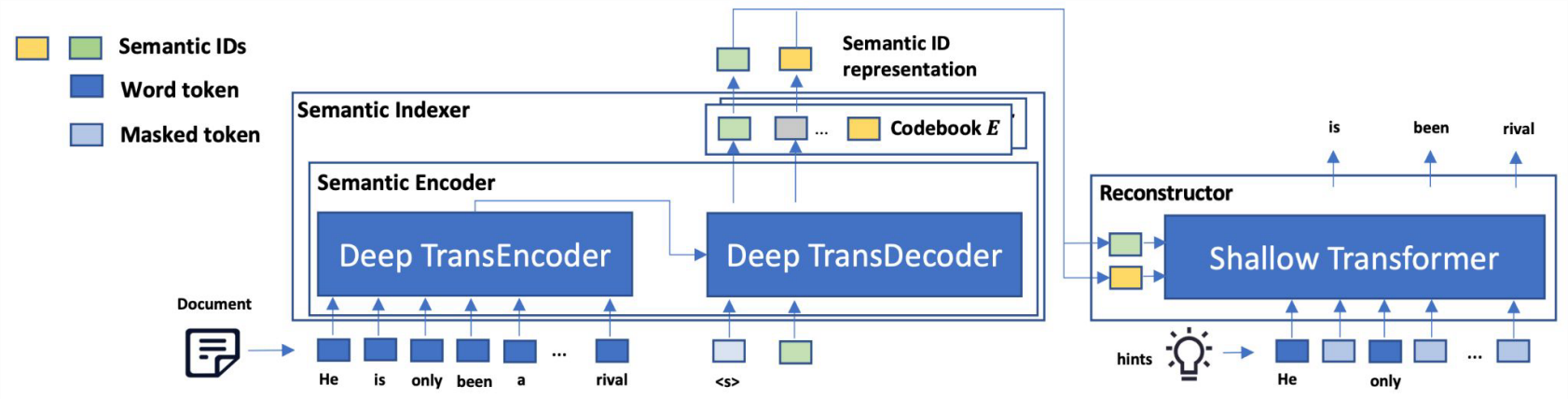
Techniques to Construct SemIDs

Hierarchical Clustering (Latent-based)



Techniques to Construct SemIDs

Language Model-based ID Generator

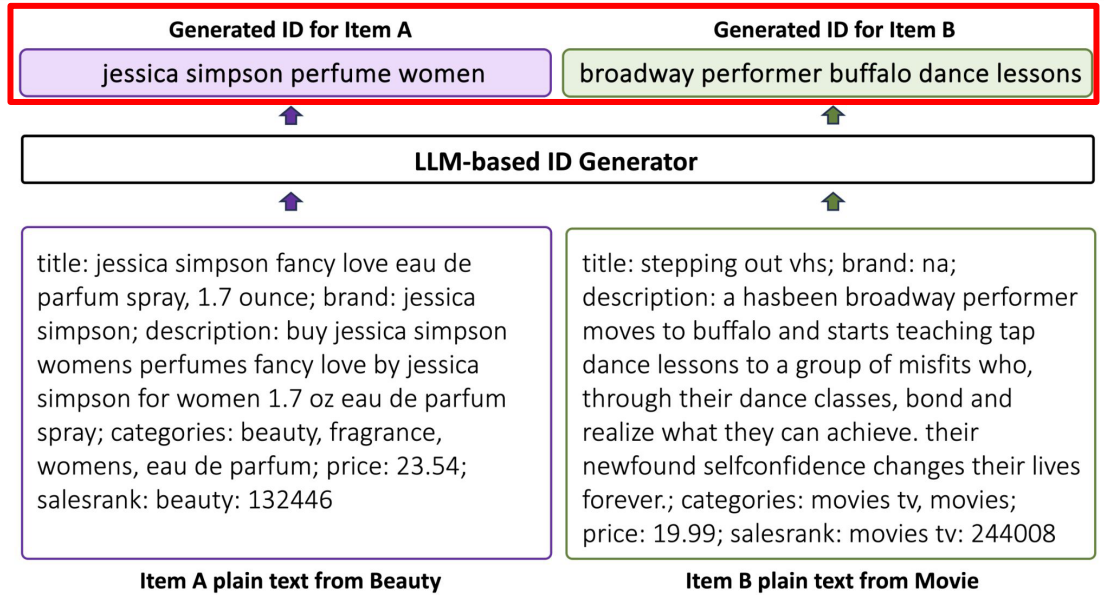


Natural language as inputs;
SemIDs as outputs

Jin et al. Language Models as Semantic Indexers. ICML 2024.

Techniques to Construct SemIDs

Language Model-based ID Generator



Words as SemIDs (like tagging)

Tan et al. IDGenRec: LLM-RecSys Alignment with Textual ID Learning. SIGIR 2024.

Techniques to Construct SemIDs

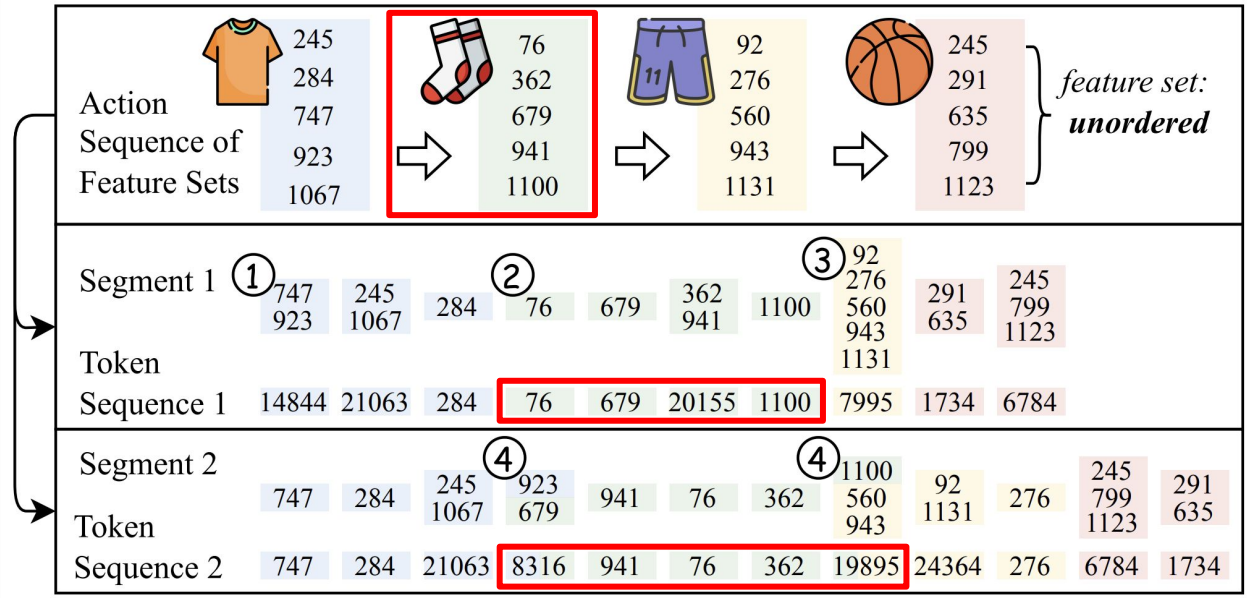
Context-independent

Action Tokenization	Example	Contextual	Unordered
Product Quantization	VQ-Rec (Hou et al., 2023)	✗	✓
Hierarchical Clustering	P5-CID (Hua et al., 2023)	✗	✗
Residual Quantization	TIGER (Rajput et al., 2023)	✗	✗
Text Tokenization	LMIndexer (Jin et al., 2024)	✗	✗
Raw Features	HSTU (Zhai et al., 2024)	✗	✗
SentencePiece	SPM-SID (Singh et al., 2024)	✗	✗

Same item ⇒ fixed semIDs in all sequences

Techniques to Construct SemIDs

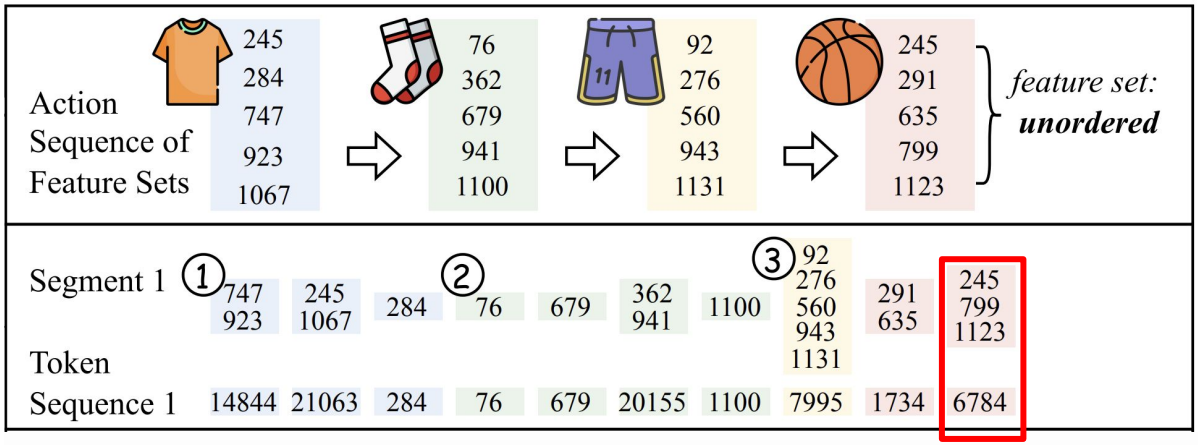
Context-independent ⇒ Context-aware



Same item ⇒ different semIDs based on context

Techniques to Construct SemIDs

Context-independent ⇒ Context-aware



Core Idea:

Merge frequently co-occurring features as new tokens

(**ActionPiece**: "WordPiece" tokenization for generative rec)

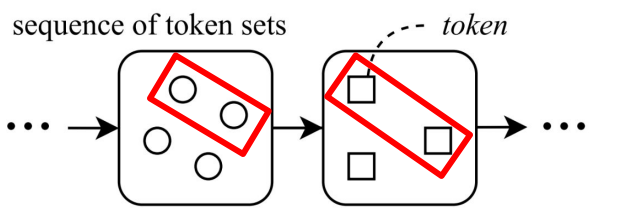
Techniques to Construct SemIDs

Context-independent ⇒ Context-aware

Algorithm 1 ActionPiece Vocabulary Construction

input Sequence corpus $\mathcal{S}'$, initial tokens $\mathcal{V}_0$, target size Q
output Merge rules $\mathcal{R}$, constructed vocabulary $\mathcal{V}$

Initialize vocabulary $\mathcal{V} \leftarrow \mathcal{V}_0$ # each initial token corresponds to one unique item feature
$\mathcal{R} \leftarrow \emptyset$
while $|\mathcal{V}| < Q$ **do**
  # ***Count:*** accumulate weighted token co-occurrences
  $\text{count}(\cdot,\cdot) \leftarrow \text{Count}(\mathcal{S}', \mathcal{V})$ # Algorithm 2
  # ***Update:*** Merge a frequent token pair into a new token
  Select $(c_u, c_v) \leftarrow \arg\max_{(c_i,c_j)} \text{count}(c_i, c_j)$
  $\mathcal{S}' \leftarrow \text{Update}(\mathcal{S}', \{(c_u, c_v) \rightarrow c_{\text{new}}\})$ # Algorithm 3
  $\mathcal{R} \leftarrow \mathcal{R} \cup \{(c_u, c_v) \rightarrow c_{\text{new}}\}$ # new merge rule
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{c_{\text{new}}\}$ # add new token to the vocabulary
end while
return $\mathcal{R}, \mathcal{V}$



$$P(\bigcirc,\bigcirc) = \frac{|\bigcirc\!\!-\!\!\bigcirc|}{|<\bigcirc,\bigcirc>|} = \frac{4-1}{\binom{4}{2}}$$

$$P(\square,\square) = \frac{|\square\!\!-\!\!\square|}{|<\square,\square>|} = \frac{3-1}{\binom{3}{2}}$$

Features co-occurring within items

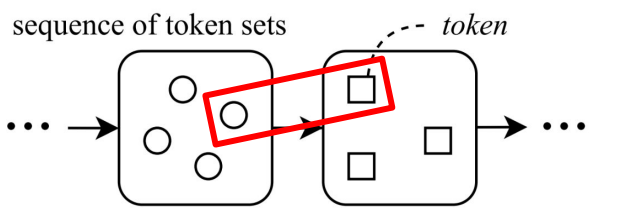
Techniques to Construct SemIDs

Context-independent ⇒ Context-aware

Algorithm 1 ActionPiece Vocabulary Construction

input Sequence corpus $\mathcal{S}'$, initial tokens $\mathcal{V}_0$, target size Q
output Merge rules $\mathcal{R}$, constructed vocabulary $\mathcal{V}$

Initialize vocabulary $\mathcal{V} \leftarrow \mathcal{V}_0$ # each initial token corresponds to one unique item feature
$\mathcal{R} \leftarrow \emptyset$
while $|\mathcal{V}| < Q$ **do**
  # ***Count:*** accumulate weighted token co-occurrences
  $\text{count}(\cdot,\cdot) \leftarrow \text{Count}(\mathcal{S}', \mathcal{V})$ # Algorithm 2
  # ***Update:*** Merge a frequent token pair into a new token
  Select $(c_u, c_v) \leftarrow \arg\max_{(c_i,c_j)} \text{count}(c_i, c_j)$
  $\mathcal{S}' \leftarrow \text{Update}(\mathcal{S}', \{(c_u, c_v) \rightarrow c_{\text{new}}\})$ # Algorithm 3
  $\mathcal{R} \leftarrow \mathcal{R} \cup \{(c_u, c_v) \rightarrow c_{\text{new}}\}$ # new merge rule
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{c_{\text{new}}\}$ # add new token to the vocabulary
end while
return $\mathcal{R}, \mathcal{V}$



$$P(\bigcirc,\bigcirc) = \frac{|\bigcirc\!\!-\!\!\bigcirc|}{|<\bigcirc,\bigcirc>|} = \frac{4-1}{\binom{4}{2}}$$

$$P(\bigcirc,\square) = \frac{|\bigcirc\!\!-\!\!\square|}{|\bigcirc| \times |\square|} = \frac{1}{4 \times 3}$$

$$P(\square,\square) = \frac{|\square\!\!-\!\!\square|}{|<\square,\square>|} = \frac{3-1}{\binom{3}{2}}$$

Features co-occurring within or across items

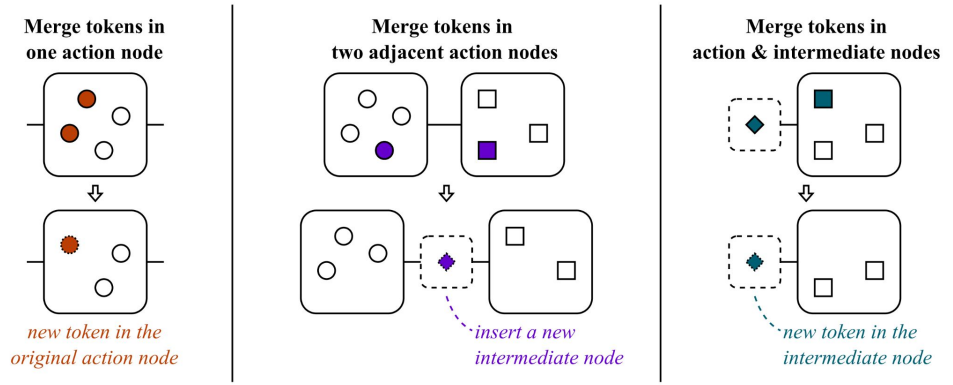
Techniques to Construct SemIDs

Context-independent ⇒ Context-aware

Algorithm 1 ActionPiece Vocabulary Construction

input Sequence corpus $\mathcal{S}'$, initial tokens $\mathcal{V}_0$, target size Q
output Merge rules $\mathcal{R}$, constructed vocabulary $\mathcal{V}$

Initialize vocabulary $\mathcal{V} \leftarrow \mathcal{V}_0$ # each initial token corresponds to one unique item feature
$\mathcal{R} \leftarrow \emptyset$
while $|\mathcal{V}| < Q$ **do**
  # ***Count:*** accumulate weighted token co-occurrences
  $\text{count}(\cdot,\cdot) \leftarrow \text{Count}(\mathcal{S}', \mathcal{V})$ # Algorithm 2
  # ***Update:*** Merge a frequent token pair into a new token
  Select $(c_u, c_v) \leftarrow \arg\max_{(c_i,c_j)} \text{count}(c_i, c_j)$
  $\mathcal{S}' \leftarrow \text{Update}(\mathcal{S}', \{(c_u, c_v) \rightarrow c_{\text{new}}\})$ # Algorithm 3
  $\mathcal{R} \leftarrow \mathcal{R} \cup \{(c_u, c_v) \rightarrow c_{\text{new}}\}$ # new merge rule
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{c_{\text{new}}\}$ # add new token to the vocabulary
end while
return $\mathcal{R}, \mathcal{V}$



Summary of
Techniques to Construct SemIDs

Context-independent

- Residual Quantization (+ regularization)
- Product Quantization
- Hierarchical Clustering
- LM-based ID Generator

Summary of
Techniques to Construct SemIDs

Context-independent

- Residual Quantization (+ regularization)
- Product Quantization
- Hierarchical Clustering
- LM-based ID Generator

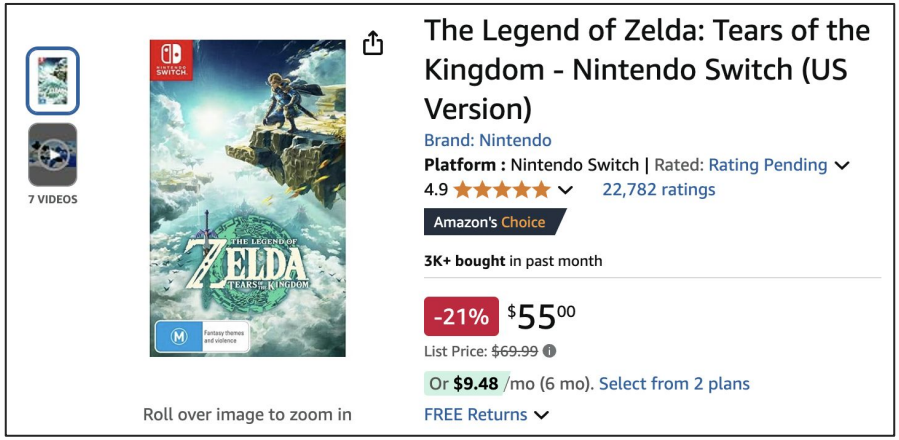
Context-aware

Part 1: Semantic ID Construction

(i) First example: TIGER and RQ-VAE-based SemIDs;
(ii) Techniques to construct SemIDs;
(iii) **Inputs** for SemID construction;

Inputs for SemID Construction

Input: all data associated with the item



Inputs for SemID Construction

Input: all data associated with the item

What exactly does “all data” mean? 🤷

Inputs for SemID Construction

Text or Multimodal Features

Text/Visual/Acoustic ➤ **Vector** ➤ **IDs**

Pretrained Encoder Quantization

ItemID **Title** **Description**

B097B2YWFX. The Legend of Zelda: Tears of the Kingdom - Nintendo Switch (US Version). An epic adventure across the land … threaten the kingdom? Video Games › Nintendo Switch › Games. Nintendo.

