

# MMGRec: Multimodal Generative Recommendation with Transformer Model

Han Liu  
Shandong University  
China  
hanliu.sdu@gmail.com

Yinwei Wei  
Monash University  
Australia  
weiyinwei@hotmail.com

Xuemeng Song  
Shandong University  
China  
sxmustc@gmail.com

Weili Guan  
Harbin Institute of Technology  
(Shenzhen)  
China  
honeyguan@gmail.com

Yuan-Fang Li  
Monash University  
Australia  
yuanfang.li@monash.edu

Liqiang Nie  
Harbin Institute of Technology  
(Shenzhen)  
China  
nieliqiang@gmail.com

## ABSTRACT

Multimodal recommendation aims to recommend user-preferred candidates based on her/his historically interacted items and associated multimodal information. Previous studies commonly employ an **embed-and-retrieve** paradigm: learning user and item representations in the same embedding space, then retrieving similar candidate items for a user via embedding inner product. However, this paradigm suffers from inference cost, interaction modeling, and false-negative issues. Toward this end, we propose a new **MMGRec** model to introduce a **generative** paradigm into multimodal recommendation. Specifically, we first devise a hierarchical quantization method Graph RQ-VAE to assign Rec-ID for each item from its multimodal and CF information. Consisting of a tuple of semantically meaningful tokens, Rec-ID serves as the unique identifier of each item. Afterward, we train a Transformer-based recommender to generate the Rec-IDs of user-preferred items based on historical interaction sequences. The generative paradigm is qualified since this model systematically predicts the tuple of tokens identifying the recommended item in an autoregressive manner. Moreover, a relation-aware self-attention mechanism is devised for the Transformer to handle non-sequential interaction sequences, which explores the element pairwise relation to replace absolute positional encoding. Extensive experiments evaluate MMGRec’s effectiveness compared with state-of-the-art methods.

## KEYWORDS

Recommender Systems, Multimodal Recommendation, Generative Recommendation, Transformer.

## 1 INTRODUCTION

With the popularity of multimedia-centric scenarios like social media and micro-video websites, multimodal recommendation systems are gaining widespread attention and adoption. These systems generally follow the **embed-and-retrieve** paradigm. The paradigm first incorporates multimodal information with collaborative filtering (CF) [26] to learn representations for users and items in the embedding stage, and then locates the preferred items for users by measuring inner product similarity in the retrieval stage. Relevant research primarily focuses on enhancing representation learning in the embedding stage. For example, ACF [6] introduces a hierarchical attention mechanism to select informative modality content, while

MMGCN [37] incorporates Graph Convolution Network (GCN) to achieve modal information aggregation and propagation.

Despite the remarkable performance, existing research overlooks the inherent issues caused by similarity calculation in the retrieval stage. First, similarity calculation becomes overwhelming. With  $I$  items and  $U$  users, capturing the top similar  $K$  items for each user incurs time complexity  $O(UID + UI\log K)$ , where  $D$  is the dimension of user/item representations. As users/items increase, the time consumption significantly grows, affecting recommendation efficiency. Secondly, linear inner product inadequately models user-item interactions’ intricate structure [15]. Although some studies model non-linear interaction with neural networks [15] and metric learning [17], they sacrifice similarity calculation speed, exacerbating efficiency issues. Lastly, the paradigm assumes that interacted items are inherently closer to user preference than uninteracted ones [25], leading to a false-negative issue [34] since the absence of interaction does not necessarily imply dislike [41].

To address the issues, recent work [24] inherits the Differentiable Search Index algorithm [30] and presents a **generative** paradigm-based recommendation method, which aims to directly generate the list of user-preferred items without similarity calculation. To achieve this goal, two components, including **Semantic ID Assignment** and **Semantic ID Generation**, are designed to answer the questions: *how to represent an item with a semantic ID* and *how to predict the semantic IDs of preferred items for a user*. Initially, the method assigns each item a unique semantic ID — a sequence of semantic tokens learned from its textual description. Then, a Transformer [31] model is trained as the recommendation agent to predict the semantic ID of the next item based on the user’s chronological interaction sequence. Each ID token is yielded with autoregressive decoding, and multiple item IDs are generated with beam search [29]. However, for multimodal recommendation, there are still two technical challenges untouched in the two components:

- **Semantic ID Assignment:** Existing ID assignment approaches employ unimodal semantic clustering, inevitably leading to ID collision among similar items. Although random distinction [24, 30] is introduced to address this, such randomness information in ID is difficult to learn with Transformer, negatively affecting the performance of generative recommendation.
- **Semantic ID Generation:** The position and order information in historical interaction sequences is important for Transformer

to understand user behavior and preference. However, this information often lacks explicit references (e.g., timestamp, rating) and varies across different users. Therefore, it is challenging to make the Transformer aware of each item’s position within the historical interaction sequence.

To overcome the challenges, we design a new ID structure — **Rec-ID**, and present a Multimodal Generative Recommendation (**MMGRec**) framework for Rec-ID assignment and generation. Specifically, Rec-ID comprises a sequence of semantic tokens followed by a popularity token, leveraging the correlation between item popularity and semantic information. For Rec-ID assignment, we devise a Graph Residual-Quantized Variational AutoEncoder (Graph RQ-VAE) to fuse item multimodal information with CF information and quantize them into a sequence of semantic tokens. Then, we rank the items with identical semantic tokens by popularity and use their ranking index as the last token to avoid the collision problem. After that, for Rec-ID generation, we train a Transformer-based model to directly generate the preferred items’ Rec-IDs based on users’ historical interactions. To capture item position information in historical interaction sequences, we design a relation-aware self-attention mechanism for Transformer. By extending the original self-attention, this mechanism constructs user-specific relations among items under the supervision of user parameters. We conduct extensive experiments on three public datasets to demonstrate the rationality and effectiveness of the proposed MMGRec.

Overall, the main contributions of our work are three-fold:

- We propose MMGRec, a novel Transformer-based recommendation framework that consists of Rec-ID assignment and generation. This is the first effort known to us to introduce the generative paradigm into multimodal recommendation.
- Technically, we design a multimodal information quantization algorithm Graph RQ-VAE for Rec-ID assignment, and a relation-aware self-attention mechanism within Transformer for Rec-ID generation.
- We conduct empirical studies on three real-world datasets. Extensive results demonstrate that MMGRec achieves state-of-the-art performance with promising inference efficiency.

## 2 PRELIMINARY

This work is inspired by TIGER (short for *Transformer Index for Generative Recommenders*), a recent generative model for sequential recommendation [24]. Its foundation lies in a novel semantic ID for item identification, consisting of an ordered semantic token tuple:

$$(c_1, \dots, c_{M-1}) = \phi(\mathbf{f}^t), \quad (1)$$

where  $\mathbf{f}^t$  is the item’s textual feature converted from descriptions using Sentence-T5 [23].  $\phi(\cdot)$  indicates the vector quantization algorithm [42] converting  $\mathbf{f}^t$  into a tuple of  $M - 1$  semantic tokens.

