

Content-Based Collaborative Generation for Recommender Systems

Yidan Wang
Shandong University
Qingdao, China
yidanwang@mail.sdu.edu.cn

Zhaochun Ren
Leiden University
Leiden, Netherlands
z.ren@liacs.leidenuniv.nl

Weiwei Sun
Shandong University
Qingdao, China
sunnweiwei@gmail.com

Jiyuan Yang
Shandong University
Qingdao, China
jiyuan.yang@mail.sdu.edu.cn

Zhixiang Liang
Zhejiang University
Hangzhou, China
zliang18@illinois.edu

Xin Chen
WeChat, Tencent
Beijing, China
andrewxchen@tencent.com

Ruobing Xie
Tencent
Beijing, China
xrbsnowing@163.com

Su Yan
WeChat, Tencent
Beijing, China
suyan@tencent.com

Xu Zhang
WeChat, Tencent
Beijing, China
xuonezhang@tencent.com

Pengjie Ren
Shandong University
Qingdao, China
renpengjie@sdu.edu.cn

Zhumin Chen
Shandong University
Qingdao, China
chenzhumin@sdu.edu.cn

Xin Xin*
Shandong University
Qingdao, China
xinxin@sdu.edu.cn

ABSTRACT

Generative models have emerged as a promising utility to enhance recommender systems. It is essential to model both item content and user-item collaborative interactions in a unified generative framework for better recommendation. Although some existing large language model (LLM)-based methods contribute to fusing content information and collaborative signals, they fundamentally rely on textual language generation, which is not fully aligned with the recommendation task. How to integrate content knowledge and collaborative interaction signals in a generative framework tailored for item recommendation is still an open research challenge.

In this paper, we propose **content-based collaborative generation for recommender systems**, namely ColaRec. ColaRec is a sequence-to-sequence framework which is tailored for directly generating the recommended item identifier. Precisely, the input sequence comprises data pertaining to the user's interacted items, and the output sequence represents the generative identifier (GID) for the suggested item. To model collaborative signals, the GIDs are constructed from a pretrained collaborative filtering model, and the user is represented as the content aggregation of interacted items. To this end, ColaRec captures both collaborative signals and content information in a unified framework. Then an item indexing task is proposed to conduct the alignment between the content-based semantic space and the interaction-based collaborative space. Besides, a contrastive loss is further introduced to ensure that items with similar collaborative GIDs have similar content representations. To

verify the effectiveness of ColaRec, we conduct experiments on four benchmark datasets. Empirical results demonstrate the superior performance of ColaRec.

CCS CONCEPTS

• Information systems → Recommender systems.

KEYWORDS

Recommender System, Generative Recommendation

ACM Reference Format:

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. Content-Based Collaborative Generation for Recommender Systems. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24)*, October 21–25, 2024, Boise, ID, USA. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/3627673.3679692>

1 INTRODUCTION

Recommender systems are widely deployed to discover user interests and provide personalized information services [21, 38, 58]. Recently, generative models, such as large language models (LLMs) [7, 62] and diffusion models [5, 57], have gained prominence in advancing artificial intelligence. In such a context, plenty of research has emerged to utilize generative models for recommendation [40, 50]. To achieve satisfying results, the recommender should be able to incorporate both user-item collaborative signals and item content information. Collaborative signals refer to the knowledge contained in the user-item interactions while item content information refers to the textual description of items, as shown in Figure 1.

LLM-based recommendation is a straightforward solution to fuse content information with collaborative signals. In general, these approaches reorganize the historical user-item interactions into text-based natural language, aka the instruction-tuning dataset. Then, the recommendation task is reformulated as language generation under designed prompts [10, 29]. Although this kind of approach utilizes the powerful text generation capability of LLMs,

*The 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.

CIKM '24, October 21–25, 2024, Boise, ID, USA

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 979-8-4007-0436-9/24/10

<https://doi.org/10.1145/3627673.3679692>

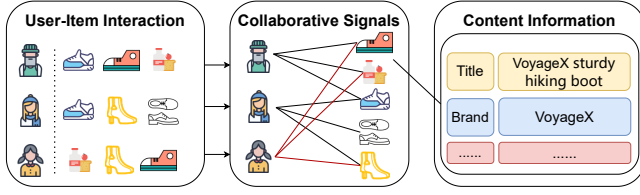


Figure 1: Illustration of collaborative signals and content information. Collaborative signals refer to the knowledge contained in user-item interactions while content information refers to the textual description of items.

there exists the inherent misalignment between the two tasks of language generation and item recommendation. For example, there is usually a non-trivial grounding stage to map the generated language to a concrete item [1, 33]. LLMs also suffer poor ranking performance to generate target itemIDs from a large candidate pool [32, 60]. The task discrepancy limits the practical usage of LLM-based methods.

Inspired by generative retrieval [47], some works [40, 44] assign each item with a unique sequence of tokens as the item’s generative identifier, aka GID, then a generative sequence-to-sequence model which is tailored for directly generating item recommendation is trained without the need of explicit language modeling. The input sequence comprises historical user-item interactions while the output sequence refers to the GID of the recommended item. In this paper, we refer such the above paradigm as **Generative Recommendation**¹. Compared with conventional itemIDs with an assigned single random token, the sequential tokens of a GID contain more explicit representation information. As shown in Figure 2, we see correlated GIDs denote correlated items, and thus help the recommender to conduct more effective generation. Generative recommendation provides an end-to-end paradigm to design generative models tailored for the recommendation task. Plenty of studies have been conducted along this research line [37, 40, 44, 46, 63].

However, existing generative recommendation methods still fail to effectively model collaborative signals and content information in a unified framework. For example, several methods utilize either the item titles or hierarchical content embeddings of items to construct GIDs [28, 40]. These methods only consider the item content information while the collaborative signals between users and items are overlooked. On the contrary, Si et al. [44] constructed GIDs using item embeddings of a pretrained SASRec [23]. Although the GID in this work contains the item-item sequential (collaborative) connections, the proposed recommendation framework in their research fails to model the item content information.

Besides, existing generative recommendation methods cannot effectively align item content information and collaborative signals. Although Hua et al. [22] proposed a combined approach that concatenates content-based semantic strings with collaborative IDs derived from the item-item co-occurrence matrix. However, this naive concatenation, lacking a proper learning process, fails to achieve effective alignment, resulting in sub-optimal performance.

¹Wang et al. [49] proposed another paradigm to directly generate new content, e.g., images, for recommendation. However, their methods are tailored for the generation of virtual content and cannot be used for concrete items. In this paper, we target on the recommendation of concrete items.

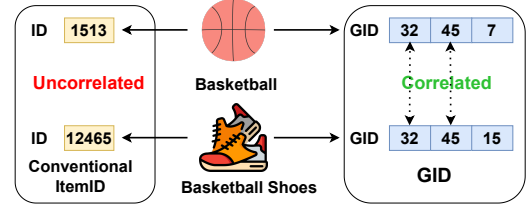


Figure 2: Comparison between conventional itemIDs and GIDs. GIDs contain more concrete correlations.

The alignment should be achieved through an explicit learning process, e.g., the mapping between the content-based semantic space and the interaction-based collaborative space.

