# Preference-Optimized Retrieval and Ranking for Efficient Multimodal Recommendation

Zhenrui Yue
University of Illinois
Urbana-Champaign
Champaign, IL, USA
zhenrui3@illinois.edu

Huimin Zeng
University of Illinois
Urbana-Champaign
Champaign, IL, USA
huiminz3@illinois.edu

Yueqi Wang
UC Berkley
Berkeley, CA, USA
yueqi@berkeley.edu

Julian McAuley
UC San Diego
La Jolla, CA, USA
jmcauley@ucsd.edu

Dong Wang
University of Illinois
Urbana-Champaign
Champaign, IL, USA
dwang24@illinois.edu

## Abstract

Large multimodal models (LMMs) exhibit enhanced capabilities in understanding and generating both textual and visual content. By leveraging item metadata, LMMs are also applied for recommendation and demonstrate improvements across diverse scenarios. However, the majority of existing methods explore static item attributes without considering additional contextual information (e.g., price, brand). Moreover, overlooking the interaction between the retrieval and ranking stages may lead to suboptimal solutions for fine-grained recommendations. In this work, we introduce PRIME: preference-optimized retrieval and ranking for efficient multimodal recommendation. PRIME operates in two stages: (i) a lightweight retriever identifies potential candidate items; (ii) an LMM learns to rank the retrieved candidates with detailed user history and multimodal features (e.g., text and image attributes). These features are incorporated into a carefully designed prompt, facilitating fine-grained transition patterns for user preference understanding. To optimize the inference efficiency of PRIME, we introduce verbalizer-based inference, which computes ranking scores for all candidate items in a single forward pass. Furthermore, we employ the LMM ranker to provide feedback on sampled candidate sets, enabling online preference optimization that refines the retriever model and improves the alignment between retrieval and ranking. As a result, PRIME can capture subtle user intentions and efficiently rank candidate items with minimal inference costs. Extensive experiments show the effectiveness and efficiency of PRIME, which consistently achieves superior performance over baseline methods.

## CCS Concepts

• **Information systems → Recommender systems**; • **Computing methodologies → Natural language processing**.

## Keywords

Large multimodal models, sequential recommendation, information retrieval, search and ranking

**ACM Reference Format:**
Zhenrui Yue, Huimin Zeng, Yueqi Wang, Julian McAuley, and Dong Wang. 2025. Preference-Optimized Retrieval and Ranking for Efficient Multimodal Recommendation . In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '25), August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3711896.3737088

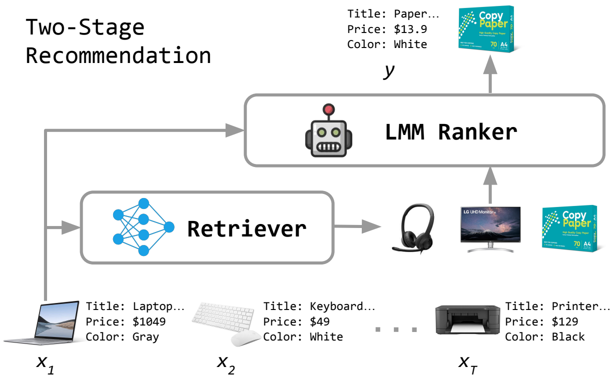


**Figure 1: Two-stage multimodal recommendation. A retriever generates candidate items based on the user's history. Then, a LMM ranker improves the quality by ranking the items.**

## 1 Introduction

Large language models (LLMs) and large multimodal models (LMMs) have led to significant improvements in understanding and generating language and visual content [6, 12, 41, 48, 50, 56]. By pretraining language models with an optional image encoder using vast amounts of text and image data, LLMs and LMMs acquire both extensive knowledge and a wide spectrum of abilities, including numerical reasoning and visual question answering. For example,

*KDD '25, Toronto, ON, Canada*

ACM ISBN 979-8-4007-1454-2/2025/08
https://doi.org/10.1145/3711896.3737088

GPT-4o excels in language and vision tasks and matches human evaluative performance across multimodal benchmarks [27]. Motivated by such advances, LLMs and LMMs are also employed in recommendation tasks like ranking and explanation generation, showcasing enhanced performance across multiple scenarios [31, 63, 73, 74, 85]. For instance, LMMs can be applied in two-stage sequential recommendation by ranking the candidate items retrieved in the initial stage [5, 25, 40, 79], as illustrated in Figure 1.

Nevertheless, the majority of existing works in sequential recommendation focus on extracting textual features from item metadata to enhance LLM recommendation performance [2, 36, 38, 73]. That is, this line of works is text-only (or even title-only), relying on fixed item attributes and often neglecting further modalities, such as images. In addition, the efficiency of applying LLMs and LMMs is largely overlooked, resulting in increased latency when generating lengthy item descriptions or over long sequences [17, 25, 68, 82]. In other words, inference through autoregressive generation lacks the speed necessary for real-time interactions, rendering such approaches impractical for real-world applications. As such, many studies focus on specific recommendation sub-tasks (e.g., CTR prediction) or employ two-stage approaches. However, in LLM / LMM-based two-stage solutions, the lack of alignment between the retrieval and ranking stages can often result in suboptimal performance for fine-grained recommendations [5, 46, 63, 79].