To address the collision problem that items with similar textual features tend to be assigned the same tokens, the method inserts an extra token at the end of the semantic tokens to ensure the uniqueness of IDs. Specifically, given  $K$  items with the same tokens  $(c_1, \dots, c_{M-1})$ , it randomly sorts the items and sets their corresponding indices in the sorted order as the extra tokens:

$$(p_1, p_2, \dots, p_K) = \text{argsort}_{\text{rand}}(\text{item}_1, \text{item}_2, \dots, \text{item}_K), \quad (2)$$

where  $\text{argsort}_{\text{rand}}(\cdot)$  is the random ranking function to sort  $K$  items.  $p_k \in \{1, 2, \dots, K\}$  denotes the index of the  $k$ -th item in the order. Then, after concatenating them at the end of semantic tokens, the semantic ID of item  $k$  can be obtained as:

$$S_k = (c_1, \dots, c_{M-1}, p_k). \quad (3)$$

With the obtained semantic ID, a Transformer-based sequence-to-sequence model is trained on the sequences of semantic IDs associated with items from a user’s chronological interactions. The objective is to predict the semantic ID of the next item after the sequence. This model autoregressively decodes the tokens of semantic ID identifying the next item, thereby qualifying as a generative recommendation.

However, this method has notable limitations. The random token lacks semantic meaning and is challenging to model using statistical machine learning techniques like Transformer. Additionally, the method struggles with the input of non-sequential interactions.

## 3 METHOD

In this section, we first elaborate on our designed MMGRec model according to its two components — Rec-ID assignment and Rec-ID generation. After that, we detail the model training and inference, followed by the discussion regarding the time complexity.

### 3.1 Rec-ID Definition

As the foundation of generative recommendation, we define a new item identifier Rec-ID equipped with two attributes: 1) **Semantics**, where the Rec-ID condenses the semantic information of the item, and 2) **Uniqueness**, where the Rec-ID is capable of distinguishing each item from others.

Towards this end, Rec-ID is designed to consist of a sequence of semantic tokens and an extra token based on the item’s popularity. The motivation is that the popularity of an item is relevant to its semantic information. Hence, during the autoregressive generation phase, the last token (i.e., popularity token) can be predicted by jointly analyzing the semantic tokens generated before.

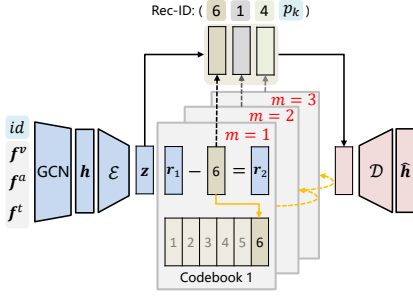
### 3.2 Rec-ID Assignment

According to the design of Rec-ID, we detail the Rec-ID assignment to describe how to allocate the semantic tokens and the popularity token for each item.

**3.2.1 Graph RQ-VAE.** To learn the semantic tokens for items, as shown in Figure 1, we devise a Graph RQ-VAE model to fuse their multimodal information and further quantize into codewords. Alongside multimodal (e.g., visual, acoustic, and textual) features, historical user-item interactions are fed to this model. These interactions not only aid in extracting relevant multimodal features for recommendation [35] but also reveal behavioral signals related to item popularity. In particular, a user-item bipartite graph is constructed, where nodes represent users and items, and edges denote interactions. To fuse these heterogeneous inputs into a unified representation, we initialize item  $i$  as follows:

$$\mathbf{h}_i^{(0)} = [\mathbf{h}_i^m \mathbf{h}_i^c], \mathbf{h}_i^m = [\mathbf{f}_i^v \mathbf{f}_i^a \mathbf{f}_i^t] \mathbf{W}, \quad (4)$$

where  $\mathbf{h}_i^m \in \mathbb{R}^D$  represents item  $i$ ’s multimodal embedding learned by concatenating its visual, acoustic, and textual features, i.e.,  $\mathbf{f}_i^v \in$



**Figure 1: An illustration of Graph RQ-VAE model architecture.**

$\mathbb{R}^{D^v}$ ,  $\mathbf{f}_i^a \in \mathbb{R}^{D^a}$ , and  $\mathbf{f}_i^t \in \mathbb{R}^{D^t}$ .  $D^v$ ,  $D^a$ , and  $D^t$  denote the dimensions of these modality features.  $\mathbf{W} \in \mathbb{R}^{(D^v+D^a+D^t) \times D}$  is a parameter matrix that maps multimodal features to the  $D$ -dimensional embedding space. Combining this multimodal embedding with a randomly initialized CF embedding  $\mathbf{h}_i^c \in \mathbb{R}^D$  yields the item representation  $\mathbf{h}_i^{(0)} \in \mathbb{R}^{2D}$ . For performing graph convolution operations on the bipartite graph, user representations are randomly initialized, such as  $\mathbf{h}_u^{(0)} \in \mathbb{R}^{2D}$  for user  $u$ .

Treating them as node embeddings at the 0-th layer, item  $i$ 's representation at the  $n$ -th GCN layer is obtained as:

$$\mathbf{h}_i^{(n)} = \text{LeakyReLU}(\mathbf{h}_i^{(n-1)} \mathbf{W}_1 + \frac{1}{|\mathcal{N}_i|} \sum_{u \in \mathcal{N}_i} \mathbf{h}_u^{(n-1)} \mathbf{W}_2), \quad (5)$$

where  $\mathcal{N}_i$  is the neighborhood set of item  $i$  (i.e., users whom item  $i$  directly interacted with), and LeakyReLU is the activation function. Analogously, the representation  $\mathbf{h}_u^{(n)}$  for user  $u$  is obtained by information propagation from its neighbor items. After stacked GCN layers, final representations are denoted as  $\mathbf{h}_u \in \mathbb{R}^D$  and  $\mathbf{h}_i \in \mathbb{R}^D$ . Note that we utilize the concise graph convolution operation [10], leaving complicated choices [21] for further exploration.

