

# Pre-training Generative Recommender with Multi-Identifier Item Tokenization

Bowen Zheng<sup>\*</sup>  
Gaoling School of Artificial  
Intelligence, Renmin  
University of China  
Beijing, China  
bwzheng0324@ruc.edu.cn

Enze Liu<sup>\*</sup>  
Gaoling School of Artificial  
Intelligence, Renmin  
University of China  
Beijing, China  
enzeliu@ruc.edu.cn

Zhongfu Chen  
Poisson Lab,  
Huawei  
Beijing, China  
chenzhongfu3@huawei.com

Zhongrui Ma  
Poisson Lab,  
Huawei  
Beijing, China  
zhongrui.ma@huawei.com

Yue Wang  
Poisson Lab,  
Huawei  
Beijing, China  
wangyue262@huawei.com

Wayne Xin Zhao<sup>✉</sup>  
Gaoling School of Artificial  
Intelligence, Renmin  
University of China  
Beijing, China  
batmanfly@gmail.com

Ji-Rong Wen  
Gaoling School of Artificial  
Intelligence, Renmin  
University of China  
Beijing, China  
jrwen@ruc.edu.cn

## Abstract

Generative recommendation has emerged as a promising paradigm that recommends the potential item by autoregressively generating its identifier. Most existing methods adopt a strict *one-to-one mapping* strategy, where each item is represented by a single identifier. However, this rigid tokenization scheme poses issues, such as sub-optimal semantic modeling for low-frequency items and limited diversity in token sequence data.

To overcome these limitations, we propose **MTGRec**, which leverages Multi-identifier item Tokenization to augment token sequence data for Generative Recommender pre-training. Our approach is built upon two core innovations: *multi-identifier item tokenization* and *curriculum recommender pre-training*. For multi-identifier item tokenization, we leverage the Residual-Quantized Variational AutoEncoder (RQ-VAE) as the tokenizer backbone and treat model checkpoints from adjacent training epochs as semantically relevant tokenizers. This allows each item to be associated with multiple identifiers, enabling a single user interaction sequence to be converted into several token sequences as different data groups. For curriculum recommender pre-training, we introduce a curriculum learning scheme guided by data influence estimation. Specifically, we estimate the influence of each tokenizer's data using first-order gradient approximation and dynamically adjust the sampling probability of each data group during recommender pre-training. After pre-training, we fine-tune the model using a single tokenizer

to ensure accurate item identification for recommendation. Extensive experiments on three public benchmark datasets demonstrate that MTGRec significantly outperforms both traditional and generative recommendation baselines in terms of effectiveness and scalability. Our code is available at <https://github.com/RUCAIBox/MTGRec>.

## CCS Concepts

• Information systems → Recommender systems.

## Keywords

Generative Recommendation, Item Tokenization

## ACM Reference Format:

Bowen Zheng<sup>\*</sup>, Enze Liu<sup>\*</sup>, Zhongfu Chen, Zhongrui Ma, Yue Wang, Wayne Xin Zhao<sup>✉</sup>, and Ji-Rong Wen. 2018. Pre-training Generative Recommender with Multi-Identifier Item Tokenization. In *Proceedings of Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '25)*. ACM, New York, NY, USA, 12 pages. <https://doi.org/XXXXXXX.XXXXXXX>

## 1 Introduction

Nowadays, sequential recommender systems [10, 33] have been widely used in various online platforms, aiming to capture users' personalized preferences based on their historical interaction behaviors. Traditional sequential recommendation approaches [10, 16, 36, 39] assign a unique ID to each item and predict the next item by measuring the similarity between the user preference and candidate items through approximate nearest neighbor (ANN) algorithms. Most recently, driven by the promising potential of large language models (LLMs) [57] and generative retrieval methods [37, 40, 43, 51], several studies proposed applying the generative paradigm as an alternative to ANN in recommender systems [15, 23, 27, 32, 58]. The core idea of generative recommendation lies in employing a list of tokens (*i.e.*, a *token sequence*) as the identifier for item representation instead of a single vanilla ID. Thus, the next-item prediction is reformulated as a sequence-to-sequence problem, aiming to autoregressively generate the identifier of the target item.

<sup>\*</sup> Equal contribution.

✉ Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).  
SIGIR '25, July 13–18, 2025, Padua, Italy

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.  
ACM ISBN 978-1-4503-XXXX-X/18/06  
<https://doi.org/XXXXXXX.XXXXXXX>

A typical generative recommendation framework consists of two key components, namely the *item tokenizer* and the *generative recommender*. The item tokenizer is crafted to associate each item with a list of tokens that implies semantic knowledge. The merit is that shared tokens between items reflect the underlying semantic similarities. Existing methods are developed by utilizing a variety of techniques such as co-occurrence matrix decomposition [15, 27], hierarchical clustering [34, 45], and multi-level codebook [29, 32, 42]. Among these, the Residual-Quantized Variational AutoEncoder (RQ-VAE)[53] is the most commonly used item tokenizer. And recent researches attempt to further improve the quality of item identifiers by incorporating collaborative signals [22, 42] or multi-behavior information [23]. The generative recommender is leveraged to autoregressively generate the target token sequence, usually employing either decoder-only (e.g., GPT [1, 30]) or encoder-decoder (e.g., T5 [31]) architectures. Furthermore, some studies focus on enhancing the generative recommender by using dual decoders [45] or incorporating contrastive learning [34].

Despite remarkable progress, previous methods typically represent each item by a single identifier, employing a strict *one-to-one mapping* for item tokenization, which leads to the following two potential issues. First, token sequence data inherits the long-tail distribution and data sparsity issues of interaction data [46, 59]. As a result, tokens associated with long-tail items are low-frequency and lack supervision signals, making it challenging to effectively learn their semantics. Second, the one-to-one mapping restricts the diversity of sequence data. Compared to all possible token permutations, mapping observed item sequences to token sequences in a one-to-one manner results in a lack of variation. Moreover, these limitations hinder the potential for performance improvement through model scaling, as observed in LLMs [4, 17, 57].

In light of these issues, our idea is to associate one item with multiple identifiers, which is achieved by incorporating multiple item tokenizers with semantic relevance. The advantages of this multi-identifier scheme are two-fold. First, associating each item with more tokens increases the exposure frequency of tokens and promotes token sharing across items, thereby facilitating the effective learning of token semantics. Second, an item interaction sequence can be tokenized into several token sequences, thereby enriching the diversity of the training data. In addition, the increased volume and diversity of token sequence data enable us to achieve performance improvement through model scaling. To develop our approach, we focus on two key challenges: (i) learning multiple semantically relevant rather than extraneous item tokenizers; and (ii) effectively training a generative recommender based on the proposed multi-identifier item tokenization.

To this end, we propose a novel framework namely **MTGRec**, which incorporates Multiple item Tokenizers to improve the effectiveness and scalability of the Generative Recommender. Overall, our approach associates each item with multiple identifiers and augments one item sequence into several token sequences, which serve as training data for generative recommender pre-training. Specifically, we focus on two key aspects, namely *multi-identifier item tokenization* and *curriculum recommender pre-training*. For multi-identifier item tokenization, we adopt the learnable RQ-VAE as the backbone and take the model checkpoints corresponding to adjacent epochs as semantically relevant item tokenizers. In this

way, we can tokenize each item to multiple identifiers and construct several groups of token sequence data. Each group is characterized by related but distinct distributions derived from different item tokenizers. For curriculum recommender pre-training, we design a data curriculum scheme to adaptively schedule the proportions of these data groups during model pre-training. As the specific technique, we measure the influence of data from each item tokenizer using first-order gradient approximation and adjust the sampling probability of each data group accordingly. Finally, we fine-tune the pre-trained model using a single item identifier to ensure precise item identification during recommendation.

In summary, our primary contributions are as follows:

- We propose a novel framework **MTGRec** that learns multiple item tokenizers for curriculum recommender pre-training to improve generative recommendation.
- We develop a multi-identifier item tokenization approach for token sequence augmentation and introduce a data curriculum scheme based on data influence estimation to enhance recommender training.
- We conduct extensive experiments on three public datasets, demonstrating the superiority of our proposed framework over both traditional and generative recommendation baselines in terms of effectiveness and scalability.

## 2 Methodology

In this section, we present our proposed generative recommender **MTGRec**, which uses multi-identifier item tokenization to augment token sequences for generative recommender pre-training.