In this paper, we propose **content-based collaborative generation for recommender systems (ColaRec)**, a model that unifies both item content information and user-item collaborative signals in a sequence-to-sequence generation framework tailored for the recommendation task. Taking a particular user as an example, the input sequence of ColaRec consists of unordered² tuples with each tuple describing the content information of one interacted item of this user. Then an encoder-decoder Transformer [48] is used to generate the GID of the target item. More precisely, we propose two tailored designs to jointly model user-item collaborative signals and item content information. Firstly, we propose to construct GIDs using item representations obtained from a pretrained collaborative filtering model. In this paper, we use LightGCN [16] as the pretrained model. The LightGCN model is trained on the user-item interaction graph and thus the constructed GID can effectively encode the user-item collaborative signals. Note that the LightGCN model can also be alternated with other models. Secondly, in ColaRec, the user is represented as the content aggregation of tuples with each tuple describing one her/his historically interacted item. This design also keeps inline with the nature of collaborative filtering. The aggregation of content-based tuples is fed to the generative Transformer model to effectively capture the item content information.

To conduct the alignment between content information and collaborative signals, we propose an auxiliary item indexing task which targets on mapping the item side information into the GID of this item through the same encoder-decoder model. Specifically, the item side information contains both the textual content information and a set of users who have interacted with this item. To this end, the indexing task maps both the item content information and user-item interaction signals into the constructed GID, achieving better alignment. Besides, we further propose a contrastive loss to ensure that items with similar collaborative GIDs are also similar in the content-based semantic space.

To demonstrate the effectiveness of the proposed ColaRec, we conduct extensive experiments on four public datasets. Experimental results show that the proposed ColaRec outperforms related state-of-the-art baselines³.

Our main contributions are as follows:

²In this paper, we focus on the general recommendation task other than sequential recommendation. To this end, we use unordered item tuples as the input.

³The code of this work is available at <https://github.com/Junewang0614/ColaRec>

- We propose ColaRec, a generative recommendation framework which utilizes an encoder-decoder model to jointly capture content information and collaborative signals for recommendation.
- We propose an auxiliary item indexing task and a contrastive loss to perform better alignment between item content information and user-item collaborative signals to further enhance the performance of generative recommendation.
- We conduct extensive experiments on four datasets to demonstrate the effectiveness of the proposed ColaRec. Experimental results show superior recommendation performance of ColaRec.

2 RELATED WORK

In this section, we review related literature on collaborative filtering and generative models for recommendation.

2.1 Collaborative Filtering

Collaborative filtering (CF) is one of the most representative methods to build a recommendation agent. CF believes that a user can be represented as the aggregation of her/his interacted items, and vice versa. The keystone to conduct collaborative filtering is the user-item interaction matrix. Early approaches [26, 42] are based on matrix factorization (MF) to jointly model the latent space for users and items. Due to the expressiveness of deep neural networks, plenty of research [8, 11, 18, 36, 64] has been conducted to enhance CF through deep learning. Besides, since the user-item interaction signals can be naturally encoded into an interaction graph, graph neural networks (GNN) also shed lights in the field of CF. Berg et al. [2] proposed to use graph convolution for matrix completion. Wang et al. [51] proposed the NGCF model to use GNN for CF. He et al. [16] proposed the LightGCN model to simplify GNN for recommender systems, leading to a simpler and linear CF model. LightGCN serves as one of the most popular GNN-based CF approaches due to its effectiveness. Lin et al. [34] proposed NCL to introduce contrastive learning into graph-based CF. Besides, content information has also been utilized to enhance the CF model. Rendle [41] proposed the notable factorization machine (FM) to extend MF for categorical contextual features. He and McAuley [15], Wei et al. [54, 55], Wu et al. [56] proposed to enhance CF with visual or text features. Li et al. [27] proposed RecFormer to model long text sequences for recommendation.

Different from existing CF methods, in this paper we focus on the new paradigm of generative recommendation, where the item is represented as a sequence of tokens, i.e., GID, and the recommendation is provided in a generative fashion.

2.2 Generative Models for Recommendation

Generative models have become a hot research topic to generate new content, such as images and text. Variational Autoencoders (VAEs) [19, 25] and Generative Adversarial Networks (GANs) [9, 13, 24], are two kinds of representative generative models. Besides, diffusion models [5, 20, 57] have also shown promising results in content generation. These generative models have also been utilized for recommender systems [4, 14, 17, 30, 31, 43, 50, 53].

Recently, Transformers [48] have shown promise in language generation, leading to notable LLMs like GPTs (Generative Pre-trained Transformers). LLM-based recommendation provides a

straightforward solution to fuse user-item interaction signals with item textual content [32, 35, 61, 63, 65]. Typically, these approaches need to construct the recommendation training data with natural language, i.e., the instruction-tuning dataset, then the recommendation task is transformed into text generation under designed language prompts [10, 29, 33, 60]. For instance, Luo et al. [35] proposed to inject collaborative signals into LLM-based recommendation through prompt augmentation. [61] and [32] proposed to fuse item embeddings into prompt embeddings for CTR prediction and recommendation. Lin et al. [33] utilized LLMs to generate multiple aspects of the recommended items including titles and attributes. LC-Rec [63] proposed to perform recommendation through conducting various language generation tasks. Despite the strong generation capabilities of LLMs, there is an inherent misalignment between language generation and item recommendation, which limits the practical usage of LLM-based methods. For example, it is usually necessary to go through a non-trivial grounding stage to map the generated language into specific items [1, 32, 33]. Besides, it is also challenging for LLMs to directly generate recommended items from a large candidate pool [32, 35, 60].

Inspired by generative retrieval, generative recommendation is a new paradigm to design generative models which is tailored for the recommendation task. Generative retrieval, which relies on the parametric memory of generative models to directly generate relevant document identifiers, has drawn increasing attention. Tay et al. [47] firstly proposed a differentiable search index (DSI) for generative retrieval. Plenty of methods have been conducted following this research line. Representative works include SEAL [3], NCI [52], and GenRet [45]. Similar to generative retrieval, generative recommendation constructs a sequence of tokens, aka generative identifier (GID), for each item, then a generative sequence-to-sequence model which is tailored for directly generating item recommendation is trained without the need of explicit language modeling. The input comprises historical user-item interactions and the output refers to the GID of the recommended item. Following this paradigm, TIGER [40] is a representative generative recommendation method which uses an RQ-VAE [59] to construct the GID and then uses an encoder-decoder based transformer to generate sequential recommendation. Si et al. [44] proposed to construct the GID from a pretrained SASRec [23] model for sequential recommendation. Hua et al. [22] further investigated the effect of item identifier construction. Tan et al. [46] utilized human language tokens to represent each item with a unique textual identifier.

Despite the rise of generative recommendation, existing generative recommendation methods still suffer from the ineffective infusion of collaborative signals and item content information. How to jointly model and align collaborative signals and content information in an end-to-end generative framework tailored for item recommendation is still an open research challenge.