Based on the final item representations, we employ Residual-Quantized Variational AutoEncoder (RQ-VAE) [42] to generate semantic tokens of Rec-IDs. As a multi-stage vector quantizer, RQ-VAE can produce a tuple of codewords by quantizing residuals across hierarchical levels. As illustrated in Figure 1, we input the learned item representation into an encoder model:

$$\mathbf{z}_i = \mathcal{E}(\mathbf{h}_i), \quad (6)$$

where  $\mathbf{z}_i$  is the output latent vector and  $\mathcal{E}$  is the encoder, comprising a two-layer Multi-Layer Perceptron (MLP). At the first level, the initial residual is  $\mathbf{r}_{i,1} = \mathbf{z}_i$ . A codebook  $\mathbf{B}_1 = \{\mathbf{b}_l\}_{l=1}^L$  is employed, where  $\mathbf{b}_l$  denotes the parameter embedding and  $L$  represents the codebook size. Then,  $\mathbf{r}_{i,1}$  is quantized by mapping it to the closest embedding from this codebook:

$$c_{i,1} = \arg\min_l \|\mathbf{r}_{i,1} - \mathbf{b}_l\|_2^2, \quad (7)$$

where  $c_{i,1}$  is the index of the closest embedding, i.e., the first codeword. Recursively, for subsequent level  $m$ , the residual is defined as  $\mathbf{r}_{i,m} = \mathbf{r}_{i,m-1} - \mathbf{b}_{c_{i,m-1}}$ , and the codeword  $c_{i,m}$  is similarly computed using the level-specific codebook  $\mathbf{B}_m$ . By iteratively performing  $M-1$  operations, a sequence of codewords is generated as follows:

$$(c_{i,1}, \dots, c_{i,M-1}) = \text{RQ-VAE}(\mathbf{h}_i), \quad (8)$$

where  $c_{i,m}$  denotes the  $m$ -th codeword used as the  $m$ -th semantic token in Rec-ID of item  $i$ . Notably, the recursive approach approximates the input in a coarse-to-fine manner. Since each codeword is sourced from a distinct codebook, the capacity of Rec-IDs to uniquely represent items equals the product of all codebook sizes.

Upon the codewords, we use a two-layer MLP decoder  $\mathcal{D}$  to reconstruct the input  $\mathbf{h}_i$  as follows:

$$\hat{\mathbf{h}}_i = \mathcal{D}(\mathbf{z}_i + \text{sg}(\hat{\mathbf{z}}_i - \mathbf{z}_i)), \quad \hat{\mathbf{z}}_i = \sum_{m=1}^{M-1} \mathbf{b}_{c_{i,m}}, \quad (9)$$

where  $\hat{\mathbf{h}}_i$  denotes the decoder output, and  $\hat{\mathbf{z}}_i$  is the quantized representation of  $\mathbf{z}_i$ .  $\text{sg}(\cdot)$  represents the stop-gradient operation. The loss function for training Graph RQ-VAE is as follows:

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{\text{bpr}} + \mathcal{L}_{\text{rqvae}}, \quad \mathcal{L}_{\text{bpr}} = \sum_{(u,i,j) \in \mathcal{R}} -\ln \sigma(\mathbf{h}_u \mathbf{h}_i^\top - \mathbf{h}_u \mathbf{h}_j^\top), \\ \mathcal{L}_{\text{rqvae}} &= \sum_{i=1}^I \|\mathbf{h}_i - \hat{\mathbf{h}}_i\|^2 + \sum_{m=1}^{M-1} \|\text{sg}(\mathbf{r}_{i,m}) - \mathbf{b}_{c_{i,m}}\|^2 \\ &\quad + \beta \|\mathbf{r}_{i,m} - \text{sg}(\mathbf{b}_{c_{i,m}})\|^2. \end{aligned} \quad (10)$$

Here  $\mathcal{L}_{\text{bpr}}$  is the pairwise BPR [25] loss to learn user and item representations.  $\mathcal{R} = \{(u, i, j) | (u, i) \in \mathcal{R}^+, (u, j) \in \mathcal{R}^-\}$ , where  $\mathcal{R}^+$  indicates the observed interactions, and  $\mathcal{R}^-$  indicates the unobserved interactions.  $\sigma(\cdot)$  is the Sigmoid function. The loss function  $\mathcal{L}_{\text{rqvae}}$  facilitates the simultaneous training of the encoder-decoder and codebooks in RQ-VAE.  $I$  denotes the number of items. We alternatively optimize one of two sub-losses to learn model parameters.

**3.2.2 Handling Collisions.** To handle the collision problem, we expand a popularity token at the end of the semantic tokens to ensure Rec-IDs' uniqueness. In particular, given  $K$  items with the same tokens  $(c_1, \dots, c_{M-1})$ , we sort the items according to their popularity, i.e., the total number of interactions with users. The corresponding indices in the sorted order are used as their popularity tokens, formally,

$$(p_1, p_2, \dots, p_K) = \text{argsort}_{\text{pop}}(\text{item}_1, \text{item}_2, \dots, \text{item}_K), \quad (11)$$

where  $\text{argsort}_{\text{pop}}(\cdot)$  is the popularity ranking function to sort  $K$  items.  $p_k \in \{1, 2, \dots, K\}$  denotes the index of the  $k$ -th item in the order. Then, after concatenating them at the end of semantic tokens, the Rec-ID of item  $k$  can be obtained as:

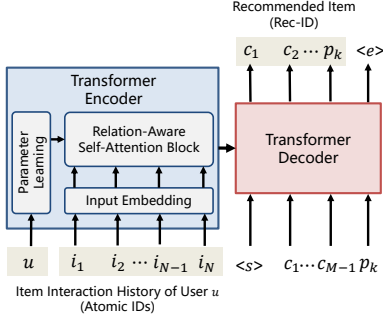
$$R_k = (c_1, \dots, c_{M-1}, p_k). \quad (12)$$

### 3.3 Rec-ID Generation

**3.3.1 Transformer Input Embedding.** To implement generative recommendation, a sequence-to-sequence Transformer model is exploited as a recommender. As shown in Figure 2, the model consists of stacked encoder and decoder layers. To adopt the model in recommendation tasks, we organize historically interacted items of each user as a sequence of tokens and feed them into the encoder.

Taking user  $u$  as an example, we gather her/his interacted items' atomic IDs together as the Transformer's input, and utilize the learned item representations for input embedding, formally,

$$[\mathbf{e}_1, \dots, \mathbf{e}_N] = [\mathbf{h}_{i_1}, \dots, \mathbf{h}_{i_N}], \quad (13)$$



**Figure 2: Schematic illustration of Transformer encoder-decoder setup for building our generative recommendation model.**

where  $\mathbf{e}_i \in \mathbb{R}^D$  denotes the embedding of the corresponding item in the sequence. It is worth noting that the input sequence length is uniformly set to  $N$ , filling the missing positions with padding token 0.

**3.3.2 Relation-Aware Self-Attention.** Following input embedding, positional encoding in Transformer is crucial for providing order information of the input sequence. Unlike sequential recommendation tasks, the input interacted item sequence is not well-ordered in our case, and its actual order implicitly and dynamically changes with different users. Thus, traditional absolute positional encoding is ineffective in this context. To overcome the problem, we develop a relation-aware self-attention mechanism for the Transformer. This mechanism explicitly models user-specific pairwise relations among input items based on user preferences, thereby encoding relative position information within interaction sequences.