### 2.1 Overview

**2.1.1 Problem Formulation.** Given the item set  $\mathcal{V}$ , let  $S = [v_1, \dots, v_t]$  denote the user's historical interacted items in chronological order. Sequential recommendation aims to capture user preferences implicit in the item sequence and predict the next potential item  $v_{t+1}$ . Generative recommendation reformulates the traditional sequential recommendation task as a sequence-to-sequence problem. In this paradigm, an item tokenizer  $T$  is learned to represent each item by a token sequence as its identifier. Formally, we refer to the above process as item tokenization, denoted as  $[c_1, \dots, c_H] = T(v)$ , where  $c_h$  represents the  $h$ -th token of  $v$  and  $H$  is the length of the identifier. Then, the interacted item sequence  $S$  and the target item  $v_{t+1}$  are tokenized into token sequences  $X = T(S) = [c_1^1, c_2^1, \dots, c_{H-1}^1, c_H^1]$  and  $Y = T(v_{t+1}) = [c_1^{t+1}, \dots, c_H^{t+1}]$  respectively, where each item is represented by  $H$  tokens. Finally, the next-item prediction is achieved by autoregressively generating the identifier of the target item (i.e.,  $Y$ ). Formally, this task can be written as:

$$P(Y|X) = \prod_{h=1}^H P(c_h^{t+1}|X, c_1^{t+1}, \dots, c_{h-1}^{t+1}). \quad (1)$$

**2.1.2 Method Overview.** Different from previous works [32, 42] that establish a *one-to-one mapping* between each item and its identifier, our idea is to associate one item with multiple identifiers to construct more massive and diverse token sequence data for recommender pre-training. To this end, we make efforts in the following two aspects:

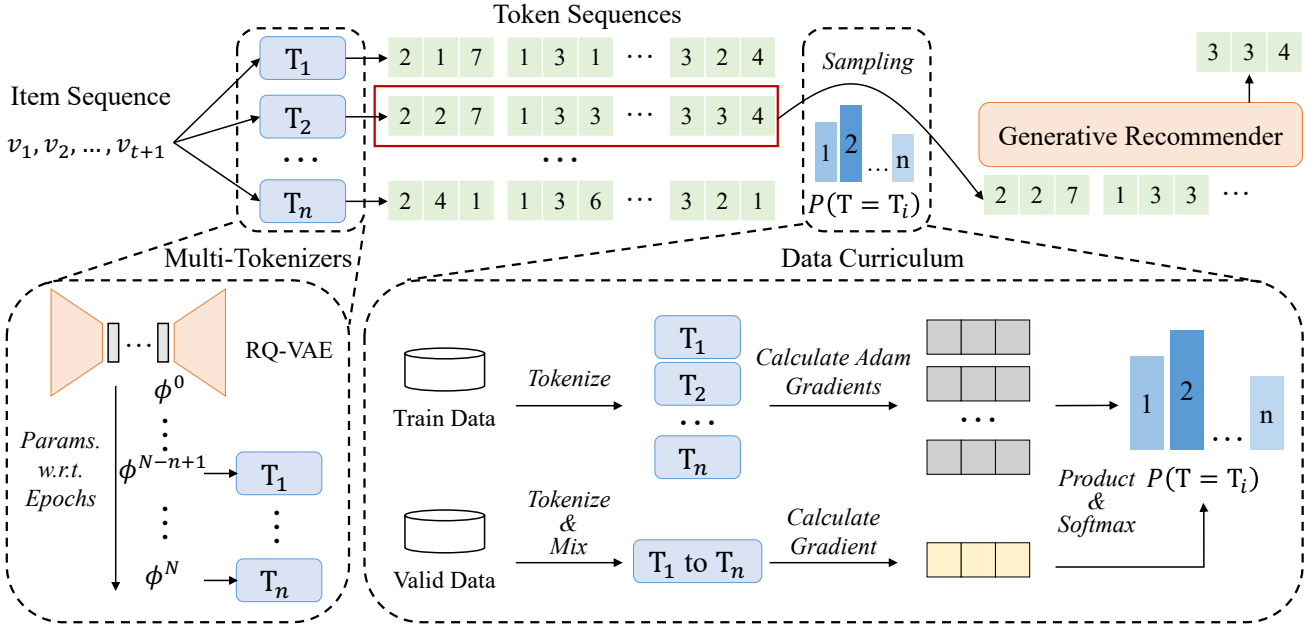


Figure 1: The overall framework of MTGRec with two key techniques. (i) We utilize RQ-VAE checkpoints of adjacent epochs as semantically relevant item tokenizers and tokenize an item sequence into multiple token sequences. (ii) We propose a data curriculum scheme based on data influence estimation, which is implemented through first-order gradient approximation.

• **Multi-Identifier Item Tokenization** (Section 2.2): We employ the learnable RQ-VAE [53] as the backbone of the item tokenizer. To obtain item tokenizers with semantic relevance, we select multiple RQ-VAE checkpoints corresponding to adjacent epochs during the training process. Applying these tokenizers, each item is associated with multiple identifiers, allowing a single item sequence to be tokenized into several token sequences. These token sequences, generated by semantically relevant tokenizers, encapsulate related yet distinct semantic knowledge.

• **Curriculum Recommender Pre-training** (Section 2.3): Given the hybrid data derived from different item tokenizers, we propose a curriculum learning method to adaptively schedule the proportions of these data groups during model pre-training. Specifically, we design a data curriculum scheme that increases the proportion of useful data while decreasing the proportion of low-quality data. To achieve this, we utilize the first-order gradient approximation to estimate the data influence, measuring whether it is “useful”, and dynamically adjust the sampling probabilities of different data groups accordingly. Finally, we fine-tune the pre-trained model based on a single item identifier to ensure precise item identification.

The overall framework of the proposed approach is shown in Figure 1. Next, we will present the details of our method.

## 2.2 Multi-Identifier Item Tokenization

As described above, we propose a multi-identifier scheme to tokenize one item sequence to several token sequences: (i) employ the learnable RQ-VAE as the backbone of item tokenizer (Section 2.2.1), (ii) select semantically relevant tokenizers from adjacent epochs

(Section 2.2.2), and (iii) tokenized the item sequence by multiple item tokenizers (Section 2.2.3).

**2.2.1 Tokenizer Backbone.** In practice, we implement the item tokenizer as a RQ-VAE [53], which is advantageous due to its efficacy in modeling item semantics and mitigating length bias [34, 42]. Initially, RQ-VAE takes the item semantic embedding  $z$  (e.g., text embedding encoded by a pre-trained language model) as input and encodes it into a latent representation  $r$ . Then,  $r$  is quantized into serialized codes (called tokens) from coarse to fine through  $H$ -level residual quantization. Each level’s codebook is denoted by  $C^h = \{e_k^h\}_{k=1}^K$ , where  $e_k^h$  is a learnable cluster center and  $K$  is the codebook size. Finally, the residual quantization is applied to  $r$ :

$$c_h = \arg \min_k \|r_h - e_k^h\|_2^2, \quad (2)$$

$$r_{h+1} = r_h - e_{c_h}^h, \quad (3)$$

where  $r_h$  is the residual vector in the  $h$ -th RQ level, and  $r_1 = r$ . Thereafter, we obtain the item quantized representation  $\tilde{r} = \sum_{h=1}^H e_{c_h}^h$  and feed it into a decoder to reconstruct the item semantic embedding. Overall, the loss of the RQ-VAE is  $\mathcal{L}_T = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{rq}}$ , where  $\mathcal{L}_{\text{recon}} = \|z - \hat{z}\|_2^2$ , and  $\mathcal{L}_{\text{rq}} = \sum_{h=1}^H \|\text{sg}[r_h] - e_{c_h}^h\|_2^2 + \beta \|r_h - \text{sg}[e_{c_h}^h]\|_2^2$ .  $\hat{z}$  is the reconstructed item embedding, and  $\text{sg}[\cdot]$  denotes the stop-gradient operation.  $\beta$  is used to balance the optimization between the encoder and codebooks, typically set to 0.25.

**2.2.2 Semantically Relevant Tokenizers.** To obtain multiple item tokenizers that associate each item with multiple identifiers, a naive method involves training multiple RQ-VAE models, each initialized with different random parameters. However, the models learned

in this operate independently, resulting in the token sequences tokenized by them being extraneous. Thus, there is no related and homogeneous knowledge among the multiple groups of token sequence data constructed by these item tokenizers, which may even lead to serious semantic conflicts.

In contrast, we propose regarding the model checkpoints corresponding to adjacent epochs within a training process as multiple semantically relevant item tokenizers. These checkpoints are derived from iterative gradient descent from the identical initialization parameters, ensuring that disparities between codebooks in adjacent epochs remain minimal. The token sequences generated by these item tokenizers encapsulate related yet distinct semantic knowledge. Formally, the learned multiple semantically relevant item tokenizers are written as:

$$\mathcal{T} = \{T_1, T_2, \dots, T_n\} \quad (4)$$

$$= \{T_{\phi^{N-n+1}}, T_{\phi^{N-n+2}}, \dots, T_{\phi^N}\}, \quad (5)$$

where  $\mathcal{T}$  denotes a set of item tokenizers, and  $n$  is the number of tokenizers.  $\phi^i$  represents the RQ-VAE parameter corresponding to the  $i$ -th epoch.  $N$  indicates the maximum number of epochs.

**2.2.3 Tokenize an Item Sequence to Multiple Token Sequences.** With the learned semantically relevant item tokenizers, a historical item sequence  $S$  and a target item  $v_{t+1}$  can be tokenized into multiple token sequences via different tokenizers:

$$X_1, X_2, \dots, X_n = T_1(S), T_2(S), \dots, T_n(S), \quad (6)$$

$$Y_1, Y_2, \dots, Y_n = T_1(v_{t+1}), T_2(v_{t+1}), \dots, T_n(v_{t+1}), \quad (7)$$

where  $X_i$  and  $Y_i$  represent the token sequence and target item identifier tokenized by  $T_i$ . Notably, we do not directly utilize all augmented token sequences for model pre-training. The reason is that when  $n$  is large, the resulting data volume becomes unmanageable, and it is infeasible to adaptively adjust the proportions of different data groups. Instead, we sample just one token sequence at a time for model optimization, which is approximately equivalent to using all the data through multiple sampling. In the following section, we detail the method for adjusting the sampling probabilities of different data groups corresponding to the item tokenizers.