Categories **Brand**

Text

Multimodal

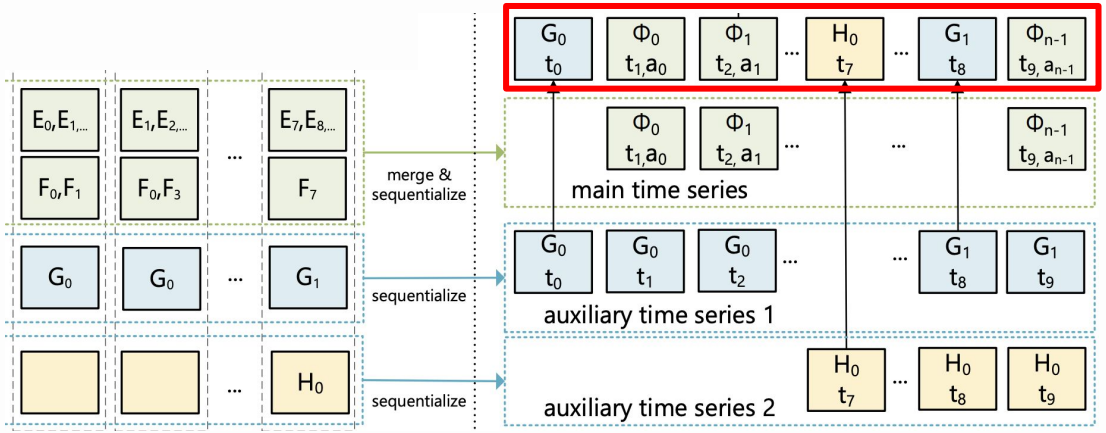
Deng et al. OneRec: Unifying Retrieve and Rank with Generative Recommender and Preference Alignment. arXiv:2502.18965.
Liu et al. MMGRec: Multimodal Generative Recommendation with Transformer Model. arXiv:2404.16555.

Inputs for SemID Construction

Categorical Features

Categorical Features ➤ IDs

Merge & Sequentialize

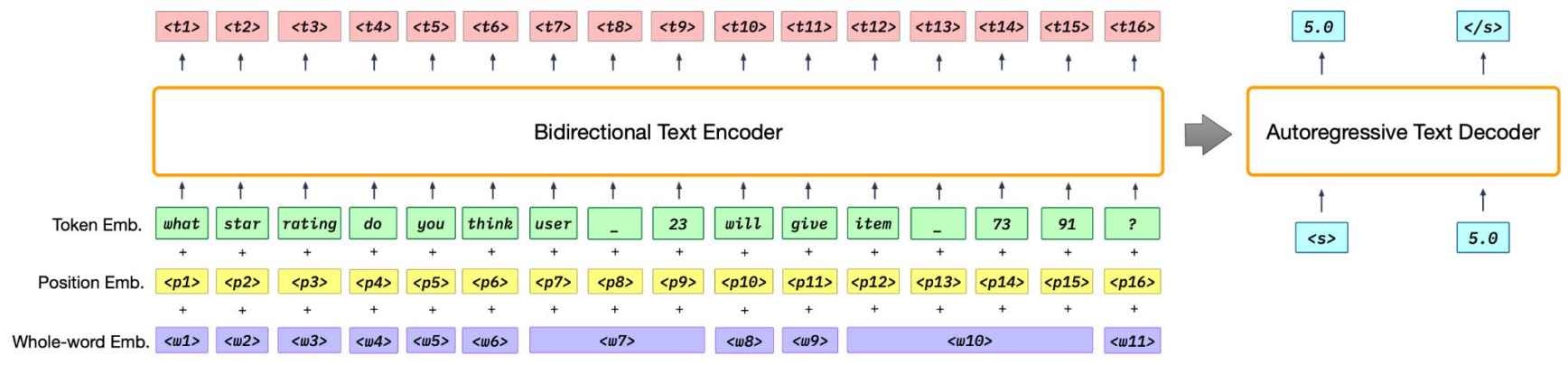


Inputs for SemID Construction

No Features

Item ID ➤ **IDs**

Text Tokenizer

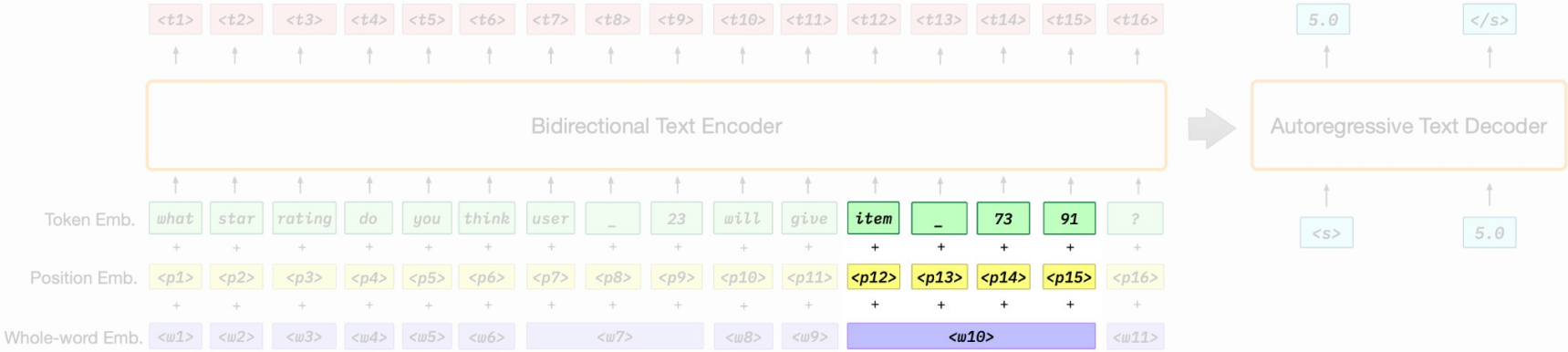


Inputs for SemID Construction

No Features

Item ID ➤ **IDs**

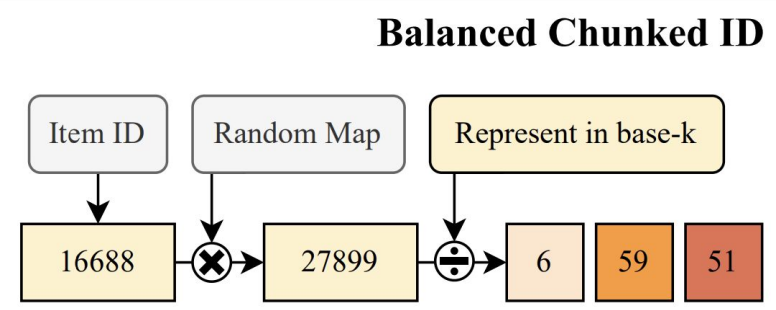
Text Tokenizer



Inputs for SemID Construction

No Features

Random IDs



Inputs for SemID Construction

Input: all data associated with the item

(1) Item Metadata

Text / Multimodal / Categorical / No Features

Inputs for SemID Construction

Input: all data associated with the item

(1) Item Metadata

Text / Multimodal / Categorical / No Features

(2) Item Metadata + Behaviors

Inputs for SemID Construction

Input: all data associated with the item

(1) Item Metadata

Text / Multimodal / Categorical / No Features

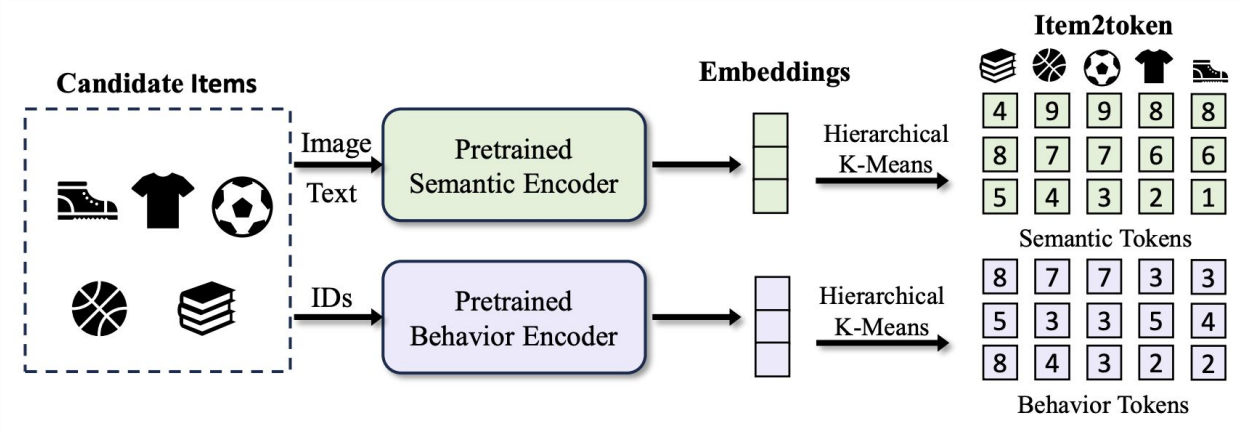
(2) Item Metadata + Behaviors

But how?

