Multimodal Product Recommendations via LLMs

Rajwinder Mahal
rsm2207@columbia.edu

Shamin Aggarwal
sa4129@columbia.edu

Vignay Chanda
vc2608@columbia.edu

Abstract

In the rapidly advancing landscape of e-commerce, effective product recommendation systems are essential for enhancing customer experience and boosting sales. Traditional recommendation systems often struggle with issues such as data sparsity, cold start problems, and overfitting. This paper introduces a novel multimodal approach that integrates the advanced capabilities of large language models (LLMs) with the rich visual information of product images, aiming to overcome these challenges. We propose a system that combines Vector Quantized Generative Adversarial Network (VQGAN) for image encoding and Sequential Recommendation with BART for processing user interactions and product attributes. This approach effectively harnesses the strengths of both content-based and collaborative filtering, leveraging LLMs to capture complex relationships between product attributes and user preferences, while utilizing product images to enhance understanding of product features and aesthetics. This paper demonstrates our results on this multimodal approach for recommendation systems and comparisons with baseline models.

1 Introduction

In the era of e-commerce, customers can choose from a wide range of products, which can make it difficult to quickly find what interests them or what they are looking for. This can lead to decision fatigue, causing frustration, abandoned shopping carts, and lost sales [2, 10]. To address this challenge, product recommendations have emerged as a crucial tool for enhancing customer experience and driving sales.

Traditional recommendation systems often rely on collaborative filtering or content-based filtering techniques to generate recommendations. More recent research has focused on deep learning based methods [7, 17], however, these approaches still fail to fully capture the nuances of user behavior and preferences, making them susceptible to data sparsity, cold start problems, and overfitting [2]. Recent advancements in large language models (LLMs) have opened up new possibilities for personalized product recommendations. These LLMs possess the ability to process and understand complex linguistic information, enabling them to extract meaningful insights from user interactions, search queries, and product descriptions [4, 9, 14, 16]. As a result, the use of large language models in recommender systems has garnered significant attention in recent research.

Numerous studies have explored the use of LLMs as recommender systems and rely on a variety of data sources, including user preferences, product descriptions, click-through data, ratings, and reviews [6, 14, 16, 18]. While these data sources can provide valuable insights, they often fail to capture the rich visual information contained in product images [3, 11]. We therefore propose a multimodal approach that uses LLMs to capture complex relationships between product attributes and user preferences, while product images enhance the understanding of product features and aesthetics [12, 19]. Our system is designed to predict user preferences more accurately by synthesizing textual and visual data, thereby providing more relevant and personalized product recommendations. In this

paper, we show that our multimodal approach provides more accurate and relevant recommendations. Our contributions can be summarized as follows:

- fine-tuned VQGAN model on H&M dataset;
- fine-tuned BART model for sequential recommendations;
- our results, analysis, and comparisons with the baseline models.

2 Related Work

Recommender systems play a pivotal role in enhancing user experiences across various online services. These systems leverage advanced deep learning techniques to capture and model user preferences [2, 7, 10, 17]. The recent surge in performance of large language models (LLMs) has led to a desire to use them for recommendations. For example, BERT4Rec [16] uses Bidirectional Encoder Representations from Transformers (BERT) for sequential recommendations. It learns a bidirectional representation of user behavior sequences and allow each item to fuse information from both left and right sides, and thereby capturing long-range dependencies and modeling user preferences accurately.

Another notable approach is P5 - A Unified Pretrain, Personalized Prompt & Predict Paradigm [6] which is trained with instruction-based prompts and reformulates all recommendation tasks as NLP tasks with the help of personalized prompts. Unlike earlier models such as the sequential recommendation model, which struggled with generalization issues and the transferability of learned knowledge, P5 adopts a novel perspective, considering that language can describe almost anything. The prompt is carefully crafted to encode the user's preferences and the desired characteristics of the recommended products. P5 is a simple and flexible model that can be easily adapted to different recommendation scenarios and has been shown to outperform traditional recommendation systems on a variety of metrics.

Other methods include LLM-Rec [14] that uses various prompting strategies tailored for personalized content recommendations and HKFR [18] that extracts and integrates heterogeneous knowledge from heterogeneous behavior information of users to create instruction datasets for recommendation tasks to fine-tune LLM.