## 2.3 Curriculum Recommender Pre-training

Based on multi-identifier item tokenization, we obtain a data mixture involving multiple groups of token sequences, from which we select instances for generative recommender pre-training. This presents a key challenge similar to that in LLM pre-training, namely how to adaptively adjust the proportions of different data groups during pre-training [57]. Inspired by the data curriculum proposed and widely employed in LLM pre-training [2, 49], we devise a curriculum pre-training scheme based on data influence estimation in MTGRec. Particularly, we estimate data influences corresponding to multiple item tokenizers via first-order gradient approximation (Section 2.3.1). Then, we dynamically adjust the sampling probabilities of different data groups in accordance with their estimated data influences for recommender pre-training (Section 2.3.2).

**2.3.1 Estimating Data Influence.** In order to more effectively utilize data from multiple item tokenizers, our idea is to increase the

proportion of useful data while decreasing the proportion of low-quality data. To measure whether the data is “useful” in a rational manner, we define the contribution of training data to the validation loss as data influence [7, 28, 48], and estimate it based on gradient information. Formally, using the first-order Taylor expansion, the validation loss can be expressed as follows:

$$\mathcal{L}(\mathcal{D}_{val}; \theta^{t+1}) = \mathcal{L}(\mathcal{D}_{val}; \theta^t) + \nabla \mathcal{L}(\mathcal{D}_{val}; \theta^t) \cdot (\theta^{t+1} - \theta^t), \quad (8)$$

where  $\mathcal{D}_{val}$  denotes the held-out data for validation, and  $\theta^t$  is the recommender parameter at time step  $t$ . The first term of the equation represents the validation loss at time step  $t$ , while the second term is the first-order derivative within the Taylor expansion. Then the update of validation loss is:

$$\mathcal{L}(\mathcal{D}_{val}; \theta^{t+1}) - \mathcal{L}(\mathcal{D}_{val}; \theta^t) = \nabla \mathcal{L}(\mathcal{D}_{val}; \theta^t) \cdot (\theta^{t+1} - \theta^t). \quad (9)$$

**Calculate Gradient of Validation Data.** Specific to the sequential recommendation scenario discussed in this paper, the validation data is acquired with the leave-one-out strategy. After item tokenization using different tokenizers, multiple groups of token sequence data are mixed into  $\mathcal{D}_{val}$ . The terms  $\mathcal{L}(\mathcal{D}_{val}; \theta)$  and  $\nabla \mathcal{L}(\mathcal{D}_{val}; \theta)$  denote the mean loss and cumulative gradient across all validation data, respectively, which can be formalized as:

$$\mathcal{L}(\mathcal{D}_{val}; \theta) = \frac{1}{|\mathcal{D}_{val}|} \sum_{X, Y \in \mathcal{D}_{val}} \mathcal{L}(X, Y; \theta), \quad (10)$$

$$\nabla \mathcal{L}(\mathcal{D}_{val}; \theta) = \frac{1}{|\mathcal{D}_{val}|} \sum_{X, Y \in \mathcal{D}_{val}} \nabla \mathcal{L}(X, Y; \theta), \quad (11)$$

where  $X, Y$  denotes a pair of token sequences corresponding to the historical interacted items and the target item.  $\mathcal{L}(\cdot, \cdot; \theta)$  is the negative log-likelihood loss in Eqn. (22).

**Calculate Adam Gradients of Training Data.** Since the generative recommender is usually trained using the Adam optimizer [18], the parameter update  $\theta^{t+1} - \theta^t$  in Eqn. (8) can be calculate as follows:

$$\theta^{t+1} - \theta^t = -\eta_t \Gamma(\mathcal{D}_{train}^i; \theta^t), \quad (12)$$

$$\Gamma(\mathcal{D}_{train}^i; \theta^t) = \frac{\mathbf{m}^{t+1}}{\sqrt{\mathbf{v}^{t+1} + \epsilon}}, \quad (13)$$

$$\mathbf{m}^{t+1} = (\beta_1 \mathbf{m}^t + (1 - \beta_1) \nabla \mathcal{L}(\mathcal{D}_{train}^i; \theta^t)) / (1 - \beta_1^t), \quad (14)$$

$$\mathbf{v}^{t+1} = (\beta_2 \mathbf{v}^t + (1 - \beta_2) \nabla \mathcal{L}(\mathcal{D}_{train}^i; \theta^t)^2) / (1 - \beta_2^t), \quad (15)$$

where  $\mathcal{D}_{train}^i$  denotes the training token sequence data tokenized by the item tokenizer  $T_i$ .  $\beta_1$  and  $\beta_2$  are the hyperparameters of the first-order and second-order momentum in Adam, which are usually set to 0.9 and 0.999 respectively.  $\eta_t$  is the learning rate at time step  $t$ . In our context, we do not estimate the influence of each individual data instance as in previous studies [7, 28, 48]. Instead, we consider a group of data from each item tokenizer as an entirety for analysis. The gradient of each group of data (i.e.,  $\nabla \mathcal{L}(\mathcal{D}_{train}^i; \theta)$ ) is calculated through gradient accumulation similar to Eqn. (11), which is equivalent to treating  $\mathcal{D}_{train}^i$  as a batch of data.

**Calculate Influence.** Based on the above analysis, we define the data influence of each item tokenizer at time step  $t$  as:

$$I(T_i; \theta^t) = \eta_t \nabla \mathcal{L}(\mathcal{D}_{val}; \theta^t) \cdot \Gamma(\mathcal{D}_{train}^i; \theta^t), \quad (16)$$

where  $I(T_i; \theta^t)$  denotes the influence of the data group associated with the tokenizer  $T_i$ . Finally, given that the training process spans multiple time steps, we calculate the cumulative influence based on multiple model checkpoints as  $\tilde{I}(T_i) = \sum_{k=1}^K I(T_i; \theta_k)$ , where  $\theta_k$  indicates the  $k$ -th checkpoint at time step  $t_k$  and  $K$  denotes the total number of checkpoints.

**2.3.2 Curriculum Pre-training.** After elaborating on how to estimate the influence of the data from each item tokenizer, we now formulate a data curriculum scheme for model pre-training by dynamically adjusting the sampling probabilities of different data groups. Specifically, we partition the training process into multiple stages, with each stage containing a specific number of epochs. At the end of each stage, we update data sampling probabilities based on the latest data influence of each item tokenizer, as determined by the current model checkpoint. Formally, given the current model checkpoint  $\theta_k$  and the cumulative data influence  $\tilde{I}_{k-1}(T_i)$  of the previous stage, the sampling probability is updated as follows:

$$\tilde{I}_k(T_i) = \tilde{I}_{k-1}(T_i) + I(T_i; \theta_k), \quad (17)$$

$$p_i^k = \frac{e^{\tilde{I}_k(T_i)/\tau}}{\sum_{j=1}^n e^{\tilde{I}_k(T_j)/\tau}}, \quad (18)$$

where  $\tau$  denotes the temperature coefficient used to control the smoothness of the distribution, and  $p_i^k$  is the sampling probability of the item tokenizer  $T_i$  for the subsequent stage. Initially, each data group is sampled with equal probability. Then, the data sampling strategy for stage  $k+1$  is defined as follows:

$$T \sim \mathcal{T} = \{T_1, T_2, \dots, T_n\}, \quad (19)$$

$$P(T = T_i) = p_i^k, \quad (20)$$

$$X = T(S), \quad Y = T(v_{t+1}). \quad (21)$$

Finally, the sampled token sequence data  $X$  and  $Y$  are subsequently fed into the generative recommender for model optimization with the negative log-likelihood loss:

$$\mathcal{L}(X, Y) = - \sum_{h=1}^H \log P(c_h^{t+1} | X, c_1^{t+1}, \dots, c_{h-1}^{t+1}). \quad (22)$$

## 2.4 Fine-tuning and Inference

**Fine-tuning for Item Identification.** In practical applications, the token sequence generated by the recommender should be able to identify the corresponding item. That is, the items and their identifiers should fulfill a one-to-one mapping within the recommender system. However, during our proposed curriculum pre-training with multiple item tokenizers, is unable to identify items since there might be multiple identifiers corresponding to the same item (i.e.,  $T_1(v), \dots, T_n(v) \mapsto v$ ). Therefore, we further fine-tune the pre-trained generative recommender based on each item tokenizer respectively and select the model with the optimal validation performance for actual deployment and testing.

**Inference.** Our objective during the inference phase is to generate the top  $K$  items from the entire item set for recommendation. To achieve this, we adopt beam search to decode  $K$  token sequences and map them to the corresponding items. Unlike some prior works [15, 42], we do not introduce a prefix tree to constrain the search process,

**Table 1: Statistics of the preprocessed datasets. Avg.len denotes the average length of item sequences.**

Dataset	#Users	#Items	#Interactions	Sparsity	Avg.len
Instrument	57,439	24,587	511,836	99.964%	8.91
Scientific	50,985	25,848	412,947	99.969%	8.10
Game	94,762	25,612	814,586	99.966%	8.60

as it would hinder parallel decoding and reduce efficiency. As for invalid identifiers, which occur only rarely [32], are simply ignored.