Specifically, as shown in Figure 3, we extend the self-attention sublayer by incorporating user-specific parameters. First, we introduce the user-specific position encoder, which aims to encode the position information of each item from different users' perspectives:

$$\mathbf{p}_j^u = \mathbf{e}_j \mathbf{W}_u^V, \quad (14)$$

where  $\mathbf{W}_u^V$  represents the parameter matrix of user  $u$ 's position encoder and  $\mathbf{p}_j^u$  is the encoded vector of item  $j$ .

After obtaining the encoded vectors, we conduct the multi-head attention operations on the input sequence. In each head, we map the input element embeddings  $[\mathbf{e}_1, \dots, \mathbf{e}_N]$  into query, key, and value spaces, and compute their new embeddings  $[\mathbf{x}_1, \dots, \mathbf{x}_N]$  where  $\mathbf{x}_i \in \mathbb{R}^D$ :

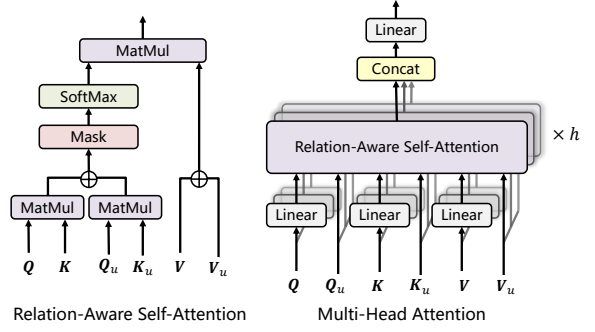
$$\mathbf{x}_i = \sum_{j=1}^N \alpha_{ij} (\mathbf{e}_j \mathbf{W}^V + \mathbf{p}_j^u), \quad (15)$$

where  $\mathbf{W}^V$  is the matrix mapping the elements to the value space.  $\alpha_{ij}$  is the weight coefficient computed with the Softmax function:

$$\alpha_{ij} = \frac{\exp(\epsilon_{ij})}{\sum_{n=1}^N \exp(\epsilon_{in})}, \quad (16)$$

wherein  $\epsilon_{ij}$  is calculated through a compatibility function that compares two input elements and considers their pairwise relation in the context of user  $u$ :

$$\epsilon_{ij} = \frac{(\mathbf{e}_i \mathbf{W}^Q)(\mathbf{e}_j \mathbf{W}^K)^\top + (\mathbf{e}_i \mathbf{W}_u^Q)(\mathbf{e}_j \mathbf{W}_u^K)^\top}{\sqrt{D}}, \quad (17)$$



**Figure 3: An illustration of relation-aware self-attention and its corresponding multi-head attention. Therein,  $Q = \mathbf{E}\mathbf{W}^Q$ ,  $Q_u = \mathbf{E}\mathbf{W}_u^Q$ , and the remaining parameters are computed analogically.**

where  $\mathbf{W}^Q$  and  $\mathbf{W}^K$  are parameter matrices mapping elements to the query and key spaces, respectively. Moreover, elements are also mapped into user-specific query and key spaces to compute the relation between each item pair.  $\mathbf{W}_u^Q$  and  $\mathbf{W}_u^K$  denote two user-specific parameter matrices for this space transition.

Considering the user number, adopting independent parameter matrices for each user will result in the issue of model parameter overload. Hence, we alternatively leverage user information to fine-tune a series of foundational parameters. Taking  $\mathbf{W}_u^V$  as an example, its acquisition is as follows:

$$\mathbf{W}_u^V = \text{MLP}(\mathbf{h}_u) \cdot \mathbf{U}^V, \quad (18)$$

where  $\mathbf{h}_u$  is the personal representation of user  $u$  learned in Graph RQ-VAE. It is projected through MLP into a scalar, measuring the bias introduced by user  $u$  on the foundational matrix parameter  $\mathbf{U}^V \in \mathbb{R}^{D \times D}$ . This strategy allows regulating the number of model parameters while maintaining user-specificity.  $\mathbf{W}_u^Q$  and  $\mathbf{W}_u^K$  can be obtained in the same way.

Notably, the relation-aware self-attention can be efficiently computed using the scaled dot product:

$$\mathbf{X} = \text{Softmax}\left(\frac{\mathbf{E}(\mathbf{W}^Q \mathbf{W}^{K^\top} + \mathbf{W}_u^Q \mathbf{W}_u^{K^\top}) \mathbf{E}^\top}{\sqrt{D}}\right) \mathbf{E}(\mathbf{W}^V + \mathbf{W}_u^V). \quad (19)$$

The parameter matrices  $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V \in \mathbb{R}^{D \times D}$  are unique to each attention head, contributing sufficient expressive capacity. While the user-specific parameters  $\mathbf{W}_u^Q, \mathbf{W}_u^K, \mathbf{W}_u^V \in \mathbb{R}^{D \times D}$  can be shared across all attention heads. To form the sublayer output, the outcomes from each head are concatenated, and a linear transformation is applied.

Overall, we use the relation-aware self-attention sublayer followed by a feed-forward sublayer as the Transformer encoder layer. Residual connections surround each sublayer, followed by layer normalization. As for the Transformer decoder, we maintain the default decoder layers and positional encoding, as the Transformer's output sequence is inherently ordered.

### 3.4 Training & Inference

During model training, each observed interaction  $(u, i)$  forms a training sample. We input the atomic ID-based interaction sequence of

user  $u$  into the Transformer encoder. To ensure generative recommendation, we deliberately exclude the ground-truth item  $i$  from the input sequence. Model parameters are learned by optimizing cross-entropy loss between Transformer decoder output and item  $i$ 's Rec-ID. This loss function is commonly used for Transformer, and the indices of Graph RQ-VAE codebooks serve as vocabulary.

In inference with a well-trained Transformer, results are output step by step, where each step's conditional probability distribution depends on the already generated results. To maximize the product of conditional probabilities for the entire output, we incorporate beam search [29] to preserve  $K$  sequences with the currently highest probabilities at each time step. The beam number is configured to  $K$  for top- $K$  recommendation. Through  $M$  steps of autoregressive decoding, beam search can ultimately generate  $K$  complete Rec-IDs. To prevent generating invalid Rec-IDs, we use constrained beam search [7] to limit the current token range based on prefixes.

### 3.5 Discussion

Moreover, we discuss the inference time complexity of MMGRec compared with the most efficient method of traditional paradigm, *i.e.*, Matrix Factorization (MF) [19]. Assuming the existence of  $I$  items, each represented by  $D$ -dimensional embeddings, MF requires a time complexity of  $O(ID)$  to complete all inner product computations and  $O(I\log K)$  to identify the top- $K$  recommendation for a single user. In contrast, MMGRec utilizes autoregressive decoding and beam search for inference. A unit decoding process with  $H$ -layer MMGRec requires  $O(H(M^2D + MD^2))$  in time complexity, where  $M$  represents the length of the output sequence (Rec-ID). Consequently, performing autoregressive decoding  $M$  times with  $K$  sequences incurs a time complexity of  $O(KH(M^3D + M^2D^2))$ . Additionally, a beam search with  $M$  steps introduces a time complexity of  $O(MKL\log K)$ , with  $L$  denoting the vocabulary size. For ease of comparison, we can express the overall complexities as  $O(KH(M^3 + M^2D)D)$  and  $O(MKL\log K)$ . Upon closer examination, it suggests that the generative paradigm may reduce the inference time in case  $I \gg KH(M^3 + M^2D) > MKL$ .