Inputs for SemID Construction

Item Metadata + Behaviors

Fused Semantic IDs

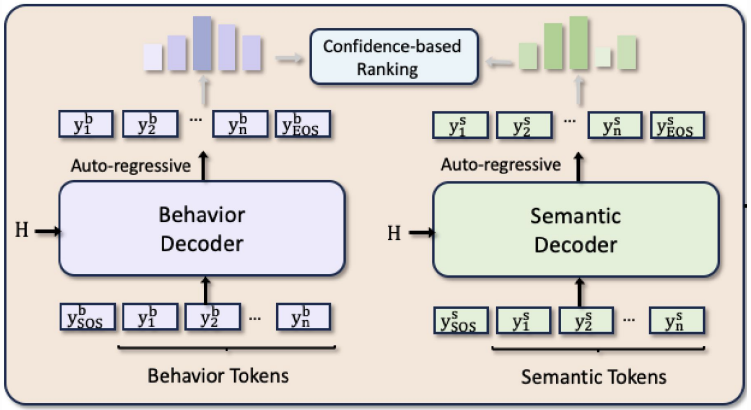


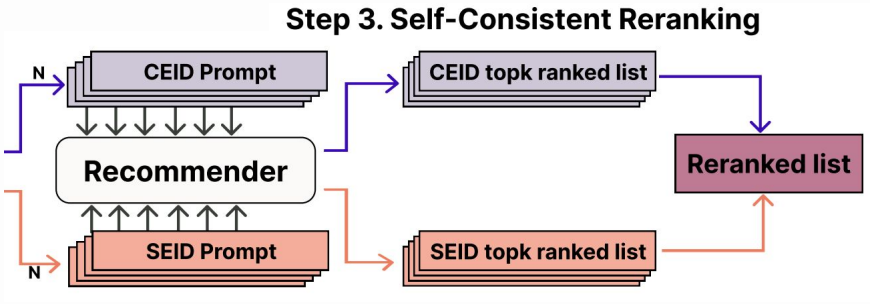
Wang et al. EAGER: Two-Stream Generative Recommender with Behavior-Semantic Collaboration. KDD 2024.

Inputs for SemID Construction

Item Metadata + Behaviors

Fused Semantic IDs + Two-stream Generation





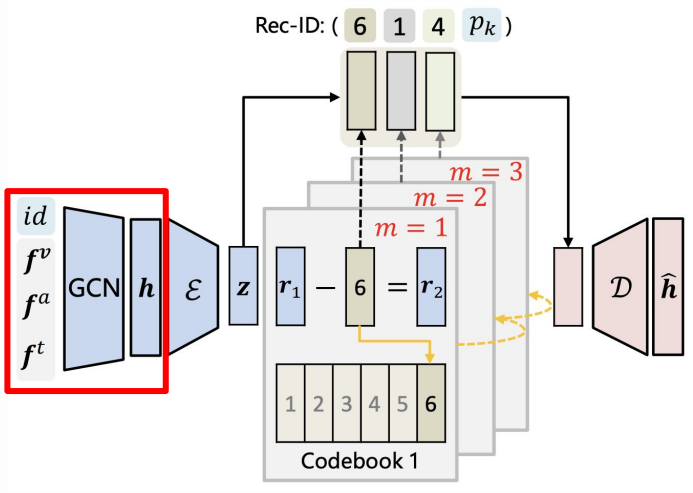
Wang et al. EAGER: Two-Stream Generative Recommender with Behavior-Semantic Collaboration. KDD 2024.
Kim et al. SC-Rec: Enhancing Generative Retrieval with Self-Consistent Reranking for Sequential Recommendation. arXiv:2408:08686.

Inputs for SemID Construction

Item Metadata + Behaviors

Fused Representations

User–Item Graph +
Semantic Features



Inputs for SemID Construction

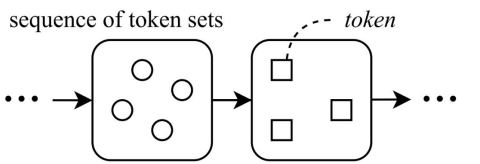
Item Metadata + Behaviors

Train Tokenizer on Behavior Sequence Corpus

Algorithm 1 ActionPiece Vocabulary Construction

input Sequence corpus $\mathcal{S}'$, initial tokens $\mathcal{V}_0$, target size Q
output Merge rules $\mathcal{R}$, constructed vocabulary $\mathcal{V}$

Initialize vocabulary $\mathcal{V} \leftarrow \mathcal{V}_0$ # each initial token corresponds to one unique item feature
$\mathcal{R} \leftarrow \emptyset$
while $|\mathcal{V}| < Q$ **do**
  # ***Count:*** accumulate weighted token co-occurrences
  $\text{count}(\cdot,\cdot) \leftarrow \text{Count}(\mathcal{S}', \mathcal{V})$ # Algorithm 2
  # ***Update:*** Merge a frequent token pair into a new token
  Select $(c_u, c_v) \leftarrow \arg\max_{(c_i,c_j)} \text{count}(c_i, c_j)$
  $\mathcal{S}' \leftarrow \text{Update}(\mathcal{S}', \{(c_u, c_v) \rightarrow c_{\text{new}}\})$ # Algorithm 3
  $\mathcal{R} \leftarrow \mathcal{R} \cup \{(c_u, c_v) \rightarrow c_{\text{new}}\}$ # new merge rule
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{c_{\text{new}}\}$ # add new token to the vocabulary
end while
return $\mathcal{R}, \mathcal{V}$



$$P(\bigcirc,\bigcirc) = \frac{|\bigcirc\!-\!\bigcirc|}{|<\bigcirc,\bigcirc>|} = \frac{4-1}{\binom{4}{2}}$$

$$P(\bigcirc,\square) = \frac{|\bigcirc\!-\!\square|}{|\bigcirc| \times |\square|} = \frac{1}{4 \times 3}$$

$$P(\square,\square) = \frac{|\square\!-\!\square|}{|<\square,\square>|} = \frac{3-1}{\binom{3}{2}}$$

Features co-occurring within or across items

Inputs for SemID Construction

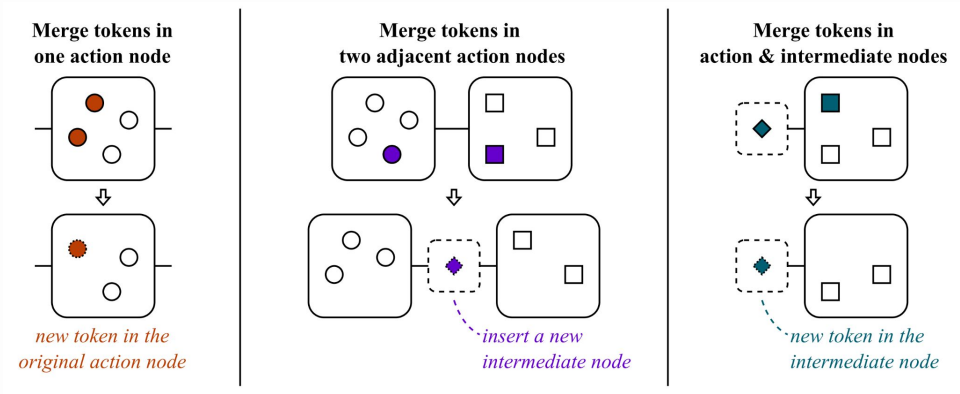
Item Metadata + Behaviors

Train Tokenizer on Behavior Sequence Corpus

Algorithm 1 ActionPiece Vocabulary Construction

input Sequence corpus $\mathcal{S}'$, initial tokens $\mathcal{V}_0$, target size Q
output Merge rules $\mathcal{R}$, constructed vocabulary $\mathcal{V}$

Initialize vocabulary $\mathcal{V} \leftarrow \mathcal{V}_0$ # each initial token corresponds to one unique item feature
$\mathcal{R} \leftarrow \emptyset$
while $|\mathcal{V}| < Q$ **do**
  # ***Count:*** accumulate weighted token co-occurrences
  $\text{count}(\cdot,\cdot) \leftarrow \text{Count}(\mathcal{S}', \mathcal{V})$ # Algorithm 2
  # ***Update:*** Merge a frequent token pair into a new token
  Select $(c_u, c_v) \leftarrow \arg\max_{(c_i,c_j)} \text{count}(c_i, c_j)$
  $\mathcal{S}' \leftarrow \text{Update}(\mathcal{S}', \{(c_u, c_v) \rightarrow c_{\text{new}}\})$ # Algorithm 3
  $\mathcal{R} \leftarrow \mathcal{R} \cup \{(c_u, c_v) \rightarrow c_{\text{new}}\}$ # new merge rule
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{c_{\text{new}}\}$ # add new token to the vocabulary
end while
return $\mathcal{R}, \mathcal{V}$