3 NOTATIONS AND TASK FORMULATION

3.1 Notations

Let u and i denote a specific user and an item, respectively. The set of interacted items of user u is denoted as \mathcal{I}_u^+ , and the set of users who have interacted with item i is denoted as \mathcal{U}_i^+ . The content description of item i is denoted as c_i . The randomly assigned single

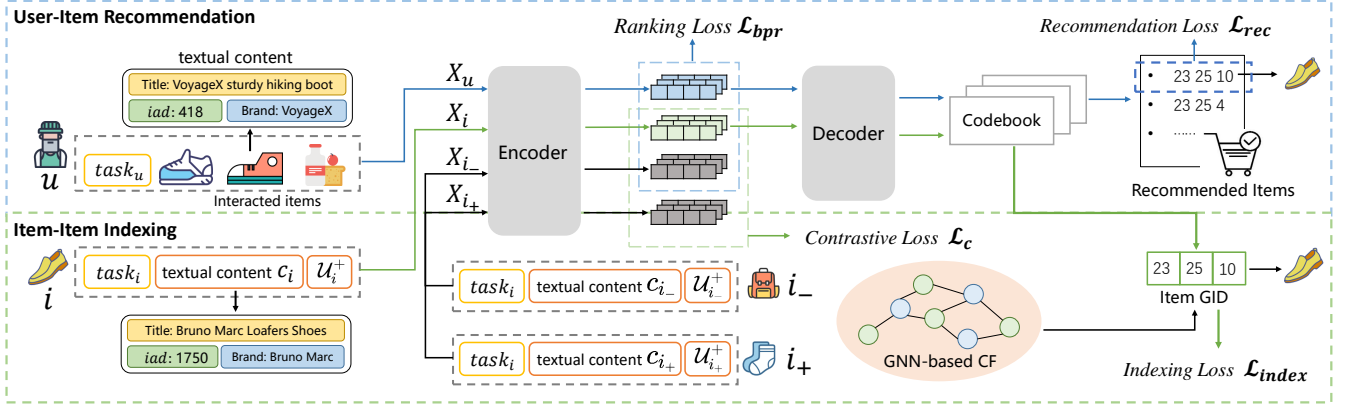


Figure 3: Overview of ColaRec. ColaRec assigns each item with a GID obtained from a GNN-based CF model. ColaRec consists two tasks. User-Item Recommendation aims to map the user’s interacted items with textual content into the GID of the recommended item, i.e., \mathcal{L}_{rec} . Item-Item Indexing targets on the mapping from item side information into the item’s GID, i.e., \mathcal{L}_{index} . Besides, a ranking loss \mathcal{L}_{bpr} and a contrastive loss \mathcal{L}_c are also introduced.

token to denote user u is the user’s atomic identifier, aka, uad_u . Similarly, the randomly assigned single token to denote item i is the item’s atom identifier iad_i . Besides iad_i , each item i is also assigned with a generative identifier $GID_i = [z_i^1, z_i^2, \dots, z_i^l]$, where l denotes the length of GID_i .

3.2 Generative Recommendation

The task of generative recommendation is given the input describing the information of \mathcal{I}_u^+ , generating a list of GIDs as the recommendation result. The GID is generated through an auto-regressive manner. The probability of recommending item i for user u is estimated as:

$$p(u, i) = \prod_{t=1}^l p(z_i^t | \mathcal{I}_u^+, z_i^1, z_i^2, \dots, z_i^{t-1}). \quad (1)$$

The recommender selects items with the top- n highest $p(u, i)$ as the recommendation list for user u .

4 METHODOLOGY

In this section, we describe the details of ColaRec. We first provide the overview of the proposed ColaRec. Then the construction of items’ GIDs is detailed. After that, the user-item recommendation task and the item-item indexing task are described. Finally, we describe the joint optimization of the above tasks.

4.1 Overview of ColaRec

Figure 3 illustrates an overview of the proposed ColaRec. ColaRec constructs the GID using a graph-based CF model, which effectively captures collaborative signals. The training of ColaRec consists of two tasks: the user-item recommendation task and the item-item indexing task. The user-item recommendation aims to map the content information of the user’s historical interacted items into the GID of the recommended item. The item-item indexing task targets on the mapping from the item side information into the item’s GID. Both the two tasks are achieved through a shared

encoder-decoder based model. The recommendation task unifies both collaborative signals and item content information for better recommendation, while the indexing task performs the alignment between collaborative signals and content information.

4.2 Generative Identifier Construction

The construction of GIDs plays a crucial role in generative recommendation. Generally speaking, GIDs should satisfy the following expectations for better recommendation: (i) GIDs need to contain knowledge about both collaborative signals and content information; (ii) Correlated items (e.g., similar items in content or items interacted by similar users) should have correlated GIDs; (iii) Each item should have one unique GID and each GID should correspond to one specific item.

To fulfill above expectations, we utilize a hierarchical clustering approach to construct GIDs from a graph-based CF model. Specifically, we first extract item representations from a pretrained LightGCN model. Then the constrained K -means algorithm is called hierarchically based on the item representations. The next level clustering is conducted with items in the current cluster as the whole item set. For the t -th level clustering with $t \in [1, l-1]$, the number of items is no more than K^{l-t} in each cluster. Regarding the last level of leaf nodes, we randomly allocate 1 to K to the items. In this way, we establish a K -ary tree to organize the item set. Each item corresponds to a leaf node, while the path from the root to the leaf node is the GID of the item. Since LightGCN is trained on the user-item interaction graph, the GID can naturally encode collaborative signals. Meanwhile, there is a codebook embedding matrix for every position of the GID, which will incorporate the content information in the item indexing task. We give the detailed description in section 4.4. To this end, the GID together with the corresponding codebook embeddings helps the recommender to model both collaborative signals and content information.

4.3 User-Item Recommendation

Model Inputs. The input sequence of user-item recommendation for user u consists of unordered tuples with each tuple describing the content information of one interacted item of this user. For the textual description of item i , we adopt the universal data format from [27]. Specifically, textual description c_i of item i is formulated as a sequence that comes from a flattened attribute dictionary consisting of key-value attribute pairs (k, v) , i.e., $[k_1:v_1, k_2:v_2, \dots]$. Besides, we also introduce the item atomic identifier iad_i into the content information to further increase model fidelity. To this end, the content tuple of item i is formulated as:

$$c_i = [iad_i, k_1:v_1, k_2:v_2, \dots]. \quad (2)$$

The key idea of CF is that users' preferences can be inferred from their interacted items. Therefore, for each user u , the input consists of the content aggregation of item tuples that u has interacted with to reinforce the collaborative signals.

Since the training of ColaRec has two tasks, in the user-item recommendation task, we augment the input with a special task token $task_u$ at the beginning of the input to inform the model that the ongoing task is the recommendation task. Thus, the input for the user-item recommendation task is:

$$X_u = [task_u, \{c_i | i \in \mathcal{I}_u^+\}]. \quad (3)$$

Item Generation. We employ an encoder-decoder based Transformer model to generate item recommendation. Given the model input X_u , the model encoder captures the semantic information of X_u and returns the hidden state $\text{Encoder}(X_u)$. After that, given the generated tokens $z^{<t}$ before the t -th generation step, the decoder generates the latent representation $\mathbf{d}_t \in \mathbb{R}^m$ for the t -th token of GID. m is the dimension of the latent representation. This process can be formulated as:

$$\mathbf{d}_t = \text{Decoder}(\text{Encoder}(X_u), z^{<t}) \quad (4)$$

The generation probability at step t is estimated by \mathbf{d}_t and the codebook embedding matrix for position t , which is formulated as:

$$p(z^t | z^{<t}, X_u) = \text{softmax}(\mathbf{d}_t \cdot \mathbf{E}_t^T), \quad (5)$$

where \mathbf{E}_t is the t -th step codebook embedding matrix.

We adopt the cross-entropy loss for model optimization. Specifically, given a (u, i) pair in the training set, the generative loss of recommendation is formulated as:

$$\mathcal{L}_{\text{rec}} = - \sum_{t=1}^l \log p(z_i^t | X_u, z_i^1, z_i^2, \dots, z_i^{t-1}). \quad (6)$$

In this work, we use a pretrained T5 [39] as the Transformer model. The parameters of T5 are also fine-tuned through back propagation to better adapt the model for recommendation. Note that our approach is model-agnostic and T5 can be replaced by other sequence-to-sequence models, even without pretraining.