To incorporate additional item attributes or modalities for recommendations, current approaches flatten key-value pairs within item metadata or use image descriptors that capture visual features as item representations [19, 44, 62, 69, 72, 73]. For instance, VIP5 employs the CLIP vision encoder to incorporate pooled image features [15]. Yet such methods are tailored for visual inputs and fail to adaptively incorporate item metadata, leading to limited flexibility in handling diverse item attributes across modalities. As for inference efficiency, existing solutions use linear projections to bypass generation or employ sorting algorithms for set-wise ranking [31, 37, 45, 87]. However, the former approach is trained for specific settings (i.e., $N$-way classification), whereas the latter requires multiple inference with additional sorting, thereby limiting their applicability in efficient multimodal recommendation.

To this end, we propose PRIME: preference-optimized retrieval and ranking for efficient multimodal recommendation. PRIME not only provides an effective solution that includes retrieval and ranking, but also outperforms existing methods with enhanced inference efficiency. In particular, we leverage a lightweight retriever model, which efficiently retrieves candidate items regardless of the sequence length. In the subsequent stage, we construct input instructions that interleave text and image data, incorporating both user history and candidate items. Such cross-modal instructions are used to train a LMM-based ranker, enabling multimodal attributes by converting image pixels into features compatible with text embeddings. For efficient inference, we design a verbalizer-based approach to transform the language modeling head output into a probability distribution over the candidate items, eliminating the need for autoregressive token sampling. To improve retrieval-ranking alignment, we refine our two-stage framework by utilizing the learned ranker to provide feedback on sampled candidate sets. This feedback iteratively updates the retriever model through online preference learning, improving overall retrieval quality. This approach strengthens the alignment between both stages within PRIME, leading to improved retrieval and ranking collaboration. As a result, PRIME can train and infer efficiently to capture nuanced user intentions and deliver high-quality recommendations.

We summarize our contributions below[1]:

(1) We introduce PRIME, a LMM-based two-stage recommendation framework that optimizes retrieval efficiency and multimodal ranking through preference-aware learning.
(2) We develop a retrieval-ranking framework that integrates a lightweight retriever with a multimodal ranker. With our verbalizer-based inference, PRIME achieves efficient inference for fine-grained recommendation.
(3) To the best of our knowledge, we are the first to incorporate LMM ranker feedback for refining the retriever through online preference optimization, which enhances the dynamics between the retrieval and ranking stages in PRIME.
(4) We demonstrate the efficacy and effectiveness of our PRIME on benchmark datasets, where the proposed PRIME consistently outperforms state-of-the-art baseline methods with considerable improvements in recommendation performance and inference speed.

# 2 Related Work

## 2.1 Large Language Models

Recent advancements in large language models (LLMs) such as ChatGPT have been driven by the increase in model sizes and the expansion of pretraining corpora [6, 48, 50, 59, 65, 83]. These language models employ encoder, encoder-decoder or decoder-only architectures to enhance language modeling across a broad spectrum of downstream tasks [4, 10, 42, 52, 58, 64]. Among recent LLMs, the majority utilizes causal attention and stacked decoder layers to predict next tokens based on the input context [52, 66]. By incorporating extensive knowledge from the pretraining corpora, LLMs exhibit significant advantages in language understanding and generation tasks. Following this paradigm, recent LLMs like Qwen 2.5 and Llama 3 [12, 76] show substantial improvements on benchmark datasets and human evaluation, and therefore stand as state-of-the-art open-source models. In this work, we use a decoder-only LLM as our base language model and design an efficient training and inference framework for two-stage sequential recommendation.

## 2.2 Large Multimodal Models

In addition to advancements in LLMs, substantial progress has been made in large multimodal models (LMMs), with recent examples including GPT-4o and Gemini 2 [27, 56]. One common strategy for learning LMMs involves using separate text and image models to align input features across modalities, enabling the integration of visual and textual information for improved multimodal performance [55, 70, 81, 84]. For instance, CLIP adopts contrastive pretraining to align crossmodal representations between image-text pairs [51]. More recently, LMMs have been optimized via next-token prediction, using image encoder to convert visual inputs into token-level embeddings. This approach enables the generation of coherent and contextually accurate outputs for multimodal

[1] Our code is publicly available at https://github.com/Yueeeeeeee/PRIME.

tasks [1, 7, 33, 67]. An open-source LMM example is Llava, which utilizes the CLIP image encoder to transform image embeddings that interleave with text embeddings [41]. Nevertheless, LMMs have not been well studied for multimodal recommender systems; thus, in this study, we aim to design a two-stage sequential recommendation framework that efficiently generates candidate items, followed by a fine-grained LMM re-ranker that captures multimodal attributes for accurate next-item prediction.