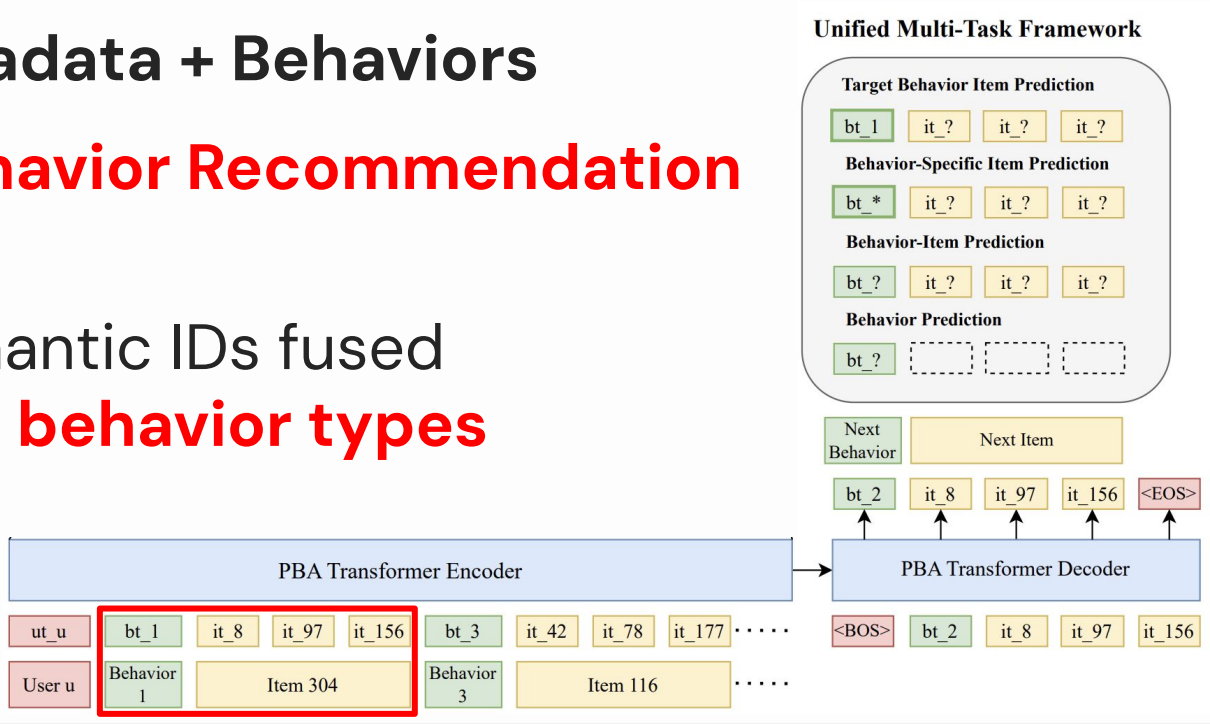


Inputs for SemID Construction

Item Metadata + Behaviors

Multi-Behavior Recommendation

Semantic IDs fused
with **behavior types**



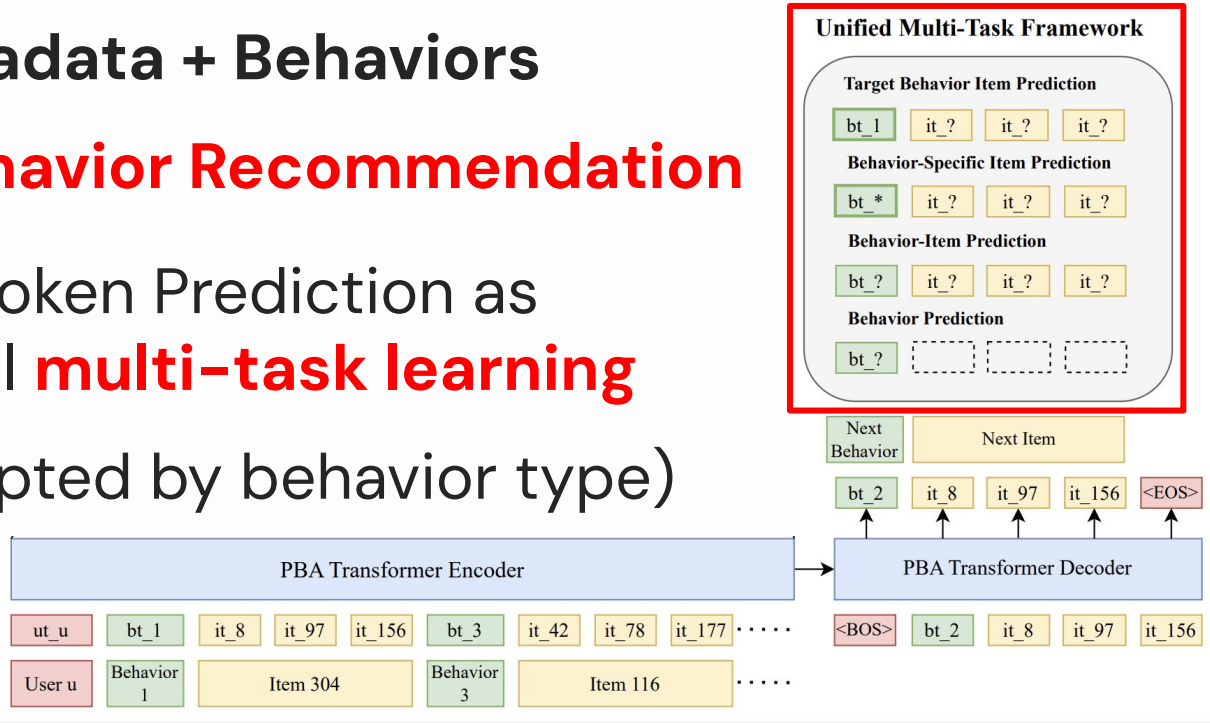
Inputs for SemID Construction

Item Metadata + Behaviors

Multi-Behavior Recommendation

Next Token Prediction as natural **multi-task learning**

(prompted by behavior type)



Inputs for SemID Construction

Input: all data associated with the item

(1) Item Metadata

Text / Multimodal / Categorical / No Features

(2) Item Metadata + Behaviors

Fused semantic IDs & Representations

Tokenizer trained on behavior sequences

Part 1 Summary – SemID Construction

(1) First Example: TIGER

(2) Construction Techniques

(3) Inputs

Part 1 Summary – SemID Construction

(1) First Example: TIGER

(2) Construction Techniques

Context-independent (PQ, RQ, Clustering, LM-based generator) -> Context-aware

(3) Inputs

Part 1 Summary – SemID Construction

(1) First Example: TIGER

(2) Construction Techniques

Context-independent (PQ, RQ, Clustering, LM-based generator) -> Context-aware

(3) Inputs

Item Metadata (Text, Multimodal, Features)

+ Behaviors (Fused SemIDs / Representations)

Part 2: SemID-based Generative Recommendation Model Architecture

SemID-based Recommender Architecture

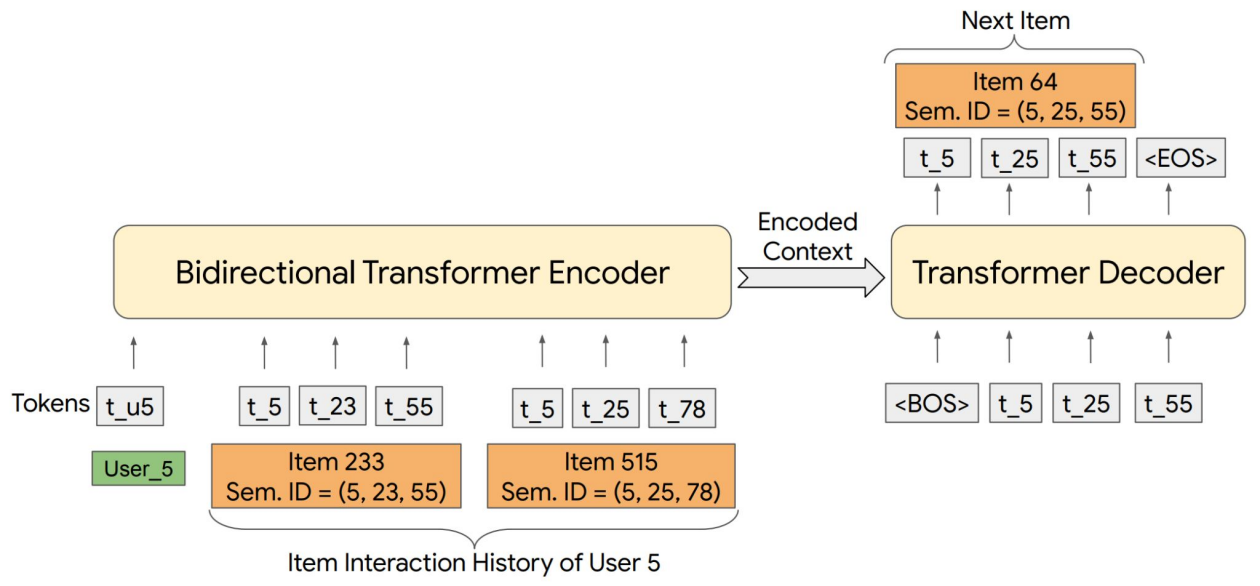
Recommendation as a **seq**-to-**seq** generation problem