4.4 Item-Item Indexing

To align collaborative signals and item content information, we introduce an item-item indexing task which conducts the mapping

from the content-based semantic space into the interaction-based collaborative space.

Model Inputs. The input sequence for item indexing contains the textual information of the item. Besides, we also introduce information of users who have interacted with this item, to further encode collaborative signals. Similar to the recommendation task, we augment the input with a special task token $task_i$ at the beginning of the input for item indexing. Therefore, the input of item indexing is formulated as:

$$X_i = [task_i, c_i, \{uad_u | u \in \mathcal{U}_i^+\}]. \quad (7)$$

Item Indexing. The indexing task is conducted through the same model and codebook embeddings as the recommendation task. The generation probability for the indexing task is formulated similarly with Eq. (4) and Eq. (5) except that the model input is X_i instead of X_u . We adopt the cross-entropy loss for parameter learning. The loss for item indexing is defined as:

$$\mathcal{L}_{\text{index}} = - \sum_{t=1}^l \log p(z_i^t | X_i, z_i^1, z_i^2, \dots, z_i^{t-1}). \quad (8)$$

4.5 Multi-Task Training

Besides the above two tasks, we further introduce a ranking loss to enhance the ranking ability of ColaRec and a contrastive loss to conduct better alignment.

Item Ranking. For a (u, i) pair in the training dataset, we randomly sample one item that the user u has not interacted with as the negative sample i_- . The BPR loss [42] is utilized to optimize the item ranking, which is formulated as:

$$\mathcal{L}_{\text{bpr}} = -\ln \sigma(\mathbf{h}(X_u) \cdot (\mathbf{h}(X_i) - \mathbf{h}(X_{i_-}))), \quad (9)$$

where $\mathbf{h}(\cdot)$ denotes the last hidden states of $\text{Encoder}(\cdot)$, and σ denotes the sigmoid function. The above loss pushes together the positive (u, i) pair and pushes away the negative (u, i_-) pair, thus helping the model to capture the ranking knowledge between items.

Contrastive Learning. To conduct better alignment between collaborative signals and content information, a contrastive loss is further introduced. The idea is that items with similar GIDs should also be similar in the content-based semantic space. To this end, for the item i , we randomly sample an item i_+ which has the overlapped prefix tokens in GIDs as the positive sample. The sampled item i_- in \mathcal{L}_{bpr} serves as the negative sample. Note that here we ensure the sampled item i_- in \mathcal{L}_{bpr} has no overlapped GID tokens with item i . The contrastive loss is defined as:

$$\mathcal{L}_c = -\ln \sigma(\mathbf{h}(X_i) \cdot (\mathbf{h}(X_{i_+}) - \mathbf{h}(X_{i_-}))), \quad (10)$$

Such a contrastive loss helps the model to learn better item input representations.

Joint Optimization. Finally, ColaRec is trained with the above described tasks jointly:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{index}} + \mathcal{L}_{\text{bpr}} + \alpha \mathcal{L}_c, \quad (11)$$

where α denotes the weight for the contrastive loss.

During the inference, to avoid the recommender from generating invalid GIDs, we employ the constrained beam search [6] to limit the generative range of the current token based on the prefix tokens.

Table 1: Statistics of four public datasets after preprocessing.

Datasets	#Users	#Items	#Interactions
Beauty	22,363	12,101	198,502
Sports	35,598	18,357	296,337
Phone	27,879	10,429	194,439
Recipe	17,813	41,240	555,618

5 EXPERIMENT

In this section, we conduct experiments to evaluate the proposed ColaRec. We aim to answer the following research questions:

- RQ1 How does the proposed ColaRec perform compared with existing recommendation methods?
- RQ2 How does the joint training of multiple tasks affect the performance of ColaRec?
- RQ3 How does the design of GIDs affect the recommendation performance?

5.1 Datasets

We use four real-world public datasets to evaluate the performance of ColaRec. Specifically, the experiments are conducted on three subcategories from Amazon Product Reviews⁴ (“Beauty”, “Sports and Outdoors”, and “Cell Phones and Accessories”) and “Recipe” from Food.com⁵. Users and items that have less than five interactions are filtered out. Table 1 shows the statistics of all four datasets. As for content information, we use “title”, “brand” and “categories” from the Amazon item metadata as the textual content information of items. For Recipe, we use “name”, “description”, and “tag” to describe item content.

5.2 Evaluation Protocols

We adopt cross-validation to evaluate the performance of recommenders. In this paper we focus on general recommendation rather than sequential recommendation. To this end, we randomly split each user’s historical interactions into the training/validation/test set with the ratio of 8:1:1. We employ two widely used metrics, recall@ n and normalized discount cumulative gain (NDCG@ n), to evaluate the model performance. Recall evaluates how many ground-truth items occur in the recommended list, while NDCG further focuses on their rankings in the list. Note that in this paper, the candidate item set is the whole item set, other than a small subset with selected items. Each experiment is conducted three times and the average score is reported.

5.3 Baselines

We compare ColaRec with several representative related baselines, including both conventional CF-based methods and generative models for recommendation. CF-based baselines include NeuMF [18], LightGCN [16], SimpleX [36], and NCL [34]. Generative models for recommendation include MultiVAE [31], DiffRec [50], DSI [47], TIGER [40] and LC-Rec[63].

⁴<https://jmcauley.ucsd.edu/data/amazon/>

⁵<https://www.kaggle.com/datasets/shuyangli94/food-com-recipes-and-user-interactions>

- **NeuMF** [18] enhances MF with multiple hidden layers to learn non-linear patterns in user-item interactions.
- **LightGCN** [16] simplifies GNN for CF and learns user and item representations via linear neighborhood aggregation.
- **SimpleX** [36] is a simple CF model with a cosine-based contrastive loss and negative sampling.
- **NCL** [34] improves LightGCN with contrastive learning.
- **MultiVAE** [31] is an autoencoder-based method, which utilizes VAEs to model the interaction signals.
- **DiffRec** [50] is a new proposed recommendation model based on diffusion models. DiffRec learns the user-item interaction knowledge through a reconstruction and denoising manner.
- **DSI** [47] is a generative document retrieval method. To adapt DSI for recommendation, we formulate the input as GIDs of the user’s historical interacted items. We use two versions of DSI. **DSI-R** refers to the DSI model with a random string as the GID of the item. **DSI-S** is a DSI model which constructs the item GIDs with a hierarchical K -means algorithm based on the item textual content embeddings from a pretrained BERT model.
- **TIGER** [40] is a generative recommendation method. Specifically, a pretrained Sentence-T5 encoder is used to obtain embeddings of the items’ textual content. These embeddings are then quantized using an RQ-VAE to build GIDs. We do not introduce sequential orders to adapt TIGER for general recommendation.
- **LC-Rec** [63] follows a similar approach to TIGER [40] for item index learning. Besides, it employs various language generation tasks under different prompts to accomplish recommendation. We don’t introduce P5-based baselines [12, 22] since these methods require the model input prompt to include candidate items for the general recommendation task. Given the limited input length, these methods cannot perform item ranking among the whole item set.