## 3 Experiments

In this section, we conduct empirical experiments and in-depth analyses on three public datasets to demonstrate the effectiveness of our proposed MTGRec.

### 3.1 Experiment Setup

**3.1.1 Dataset.** We evaluated the proposed approach on three subsets of the latest Amazon 2023 review dataset [12], i.e., “Musical Instruments”, “Industrial and Scientific” and “Video Games”. These datasets contain user review data spanning from May 1996 to September 2023. In line with the preprocessing steps outlined in previous studies [32, 59], we filter out low-activity users and items with less than five interaction records. Subsequently, we group the historical item sequences by users and sort them in chronological order, with a maximum sequence length limit of 20 items. The detailed statistics of preprocessed datasets are presented in Table 1.

**3.1.2 Baseline Models.** To facilitate a comprehensive comparison, we categorize the baseline models into the following two groups:

#### (1) Traditional sequential recommendation models:

- **Caser** [39] leverages convolutional neural networks to capture spatial and positional patterns in user behavior sequences.
- **HGN** [24] uses feature-level and instance-level gating mechanisms to model user preference.
- **GRU4Rec** [10] employs GRUs to capture sequential patterns in user interactions.
- **BERT4Rec** [36] utilizes a bi-directional self-attentive model with mask prediction objective for sequence modeling.
- **SASRec** [16] adopts a unidirectional self-attention network for user behavior modeling.
- **FMLP-Rec** [60] proposes an all-MLP model with learnable filters to reduce noise and model user preference.
- **HSTU** [54] incorporates user actions and timestamps into the next item prediction and proposes hierarchical sequential transducers with significant scalability. Note that it is still an ID-based method.
- **FDSA** [55] introduces a dual-stream self-attention framework that independently models item-level and feature-level sequence for recommendation.
- **S<sup>3</sup>-Rec** [59] enhances sequential recommendation models by leveraging feature-item correlations as self-supervised signals.

#### (2) Generative recommendation models:

- **TIGER** [32] employs RQ-VAE to quantize the item embedding into semantic IDs serving as the item identifier and adopts the generative retrieval paradigm for sequential recommendation.
- **LETTER** [42] extends TIGER by integrating collaborative and diversity regularization into RQ-VAE.
- **TIGER++** [32] employs representation whitening and exponential moving average (EMA) techniques to enhance codebook learning and improve the quality of semantic IDs. For implementation details, please refer to Section 3.1.4.

**3.1.3 Evaluation Settings.** We adopt top- $K$  Recall and Normalized Discounted Cumulative Gain (NDCG), with  $K$  set to 5 and 10, to evaluate the model performance in sequential recommendation. Following prior studies [16, 32, 59], we apply the *leave-one-out* strategy to split training, validation, and test sets. For each user interaction sequence, the final item he/she interacted is used as the test data, the second most recent item is used as the validation data, and all other items are used for training. For the sake of rigorous comparison, we conduct full ranking evaluation over the entire item set rather than sample-based evaluation. Additionally, the beam size of autoregressive decoding is set to 50 for all generative recommendation models.

**3.1.4 Implementation Details.** In this part, we introduce the implementation details of the item tokenizer and generative recommender in MTGRec respectively.

**Item Tokenizer.** In accordance with TIGER [32], we leverage Sentence-T5 [25] to encode the textual information associated with each item as its semantic embedding. Subsequently, we learn an RQ-VAE model with 3 codebooks of size 256 and an extra codebook for collision handling. In addition, the following three techniques are introduced to enhance codebook learning: (i) PCA with representation whitening [35] is applied to enhance the quality of item semantic embeddings. (ii) As previous studies [23, 58], a deeper MLP with hidden layer sizes of [2048, 1024, 512, 256] is utilized as the encoder and decoder in RQ-VAE. The codebook dimension is set to 128. (iii) Apply exponential moving averages (EMA) instead of gradient descent for codebook learning, which is more stable and effective [41]. We denote the method after applying these techniques as TIGER++ and use the same techniques to learn RQ-VAE in our MTGRec. The model is optimized by Adagrad optimizer over 10K epochs, employing a learning rate of 0.001 and a batch size of 2048. We select the RQ-VAE checkpoints of the final  $n$  epochs as the semantically relevant item tokenizers in our proposed approach, and  $n$  is tuned between 5 and 30 with an interval of 5.

**Generative Recommender.** We adopt T5 [31] as the backbone of the recommender, which has a model dimension of 128, an inner dimension of 512, 4 attention heads with a dimension of 64, and employs the ReLU activation function. We tune the number of model layers  $L$  within {1, 2, 3, 4, 5, 6, 7, 8}, and both the number of encoder layers and decoder layers are set to  $L$ . We set the batch size on each GPU to 256 and use 4 GPUs to pre-train the model for 200 epochs on all datasets. As for the number of epochs in each stage for curriculum pre-training, we first train 60 epochs for gradient feature warmup and then perform sampling probability update

every 20 epochs. Furthermore, the temperature coefficient  $\tau$  tuned in {0.1, 0.3, 1.0, 3.0, 5.0, 10.0}. The AdamW optimizer is used for both pre-training and fine-tuning, with the learning rates set to 0.005 and 0.0002 respectively. In addition, a cosine scheduler is utilized to adjust the learning rate.

We implement all traditional sequential recommendation models based on RecBole [56], which is a user-friendly open-source library for recommender systems. For a fair comparison, we set the embedding dimension of all models to 128 and obtain the best performance through hyperparameter grid search. For all generative baseline models, we use the same model architecture as our MTGRec and tune  $L$  between 1 to 8.

## 3.2 Overall Performance

We compare MTGRec with both traditional and generative baseline models on three public recommendation benchmarks. The overall results are presented in Table 2. From these results, we can find:

For traditional sequential recommendation models, FMLP-Rec and HSTU attain better results than SASRec by introducing more advanced model architectures.  $S^3$ -Rec integrates auxiliary features for self-supervised pre-training, achieving outstanding results on the Game dataset. Moreover, FDSA demonstrates superior performance compared to other models (*i.e.*, Caser, HGN, GRU4Rec, BERT4Rec, SASRec, FMLP-Rec, HSTU) that only involve item ID and collaborative information on three datasets. This observation implies that incorporating item textual features as supplementary information can significantly boost recommendation efficacy.

Regarding generative recommendation models, they generally outperform the traditional sequential recommendation models, benefiting from the item identifiers implying semantics and the generative paradigm. Among them, LETTER and TIGER++ show better performance than TIGER, which is attributed to their improvements on the item tokenizer. LETTER introduces collaborative and diversity regularization to integrate collaborative signals and alleviate code assignment bias for RQ-VAE. TIGER++ applies representation whitening and EMA techniques to improve the item tokenizer in terms of item embedding quality and model optimization.

Finally, our proposed MTGRec consistently maintains the optimal performance in all cases, achieving substantial improvements over both traditional and generative baseline models. Different from prior generative recommendation models, we introduce multiple semantically relevant item tokenizers for token sequence augmentation and design a model pre-training approach with data curriculum. By pre-training the generative recommender on larger and more diverse sequence data derived from multiple item tokenizers, we significantly improve the model’s scalability and effectiveness.

## 3.3 Ablation Study

To investigate the contributions of the various techniques included in our MTGRec, we conduct ablation studies on the Instrument and Scientific datasets and present the results in Table 3. Specifically, we compare MTGRec with the following three variants:

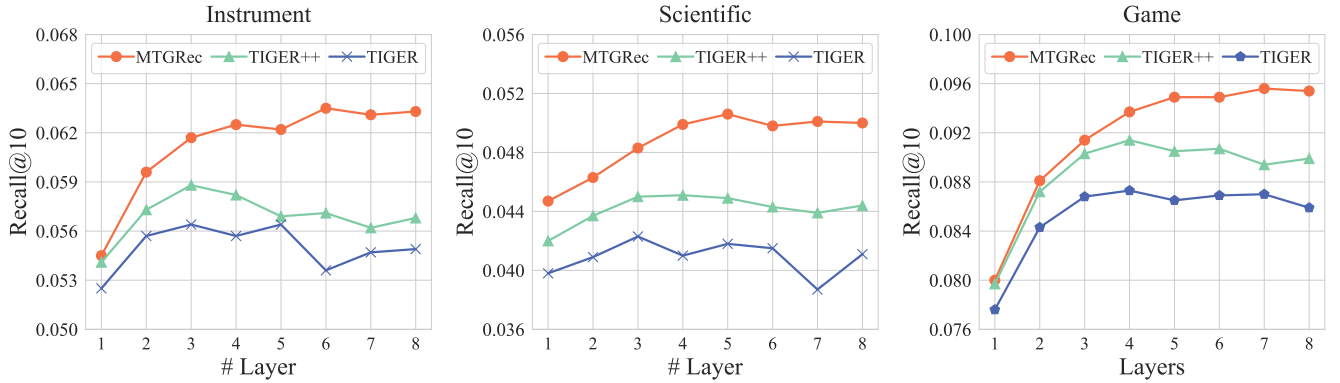
- (1) w/o Data curriculum without data curriculum based on influence estimation and samples data from different item tokenizers with equal probability. We can see that this variant performs worse than MTGRec across all datasets, which indicates that introducing

**Table 2: The overall performance comparisons between different baseline methods and MTGRec. The best and second-best results are highlighted in bold and underlined font, respectively.**