## 4 EXPERIMENTAL SETUP

### 4.1 Datasets & Evaluation Metrics

To evaluate the effectiveness of our proposed model, we conduct extensive experiments on three benchmark datasets: MovieLens<sup>1</sup>, TikTok<sup>2</sup>, and Kwai<sup>3</sup>, which are widely used in multimodal recommendation study. The dataset statistics are summarized in Table 1.

- **MovieLens**: This dataset was created for research by MMGCN authors. They collected movie trailers and descriptions based on the original dataset. Visual, acoustic, and textual features are extracted from frames, audio tracks, and descriptions using pre-trained models [1, 12, 16], respectively.
- **TikTok**: This dataset was released by the short-video sharing platform TikTok, including user-item interactions and multimodal features (visual, acoustic, textual) of items. According to the publisher's statement, the multimodal features are extracted from the raw data of short videos.

<sup>1</sup><https://movielens.org/>.

<sup>2</sup><https://www.tiktok.com/>.

<sup>3</sup><https://www.kwai.com/>.

**Table 1: Statistics of the evaluation datasets. ( $D^v$ ,  $D^a$ , and  $D^t$  denote the dimensions of visual, acoustic, and textual modality feature data, respectively.)**

Dataset	#Users	#Items	#Inter.	$D^v$	$D^a$	$D^t$
MovieLens	55,485	5,986	1,239,508	2048	128	100
TikTok	36,656	76,085	726,065	128	128	128
Kwai	7,010	86,483	298,492	2048	-	128

- **Kwai**: The dataset was released by the short-video sharing platform Kwai, comprising user behavioral records and short videos' multimodal features (excluding acoustic features). Likewise, the multimodal features are obtained by pre-trained extractors.

For each dataset, we randomly select 80% of the historical interactions per user to constitute the training set, 10% for validation, and the remaining 10% for testing. The validation and testing sets are used for hyper-parameter tuning and performance evaluation, respectively. We adopt Recall (R@K) and Normalized Discounted Cumulative Gain (N@K) as evaluation metrics for top- $K$  recommendation and preference ranking. Defaulting  $K = 10$ , we report the average metric value for all users in the testing set.

### 4.2 Baselines

To demonstrate the effectiveness, we compare our method with the following state-of-the-art baselines, briefly divided into CF-based (*i.e.*, GraphSAGE, NGCF, GAT, and LightGCN) and multimodal (*i.e.*, VBPR, MMGCN, GRCN, LATTICE, InvRL, and LightGT).

- **GraphSAGE** [10] passes information along the graph structure and aggregates them to update each node's representation.
- **NGCF** [33] encodes CF signals into representation learning by exploiting high-order connectivity from user-item interactions.
- **GAT** [32] automatically learns weights for each node's neighbors and alleviates noisy information to improve GCN performance.
- **LightGCN** [14] simplifies GCN components, utilizing a weighted sum aggregator as the graph convolution operation.
- **VBPR** [13] incorporates multimodal information with MF framework to predict the interactions between users and items.
- **MMGCN** [37] allocates an independent GCN for each modality, which learns modality-specific user preferences and item representations via the propagation of modality information.
- **GRCN** [36] adaptively adjusts the interaction graph's structure according to the model training status, then applies graph convolution layers to distill informative signals on user preference.
- **LATTICE** [43] discovers item relationships via multimodal features to learn a graph structure. Graph convolutions aggregate high-order affinities along the structure for recommendation.
- **InvRL** [8] eliminates spurious correlations via heterogeneous environments to learn consistent invariant item representations across diverse settings, improving predictions of interactions.
- **LightGT** [35] is a state-of-the-art Transformer-based model. It designs modal-specific embedding and layer-wise position encoding for effective feature distillation, learning superior user preference on item content for interaction prediction.

To ensure consistency, we adopt the publicly available implementations of the baselines. Note that we did not select the existing generative recommendation model TIGER [24] as baseline, since it only works with sequential recommendation rather than our task.

### 4.3 Parameter Settings

We implement our MMGRec using Pytorch<sup>4</sup> and Pytorch Geometric<sup>5</sup>. Model parameters are initialized with Xavier approach [9] and optimized using the SGD optimizer [3] with a batch size selected from {500, 1,000, 2,000, 3,000}. Hyper-parameters are tuned through grid search based on the results from the validation set. The optimal learning rate is searched from {0.0001, 0.0005, 0.001, 0.005} and ultimately set to 0.001. The  $L_2$  normalization coefficient is searched within  $\{10^{-6}, 10^{-5}, \dots, 10^{-1}, 1\}$ , with the optimal value set to  $10^{-5}$ . Moreover, early stopping is adopted if Recall@10 on the validation set does not rise for 20 successive epochs. Without specification, user/item representations default to a size of 64 in all methods for fairness. As for the Graph RQ-VAE model, we adopt three-stage quantization to assign a three-tuple Rec-ID for each item. To avoid Rec-ID collision, we add a popularity token as the fourth. The size of each level codebook is tuned in {64, 128, 256}. When computing the loss  $\mathcal{L}_{\text{rqvae}}$ , we set  $\beta = 0.25$ .

## 5 EXPERIMENTAL RESULTS

We conduct extensive quantitative and qualitative experiments to answer the following research questions:

- **RQ1:** How does MMGRec perform on the multimodal recommendation task compared with state-of-the-art baselines?
- **RQ2:** Is the technical choice of each component (*i.e.*, item representations, Graph RQ-VAE, and relation-aware self-attention) effective for MMGRec?
- **RQ3:** How do hyper-parameter settings (*e.g.*, depth of layer and number of heads) affect MMGRec?
- **RQ4:** How does MMGRec compare with traditional paradigm methods regarding inference efficiency?