5.4 Implementation Details

For all datasets, the length of GIDs is set to $l = 3$, and the number of clusters in hierarchical K -means is set as $K = 32$, except for Recipe in which the cluster number is set as 48. Each user is represented through the aggregation of randomly sampled interacted item tuples, while each item introduces one randomly sampled user who have interacted with it in the indexing task. We use a uniform distribution to sample negative instances for \mathcal{L}_{bpr} and \mathcal{L}_c to avoid the effect of different negative sampling strategies. We leave the investigation of negative sampling for generative recommendation as the future work. To be consistent with the word embeddings of the pretrained T5-small model, the embedding dimensions of u_{ad} , i_{ad} and codebooks in ColaRec are set to 512. The values of the contrastive loss coefficient, i.e., α are set to $\{0.02, 0.08, 0.1, 0.05\}$ in Beauty, Sports, Phone and Recipe respectively. We optimize the model using AdamW with $5e-4$ as the learning rate. The batch size is set to 128. For baselines, we carefully search the hyper-parameters except for user and item embedding sizes, which are set to 512 to ensure a fair comparison with ColaRec.

5.5 Performance Comparison (RQ1)

To answer RQ1, we conduct a comparative analysis of the proposed ColaRec on both overall users and long-tail users.

Table 2: Performance comparison on four public datasets. The best and the second-best scores are marked in bold and underlined fonts, respectively. * denotes the paired t-test with significance p-value < 0.1. R and N stand for Recall and NDCG.

Datasets	Metric	CF-based Methods				Generative Models for Recommendation						Ours
		NeuMF	LightGCN	SimpleX	NCL	MutiVAE	DiffRec	DSI-R	DSI-S	TIGER	LC-Rec	
Beauty	R@5	0.0447	0.0649	0.0551	<u>0.0650</u>	0.0530	0.0524	0.0128	0.0451	0.0519	0.0492	0.0667 *
	R@10	0.0653	<u>0.0952</u>	0.0831	0.0940	0.0776	0.0741	0.0228	0.0705	0.0799	0.0770	0.0993 *
	R@20	0.0889	0.1314	0.1193	<u>0.1320</u>	0.1093	0.1016	0.0360	0.1018	0.1154	0.1104	0.1371 *
	N@5	0.0315	<u>0.0450</u>	0.0377	0.0452	0.0362	0.0378	0.0084	0.0305	0.0350	0.0326	0.0449
	N@10	0.0383	<u>0.0549</u>	0.0469	0.0547	0.0443	0.0450	0.0117	0.0385	0.0443	0.0415	0.0556 *
	N@20	0.0445	0.0643	0.0563	<u>0.0646</u>	0.0526	0.0521	0.0151	0.0470	0.0534	0.0499	0.0654 *
Sports	R@5	0.0206	0.0418	0.0355	<u>0.0427</u>	0.0314	0.0273	0.0117	0.0320	0.0374	0.0397	0.0442 *
	R@10	0.0321	0.0623	0.0557	<u>0.0631</u>	0.0476	0.0403	0.0178	0.0497	0.0572	0.0617	0.0660 *
	R@20	0.0471	0.0901	0.0836	0.0908	0.0713	0.0569	0.0284	0.0766	0.0881	<u>0.0931</u>	0.0964 *
	N@5	0.0140	0.0288	0.0240	<u>0.0294</u>	0.0208	0.0193	0.0079	0.0225	0.0249	0.0264	0.0294 *
	N@10	0.0177	0.0355	0.0306	<u>0.0359</u>	0.0261	0.0235	0.0099	0.0284	0.0313	0.0335	0.0364 *
	N@20	0.0215	0.0426	0.0377	<u>0.0431</u>	0.0321	0.0278	0.0126	0.0350	0.0392	0.0413	0.0442 *
Phone	R@5	0.0410	0.0713	0.0643	<u>0.0717</u>	0.0569	0.0470	0.0187	0.0412	0.0601	0.0615	0.0745 *
	R@10	0.0603	<u>0.1052</u>	0.0976	0.1043	0.0855	0.0668	0.0341	0.0625	0.0895	0.0919	0.1121 *
	R@20	0.0871	<u>0.1487</u>	0.1420	0.1481	0.1233	0.0928	0.0564	0.0966	0.1299	0.1354	0.1587 *
	N@5	0.0282	0.0481	0.0423	<u>0.0486</u>	0.0378	0.0315	0.0121	0.0282	0.0403	0.0408	0.0490 *
	N@10	0.0344	0.0590	0.0530	<u>0.0593</u>	0.0470	0.0379	0.0170	0.0347	0.0498	0.0506	0.0611 *
	N@20	0.0412	0.0700	0.0643	<u>0.0704</u>	0.0566	0.0445	0.0225	0.0431	0.0600	0.0615	0.0729 *
Recipe	R@5	0.0118	0.0188	0.0114	<u>0.0192</u>	0.0167	0.0142	0.0142	0.0157	0.0168	0.0174	0.0198 *
	R@10	0.0210	0.0296	0.0202	<u>0.0298</u>	0.0285	0.0235	0.0248	0.0270	0.0292	0.0289	0.0306 *
	R@20	0.0339	0.0454	0.0328	0.0459	0.0462	0.0343	0.0403	0.0436	<u>0.0464</u>	0.0454	0.0482 *
	N@5	0.0088	<u>0.0149</u>	0.0093	<u>0.0149</u>	0.0128	0.0105	0.0107	0.0122	0.0137	0.0138	0.0151 *
	N@10	0.0119	<u>0.0182</u>	0.0122	<u>0.0182</u>	0.0167	0.0135	0.0141	0.0158	0.0176	0.0175	0.0185 *
	N@20	0.0154	0.0223	0.0156	<u>0.0224</u>	0.0214	0.0165	0.0182	0.0202	0.0221	0.0218	0.0232 *

5.5.1 Comparison on whole users. Table 2 shows the performance comparison of overall users. From these results, we make the following observations. Firstly, ColaRec achieves the best recommendation performance on all datasets except for NDCG@5 in Beauty, which achieves comparable scores with the best NCL baseline. In particular, ColaRec consistently outperforms previous CF-based and generative models in Recall@20, achieving a relative improvement of 3.87%, 3.54%, 6.72% and 3.88% on Beauty, Sports, Phone and Recipe, respectively. These results demonstrate the effectiveness of ColaRec and its generalization across different domains.

Secondly, compared with DSI-R and DSI-S, which directly adapt generative retrieval methods to the recommendation task, ColaRec achieves a notable 47.89%, 38.13%, 80.82% and 26.11% relative improvement in terms of Recall@5 on four datasets respectively, demonstrating the effectiveness of the proposed ColaRec. Such results also demonstrate that naively transferring the generative retrieval methods for recommendation cannot achieve satisfying results. Furthermore, our method consistently outperforms existing representative generative methods TIGER and LC-Rec. For TIGER, the reason may be that it only considers item content information but the collaborative signals are overlooked without an explicit alignment process. While for LC-Rec, the recommendation is conducted through language modeling and generation under designed prompts, which is not fully aligned with the recommendation task, leading to sub-optimal results.

Lastly, existing generative methods (e.g., DiffRec and TIGER), while exhibiting promising results in some specific scenarios [30, 40, 50], still underperform the strong CF-based methods (e.g., NCL) in the general recommendation task. In contrast, ColaRec achieves competitive results compared to these CF methods on all datasets, demonstrating the potential of infusing content information for collaborative generation of recommender systems.