Methods	Instrument				Scientific				Game			
	Recall@5	Recall@10	NDCG@5	NDCG@10	Recall@5	Recall@10	NDCG@5	NDCG@10	Recall@5	Recall@10	NDCG@5	NDCG@10
Caser	0.0241	0.0386	0.0151	0.0197	0.0159	0.0257	0.0101	0.0132	0.0330	0.0553	0.0209	0.0281
HGN	0.0321	0.0517	0.0202	0.0265	0.0212	0.0351	0.0131	0.0176	0.0424	0.0687	0.0271	0.0356
GRU4Rec	0.0324	0.0501	0.0209	0.0266	0.0202	0.0338	0.0129	0.0173	0.0499	0.0799	0.0320	0.0416
BERT4Rec	0.0307	0.0485	0.0195	0.0252	0.0186	0.0296	0.0119	0.0155	0.0460	0.0735	0.0298	0.0386
SASRec	0.0333	0.0523	0.0213	0.0274	0.0259	0.0412	0.0150	0.0199	0.0535	0.0847	0.0331	0.0438
FMLP-Rec	0.0339	0.0536	0.0218	0.0282	0.0269	0.0422	0.0155	0.0204	0.0528	0.0857	0.0338	0.0444
HSTU	0.0343	0.0577	0.0191	0.0271	0.0271	0.0429	0.0147	0.0198	0.0578	0.0903	0.0334	0.0442
FDSA	0.0347	0.0545	0.0230	0.0293	0.0262	0.0421	0.0169	0.0213	0.0544	0.0852	0.0361	0.0448
S <sup>3</sup> -Rec	0.0317	0.0496	0.0199	0.0257	0.0263	0.0418	0.0171	0.0219	0.0485	0.0769	0.0315	0.0406
TIGER	0.0370	0.0564	0.0244	0.0306	0.0264	0.0422	0.0175	0.0226	0.0559	0.0868	0.0366	0.0467
LETTER	0.0372	0.0580	0.0246	0.0313	0.0279	0.0435	0.0182	0.0232	0.0563	0.0877	0.0372	0.0473
TIGER++	<u>0.0380</u>	<u>0.0588</u>	<u>0.0249</u>	<u>0.0316</u>	<u>0.0289</u>	<u>0.0450</u>	<u>0.0190</u>	<u>0.0241</u>	<u>0.0580</u>	<u>0.0914</u>	<u>0.0377</u>	<u>0.0485</u>
MTGRec	<b>0.0413</b>	<b>0.0635</b>	<b>0.0275</b>	<b>0.0346</b>	<b>0.0322</b>	<b>0.0506</b>	<b>0.0212</b>	<b>0.0271</b>	<b>0.0621</b>	<b>0.0956</b>	<b>0.0410</b>	<b>0.0517</b>
Improve	+8.68%	+7.99%	+10.44%	+9.49%	+11.42%	+12.44%	+11.58%	+12.45%	+7.07%	+4.60%	+8.75%	+6.60%

**Table 3: Ablation study of our method in the Instrument and Scientific datasets.**

Methods	Instrument				Scientific			
	Recall@5	Recall@10	NDCG@5	NDCG@10	Recall@5	Recall@10	NDCG@5	NDCG@10
(0) MTGRec	<b>0.0413</b>	<b>0.0635</b>	<b>0.0275</b>	<b>0.0346</b>	<b>0.0322</b>	<b>0.0506</b>	<b>0.0212</b>	<b>0.0271</b>
(1) w/o Data curriculum	0.0406	0.0618	0.0268	0.0338	0.0312	0.0487	0.0205	0.0263
(2) w/o Relevant tokenizers	0.0350	0.0548	0.0226	0.0290	0.0249	0.0404	0.0158	0.0208
(3) w/o Pre-training	0.0380	0.0571	0.0247	0.0309	0.0285	0.0443	0.0181	0.0236

**Figure 2: Performance Comparison w.r.t. Model Scale. The x-axis coordinates are the number of encoder and decoder layers in the generative recommender. All reported results for MTGRec are the best results obtained using various numbers of tokenizers.**

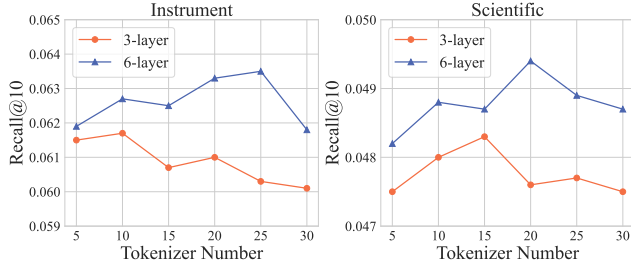
curriculum learning into generative recommender pre-training can effectively improve performance.

(2) *w/o Relevant tokenizers* learns multiple item tokenizers initialized with different random parameters for item tokenization, which are irrelevant and extraneous. The extraneous tokenizers cause serious semantic conflicts during pre-training, leading to the collapse of model learning and resulting in significant performance

degradation. This observation underscores the essence of selecting RQ-VAE checkpoints from adjacent epochs during a training process as semantically relevant item tokenizers.

(3) *w/o Pre-training* without pre-training on augmented sequence data derived from multiple semantically relevant item tokenizers, the generative recommender is learned based on a single item tokenizer (*i.e.*, TIGER++). The results show that pre-training based





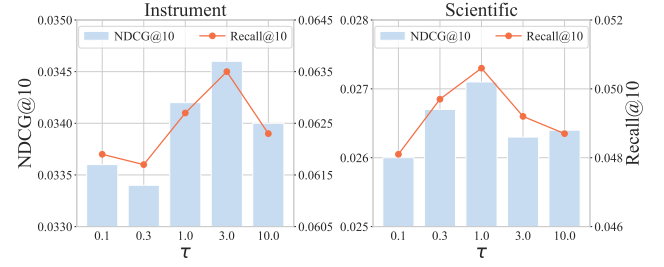
**Figure 3: Performance Comparison w.r.t. Tokenizer Number on the Instrument and Scientific datasets.**

on multiple item tokenizers serves as the critical element for the effectiveness of our framework.

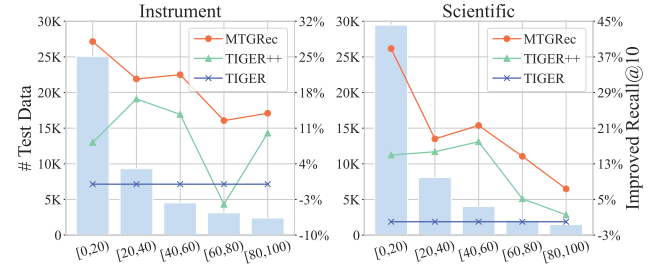
### 3.4 Further Analysis

**3.4.1 Performance Comparison w.r.t. Model Scale.** In our framework, sequence augmentation via multiple item tokenizers provides us with more massive and diverse data. The large-scale data motivates us to pursue enhanced performance via model scaling similar to that in LLMs. Therefore, in this section, we endeavor to explore the influence of different model scales on recommendation performance. Specifically, we start from a single layer and progressively increase the number of encoder and decoder layers of the generative recommender up to 8 layers. For models of distinct scales, we experiment with different numbers of tokenizers for pre-training to attain optimal performance and present the results in Figure 2. It is evident that MTGRec outperforms the baseline models (*i.e.*, TIGER, TIGER++) in all cases. Furthermore, the performance of baseline models is positively correlated with model scale only at shallow layers; as the model scale increases slightly (*e.g.*, with 4 or 5 layers), performance may degrade due to overfitting. In contrast, the performance of MTGRec generally shows an upward tendency as the model scales. However, we acknowledge that this positive correlation is constrained, unlike in LLMs, where there remains room for improvement even as the model scale reaches 100B [4, 17, 57]. This limitation may arise from the token sequence data being essentially constructed based on a finite set of observed user interactions. As larger models require more data for effective optimization, it becomes challenging to achieve a trade-off between the quality and quantity of augmented data through multiple item tokenizers. Specifically, the method fails to generate sufficient token sequences while maintaining the semantic relevance of data derived from RQ-VAE checkpoints separated by many training epochs. We leave addressing such issues to future work.

**3.4.2 Performance Comparison w.r.t. Tokenizer Number.** In addition to model scale, we further investigate how the number of item tokenizers employed for model pre-training affects recommendation performance. Specifically, we experiment with two scales of generative recommenders (*i.e.*, 3-layer and 6-layer models), pre-training them on datasets constructed with 5 to 30 item tokenizers. As shown in Figure 3, pre-training with fewer tokenizers offers only marginal advancement, which can be attributed to the insufficient diversity and volume of sequence data for deep model optimization.



**Figure 4: Performance Comparison w.r.t. Temperature Coefficient on the Instrument and Scientific datasets.**



**Figure 5: Performance Comparison w.r.t. Long-tail Items on the Instrument and Scientific datasets.. The bar graph illustrates the number of interactions in the test data for each group, while the line chart displays the improvement ratios for Recall@10 in comparison to TIGER.**

Furthermore, an excessively large number of item tokenizers will also lead to suboptimal performance. We hypothesize that when the intervals between item tokenizers span too many epochs, the semantic relevance between tokenizers weakens or even conflicts, thereby hindering effective model learning. Thus, it is crucial for MTGRec to select an appropriate number of item tokenizers to achieve a trade-off between data volume and semantic relevance. Besides, we observe that the optimal number of tokenizers increases with model scale (*i.e.*, for 6-layer models), suggesting that larger models benefit from more extensive and diverse sequence data for effective training.