With the proliferation of multimedia content in online platforms, ranging from short videos to news articles, the need for recommender systems to comprehend and recommend such diverse content has become increasingly crucial. As a result, more recent methods focused on capturing the visual features from product features to provide recommendations [3, 11, 15, 17]. For example, [15] builds visually-aware representations on top of interpretable visual features based on fine-grained parsing of product images to provide clothing recommendations.

3 Data

Our project uses H&M Personalized Fashion Recommendations dataset [1], released as part of the Kaggle Competition by H&M. The dataset consists of customer purchase history over time with detailed metadata of purchased items, product images, and customer demographics.

1. **Product Metadata:** This dataset encompasses a detailed description of each article, including its name and various categorical attributes such as the product's color, department, garment group, and descriptive text.
2. **Customer Metadata:** It contains essential information about the customers, such as their status in the store's membership club, their subscription status to fashion news, and their age group.
3. **Historical Transaction Data:** This dataset records past transactions, offering a comprehensive view of customer purchasing behaviors and preferences over time.
4. **Article Images:** A collection of images corresponding to the articles/product items.

The combination of these transactions and product attributes provide a multifaceted view of the e-commerce environment, enabling a deeper understanding of both customer behavior and product characteristics.

Description	Count
No. of product items	105,542
No. of customers	1,371,980
No. of historical transactions	31,788,324

Table 1: Statistics on H&M dataset.

4 Methods

We design our approach based on the idea that (1) LLMs possess the ability to process and understand complex linguistic information, enabling them to capture complex relationships between product attributes and user preferences, and (2) product images provide additional understanding of product features and aesthetics, thereby capturing the visual appeal of products. We believe that combining these two will result in more accurate and relevant recommendations.

4.1 Baselines

4.1.1 POP

The POP or Popularity method is a widely used and very simple baseline for recommender models. It involves calculating the frequency of occurrence of each distinct product in the dataset and sorting them in descending order of frequency, hence getting the universal 'popularity' of each product. To test recommendations for any customer, we simply return a pre-defined number of the most frequently seen products, and compare it against the test data to get a recommendation accuracy score.

4.1.2 GRU4Rec

GRU4Rec [7] is a session-based recommendation approach that uses RNNs to capture sequential dependencies and make predictions. The model employs Gated Recurrent Unit (GRU)-based RNNs for providing session-based recommendations. The architecture processes the state of user sessions, predicting the next item in a session. A weighted representation of session events is used, discounting earlier events and normalizing the input vector for stability. This aids the model in emphasizing local ordering constraints, which might not be captured by the RNN's longer memory span.

At the core of the GRU4Rec model are one or more GRU layers, where each of the layer's hidden state serves as an input to the next layer. This kind of stacked layout allows the model to capture complex item transitions within the sessions. Feed-forward layers may be interspersed between the GRU layers and the output layer to refine the prediction. The output of the network reflects the predicted preference for each item, essentially estimating the likelihood of an item being the next event in the session. Through this architecture, GRU4Rec captures both the immediate context of a session and the general sequence of events, providing a robust recommendation system suited for the dynamic nature of session-based interactions.

4.2 VQGAN

The Vector Quantized Generative Adversarial Network (VQGAN) is designed to synthesize high-resolution images by integrating the efficiency of convolutional neural networks (CNNs) with the expressivity of transformer models. [5]

The main idea of this model is to use a convolutional approach to construct a discrete code-book of visual parts of an image which can effectively capture the richness of the image constituents. This is accomplished through a two-part model comprising an encoder E and a decoder G, which together learn to represent images as a collection of codebook entries. The spatial arrangement of these codebook entries allows any image to be represented by a sequence of indices, specifying the corresponding codebook entries, as illustrated in Figure 1

Autoregressive Modeling with Transformers: With the encoder-decoder structure in place, VQGAN then uses the transformer architecture to model the composition of the obtained codebook indices. The quantized encoding of the images are processed by the transformer to predict the next index in the sequence. This forms an autoregressive model that predicts the distribution of possible next

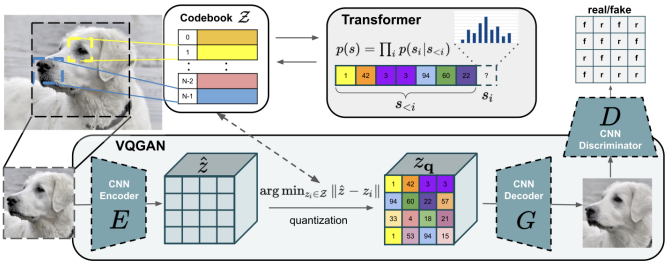


Figure 1: VQGAN framework leveraging a convolutional encoder and transformer-based autoregression for high-fidelity image synthesis.