5.5.2 Comparison on long-tail users. We also conduct experiments to verify the recommendation performance of ColaRec on long-tail users with sparse interactions. In this experiment, the ratio between head users and long-tail users is set as 20%:80%. Table 3 reports the results of Recall on four datasets. Results of NDCG show similar trends and are omitted due to the reason of space. We can see that ColaRec significantly outperforms all baselines when generating recommendation for long-tail users. The reason is that ColaRec models both user-item interactions and item content information. Given the long-tail users with less interaction knowledge, ColaRec gains better performance with the help of the content information.

To conclude, the proposed ColaRec is effective to yield better performance compared with existing baselines. This improvement is more significant on long-tail users.

Table 3: Performance comparison of long-tail users. The best and the second-best scores are marked in bold and underlined fonts, respectively. ** denotes the improvements are significant with p-value < 0.05. R stands for Recall.

Datasets	Metric	CF-based Methods				Generative Models for Recommendation						Ours
		NeuMF	LightGCN	SimpleX	NCL	MutiVAE	DiffRec	DSI-R	DSI-S	TIGER	LC-Rec	
Beauty	R@5	0.0416	0.0636	0.0555	<u>0.0639</u>	0.0510	0.0464	0.0131	0.0415	0.0487	0.0492	0.0660 **
	R@10	0.0604	<u>0.0922</u>	0.0825	0.0907	0.0742	0.0662	0.0228	0.0653	0.0745	0.0772	0.0975 **
	R@20	0.0817	0.1253	0.1160	<u>0.1264</u>	0.1039	0.0917	0.0354	0.0940	0.1084	0.1107	0.1327 **
Sports	R@5	0.0209	0.0433	0.0355	<u>0.0440</u>	0.0329	0.0267	0.0116	0.0307	0.0380	0.0397	0.0456 **
	R@10	0.0317	0.0639	0.0562	<u>0.0645</u>	0.0495	0.0394	0.0170	0.0472	0.0581	0.0617	0.0674 **
	R@20	0.0468	0.0904	0.0836	0.0908	0.0725	0.0553	0.0273	0.0728	0.0882	<u>0.0929</u>	0.0976 **
Phone	R@5	0.0405	0.0723	0.0660	<u>0.0727</u>	0.0571	0.0451	0.0206	0.0404	0.0602	0.0612	0.0756 **
	R@10	0.0590	<u>0.1054</u>	0.0986	0.1043	0.0861	0.0641	0.0371	0.0623	0.0898	0.0918	0.1131 **
	R@20	0.0855	<u>0.1482</u>	0.1418	0.1473	0.1228	0.0899	0.0600	0.0939	0.1293	0.1353	0.1590 **
Recipe	R@5	0.0128	0.0204	0.0121	<u>0.0210</u>	0.0182	0.0172	0.0157	0.0171	0.0181	0.0189	0.0219 **
	R@10	0.0229	0.0320	0.0212	<u>0.0322</u>	0.0309	0.0269	0.0274	0.0295	0.0316	0.0313	0.0334 **
	R@20	0.0371	0.0487	0.0343	0.0490	0.0499	0.0412	0.0443	0.0475	<u>0.0504</u>	0.0493	0.0528 **

Table 4: Effect of multi-task learning and different types of GIDs. The best scores are marked in bold.

	Beauty		Sports		Phone		Recipe	
	Recall@5	NDCG@5	Recall@5	NDCG@5	Recall@5	NDCG@5	Recall@5	NDCG@5
ColaRec	0.0667	0.0449	0.0442	0.0294	0.0745	0.0490	0.0198	0.0151
(1) w/o textual content	0.0527	0.0346	0.0364	0.0239	0.0636	0.0426	0.0181	0.0141
(2) w/o indexing	0.0637	0.0428	0.0422	0.0278	0.0728	0.0487	0.0179	0.0142
(3) w/o \mathcal{L}_{bpr}	0.0612	0.0412	0.0424	0.0282	0.0719	0.0486	0.0184	0.0140
(4) w/o \mathcal{L}_c	0.0657	0.0434	0.0422	0.0279	0.0731	0.0485	0.0188	0.0145
(1) <i>iad</i>	0.0658	0.0437	0.0428	0.0285	0.0719	0.0474	0.0189	0.0145
(2) <i>Random</i>	0.0600	0.0401	0.0411	0.0272	0.0667	0.0443	0.0190	0.0149
(3) <i>Content</i>	0.0662	0.0440	0.0423	0.0278	0.0716	0.0477	0.0183	0.0141

5.6 Ablation Study (RQ2)

In this section, we conduct ablation studies to analyse the effectiveness of each component in ColaRec. We implement four ablative variants of ColaRec, including: (1) *w/o textual content* deletes all textual content information in the model input; (2) *w/o indexing* removes the item-item indexing task; (3) *w/o \mathcal{L}_{bpr}* removes the ranking loss \mathcal{L}_{bpr} ; and (4) *w/o \mathcal{L}_c* removes the contrastive loss \mathcal{L}_c .

The results are shown in the upper part of Table 4. These results clearly indicate that the removal of any component from our proposed method results in a noticeable decline in overall performance. From the results of variant (1), we see that removing textual content information leads to the significant performance downgrade. Specifically, a 20.99%, 17.65%, 14.63% and 8.59% drop in Recall@5 is observed across the four datasets, respectively. This suggests the importance of incorporating textual content information to enhance the models' understanding of items. Furthermore, the variant (3), which is trained without \mathcal{L}_{bpr} , exhibits a notable performance decrease compared to ColaRec. This highlights the effectiveness of the pairwise ranking objective, which focuses on the relative

prioritization of positive items within the generative recommendation paradigm. Besides, the results of variant (2) and (4) verify the effectiveness of the proposed item-item indexing task and the contrastive objective \mathcal{L}_c . A consistent performance reduction is observed on the four datasets when either of the two techniques is removed. This suggests that aligning item content information with user-item collaborative signals not only facilitates mutual reinforcement but also enables the learning of more comprehensive and effective representations.

In conclusion, each component of ColaRec is essential to improve the recommendation performance.

5.7 GID Investigation (RQ3)

The construction of GIDs plays a crucial role in generative recommendation as it defines the model's search space for generation. To answer RQ3, we conduct two analytical experiments about GIDs: (1) an analysis of different GID types (e.g., the single token *iad*, the random assigned GID, and the content-based GID), and (2) an

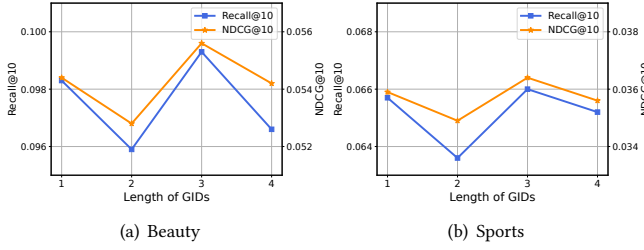


Figure 4: Impact of the length of GIDs.

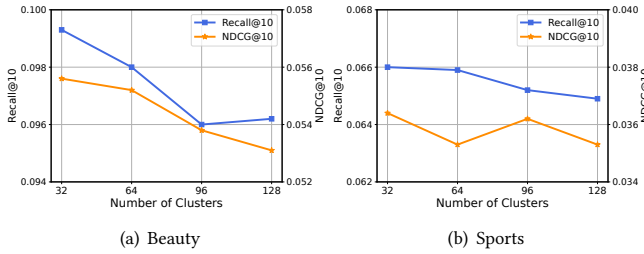


Figure 5: Impact of the number of clusters.

analysis of hyper-parameters when constructing GIDs (e.g., the length and the K -means cluster number).