### 5.1 Performance Comparison (RQ1)

Table 2 presents the results of MMGRec and baselines over three experimental datasets. Besides, it reports the improvements and statistical significance test, which are calculated between our method and the strongest baseline (highlighted with underline). From the results, the main observations are as follows:

- MMGRec consistently achieves the best performance in all cases. In particular, improvements over the strongest competitor *w.r.t.* NDCG@10 are 7.17%, 6.79%, and 6.58% in MovieLens, TikTok, and Kwai, respectively. Additionally, we conduct one-sample t-tests, which reveal that the improvements of MMGRec are statistically significant ( $p\text{-value} < 0.05$ ). The primary difference between MMGRec and baselines lies in whether the recommendation paradigm is generative or traditional. Hence, we attribute the significant improvements to introducing a Transformer-based generative paradigm in multimodal recommendation.
- MMGRec and the strongest baseline LightGT outperform other baselines by a large margin. This phenomenon validates that the Transformer is applicable to the multimodal recommendation. Nevertheless, LightGT focuses on improving representation

**Table 2: Overall performance comparison between our model and the baselines on three datasets.**

Methods	MovieLens		TikTok		Kwai	
	R@10	N@10	R@10	N@10	R@10	N@10
GraphSAGE	0.2129	0.1388	0.0778	0.0476	0.0424	0.0344
NGCF	0.2340	0.1383	0.0906	0.0547	0.0427	0.0363
GAT	0.2342	0.1589	0.0945	0.0575	0.0441	0.0369
LightGCN	0.2381	0.1592	0.0988	0.0603	0.0502	0.0411
VBPR	0.1927	0.1207	0.0600	0.0397	0.0302	0.0221
MMGCN	0.2453	0.1523	0.0645	0.0579	0.0456	0.0374
GRCN	0.2520	0.1683	0.0952	0.0584	0.0473	0.0403
LATTICE	0.2520	0.1680	0.0984	0.0611	0.0483	0.0400
InvRL	0.2518	0.1666	0.1002	0.0612	0.0486	0.0402
LightGT	<u>0.2650</u>	<u>0.1771</u>	<u>0.1213</u>	<u>0.0751</u>	<u>0.0546</u>	<u>0.0441</u>
<b>MMGRec</b>	<b>0.2804</b>	<b>0.1898</b>	<b>0.1269</b>	<b>0.0802</b>	<b>0.0567</b>	<b>0.0470</b>
% Improv.	5.81%	7.17%	4.62%	6.79%	3.85%	6.58%
p-value	8.15e-4	2.36e-4	3.83e-2	3.14e-3	5.70e-3	1.27e-2

**Table 3: Performance comparison between our model and the variants without item representations.**

Methods	MovieLens		TikTok		Kwai	
	R@10	N@10	R@10	N@10	R@10	N@10
ID w/o	0.1934	0.1238	0.0811	0.0516	0.0417	0.0361
Emb w/o	0.2780	0.1890	0.1206	0.0792	0.0553	0.0460
Ours	<b>0.2804</b>	<b>0.1898</b>	<b>0.1269</b>	<b>0.0802</b>	<b>0.0567</b>	<b>0.0470</b>

learning by only utilizing the Transformer encoder. In comparison, our model exploits the generative capability of the complete Transformer, thus achieving superior performance.

- Comparing the performance of MMGRec on two evaluation metrics, we find larger improvements in terms of NDCG. Namely, MMGRec’s recommendation result is more accurate with respect to preference ranking. This illustrates that the generative paradigm is especially good at inferring the user’s favorite items.

### 5.2 Ablation Study (RQ2)

**5.2.1 Effect of Item Representation.** To verify the necessity of item representations for MMGRec, we adopt two variants:

**ID w/o:** Without (w/o) item representation, the variant concatenates multimodal features as the input of RQ-VAE to assign Rec-ID.

**Emb w/o:** The variant replaces item representations in the Transformer encoder’s input embedding with parameter embeddings.

Table 3 displays the performance comparison between MMGRec and its variants, and we have the following observations:

- The poor performance of **ID w/o** suggests that heterogeneous multimodal features cannot be directly used for Rec-ID assignment. The likely reason is the complexity of semantic relations among different modalities, necessitating prior integration through multimodal fusion. This aligns with the prior study about the importance of multimodal fusion in semantic comprehension [2].
- Despite parameterizing input embedding, **Emb w/o** underperforms and is inferior to utilizing well-trained item representations. Hence, we infer that the Transformer struggles to learn suitable input embeddings directly from the generative recommendation task without pre-training. This is consistent with many

<sup>4</sup><https://pytorch.org/>.

<sup>5</sup><https://pytorch-geometric.readthedocs.io/en/latest/>.

**Table 4: Performance comparison among different Rec-ID assignment approaches.**

Methods	MovieLens		TikTok		Kwai	
	R@10	N@10	R@10	N@10	R@10	N@10
HK-Means	0.2520	0.1683	0.1137	0.0742	0.0469	0.0402
Random	0.2762	0.1872	0.1199	0.0785	0.0544	0.0452
Ours	<b>0.2804</b>	<b>0.1898</b>	<b>0.1269</b>	<b>0.0802</b>	<b>0.0567</b>	<b>0.0470</b>

Transformer-based NLP models that benefit from pre-trained word embedding like Word2Vec and Glove.

**5.2.2 Effect of Graph RQ-VAE.** To study the importance of Graph RQ-VAE, we compare it with Hierarchical K-Means Clustering (**HK-Means**) [5] in Rec-ID assignment. HK-Means aims at establishing a cluster hierarchy, employing a top-down methodology that initiates with a single cluster including all items, then iteratively divides it into smaller clusters according to item representations. The hierarchical cluster indices of an item can be concatenated as the Rec-ID. Experimentally, we adjust the clustering number to obtain optimal performance and comparable cardinality.

- Table 4 shows the performance comparison between employing Graph RQ-VAE and HK-Means. Graph RQ-VAE consistently outperforms HK-Means, illustrating the advantageous quantization ability. To explore the reason, we find that more ID collisions exist in HK-Means due to the clustering characteristic. This illustrates that collisions negatively affect performance and should be handled reasonably.
- Therefore, we also compare our collision solution with the previous method (denoted by **Random**) that randomly extends one token to make Rec-ID unique. As the results recorded in Table 4, it is obvious that such a random method is consistently inferior to our solution. This verifies that the Transformer cannot accurately generate the last random token, while our method relatively circumvents this weakness.

**5.2.3 Effect of Relation-Aware Self-Attention.** To investigate whether MMGRec can benefit from relation-aware self-attention, we consider two variants of our model:

**Default PE:** This variant preserves the default sinusoid-based positional encoding of Transformer.

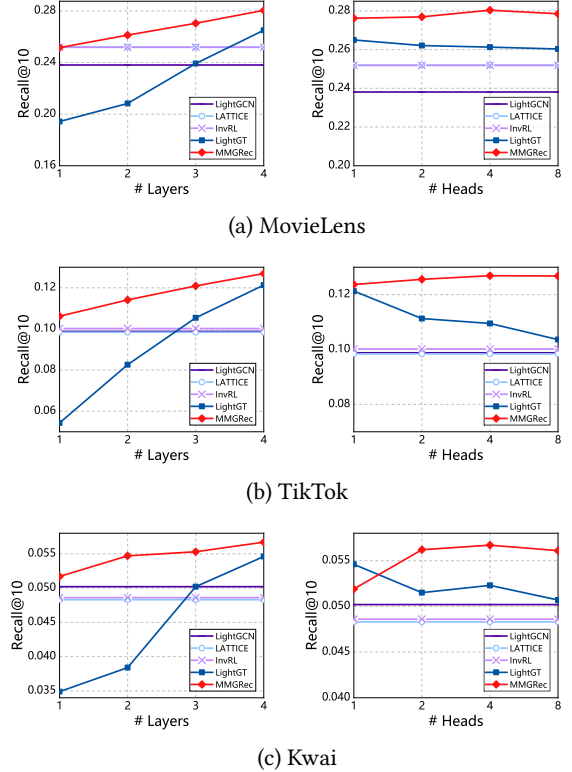
**w/o PE:** This variant entirely discards positional encoding.