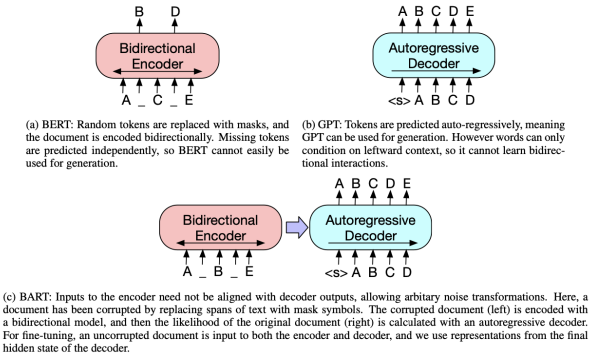
indices. This setup allows the direct computation of the likelihood for the entire representation of the image as a sequence of indices.

Discriminator: Additionally a *patch-based discriminator* is integrated into the VQGAN architecture to enable robust compression while retaining high perceptual quality at the same time. This discriminator differentiates between real and reconstructed images, guiding the encoder-decoder network to produce visually compelling synthetic images.

In summary, VQGAN tames transformers for high-resolution image synthesis by combining a context-rich vocabulary of image constituents with the global compositional modeling capabilities of transformers, facilitated by a discrete codebook that serves as the interface between the two.

4.3 BART

The BART (Bidirectional and Auto-Regressive Transformers) model is a denoising autoencoder specifically designed for pretraining sequence-to-sequence models. The model is built upon the traditional sequence-to-sequence Transformer model, with modifications inspired by GPT. [8]



(a) BERT: Random tokens are replaced with masks, and the document is encoded bidirectionally. Missing tokens are predicted independently, so BERT cannot easily be used for generation.

(b) GPT: Tokens are predicted auto-regressively, meaning GPT can be used for generation. However words can only condition on leftward context, so it cannot learn bidirectional interactions.

(c) BART: Inputs to the encoder need not be aligned with decoder outputs, allowing arbitrary noise transformations. Here, a document has been corrupted by replacing spans of text with mask symbols. The corrupted document (left) is encoded with a bidirectional model, and then the likelihood of the original document (right) is calculated with an autoregressive decoder. For fine-tuning, an uncorrupted document is input to both the encoder and decoder, and we use representations from the final hidden state of the decoder.

Figure 2: Schematic representation of BART model, comparing its components with BERT and GPT.

The main architectural components of the BART are as follows:

Bidirectional Encoder: The encoder within BART is bidirectional which helps in enabling each token within the input sequence to attend to all other tokens. This is similar to the encoder architecture in BERT (Bidirectional Encoder Representations from Transformers) and is crucial for understanding the full context of the input.

Cross-Attention Mechanism: The main difference of BART's Encoder, setting it apart from BERT, is that each of it's decoder layers additionally performs cross-attention over the final hidden layer of the encoder (as in the transformer Seq2Seq model), making the model effective for tasks that require an understanding of the entire input sequence.

Autoregressive Decoder: BART uses an autoregressive decoder which generates output text in a left-to-right fashion. Every token generated by the decoder is conditioned on the previous tokens, making it similar in behavior to the GPT (Generative Pre-trained Transformer) model.

GeLU Activation and Parameter Initialization: Inspired by GPT, BART employs Gaussian Error Linear Unit (GeLU) activation functions and specific parameter initializations to optimize its performance.

These architectural features enable BART to effectively perform tasks that involve both understanding (comprehension) and generating (production) of text, making it versatile for various natural language processing applications.