**3.4.3 Performance Comparison w.r.t. Temperature Coefficient.** In the sampling probabilities of different data groups defined in Eqn. (18), the temperature coefficient  $\tau$  is used to regulate the smoothness of the distribution. To evaluate the impact of  $\tau$ , we vary its value from 0.1 to 10 and report the results in Figure 4. The results demonstrate that an appropriate  $\tau$  can significantly improve the performance of MTGRec. Specifically, the optimal values of  $\tau$  on the Instrument and Scientific datasets are 3 and 1, respectively. A smaller  $\tau$  leads the model to be more inclined towards high-probability item tokenizers while a larger  $\tau$  results in the data curriculum degenerating into uniform sampling. Both extremes adversely affect the effectiveness of curriculum pre-training.

**3.4.4 Performance Comparison w.r.t. Long-tail Items.** One of the key motivations for developing a pre-training approach based on



**Table 4: Performance comparison on other generative recommendation methods. Our MTGRec significantly improves the performance of all models.**

Methods	Instrument		Scientific	
	Recall@10	NDCG@10	Recall@10	NDCG@10
TIGER	0.0568	0.0307	0.0423	0.0225
+MTGRec	<b>0.0598</b>	<b>0.0329</b>	<b>0.0465</b>	<b>0.0245</b>
LETTER	0.0580	0.0313	0.0435	0.0232
+MTGRec	<b>0.0614</b>	<b>0.0335</b>	<b>0.0481</b>	<b>0.0255</b>
TIGER++	0.0588	0.0316	0.045	0.0241
+MTGRec	<b>0.0635</b>	<b>0.0346</b>	<b>0.0506</b>	<b>0.0271</b>

**Table 5: Item identifier difference w.r.t. different intervals.**

Intervals	Instrument		Scientific		Game	
	First	Any	First	Any	First	Any
1	0.39%	13.58%	0.27%	11.4%	0.36%	9.36%
5	0.44%	21.26%	0.58%	22.54%	0.58%	21.22%
10	0.51%	29.75%	0.51%	30.68%	0.57%	30.43%
20	0.75%	44.09%	0.71%	47.33%	0.79%	47.42%
30	0.87%	54.94%	0.85%	58.29%	1.14%	59.95%

multiple item tokenizers is to enhance the generalization of the generative recommender and prevent it from disregarding long-tail items. To verify the merit of our approach in recommendation involving long-tail items, we evaluate MTGRec on item groups with varying numbers of interactions. Specifically, following prior works [13], we split the test data into different groups according to the popularity of target items and present the Recall@10 improvement over TIGER in Figure 5. We can see that MTGRec consistently outperforms the baseline model across all item groups. Especially when target items are unpopular, *e.g.*, group [0, 20), MTGRec shows superior performance and more significant improvement than TIGER and TIGER++. This phenomenon indicates that long-tail items can benefit from pre-training with multiple item tokenizers, as this approach provides increased exposure and incorporates more knowledge from shared tokens.

**3.4.5 Applying MTGRec on Other Generative Recommendation Methods.** Furthermore, the proposed approach can be seamlessly integrated to other generative recommendation methods, such as the original TIGER and LETTER, with the only prerequisite being a trainable item tokenizer. To evaluate its general applicability, we applied MTGRec to additional generative recommendation methods on the Instrument and Scientific datasets. As shown in Table 4, the results demonstrate that the proposed method can consistently improve the performance of the base models, further verifying its effectiveness. This confirms that selecting model checkpoints from adjacent epochs as semantically relevant item tokenizers the generation of sequence data with homogeneous knowledge across multiple methods.

**Table 6: Efficiency of different Methods.**

Intervals	Instrument		Scientific		Game	
	Time	Epoch	Time	Epoch	Time	Epoch
TIGER	1.33 h	186	1.04 h	184	2.19 h	253
TIGER++	1.22 h	178	1.02 h	187	2.23 h	264
MTGRec	1.41 h	209	1.21 h	217	2.11 h	248

**3.4.6 Multiple Identifier Difference Analysis.** In this section, we analyze the relevance and differences of item identifiers generated by item tokenizers at different training epochs. Specifically, given two sets of item identifiers generated by two item tokenizers, we calculate two metrics: (1) the proportion of items whose first token changed, and (2) the proportion of items with any token changes in the item identifier. The results are shown in Table 5. We observe that for two adjacent tokenizers (*i.e.*, with an epoch interval of 1), the item identifiers on the three datasets exhibit minimal changes, indicating strong semantic consistency. As the epoch interval between tokenizers increases, a larger number of item identifiers change (*e.g.*, 59.95% in the Game dataset), leading to a higher risk of semantic conflict. It is worth noting that even when the interval is large and many identifiers differ, the proportion of changes in the first token typically remains below 1%, thereby preserving the core semantic information.

**3.4.7 Efficiency Analysis.** In this section, we further investigate the efficiency of the proposed method. As shown in Table 6, we measure the training time and the number of epochs required for model convergence across different methods under the same settings and the same hardware environment. Our method, MTGRec, consists of a pre-training phase of 200 epochs, followed by a low-cost fine-tuning phase. The results demonstrate that our multi-identifier pre-training strategy does not introduce excessive training time costs compared to baseline methods, while achieving significant performance gains. Furthermore, the proposed curriculum learning scheme accelerates model convergence on the Game dataset.

## 4 Related Work

In this section, we review related work from two aspects, namely, sequential recommendation and generative recommendation.

### 4.1 Sequential Recommendation

Sequential recommendation [10, 16] aims to capture user preferences based on the historical behavior sequence and predict the next item that the user is most likely to interact with. Early studies [9, 33] employ the Markov Chain to model item sequences and learn the transformation relationship between items. With the rapid advancements in deep learning, deep neural networks have emerged as a powerful tool for sequence modeling. Therefore, recent works propose various sequential recommenders based on neural networks, including convolutional neural network (CNN) [39], recurrent neural network (RNN) [10, 38], graph neural network (GNN) [47, 50], and Transformer [16, 36]. However, these approaches are primarily developed based on item IDs and collaborative filtering relations, while overlooking the wealth of information embedded in item

content (*i.e.*, title, description, category). More recently, there have been several attempts [55, 59] that leverage additional information related to items to enhance ID sequence modeling. Furthermore, pre-trained language models have been widely utilized to encode item textual features as semantic embeddings to improve performance and generalization [11, 13, 20].

## 4.2 Generative Recommendation

Generative recommendation [3, 19, 26, 32, 52], as an emerging and promising paradigm, has demonstrated superior performance on sequential recommendation tasks compared to traditional recommender systems. In the generative paradigm, each item is indexed with an identifier represented by a list of tokens. This process, known as item tokenization, plays a critical role in generative recommendation. Existing item tokenization methods can be broadly categorized into three groups: heuristic approach, text-based approach, and codebook-based approach. Heuristic approach mainly relies on manually defined rules or techniques, such as time order [15], item clustering [34, 45], and matrix decomposition [15, 27], to construct item identifiers. While these methods are easy to implement, they often fail to capture the implicit relationships between items, limiting their effectiveness. Text-based approach directly utilizes item attributes, such as title, features, and description as identifiers [5, 8, 14, 21]. These methods are usually designed to leverage the internal knowledge of pre-trained language models to improve recommendation performance. However, they suffer from problems such as inconsistent length, semantic ambiguity, and a lack of collaborative information. In contrast, codebook-based approach [6, 29, 32, 44] adopt learnable codebooks to quantize item embeddings, thereby constructing fixed-length, semantically rich item identifiers. In addition, several recent studies have focused on enhancing codebook learning to better adapt it to recommendation systems. Notable examples include the introduction of collaborative and diversity regularization [42], as well as alignment between the item tokenizer and the generative recommender [22].

Reviewing the existing generative recommendation methods, most of them establish a one-to-one mapping between items and their identifiers, which results in challenges such as long-tail distribution, data sparsity, and insufficient diversity in token sequence data. In contrast, in this paper, we introduce multiple semantically relevant item tokenizers to construct more massive and diverse data for generative recommender pre-training, aiming to enhance model scalability and performance.

## 5 Conclusion

In this paper, we introduced MTGRec, a framework that leverages multi-identifier item tokenization for generative recommender pre-training. Compared with previous methods that establish a *one-to-one mapping* between items and their identifiers, MTGRec incorporates multiple item tokenizers to associate each item with several identifiers. Specifically, we first detailed the concept of multi-identifier item tokenization and then enhanced the generative recommender through curriculum pre-training. For multi-identifier item tokenization, we proposed using RQ-VAE checkpoints corresponding to adjacent epochs as semantically relevant item tokenizers. These tokenizers enable the augmentation of item

sequence data into multiple groups of token sequences, each with related but distinct semantic distributions. For curriculum recommender pre-training, we designed a data curriculum scheme based on data influence estimation to dynamically adjust the sampling probabilities of different data groups. Finally, to ensure accurate item identification during recommendation, we fine-tuned the pre-trained model on each item tokenizer and selected the best model for deployment and testing. Extensive experiments and in-depth analyses on three public datasets demonstrated the superior performance of our proposed framework over both traditional and generative recommendation baselines.