Note that our investigation of positional encoding is limited to the Transformer’s encoder, while the decoder’s requires no changes. Table 5 exhibits MMGRec’s performance with different position encoding approaches. We find that:

- The variant without any positional encoding achieves poor performance. Even using absolute positions of input elements for positional encoding can improve the performance. This indicates the importance of positional information for the Transformer [27].
- Compared to default positional encoding, relation-aware self-attention yields remarkable improvements, suggesting the effectiveness of leveraging pairwise item relations as positional information. This also validates the necessity of utilizing user information to model user-specific pairwise item relations, distinguishing our design from a standard self-attention head.

**Table 5: Performance comparison among different positional encoding approaches.**

Methods	MovieLens		TikTok		Kwai	
	R@10	N@10	R@10	N@10	R@10	N@10
w/o PE	0.2696	0.1850	0.1152	0.0765	0.0517	0.0427
Default PE	0.2732	0.1853	0.1180	0.0778	0.0545	0.0441
Ours	<b>0.2804</b>	<b>0.1898</b>	<b>0.1269</b>	<b>0.0802</b>	<b>0.0567</b>	<b>0.0470</b>

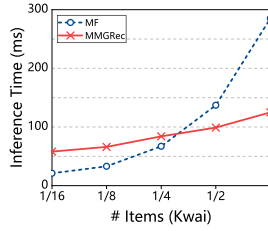


**Figure 4: Effect of layer and head numbers.**

### 5.3 Hyper-Parameter Analysis (RQ3)

To investigate the impact of Transformer layers on our model, we vary the layer number in the range of {1, 2, 3, 4} when fixing the self-attention head number to optimal. As a comparison, we test the LightGT with the same setting. Left of Figure 4 plots the results *w.r.t.* Recall@10 on three datasets, we observe that: First, MMGRec outperforms LightGT as varying layer numbers across all datasets. This validates the stable superiority derived from the generative recommendation. Second, MMGRec can achieve quite good performance with fewer layers. This indicates that our generative model can be effective in a fundamental setting, while the embedding-based method needs stacking Transformer layers to guarantee performance.

Moreover, we investigate the impact of multi-head attention on our model by comparing 1-head, 2-head, 4-head, and 8-head self-attention blocks with LightGT. As shown in the right of Figure 4, we observe that MMGRec consistently outperforms LightGT. As the head number increases, the performance of MMGRec tends to



**Figure 5: Inference time on different-scale Kwai datasets. The unit of time is milliseconds (ms).**

stabilize after rising. In comparison, the performance of LightGT shows a continued downward trend. Theoretically, one head of self-attention models the relation among elements in a single aspect. Therefore, we infer that the generative model can handle more kinds of relations than the embedding-based method.

#### 5.4 Efficiency Study (RQ4)

Instead of computing and ranking similarity, MMGRec employs autoregressive decoding to directly generate Rec-IDs identifying items during inference. To assess the efficiency of the two inference approaches, we conduct comparative experiments using identical computing configurations and record their inference time. Figure 5 presents an efficiency comparison between MMGRec and MF regarding average inference time per user across different-scale Kwai datasets. **1/8 Kwai** denotes randomly selecting one in eight items from the complete dataset. The main observations are as follows:

In scenarios where the item count is low (e.g.,  $1/16 \sim 1/4$  Kwai), MMGRec exhibits inefficiency, primarily due to the basic computational demands of autoregressive decoding. However, as the number of items increases (e.g.,  $1/4 \sim 1$  Kwai), the inference time of MF experiences a significant rise. This escalation is attributed to the fact that item count linearly impacts the time complexity of inner product computation and sorting. In contrast, MMGRec demonstrates enhanced efficiency as its inference time remains relatively stable. This stability arises from the unchanged number of autoregressive decoding steps, even with an increase in the number of items. These findings suggest that the generative paradigm holds the potential to deliver more efficient inference for large-scale recommendation.

## 6 RELATED WORK

### 6.1 Multimodal Recommendation

Research on multimodal recommendation aims to enhance recommender systems by leveraging multimodal information. Existing methodologies typically employ a combination of CF and content-based recommendation techniques [21]. Early studies aim to integrate multimodal content for more rational representation learning. For example, the pioneering work VBPR [13] maps visual information to the representation space and concatenates it with CF representations. Subsequent work like ACF [6] and UVCAN [22] adaptively select multimodal features using different levels of attention mechanisms to learn representations of users and items. In recent years, the research trend turns to exploring the relationship between modal information and users/items. For instance, the representative work MMGCN [37] introduces a modality-aware GCN into multimodal recommendation tasks to aggregate and propagate

multimodal information on the user-item graph. HHFAN [4] utilizes a modality-aware heterogeneous information graph to explore more complex relationships among users, micro-videos, and multimodal information, thereby improving recommendation performance.

Despite their effectiveness, existing methods suffer from inherent limitations, such as overwhelming inference cost, insufficient interaction modeling, and false-negative issues, stemming from their embed-and-retrieve paradigm. To address these limitations, this study proposes a generative multimodal recommendation method.

### 6.2 Transformer-Based Recommendation Model

Since its inception, Transformer has demonstrated outstanding performance across various domains, such as natural language processing [38] and computer vision [11], yielding remarkable results. Owing to its proficiency in modeling contextual information, Transformer has emerged as a central focus in recommendation research. Researchers have developed Transformer-based recommendation models to capture item relationships and user behavior patterns in historical interactions [39, 40]. PEAR [20] leverages the Transformer to capture interaction and contextual information from the history of clicked items. LightGT [35] lightens the Transformer and combines it with GCN to enhance user preference modeling by considering item multimodal correlations. Recently, Transformer has emerged as a significant contributor to the development of generative recommendation. DSI [30] signifies a pivotal transition in information retrieval towards a generative paradigm, being the first successful end-to-end Transformer application for retrieval tasks. Relying on a novel semantic ID representation, TIGER [24] implements a Transformer-based sequence-to-sequence model capable of directly predicting the semantic ID of the next item to achieve generative recommendation.

Existing Transformer-based approaches are primarily limited to traditional representation learning or sequential generative recommendation. This study firstly introduces the Transformer with a generative paradigm to the field of multimodal recommendation.