5.7.1 Effect of different GID types. To further evaluate our GID generation strategy, we conduct an ablation study comparing it with three techniques: *iad*-based GID, *Random* GID, and *Content* GID. Specifically, the *iad*-based GID assigns a unique single token to each item to represent it using the corresponding vector in the item embedding matrix. The *Random* GID assigns a random string to each item as the identifier, without considering any prior knowledge. The *Content* GID represents the GID constructed from item content information. Unlike the collaborative GID used in ColaRec, the content-based GID employs a hierarchical K -means clustering algorithm [47] to cluster items based on their textual representation derived from a pretrained BERT model. To make a fair comparison with ColaRec, the length and the codebook size of both *Random* GIDs and *Content* GIDs are identical to those in ColaRec.

The bottom part of Table 4 details our results on the four datasets, where ColaRec attains the best performance. This indicates that our GID construction method, based on collaborative signals, is highly effective for generative recommendation. The improvement over the *Content* GID highlights the effectiveness of collaborative signals in recommendation tasks. Furthermore, ColaRec’s superior performance compared to the *iad* method demonstrates the benefit of explicitly introducing item correlations into item GIDs. In addition, the *Random* method leads to the lowest performance as the random string could further introduce noisy signals in the learning process. These results illustrate the importance of constructing effective GIDs for generative recommenders.

5.7.2 Impact of GID Hyper-parameters. In this section, we investigate the impact of hyper-parameters l and K in the GID construction

process. Figure 4 shows the results of ColaRec with different GID lengths l , ranging from 1 to 4. Note that to include the whole item set, shorter GIDs indicate larger search spaces for each GID position, and vice versa. We can see that as l varies, the recommendation performance exhibits some fluctuations. For the performance drop from $l = 1$ to $l = 2$ in Beauty and Sports, the reason is that compared with single token GIDs, GIDs with $l = 2$ increase one decoding step but the search space for each step is still large, increasing the search difficulty. When $l = 3$, ColaRec achieves a better trade-off between decoding steps and the search space for each step, leading to the best performance in most cases. When $l = 4$, a too long GID means more autoregressive decoding steps in model generation, which increases generation difficulty and inference latency. Therefore, we have selected $l = 3$ as the default setting in this paper.

To assess how the number of clusters K affects performance, we fix the GID length l as $l = 3$ and vary K with values 32, 64, 96, and 128. As shown in Figure 5, a higher K typically results in a slight decrease in overall performance. The decrease in Beauty is more notable. The reason is that a higher K indicates a larger search space in the decoding process, and thus increases the generation difficulty. To this end, choosing a suitable clustering number according to the number of items is important for generative recommendation. The principle is that K needs to be large enough to encode the whole item set, after that K needs to be controlled to limit the search space.

6 CONCLUSIONS AND FUTURE WORK

This paper proposes ColaRec, a novel framework to conduct content-based collaborative generation for recommender systems. As an end-to-end generative recommender, ColaRec effectively integrates both item content information and user-item collaborative signals within a unified framework through a generative model. In addition, an auxiliary item indexing task and a contrastive loss are proposed to further align the model’s representations of item content and user-item collaborative signals. We have conducted extensive experiments on four real-world datasets and empirical results demonstrate the effectiveness of ColaRec.

In the future, we intend to investigate more methods to construct generative identifiers (GIDs) and adopt more effective approaches to better align content information and collaborative signals. Negative sampling for generative recommendation is also one of our future works, either using GIDs to sample more informative negative samples or utilizing generative models to generate synthetic negative instances. Besides, how to improve model efficiency for generative recommendation is also a potential research direction.

ACKNOWLEDGEMENTS

This work was supported by the Tencent WeChat Rhino-Bird Focused Research Program (WYG-FR-2023-07), the Natural Science Foundation of China (62202271, 62372275, 62272274, T2293773, 62102234, 62072279), the National Key R&D Program of China with grant No.2022YFC3303004, the Natural Science Foundation of Shandong Province (ZR2021QF129) and the Young Elite Scientists Sponsorship Program by CAST (2023QNRC001). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

- [1] Keqin Bao, Jizhi Zhang, Wenjie Wang, Yang Zhang, Zhengyi Yang, Yancheng Luo, Fuli Feng, Xiangnan He, and Qi Tian. 2023. A bi-step grounding paradigm for large language models in recommendation systems. *arXiv preprint arXiv:2308.08434* (2023).
- [2] Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263* (2017).
- [3] Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive search engines: Generating substrings as document identifiers. *Advances in Neural Information Processing Systems* 35 (2022), 31668–31683.
- [4] Zefeng Cai and Zerui Cai. 2022. PEVAE: A Hierarchical VAE for Personalized Explainable Recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*. ACM, 692–702. <https://doi.org/10.1145/3477495.3532039>
- [5] Hanqun Cao, Cheng Tan, Zhangyang Gao, Yilun Xu, Guangyong Chen, Pheng-Ann Heng, and Stan Z Li. 2022. A survey on generative diffusion model. *arXiv preprint arXiv:2209.02646* (2022).
- [6] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive Entity Retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. <https://openreview.net/forum?id=5k8f6U39V>
- [7] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Kaijie Zhu, Hao Chen, Linyi Yang, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2023. A survey on evaluation of large language models. *arXiv preprint arXiv:2307.03109* (2023).
- [8] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural Attentional Rating Regression with Review-level Explanations. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. ACM, 1583–1592. <https://doi.org/10.1145/3178876.3186070>
- [9] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sen Gupta, and Anil A Bharath. 2018. Generative adversarial networks: An overview. *IEEE signal processing magazine* 35, 1 (2018), 53–65.
- [10] Wenqi Fan, Zihuai Zhao, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Jiliang Tang, and Qing Li. 2023. Recommender systems in the era of large language models (llms). *arXiv preprint arXiv:2307.02046* (2023).
- [11] Jingyue Gao, Yang Lin, Yasha Wang, Xiting Wang, Zhao Yang, Yuanduo He, and Xu Chu. 2020. Set-Sequence-Graph: A Multi-View Approach Towards Exploiting Reviews for Recommendation. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*. ACM, 395–404. <https://doi.org/10.1145/3340531.3411939>
- [12] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as Language Processing (RLP): A Unified Pretrain, Personalized Prompt & Predict Paradigm (P5). In *RecSys '22: Sixteenth ACM Conference on Recommender Systems, Seattle, WA, USA, September 18 - 23, 2022*. ACM, 299–315. <https://doi.org/10.1145/3523227.3546767>
- [13] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*. 2672–2680. <https://proceedings.neurips.cc/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html>
- [14] Guibing Guo, Huan Zhou, Bowei Chen, Zhirong Liu, Xiao Xu, Xu Chen, Zhenhua Dong, and Xiuqiang He. 2022. IPGAN: Generating Informative Item Pairs by Adversarial Sampling. *IEEE Trans. Neural Networks Learn. Syst.* 33, 2 (2022), 694–706. <https://doi.org/10.1109/TNNLS.2020.3028572>
- [15] Ruining He and Julian McAuley. 2016. VBPR: visual bayesian personalized ranking from implicit feedback. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 30.
- [16] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 639–648.
- [17] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial Personalized Ranking for Recommendation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*. ACM, 355–364. <https://doi.org/10.1145/3209978.3209981>
- [18] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*. 173–182.
- [19] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2016. beta-va: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations*.
- [20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. <https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html>
- [21] Yujing Hu, Qing Da, Anxiang Zeng, Yang Yu, and Yinghui Xu. 2018. Reinforcement learning to rank in e-commerce search engine: Formalization, analysis, and application. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 368–377.
- [22] 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*. ACM, 195–204. <https://doi.org/10.1145/3624918.3625339>
- [23] 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>
- [24] Tero Karras, Samuli Laine, and Timo Aila. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 4401–4410.
- [25] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. <http://arxiv.org/abs/1312.6114>
- [26] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.
- [27] 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*. ACM, 1258–1267. <https://doi.org/10.1145/3580305.3599519>
- [28] Jiming Li, Wentao Zhang, Tian Wang, Guanglei Xiong, Alan Lu, and Gerard Medioni. 2023. GPT4Rec: A generative framework for personalized recommendation and user interests interpretation. *arXiv preprint arXiv:2304.03879* (2023).
- [29] Yongqi Li, Xinyu Lin, Wenjie Wang, Fuli Feng, Liang Pang, Wenjie Li, Liqiang Nie, Xiangnan He, and Tat-Seng Chua. 2024. A Survey of Generative Search and Recommendation in the Era of Large Language Models. *arXiv preprint arXiv:2404.16924* (2024).
- [30] Zihao Li, Aixin Sun, and Chenliang Li. 2023. DiffuRec: A Diffusion Model for Sequential Recommendation. *arXiv preprint arXiv:2304.00686* (2023).
- [31] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. ACM, 689–698. <https://doi.org/10.1145/3178876.3186150>
- [32] Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2023. Llora: Aligning large language models with sequential recommenders. *arXiv preprint arXiv:2312.02445* (2023).
- [33] Xinyu Lin, Wenjie Wang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2023. A multi-facet paradigm to bridge large language model and recommendation. *arXiv preprint arXiv:2310.06491* (2023).
- [34] Zihan Lin, Changxin Tian, Yupeng Hou, and Wayne Xin Zhao. 2022. Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In *Proceedings of the ACM Web Conference 2022*. 2320–2329.
- [35] Sichun Luo, Yuxuan Yao, Bowei He, Yinya Huang, Aojun Zhou, Xinyi Zhang, Yuanzhang Xiao, Mingjie Zhan, and Linqi Song. 2024. Integrating Large Language Models into Recommendation via Mutual Augmentation and Adaptive Aggregation. *arXiv preprint arXiv:2401.13870* (2024).
- [36] Kelong Mao, Jieming Zhu, Jinpeng Wang, Quanyu Dai, Zhenhua Dong, Xi Xiao, and Xiuqiang He. 2021. SimpleX: A simple and strong baseline for collaborative filtering. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 1243–1252.
- [37] 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>
- [38] Tao Qi, Fangzhao Wu, Chuhan Wu, Yongfeng Huang, and Xing Xie. 2020. Privacy-Preserving News Recommendation Model Learning. In *EMNLP (Findings) (Findings of ACL, Vol. EMNLP 2020)*. Association for Computational Linguistics, 1423–1432.
- [39] 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>
- [40] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan H Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q Tran, Jonah Samost, et al. 2023. Recommender Systems with Generative Retrieval. *arXiv preprint arXiv:2305.05065* (2023).
- [41] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International conference on data mining*. IEEE, 995–1000.
- [42] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