4.4 Our Architecture

We design our approach based on Vector Quantized Generative Adversarial Network (VQGAN) [5] and Sequential Recommendation with Bidirectional Encoder Representations from Transformer (BERT4Rec) [16]. We will use VQGAN to encode images into a fixed length tokens and then pass them to Bidirectional and Auto-Regressive Transformers (BART) [8] along with user interactions to generate predictions.

We will first fine-tune VQGAN on our dataset. Then, we will fine-tune BART as follows:

- construct sequential user history with product features and images;
- encode product images using VQGAN;
- encode product attributes using BART encoder;
- concatenate both product image and attributes embeddings;
- the task of BART model is to try to predict the next product items given the sequence of historical items;
- we then use output embeddings to find items that are similar to both product images and attributes. This is done using a vector database.

5 Experiments

In this section, we describe in more details our process for data preprocessing, fine-tuning our models, and evaluation methodology.

5.1 Experimental Details

5.1.1 VQGAN

We used the original VQGAN implementation by Esser et al [5]. We used a pretrained checkpoint that is trained ImageNet for 12 epochs. We used this as a starting point to fine-tune the model on our H&M dataset.

Most of the architecture related code is adapted from the Taming Transformers repository, however, we end up making many adjustments to code to decouple components and implement distributed training script using PyTorch Lightning. We fine-tuned it on 100k images for 1 epoch, with 4 Tesla L4 GPUs, 8 data loading workers per GPU process, and 4.5e-6 learning rate which is adopted from the pretrained checkpoint. To maximize the GPU memory use, we used a batch size of 6 per GPU with an effective batch size 24 (4 GPUs).

5.1.2 BART

We used a standard Transformer encoder-decoder model, pre-trained BART base [8], consisting of 140M parameters with 6 encoder and decoder layers, 1024 hidden states, and 16 attention heads.

We used HuggingFace Transformers library for fine-tuning our models with a learning rate of $5e-5$ for 2 epochs. We used Adam with weight decay [13] for parameter optimization. We used 4 Tesla L4 GPUs for each experiment, each with a batch size of 4 and 4 gradient accumulation steps; effective

```
  | Name            | Type                        | Params
-------------------------------------------------------------
0 | encoder         | Encoder                     | 29.3 M
1 | decoder         | Decoder                     | 42.4 M
2 | loss            | VQLPIPSWithDiscriminator    | 15.4 M
3 | quantize        | VectorQuantizer2            | 4.2 M
4 | quant_conv      | Conv2d                      | 65.8 K
5 | post_quant_conv | Conv2d                      | 65.8 K
-------------------------------------------------------------
76.7 M    Trainable params
14.7 M    Non-trainable params
91.5 M    Total params
```

Figure 3: VQGAN model summary.

batch size of 64. It is common practice to set the number of data loading workers to the number of CPU cores. In our case, we used 8 data loading workers per GPU process to reduced GPU idle time.

We are fine-tuning 3 different BART models:

- model with small number of historical items as input (no images);
- model with large number of history items as input (no images);
- model with historical items with encoded images as input. We cannot experiment with a larger history items as input for this multimodal approach as our input restricted by model input size and image tokens results in 256 tokens per history item.

5.1.3 GRU4REC

For the GRU4Rec model implementation we followed the underlying paper by Balázs Hidasi et al. [7] very closely. We adapted the author-provided code from the associated GRU4Rec PyTorch Official GitHub repository, making changes to the architecture as required to modify it according to our dataset schema. One of these major changes was to accommodate our custom-defined sessions (described below).

To load the dataset we used 'datatable' Python library which is optimized to handle large datasets, and in conjunction with Pandas, we used it to perform the further required preprocessing. This involved mainly grouping the rows by 'customer_id' and for each group, dividing all the transactions into week-long windows. Thus, for each distinct customer, each of these windows represented one session. Subsequently, we added a 'SessionId' column to the dataset and all the transaction data belonging to one session was assigned a unique ID.

We trained the model using PyTorch from scratch on our preprocessed dataset on a Google Cloud Platform Linux VM, utilizing one NVIDIA Tesla P4 GPU. We trained for 10 epochs and experimented around with the hyperparameter tuning; the results presented in this paper used a learning rate of 0.05, a sample size of 2048 and batch size 64.