## 7 CONCLUSION

In this work, we propose MMGRec, a novel Transformer-based model for multimodal recommendation. MMGRec explores the generative paradigm to address limitations inherent in the traditional paradigm, consisting of two essential components — Rec-ID assignment and Rec-ID generation. In Rec-ID assignment, we integrate item multimodal data with CF information and develop a Graph RQ-VAE to quantize them into Rec-ID for each item. In Rec-ID generation, a Transformer model is trained to directly predict item Rec-IDs for recommendation based on a user’s interaction sequence. Experimental results demonstrate that MMGRec achieves state-of-the-art performance while maintaining conditionally efficient inference.

For future work, we plan to enhance MMGRec’s performance by developing a more effective vector quantization method to mitigate Rec-ID assignment collisions. Additionally, we intend to explore the integration of large models [18], such as ChatGPT, to improve user and item representations. Furthermore, we are interested in leveraging MMGRec’s generative capability to tackle the cold-start problem in recommendation [28].

## REFERENCES

- [1] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *International Conference on Learning Representations*. 1–16.
- [2] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2018. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41, 2 (2018), 423–443.
- [3] Léon Bottou. 2010. Large-scale machine learning with stochastic gradient descent. In *Proceedings of International Conference on Computational Statistics*. 177–186.
- [4] Desheng Cai, Shengsheng Qian, Quan Fang, and Changsheng Xu. 2021. Heterogeneous hierarchical feature aggregation network for personalized micro-video recommendation. *IEEE Transactions on Multimedia* 24 (2021), 805–818.
- [5] Bernard Chen, Phang C Tai, Robert Harrison, and Yi Pan. 2005. Novel hybrid hierarchical-K-means clustering method (HK-means) for microarray analysis. In *IEEE Computational Systems Bioinformatics Conference-Workshops*. 105–108.
- [6] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item- and component-level attention. In *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval*. 335–344.
- [7] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2020. Autoregressive entity retrieval. In *International Conference on Learning Representations*.
- [8] Xiaoyu Du, Zike Wu, Fuli Feng, Xiangnan He, and Jinhui Tang. 2022. Invariant representation learning for multimedia recommendation. In *Proceedings of ACM International Conference on Multimedia*. 619–628.
- [9] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of International Conference on Artificial Intelligence and Statistics*. 249–256.
- [10] Will Hamilton, Zhitaoying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Conference on Neural Information Processing Systems*. 1024–1034.
- [11] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. 2022. A survey on vision transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 1 (2022), 87–110.
- [12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 770–778.
- [13] Ruining He and Julian McAuley. 2016. VBPR: Visual bayesian personalized ranking from implicit feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 144–150.
- [14] 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 International ACM SIGIR Conference on Research and Development in Information Retrieval*. 639–648.
- [15] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of International World Wide Web Conference*. 173–182.
- [16] Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al. 2017. CNN architectures for large-scale audio classification. In *International Conference on Acoustics, Speech and Signal Processing*. 131–135.
- [17] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. 2017. Collaborative metric learning. In *Proceedings of International Conference on World Wide Web*. 193–201.
- [18] Wenyue Hua, Lei Li, Shuyuan Xu, Li Chen, and Yongfeng Zhang. 2023. Tutorial on large language models for recommendation. In *Proceedings of ACM Conference on Recommender Systems*. 1281–1283.
- [19] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.
- [20] Yi Li, Jieming Zhu, Weiwen Liu, Liangcai Su, Guohao Cai, Qi Zhang, Ruiming Tang, Xi Xiao, and Xiuqiang He. 2022. PEAR: Personalized Re-ranking with Contextualized Transformer for Recommendation. In *Proceedings of International World Wide Web Conference*. 62–66.
- [21] Han Liu, Yinwei Wei, Fan Liu, Wenjie Wang, Liqiang Nie, and Tat-Seng Chua. 2023. Dynamic Multimodal Fusion via Meta-Learning Towards Micro-Video Recommendation. *ACM Transactions on Information Systems* 42, 2 (2023), 1–26.
- [22] Shang Liu, Zhenzhong Chen, Hongyi Liu, and Xinghai Hu. 2019. User-video co-attention network for personalized micro-video recommendation. In *Proceedings of International World Wide Web Conference*. 3020–3026.
- [23] Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith 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*. 1864–1874.
- [24] 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. In *Proceedings of International Conference on Neural Information Processing Systems*.
- [25] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of Conference on Uncertainty in Artificial Intelligence*. 452–461.
- [26] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of International Conference on World Wide Web*. 285–295.
- [27] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-Attention with Relative Position Representations. In *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 464–468.
- [28] Changfeng Sun, Han Liu, Meng Liu, Zhaochun Ren, Tian Gan, and Liqiang Nie. 2020. LARA: Attribute-to-feature adversarial learning for new-item recommendation. In *Proceedings of International Conference on Web Search and Data Mining*. 582–590.
- [29] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of International Conference on Neural Information Processing Systems*. 3104–3112.
- [30] 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.
- [31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of International Conference on Neural Information Processing Systems*. 6000–6010.
- [32] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*. 1–12.
- [33] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of International ACM SIGIR conference on Research and Development in Information Retrieval*. 165–174.
- [34] Xiang Wang, Yaokun Xu, Xiangnan He, Yixin Cao, Meng Wang, and Tat-Seng Chua. 2020. Reinforced negative sampling over knowledge graph for recommendation. In *Proceedings of International Conference on World Wide Web*. 99–109.
- [35] Yinwei Wei, Wenqi Liu, Fan Liu, Xiang Wang, Liqiang Nie, and Tat-Seng Chua. 2023. LightGT: A light graph transformer for multimedia recommendation. In *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1508–1517.
- [36] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, and Tat-Seng Chua. 2020. Graph-refined convolutional network for multimedia recommendation with implicit feedback. In *Proceedings of ACM International Conference on Multimedia*. 3541–3549.
- [37] 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 ACM International Conference on Multimedia*. 1437–1445.
- [38] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*. 38–45.
- [39] Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. 2020. SSE-PT: Sequential recommendation via personalized transformer. In *Proceedings of ACM Conference on Recommender Systems*. 328–337.
- [40] Lianghao Xia, Chao Huang, Yong Xu, Peng Dai, Bo Zhang, and Liefeng Bo. 2020. Multiplex behavioral relation learning for recommendation via memory augmented transformer network. In *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2397–2406.
- [41] Zhengyi Yang, Jiancan Wu, Zhicai Wang, Xiang Wang, Yancheng Yuan, and Xiangnan He. 2023. Generate what you prefer: Reshaping sequential recommendation via guided diffusion. In *Proceedings of International Conference on Neural Information Processing Systems*.
- [42] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. 2021. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 30 (2021), 495–507.
- [43] Jinghao Zhang, Yanqiao Zhu, Qiang Liu, Shu Wu, Shuhui Wang, and Liang Wang. 2021. Mining Latent Structures for Multimedia Recommendation. In *Proceedings of ACM International Conference on Multimedia*. 3872–3880.