For future work, we will adapt the multi-identifier item tokenization to more generalized recommendation scenarios such as transferable recommendation and multi-domain recommendation. Additionally, we will attempt to further scale the model parameters to the billion level and investigate the scaling effect when the model parameters are increased.

## References

- [1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6–12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). <https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfc4967418bfb8ac142f64a-Abstract.html>
- [2] Mayee F. Chen, Nicholas Roberts, Kush Bhatia, Jue Wang, Ce Zhang, Fred-eric Sala, and Christopher Ré. 2023. Skill-it! A data-driven skills framework for understanding and training language models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 – 16, 2023*, Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (Eds.). [http://papers.nips.cc/paper\\_files/paper/2023/hash/70b8505ac79e3e131756f793cd80eb8d-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2023/hash/70b8505ac79e3e131756f793cd80eb8d-Abstract-Conference.html)
- [3] Runjin Chen, Mingxuan Ju, Ngoc Bui, Dimosthenis Antypas, Stanley Cai, Xiaopeng Wu, Leonardo Neves, Zhangyang Wang, Neil Shah, and Tong Zhao. 2024. Enhancing Item Tokenization for Generative Recommendation through Self-Improvement. *arXiv:2412.17171 [cs.LG]* <https://arxiv.org/abs/2412.17171>
- [4] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Y. Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2024. Scaling Instruction-Finetuned Language Models. *J. Mach. Learn. Res.* 25 (2024), 70:1–70:53. <https://jmlr.org/papers/v25/23-0870.html>
- [5] Dario Di Palma. 2023. Retrieval-augmented Recommender System: Enhancing Recommender Systems with Large Language Models. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023, Singapore, Singapore, September 18–22, 2023*, Jie Zhang, Li Chen, Shlomo Berkovsky, Min Zhang, Tommaso Di Noia, Justin Basilico, Luiz Pizzato, and Yang Song (Eds.). ACM, 1369–1373. <https://doi.org/10.1145/3604915.3608889>
- [6] Yijie Ding, Yupeng Hou, Jiacheng Li, and Julian McAuley. 2024. Inductive Generative Recommendation via Retrieval-based Speculation. *arXiv preprint arXiv:2410.02939* (2024).
- [7] Xiaochuang Han, Daniel Simig, Todor Mihaylov, Yulia Tsvetkov, Asli Celikyilmaz, and Tianlu Wang. 2023. Understanding In-Context Learning via Supportive Pretraining Data. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9–14, 2023*, Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, 12660–12673. <https://doi.org/10.18653/V1/2023.ACL-LONG.708>

- [8] Jesse Harte, Wouter Zorgdrager, Panos Louridas, Asterios Katsifodimos, Dietmar Jannach, and Marios Fragkoulis. 2023. Leveraging Large Language Models for Sequential Recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023, Singapore, Singapore, September 18–22, 2023*, Jie Zhang, Li Chen, Shlomo Berkovsky, Min Zhang, Tommaso Di Noia, Justin Basilico, Luiz Pizzato, and Yang Song (Eds.). ACM, 1096–1102. <https://doi.org/10.1145/3604915.3610639>
- [9] Ruining He and Julian J. McAuley. 2016. Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. In *IEEE 16th International Conference on Data Mining, ICDM 2016, December 12–15, 2016, Barcelona, Spain*, Francesco Bonchi, Josep Domingo-Ferrer, Ricardo Baeza-Yates, Zhi-Hua Zhou, and Xindong Wu (Eds.). IEEE Computer Society, 191–200. <https://doi.org/10.1109/ICDM.2016.0030>
- [10] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2–4, 2016, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). <http://arxiv.org/abs/1511.06939>
- [11] Yupeng Hou, Zhankui He, Julian J. McAuley, and Wayne Xin Zhao. 2023. Learning Vector-Quantized Item Representation for Transferable Sequential Recommenders. In *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 – 4 May 2023*, Ying Ding, Jie Tang, Juan F. Sequeda, Lora Aroyo, Carlos Castillo, and Geert-Jan Houben (Eds.). ACM, 1162–1171. <https://doi.org/10.1145/3543507.3583434>
- [12] Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian J. McAuley. 2024. Bridging Language and Items for Retrieval and Recommendation. *CoRR abs/2403.03952* (2024). <https://doi.org/10.48550/ARXIV.2403.03952> arXiv:2403.03952
- [13] Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2022. Towards Universal Sequence Representation Learning for Recommender Systems. In *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 – 18, 2022*, ACM, 585–593. <https://doi.org/10.1145/3534678.3539381>
- [14] Yupeng Hou, Jianmo Ni, Zhankui He, Naveen Sachdeva, Wang-Cheng Kang, Ed H. Chi, Julian J. McAuley, and Derek Zhiyuan Cheng. 2025. Action-Piece: Contextually Tokenizing Action Sequences for Generative Recommendation. *CoRR abs/2502.13581* (2025). <https://doi.org/10.48550/ARXIV.2502.13581> arXiv:2502.13581
- [15] Wenyue Hua, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. 2023. How to Index Item IDs for Recommendation Foundation Models. In *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region, SIGIR-AP 2023, Beijing, China, November 26–28, 2023*, Qingyao Ai, Yiqin Liu, Alistair Moffat, Xuanjing Huang, Tetsuya Sakai, and Justin Zobel (Eds.). ACM, 195–204. <https://doi.org/10.1145/3624918.3625339>
- [16] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17–20, 2018*, IEEE Computer Society, 197–206. <https://doi.org/10.1109/ICDM.2018.00035>
- [17] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling Laws for Neural Language Models. *CoRR abs/2001.08361* (2020). <https://arxiv.org/abs/2001.08361>
- [18] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). <http://arxiv.org/abs/1412.6980>
- [19] Guanghan Li, Xun Zhang, Yufei Zhang, Yifan Yin, Guojun Yin, and Wei Lin. 2024. Semantic Convergence: Harmonizing Recommender Systems via Two-Stage Alignment and Behavioral Semantic Tokenization. arXiv:2412.13771 [cs.LR] <https://arxiv.org/abs/2412.13771>
- [20] Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian J. McAuley. 2023. Text Is All You Need: Learning Language Representations for Sequential Recommendation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6–10, 2023*, Ambuj K. Singh, Yizhou Sun, Leman Akoglu, Dimitrios Gunopulos, Xifeng Yan, Ravi Kumar, Fatma Özcan, and Jieping Ye (Eds.). ACM, 1258–1267. <https://doi.org/10.1145/3580305.3599519>
- [21] Jinming Li, Wentao Zhang, Tian Wang, Guanglei Xiong, Alan Lu, and Gerard Medioni. 2023. GPT4Rec: A Generative Framework for Personalized Recommendation and User Interests Interpretation. In *Proceedings of the 2023 SIGIR Workshop on eCommerce co-located with the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2023), Taipei, Taiwan, July 27, 2023 (CEUR Workshop Proceedings, Vol. 3589)*, Surya Kallumadi, Yubin Kim, Tracy Holloway King, Shervin Malmasi, Maarten de Rijke, and Jacopo Tagliabue (Eds.). CEUR-WS.org. [https://ceur-ws.org/Vol-3589/paper\\_2.pdf](https://ceur-ws.org/Vol-3589/paper_2.pdf)
- [22] Enze Liu, Bowen Zheng, Cheng Ling, Lantao Hu, Han Li, and Wayne Xin Zhao. 2024. End-to-End Learnable Item Tokenization for Generative Recommendation. *CoRR abs/2409.05546* (2024). <https://doi.org/10.48550/ARXIV.2409.05546> arXiv:2409.05546
- [23] Zihan Liu, Yupeng Hou, and Julian J. McAuley. 2024. Multi-Behavior Generative Recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management, CIKM 2024, Boise, ID, USA, October 21–25, 2024*, Edoardo Serra and Francesca Spezzano (Eds.). ACM, 1575–1585. <https://doi.org/10.1145/3627673.3679730>
- [24] Chen Ma, Peng Kang, and Xue Liu. 2019. Hierarchical Gating Networks for Sequential Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4–8, 2019*, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (Eds.). ACM, 825–833. <https://doi.org/10.1145/3292500.3330984>
- [25] Jianmo Ni, Gustavo Hernández Ábreo, Noah Constant, Ji Ma, Keith B. Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-T5: Scalable Sentence Encoders from Pre-trained Text-to-Text Models. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22–27, 2022*, Association for Computational Linguistics, 1864–1874. <https://doi.org/10.18653/V1/2022.FINDINGS-ACL.146>
- [26] Fabian Paischer, Liu Yang, Linfeng Liu, Shuai Shao, Kaveh Hassani, Jiacheng Li, Ricky Chen, Zhang Gabriel Li, Xialo Gao, Wei Shao, Xue Feng, Nima Noorshams, Sem Park, Bo Long, and Hamid Eghbalzadeh. 2024. Preference Discerning with LLM-Enhanced Generative Retrieval. *CoRR abs/2412.08604* (2024). <https://doi.org/10.48550/ARXIV.2412.08604> arXiv:2412.08604
- [27] Aleksandr V. Petrov and Craig Macdonald. 2023. Generative Sequential Recommendation with GPTRec. *CoRR abs/2306.11114* (2023). <https://doi.org/10.48550/ARXIV.2306.11114> arXiv:2306.11114
- [28] Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. 2020. Estimating Training Data Influence by Tracing Gradient Descent. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6–12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). <https://proceedings.neurips.cc/paper/2020/hash/e6385d39ec9394f2f3a354d9d2b88ec-Abstract.html>
- [29] Haohao Qu, Wenqi Fan, Zihui Zhao, and Qing Li. 2024. TokenRec: Learning to Tokenize ID for LLM-based Generative Recommendation. *CoRR abs/2406.10450* (2024). <https://doi.org/10.48550/ARXIV.2406.10450> arXiv:2406.10450
- [30] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.
- [31] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.* 21 (2020), 140:1–140:67. <http://jmlr.org/papers/v21/20-074.html>
- [32] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Mahesh Sathiamoorthy. 2023. Recommender Systems with Generative Retrieval. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 – 16, 2023*. [http://papers.nips.cc/paper\\_files/paper/2023/hash/20dcab0f14046a5c6b02b61da9f13229-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2023/hash/20dcab0f14046a5c6b02b61da9f13229-Abstract-Conference.html)
- [33] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26–30, 2010*, Michael Rappa, Paul Jones, Juliana Freire, and Soumen Chakrabarti (Eds.). ACM, 811–820. <https://doi.org/10.1145/1772690.1772773>
- [34] Zihua Si, Zhongxiang Sun, Jiale Chen, Guozhang Chen, Xiaoxue Zang, Kai Zheng, Yang Song, Xiao Zhang, and Jun Xu. 2023. Generative Retrieval with Semantic Tree-Structured Item Identifiers via Contrastive Learning. *CoRR abs/2309.13375* (2023). <https://doi.org/10.48550/ARXIV.2309.13375> arXiv:2309.13375
- [35] Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening Sentence Representations for Better Semantics and Faster Retrieval. *CoRR abs/2103.15316* (2021). <https://arxiv.org/abs/2103.15316>
- [36] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3–7, 2019*, Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu (Eds.). ACM, 1441–1450. <https://doi.org/10.1145/3357384.3357895>
- [37] Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten de Rijke, and Zhaochun Ren. 2023. Learning to Tokenize for Generative Retrieval. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 – 16, 2023*, Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (Eds.). [http://papers.nips.cc/paper\\_files/paper/2023/hash/](http://papers.nips.cc/paper_files/paper/2023/hash/)