5.2 Data Preprocessing

Our data preprocessing include removing items with null values, dropping columns, converting values to numerical values, etc. Since we have a large dataset, we chose to simply remove items with null values. We also removed columns that have imbalanced distribution. For example, over 90% of the customers were H&M membership holders, so we removed this column. Similarly, we converted postal codes to ordinal encoding. We also dropped columns that has high percentage of null values.

5.3 Evaluation Metric

For evaluating our fine-tuned VQGAN, we used Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS). PSNR measures the difference between the original and reconstructed images in terms of their signal-to-noise ratio. SSIM measures the structural similarity between the two images. LPIPS measures the perceptual distance between the generated image, in our case a reconstructed image, and the ground truth image in terms of their perceptual similarity.

For evaluating our fine-tuned BART models, we used BERTScore to measure the precision, recall, and f1 scores. We measured the number of correctly predicted items given a customer's k historical purchases.

6 Results

The POP baseline did not show good results, giving an accuracy of 0.17%. This might be expected when using such large datasets, as the probability that even the most popular product would be purchased by a customer within a 2 week period is negligible, when considering the fact that the dataset contains almost 100,000 distinct products.

We evaluated the GRU4Rec baseline model by testing Recall and MRR (Mean Reciprocal Rank), two very common metrics for recommender models. We used a range of different cutoffs, and the results are displayed in Table 2.

Comparing with the results of the Kaggle competition for which this dataset was released, we observed decent results in terms of the recommendation accuracy of our implementation, however its difficult to compare directly as they didn't post complete breakdowns of their results. We could also compare against the original GRU4Rec paper [7]; even though it was tested on different datasets, we can still get a sense of the expected range of accuracy. The authors tested their model on a similar e-commerce dataset called "RecSys Challenge 2015" and using a cutoff of 20, observed a Recall of 24.82% which is comparable to ours of 25.31%. We did observe lower MRR results for our implementation, it is unclear whether this is a consequence of the differences in the underlying datasets used or a lack of optimization on our part.

Our evaluation of the GRU4Rec model revealed several inherent limitations: it lacks explicit user modeling, neglecting user attributes like demographics or preferences. This omission led to minimal improvement in accuracy, even with product and customer metadata integration. Additionally, the model's high computational demand, relative to its low performance gains, made processing these extra features impractical. Finally, GRU4Rec's session-based design appears to bias it towards recent interactions, limiting its ability to reflect users' long-term interests and evolving preferences.

Cutoff	Recall	MRR
1	0.1131	0.1131
3	0.1634	0.1353
5	0.1857	0.1404
10	0.2179	0.1447
20	0.2532	0.1471

Table 2: Evaluation results from the GRU4Rec model.

	PSNR ↑	SSIM ↑	LPIPS ↓
Original Pretrained model	26.4767	0.8016	0.00055
Our Fine-tuned model	27.3539	0.8220	0.00033

Table 3: Evaluation results from the fine-tuned VQGAN model.

Model	Precision	Recall	f1
BART (small history)	0.9489	0.9294	0.9390
BART (large history)	0.9492	0.9301	0.9396
VQGAN+BART	0.9464	0.9293	0.9377

Table 4: Evaluation results from the fine-tuned BART models.

In our experiments, we fine-tuned three distinct BART models, each tailored to different input configurations for product recommendation, as mentioned in Section 5.1.2 and found some notable

results (See Table 4). The *BART model with a small history input*, achieved impressive results with a precision of 94.89%, a recall of 92.94%, and an F1 score of 93.90%. We can also see that the *BART model with a large history input* demonstrated a slight improvement, recording an F1 score of 93.96%. This enhancement indicates the model's capability to benefit from an expanded historical context. Furthermore, our *VQGAN-enhanced BART model*, which combined encoded images with historical data, demonstrated a marginally lower F1 score of 93.77%. This outcome can be attributed to the model's input constraints - specifically, the inability to incorporate a larger history of items due to the limitation imposed by the model's input size and the substantial token allocation for image data, which amounts to 256 tokens per historical item.