Input: user interacted items $\{c_{11}, c_{12}, c_{13}, c_{14}, c_{21}, c_{22}, \ldots\}$

Output: next item $\{c_{t1}, c_{t2}, c_{t3}, c_{t4}\}$

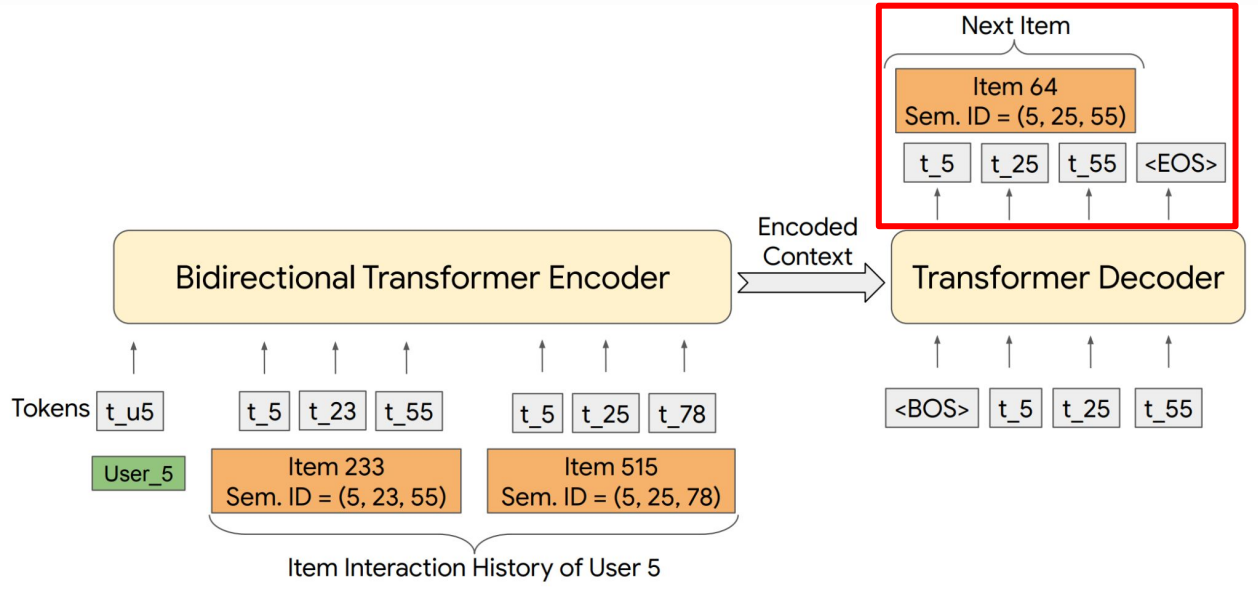
SemID-based Recommender Architecture

Architecture: Decoder-Only / Encoder-Decoder



SemID-based Recommender Architecture

Objective: Next-Token Prediction



SemID-based Recommender Architecture

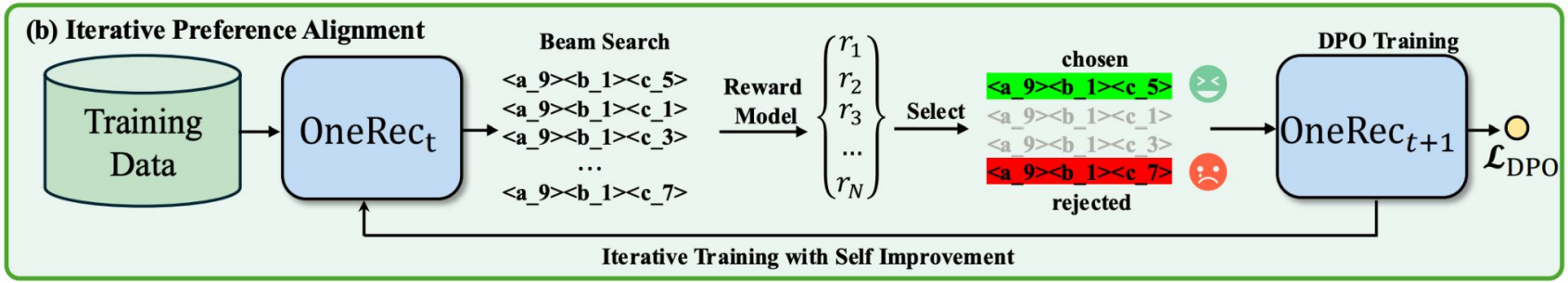
Objective: Next-Token Prediction

Could we add negative samples like BPR loss?

SemID-based Recommender Architecture

Objective: Preference Alignment Objective

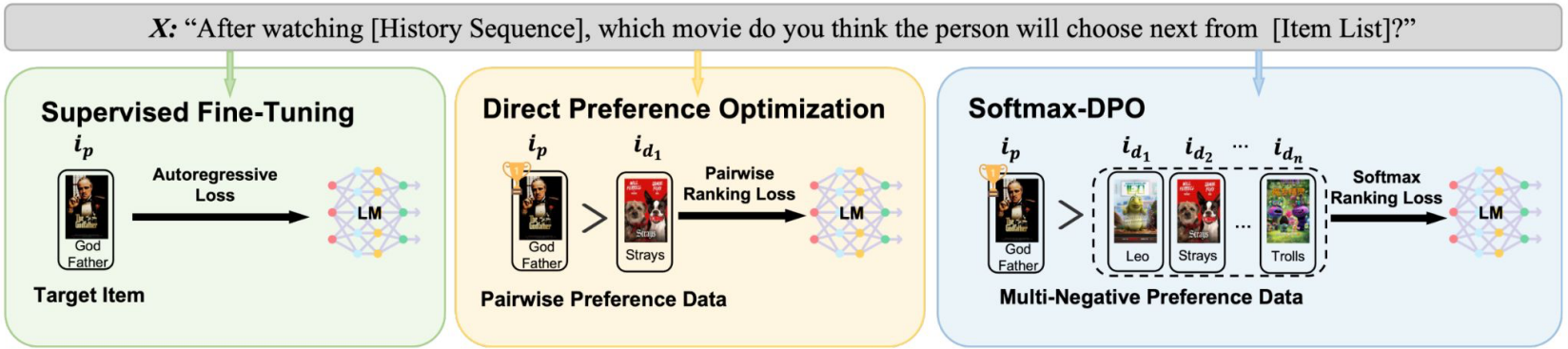
One negative sample per instance



SemID-based Recommender Architecture

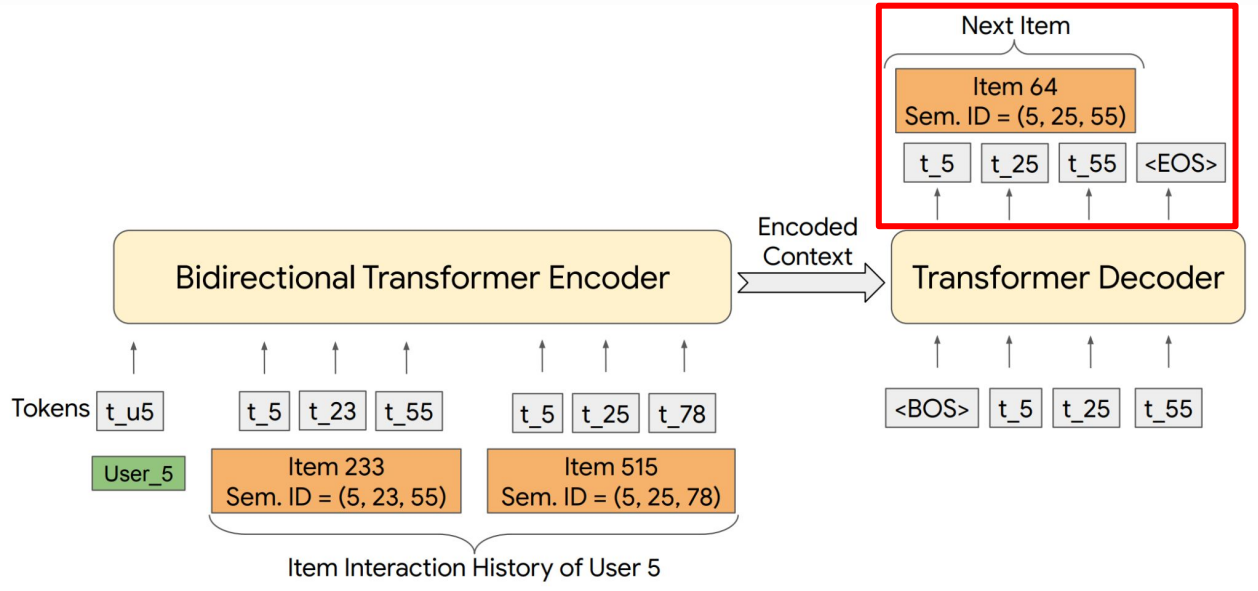
Objective: Preference Alignment Objective

Multiple negative samples per instance



SemID-based Recommender Architecture

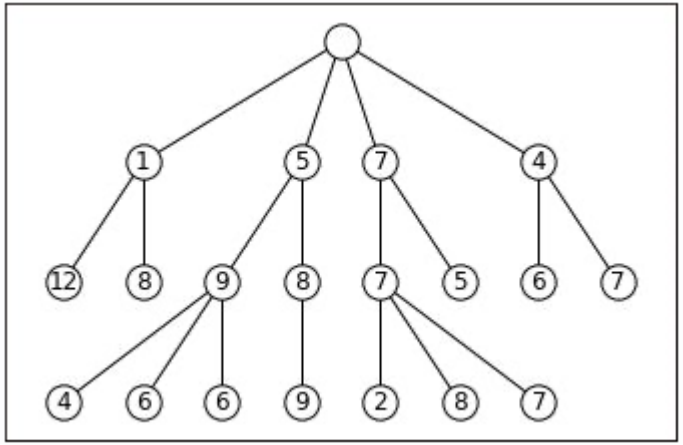
Inference: How to get a ranking list?