- 91228b942a4528cdae031c1b68b127e8-Abstract-Conference.html
- [38] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved Recurrent Neural Networks for Session-based Recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, DLRS@RecSys 2016, Boston, MA, USA, September 15, 2016*. ACM, 17–22. <https://doi.org/10.1145/2988450.2988452>
- [39] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5–9, 2018*, Yi Chang, Chengxiang Zhai, Yan Liu, and Yoelle Maarek (Eds.). ACM, 565–573. <https://doi.org/10.1145/3159652.3159656>
- [40] Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Prakash Gupta, Tal Schuster, William W. Cohen, and Donald Metzler. 2022. Transformer Memory as a Differentiable Search Index. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28–December 9, 2022*, Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (Eds.). [http://papers.nips.cc/paper\\_files/paper/2022/hash/892840a6123b5ec99ebaab8be1530fba-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2022/hash/892840a6123b5ec99ebaab8be1530fba-Abstract-Conference.html)
- [41] Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural Discrete Representation Learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4–9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 6306–6315. <https://proceedings.neurips.cc/paper/2017/hash/7a98af17e63a0ac09ce2e96d03992fbc-Abstract.html>
- [42] Wenjie Wang, Honghui Bao, Xinyu Lin, Jizhi Zhang, Yongqi Li, Fuli Feng, Seokiong Ng, and Tat-Seng Chua. 2024. Learnable Item Tokenization for Generative Recommendation. In *International Conference on Information and Knowledge Management*.
- [43] Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, Xing Xie, Hao Sun, Weiwei Deng, Qi Zhang, and Mao Yang. 2022. A Neural Corpus Indexer for Document Retrieval. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28–December 9, 2022*, Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (Eds.). [http://papers.nips.cc/paper\\_files/paper/2022/hash/a46156bd3579c3b268108ea6aca71d13-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2022/hash/a46156bd3579c3b268108ea6aca71d13-Abstract-Conference.html)
- [44] Yidan Wang, Zhaochun Ren, Weiwei Sun, Jiyuan Yang, Zhixiang Liang, Xin Chen, Ruobing Xie, Su Yan, Xu Zhang, Pengjie Ren, Zhumin Chen, and Xin Xin. 2024. Enhanced Generative Recommendation via Content and Collaboration Integration. *CoRR* abs/2403.18480 (2024). <https://doi.org/10.48550/ARXIV.2403.18480> arXiv:2403.18480
- [45] Ye Wang, Jiahao Xun, Mingjie Hong, Jieming Zhu, Tao Jin, Wang Lin, Haoyuan Li, Linjun Li, Yan Xia, Zhou Zhao, and Zhenhua Dong. 2024. EAGER: Two-Stream Generative Recommender with Behavior-Semantic Collaboration. *CoRR* abs/2406.14017 (2024). <https://doi.org/10.48550/ARXIV.2406.14017> arXiv:2406.14017
- [46] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised Graph Learning for Recommendation. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11–15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 726–735. <https://doi.org/10.1145/3404835.3462862>
- [47] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-Based Recommendation with Graph Neural Networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27–February 1, 2019*. AAAI Press, 346–353. <https://doi.org/10.1609/AAAI.V33I01.3301346>
- [48] Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. LESS: Selecting Influential Data for Targeted Instruction Tuning. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21–27, 2024*. OpenReview.net. <https://openreview.net/forum?id=PG5fV50maR>
- [49] Canwen Xu, Corby Rosset, Luciano Del Corro, Shweti Mahajan, Julian J. McAuley, Jennifer Neville, Ahmed Hassan Awadallah, and Nikhil Rao. 2023. Contrastive Post-training Large Language Models on Data Curriculum. *CoRR* abs/2310.02263 (2023). <https://doi.org/10.48550/ARXIV.2310.02263> arXiv:2310.02263
- [50] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph Contextualized Self-Attention Network for Session-based Recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10–16, 2019*. ijcai.org, 3940–3946. <https://doi.org/10.24963/IJCAI.2019/547>
- [51] Tianchi Yang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, and Qi Zhang. 2023. Auto Search Indexer for End-to-End Document Retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6–10, 2023*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, 6955–6970. <https://doi.org/10.18653/V1/2023.FINDINGS-EMNLP.464>
- [52] Jun Yin, Zhengxin Zeng, Mingzheng Li, Hao Yan, Chaozhao Li, Weihao Han, Jianjin Zhang, Ruochen Liu, Allen Sun, Denvy Deng, Feng Sun, Qi Zhang, Shirui Pan, and Senzhang Wang. 2024. Unleash LLMs Potential for Recommendation by Coordinating Twin-Tower Dynamic Semantic Token Generator. *CoRR* abs/2409.09253 (2024). <https://doi.org/10.48550/ARXIV.2409.09253> arXiv:2409.09253
- [53] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. 2022. SoundStream: An End-to-End Neural Audio Codec. *IEEE ACM Trans. Audio Speech Lang. Process.* 30 (2022), 495–507. <https://doi.org/10.1109/TASLP.2021.3129994>
- [54] Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Jiayuan He, Yinghai Lu, and Yu Shi. 2024. Actions Speak Louder than Words: Trillion-Parameter Sequential Transducers for Generative Recommendations. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21–27, 2024*. OpenReview.net. <https://openreview.net/forum?id=xye7iNsgXn>
- [55] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. 2019. Feature-level Deeper Self-Attention Network for Sequential Recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10–16, 2019*, Sarit Kraus (Ed.). ijcai.org, 4320–4326. <https://doi.org/10.24963/IJCAI.2019/600>
- [56] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, Yingqian Min, Zhichao Feng, Xinyan Fan, Xu Chen, Pengfei Wang, Wendi Ji, Yaliang Li, Xiaoling Wang, and Ji-Rong Wen. 2021. RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1–5, 2021*. ACM, 4653–4664.
- [57] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A Survey of Large Language Models. *CoRR* abs/2303.18223 (2023). <https://doi.org/10.48550/ARXIV.2303.18223> arXiv:2303.18223
- [58] Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. 2024. Adapting Large Language Models by Integrating Collaborative Semantics for Recommendation. In *40th IEEE International Conference on Data Engineering, ICDE 2024, Utrecht, The Netherlands, May 13–16, 2024*. IEEE, 1435–1448. <https://doi.org/10.1109/ICDE60146.2024.00118>
- [59] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-Rec: Self-Supervised Learning for Sequential Recommendation with Mutual Information Maximization. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19–23, 2020*, Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (Eds.). ACM, 1893–1902. <https://doi.org/10.1145/3340531.3411954>
- [60] Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Filter-enhanced MLP is All You Need for Sequential Recommendation. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25–29, 2022*, Frédérique Laforest, Raphaël Troncy, Elena Simperl, Deepak Agarwal, Aristides Gionis, Ivan Herman, and Lionel Médini (Eds.). ACM, 2388–2399. <https://doi.org/10.1145/3485447.3512111>