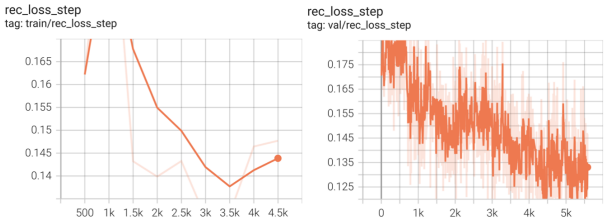


Figure 4: VQGAN image reconstruction loss.

Figure 5: Reconstructed image by our fine-tuned VQGAN.

7 Conclusion and Future Work

Our experiments demonstrate the efficacy of combining textual and visual modalities for product recommendations. The fine-tuned VQGAN model showed good improvements in image reconstruction quality, while the fine-tuned BART models excelled in sequential recommendation tasks. While the GRU4Rec baseline comparisons illustrates the advantage of our approach in handling the session-based interactions.

Given the limitation of input size due to large image tokens (256 per history item), future iterations of our model could explore iterative or recursive methods to accommodate longer sequences of historical items, potentially enhancing the depth and accuracy of our recommendations.

Additionally, while the current system focuses on content-based filtering, future work could explore a hybrid model incorporating collaborative filtering to enhance recommendation quality further. Such an approach could take advantage of the user interaction data more effectively, potentially offering a comprehensive solution to the cold start problem and improving personalized recommendations.

The integration of multimodal data represents a significant step forward in recommendation systems. By capturing the subtleties of user preferences and product features, our study shows that there can me more nuanced and satisfying shopping experience for users which can benefit e-commerce platforms through increased engagement and sales.

References

[1] Hm personalized fashion recommendations. `https://www.kaggle.com/competitions/h-and-m-personalized-fashion-recommendations/data`, 2022. [Online; accessed 18-December-2023].

[2] Rand Jawad Kadhim Almahmood and Adem Tekerek. Issues and solutions in deep learning-enabled recommendation systems within the e-commerce field. *Applied Sciences*, 12(21), 2022.

[3] Xu Chen, Hanxiong Chen, Hongteng Xu, Yongfeng Zhang, Yixin Cao, Zheng Qin, and Hongyuan Zha. Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 765–774, New York, NY, USA, 2019. Association for Computing Machinery.

[4] Dario Di Palma. Retrieval-augmented recommender system: Enhancing recommender systems with large language models. In *Proceedings of the 17th ACM Conference on Recommender Systems*, RecSys '23, page 1369–1373, New York, NY, USA, 2023. Association for Computing Machinery.

[5] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis, 2021.

[6] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (rlp): A unified pretrain, personalized prompt predict paradigm (p5), 2023.

[7] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks, 2016.

[8] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019.

[9] Lei Li, Yongfeng Zhang, and Li Chen. Prompt distillation for efficient llm-based recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, CIKM '23, page 1348–1357, New York, NY, USA, 2023. Association for Computing Machinery.

[10] Zhijie Lin. An empirical investigation of user and system recommendations in e-commerce. *Decision Support Systems*, 68:111–124, 2014.

[11] Qiang Liu, Shu Wu, and Liang Wang. Deepstyle: Learning user preferences for visual recommendation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 841–844, New York, NY, USA, 2017. Association for Computing Machinery.

[12] Qidong Liu, Jiaxi Hu, Yutian Xiao, Jingtong Gao, and Xiangyu Zhao. Multimodal recommender systems: A survey, 2023.

[13] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

[14] Hanjia Lyu, Song Jiang, Hanqing Zeng, Qifan Wang, Si Zhang, Ren Chen, Chris Leung, Jiajie Tang, Yinglong Xia, and Jiebo Luo. Llm-rec: Personalized recommendation via prompting large language models, 2023.

[15] Charles Packer, Julian McAuley, and Arnau Ramisa. Visually-aware personalized recommendation using interpretable image representations, 2018.

[16] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer, 2019.

[17] Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding, 2018.

[18] Bin Yin, Junjie Xie, Yu Qin, Zixiang Ding, Zhichao Feng, Xiang Li, and Wei Lin. Heterogeneous knowledge fusion: A novel approach for personalized recommendation via llm, 2023.

[19] Hongyu Zhou, Xin Zhou, Zhiwei Zeng, Lingzi Zhang, and Zhiqi Shen. A comprehensive survey on multimodal recommender systems: Taxonomy, evaluation, and future directions, 2023.