SemID-based Recommender Architecture

Inference: How to get a ranking list?

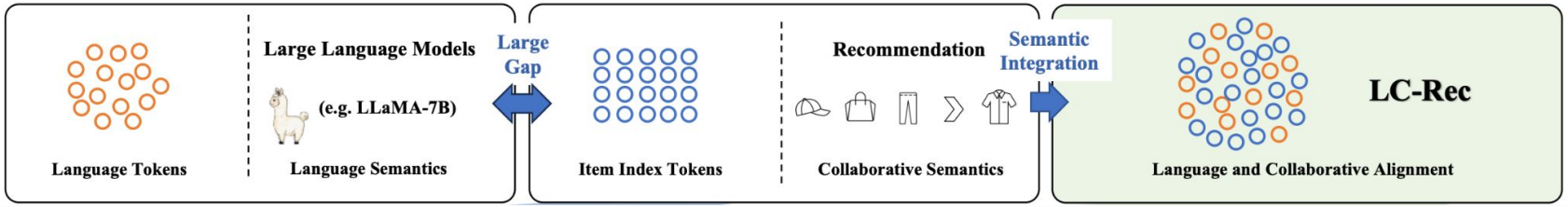
(Constrained) Beam Search



https://en.wikipedia.org/wiki/Beam_search

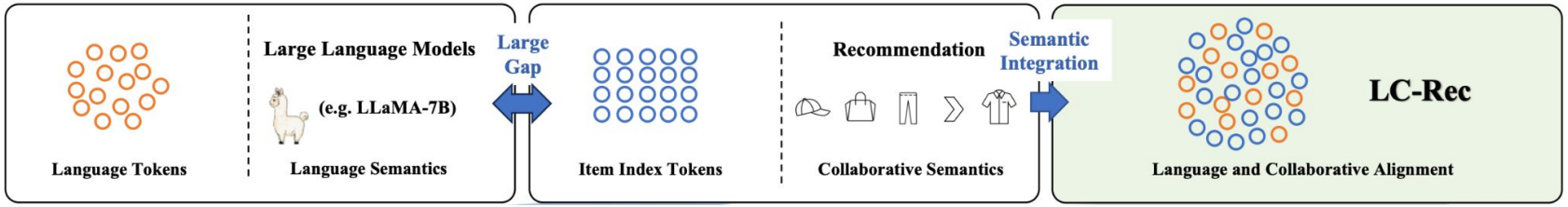
SemID-based Recommender Architecture

Align with LLMS – LC-Rec



SemID-based Recommender Architecture

Align with LLMS – LC-Rec



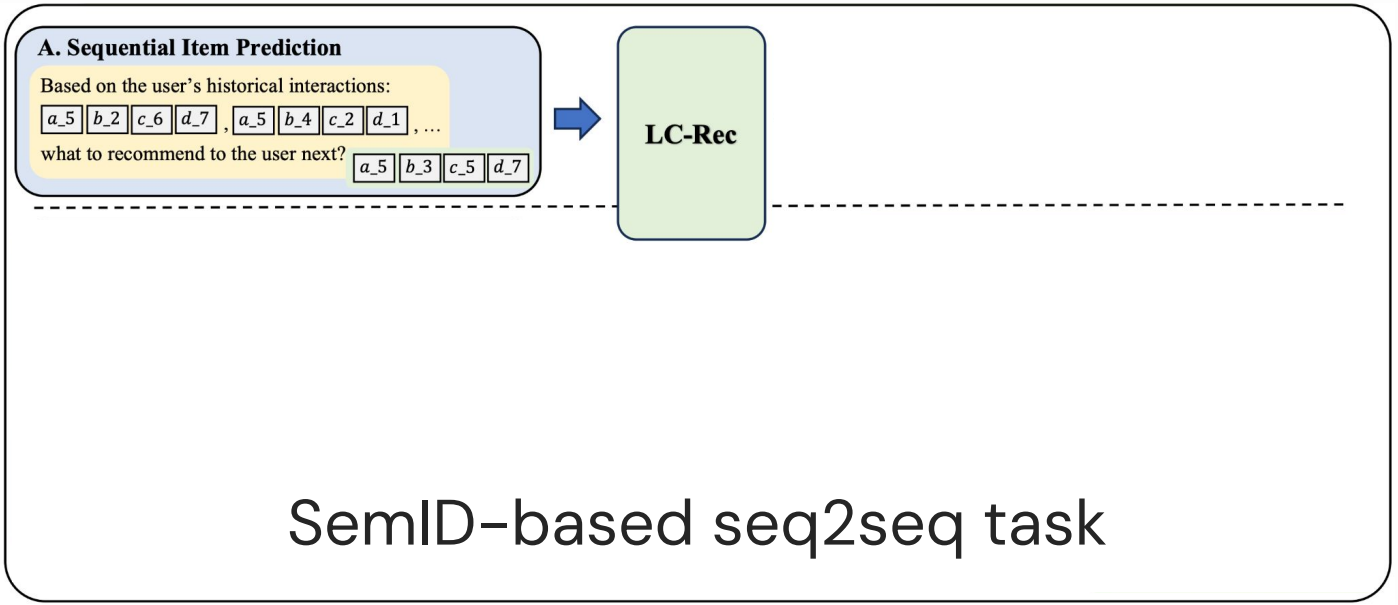
Core Idea:

Construct instructions containing both Semantic IDs and language tokens

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

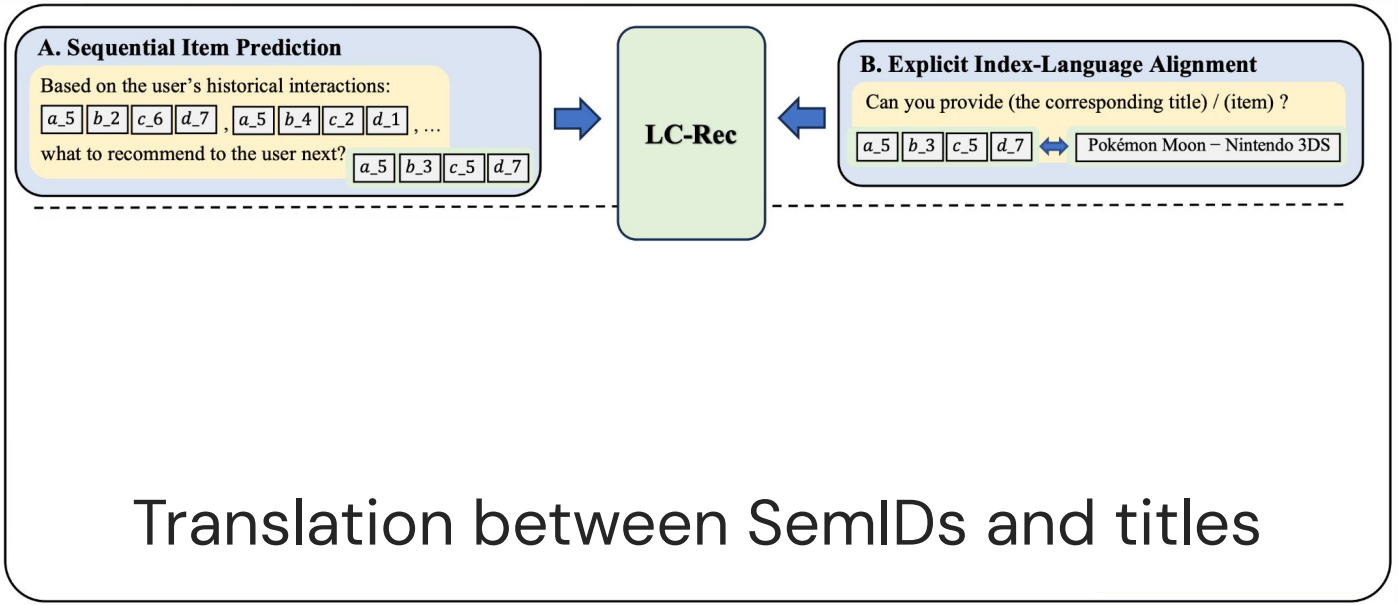
SemID-based Recommender Architecture

Align with LLMS – LC-Rec



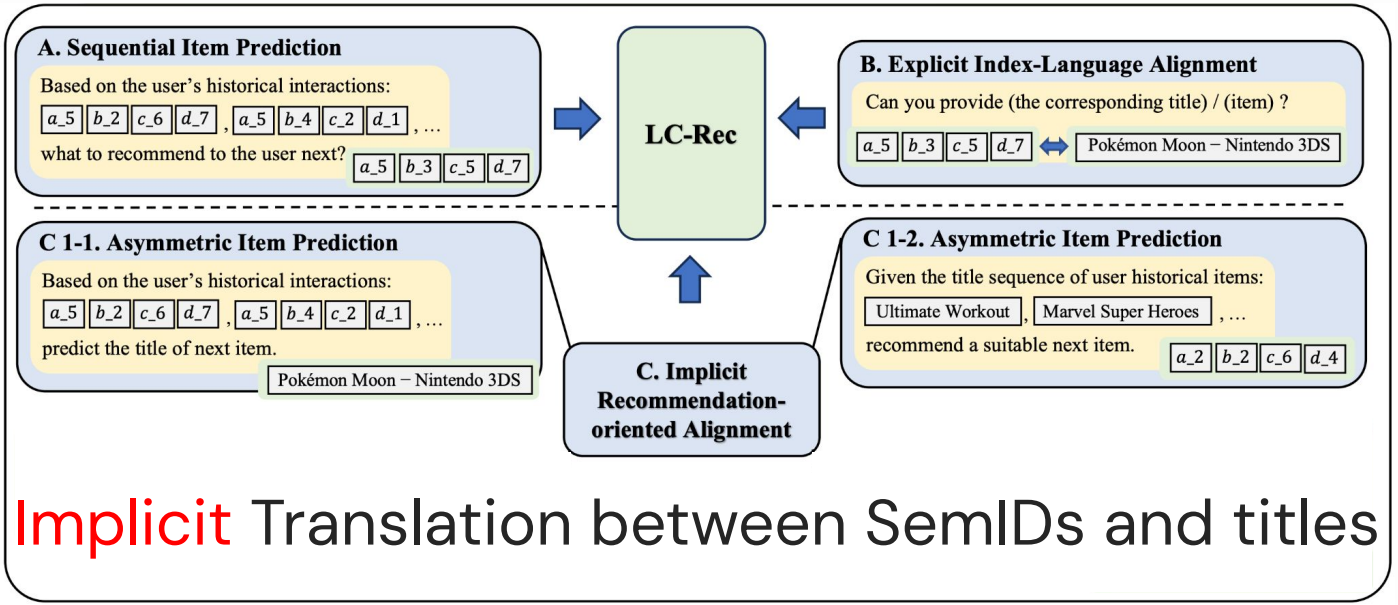
SemID-based Recommender Architecture

Align with LLMS – LC-Rec



SemID-based Recommender Architecture

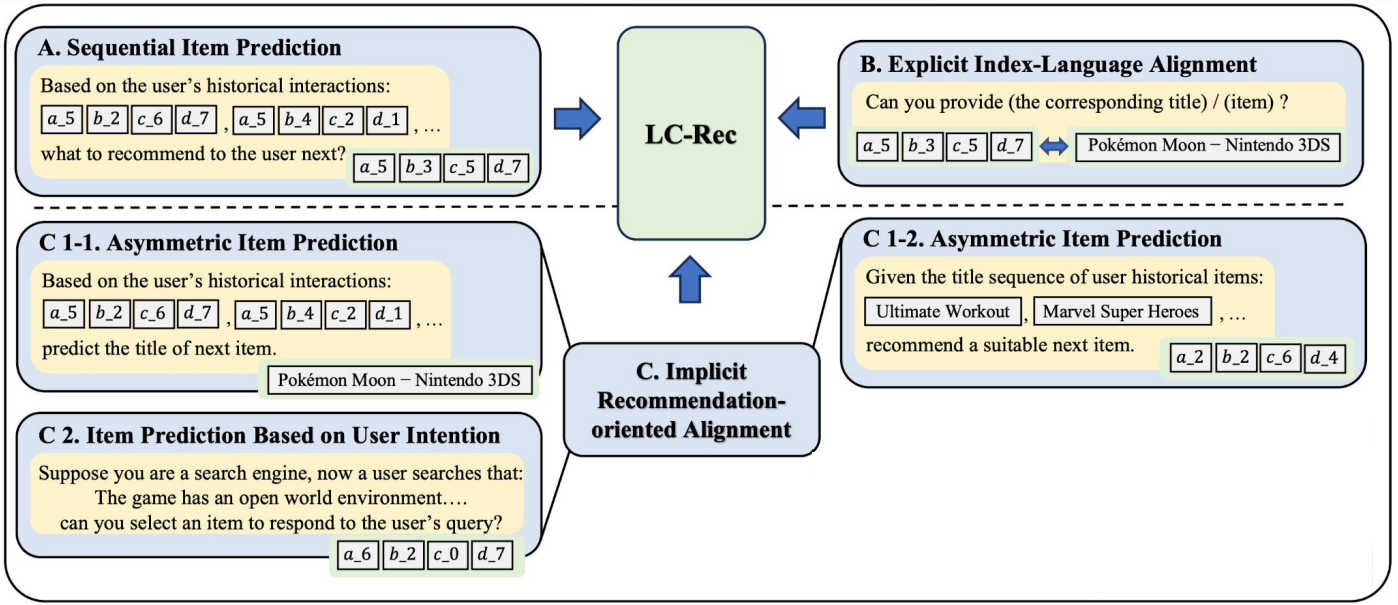
Align with LLMS – LC-Rec



Implicit Translation between SemIDs and titles

SemID-based Recommender Architecture

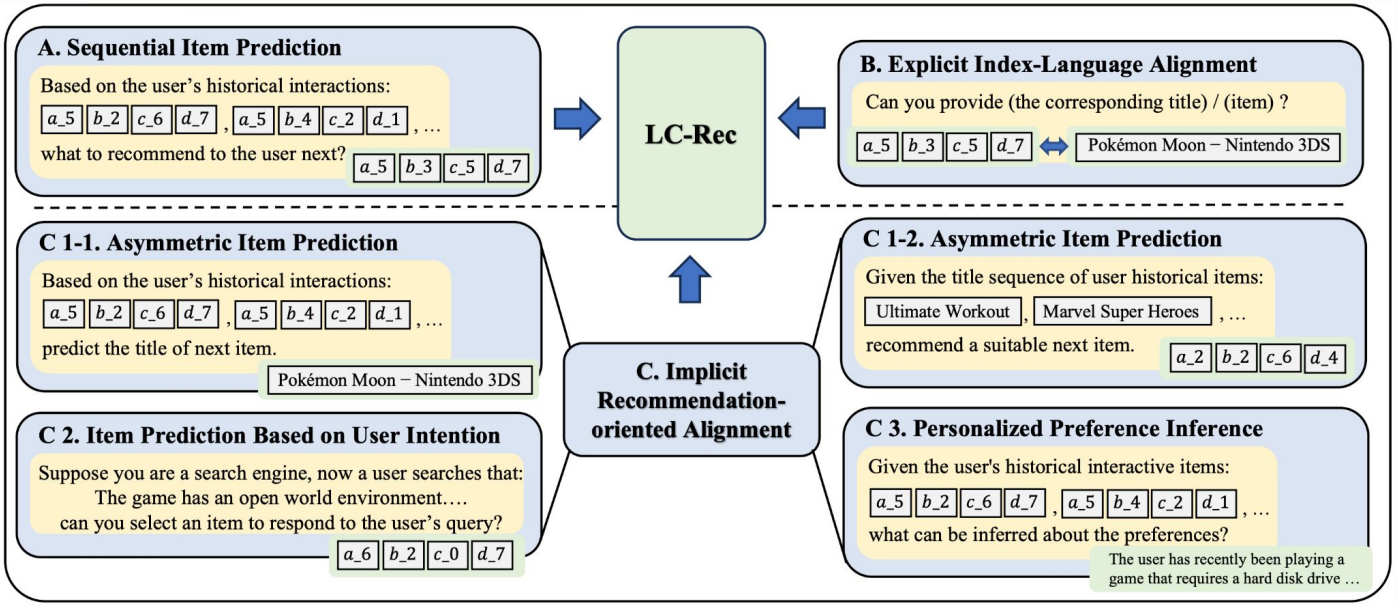
Align with LLMS – LC-Rec



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

SemID-based Recommender Architecture

Align with LLMS – LC-Rec



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

Part 2 Summary – Architecture

(1) Train from Scratch

(2) Align with LLMs

Part 2 Summary – Architecture

(1) Train from Scratch

Objective (NTP, DPO, S-DPO)

Inference (Beam Search)

(2) Align with LLMs

Part 2 Summary – Architecture

(1) Train from Scratch

Objective (NTP, DPO, S-DPO)

Inference (Beam Search)

(2) Align with LLMs

LC-Rec: Instructions containing both semIDs and language tokens