- [43] Ilya Shenbin, Anton Alekseev, Elena Tutubalina, Valentin Malykh, and Sergey I. Nikolenko. 2020. RecVAE: A New Variational Autoencoder for Top-N Recommendations with Implicit Feedback. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining*, Houston, TX, USA, February 3–7, 2020. ACM, 528–536. <https://doi.org/10.1145/3336191.3371831>
- [44] 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. *arXiv preprint arXiv:2309.13375* (2023).
- [45] 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*.
- [46] Juntao Tan, Shuyuan Xu, Wenyue Hua, Yingqiang Ge, Zelong Li, and Yongfeng Zhang. 2024. Towards LLM-RecSys Alignment with Textual ID Learning. *arXiv preprint arXiv:2403.19021* (2024).
- [47] Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. 2022. Transformer memory as a differentiable search index. *Advances in Neural Information Processing Systems* 35 (2022), 21831–21843.
- [48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).
- [49] Wenjie Wang, Xinyu Lin, Fuli Feng, Xiangnan He, and Tat-Seng Chua. 2023. Generative recommendation: Towards next-generation recommender paradigm. *arXiv preprint arXiv:2304.03516* (2023).
- [50] Wenjie Wang, Yiyang Xu, Fuli Feng, Xinyu Lin, Xiangnan He, and Tat-Seng Chua. 2023. Diffusion Recommender Model. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23–27, 2023*. ACM, 832–841. <https://doi.org/10.1145/3539618.3591663>
- [51] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*. 165–174.
- [52] 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 *NeurIPS*. http://papers.nips.cc/paper_files/paper/2022/hash/a46156bd3579c3b268108ea6aca71d13-Abstract-Conference.html
- [53] Zhidan Wang, Wenwen Ye, Xu Chen, Wenqiang Zhang, Zhenlei Wang, Lixin Zou, and Weidong Liu. 2022. Generative Session-based Recommendation. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25–29, 2022*. ACM, 2227–2235. <https://doi.org/10.1145/3485447.3512095>
- [54] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, and Tat-Seng Chua. 2020. Graph-Refined Convolutional Network for Multimedia Recommendation with Implicit Feedback. In *MM '20: The 28th ACM International Conference on Multimedia, Virtual Event / Seattle, WA, USA, October 12–16, 2020*. ACM, 3541–3549. <https://doi.org/10.1145/3394171.3413556>
- [55] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. 2019. MMGCN: Multi-modal Graph Convolution Network for Personalized Recommendation of Micro-video. In *Proceedings of the 27th ACM International Conference on Multimedia, MM 2019, Nice, France, October 21–25, 2019*. ACM, 1437–1445. <https://doi.org/10.1145/3343031.3351034>
- [56] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, and Alexander J. Smola. 2017. Joint Training of Ratings and Reviews with Recurrent Recommender Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Workshop Track Proceedings*. OpenReview.net. <https://openreview.net/forum?id=Bkv9FyHYx>
- [57] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. 2023. Diffusion models: A comprehensive survey of methods and applications. *Comput. Surveys* 56, 4 (2023), 1–39.
- [58] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. 2019. A Simple Convolutional Generative Network for Next Item Recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 582–590.
- [59] 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>
- [60] Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2023. Recommendation as instruction following: A large language model empowered recommendation approach. *arXiv preprint arXiv:2305.07001* (2023).
- [61] Yang Zhang, Fuli Feng, Jizhi Zhang, Keqin Bao, Qifan Wang, and Xiangnan He. 2023. Collm: Integrating collaborative embeddings into large language models for recommendation. *arXiv preprint arXiv:2310.19488* (2023).
- [62] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).
- [63] Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Adapting large language models by integrating collaborative semantics for recommendation. *arXiv preprint arXiv:2311.09049* (2023).
- [64] Lei Zheng, Vahid Noroozi, and Philip S. Yu. 2017. Joint Deep Modeling of Users and Items Using Reviews for Recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6–10, 2017*. ACM, 425–434. <https://doi.org/10.1145/3018661.3018665>
- [65] Yaochen Zhu, Liang Wu, Qi Guo, Liangjie Hong, and Jundong Li. 2023. Collaborative large language model for recommender systems. *arXiv preprint arXiv:2311.01343* (2023).