## 2.3 LLM- and LMM-based Recommender Systems

LLMs are applied as recommender systems to understand user preferences and item characteristics for enhanced recommendation performance [14, 23, 24]. Current LLM-based recommenders leverage LLMs / embedding models to generate or re-rank recommended items [5, 35, 45, 63, 68, 75]. For example, Chat-REC utilizes ChatGPT to understand user preferences and improve both the interactivity and explainability of recommendations [13]. Another stream of LLM-based models focuses on developing learning strategies to further enhance user / item representations [11, 29, 38, 57, 73, 85]. For instance, RecFormer leverages a pretrained transformer to learn informative item and user history features to improve recommendation performance [34]. In contrast, LMM-based methods remain under-explored as recommender systems [43, 71, 77], with VIP5 being a notable example that incorporates image features from the CLIP image encoder for recommendation tasks [15]. However, such LLM- and LMM-based models are not tailored for flexible item attributes / modalities, nor are they optimized towards long sequences. Moreover, existing methods often suffer from efficiency issues due to the prolonged autoregressive generation in inference. As such, we propose a two-stage recommendation framework that leverages *multimodal* and *cross-attribute* item features to enhance user preference modeling, boosting both *performance* and *efficiency*.

# 3 Methodology

## 3.1 Setup

Our two-stage framework is based on sequential recommendation, in which the model $\boldsymbol{f}$ takes user interaction history $\boldsymbol{x}$ from dataset $\mathcal{X}$ as input. In particular, $\boldsymbol{x}$ is a sequence of interacted items $[x_1, x_2, \ldots, x_T]$ in chronological order, in which each item is defined in the item space $\mathcal{I}$ ($x_i \in \mathcal{I}, i = 1, 2, \ldots, T$). For each item $x$, its item ID and metadata are provided, with the metadata consisting of key-value pairs across text and image modalities (e.g., image, title, price, etc.). The output of the model is $\hat{\boldsymbol{y}}$, the top-$k$ recommended item scores $\{(x_i, s_i)|x_i \in \mathcal{I}, s_i \in \mathbb{R}\}_{i=1}^{k}$. Ideally, the highest ranked item in $\hat{\boldsymbol{y}}$ should be the ground truth item $y$ (i.e., $y = \arg\max \hat{\boldsymbol{y}}$). In our two-stage framework, the items are represented in different ways to maximize efficiency:

- *Retrieval*: In the retrieval stage, items are represented with unique item IDs (i.e., 1, 2, 3, ...), which are mapped into learnable item embedings to facilitate efficient candidate retrieval over large volumes of product data.
- *Ranking*: To perform fine-grained ranking, all item-level metadata (e.g., image, title, etc.) is accessible, enabling an in-depth analysis of multimodal item attributes and thereby enabling improved understanding of user preferences.

The two-stage recommender model $\boldsymbol{f}$ is parameterized by $\boldsymbol{\theta}$ (i.e., $\hat{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{\theta}; \boldsymbol{x})$). As mentioned, $\boldsymbol{f}$ consists of an efficient retriever model $\boldsymbol{f}_{\text{retr}}$ and a LMM-based ranker model $\boldsymbol{f}_{\text{rank}}$ ($\boldsymbol{f} = \boldsymbol{f}_{\text{rank}} \circ \boldsymbol{f}_{\text{retr}}$). Upon user history interactions, the retriever model (parametrized by $\boldsymbol{\theta}_{\text{retr}}$) returns a set of candidate items $\boldsymbol{c}$ (i.e., $\boldsymbol{c} = \boldsymbol{f}_{\text{retr}}(\boldsymbol{\theta}_{\text{retr}}; \boldsymbol{x})$). Next, the candidate items and the user's history serve as the input to the ranker model to produce the final ranking results. In other words, the multimodal ranker (parametrized by $\boldsymbol{\theta}_{\text{rank}}$) generates $\hat{\boldsymbol{y}}$ with $\hat{\boldsymbol{y}} = \boldsymbol{f}_{\text{rank}}(\boldsymbol{\theta}_{\text{rank}}; \boldsymbol{x}, \boldsymbol{c})$. The objective of our framework is to maximize the ground truth item score. That is, we seek to minimize the negative log likelihood loss $\mathcal{L}$ w.r.t. parameters $\boldsymbol{\theta}$ over $\mathcal{X}$:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x}, y) \sim \mathcal{X}} [\mathcal{L}(\boldsymbol{f}(\boldsymbol{\theta}; \boldsymbol{x}), y)]. \tag{1}$$

## 3.2 The Proposed PRIME

*3.2.1 Retriever Model.* For PRIME, it is possible to select an arbitrary model for the retrieval stage. In this work, we adopt the linear recurrent units for sequential recommendation (LRURec) as our retriever model $\boldsymbol{f}_{\text{retr}}$ [80], with comparison for different retriever models provided in Section 4. LRURec is a state-of-the-art ID-based recommender model that utilizes state space modeling (SSM) to efficiently train and infer on long input sequences. We present a recursive formulation of LRURec at the time step $t$:

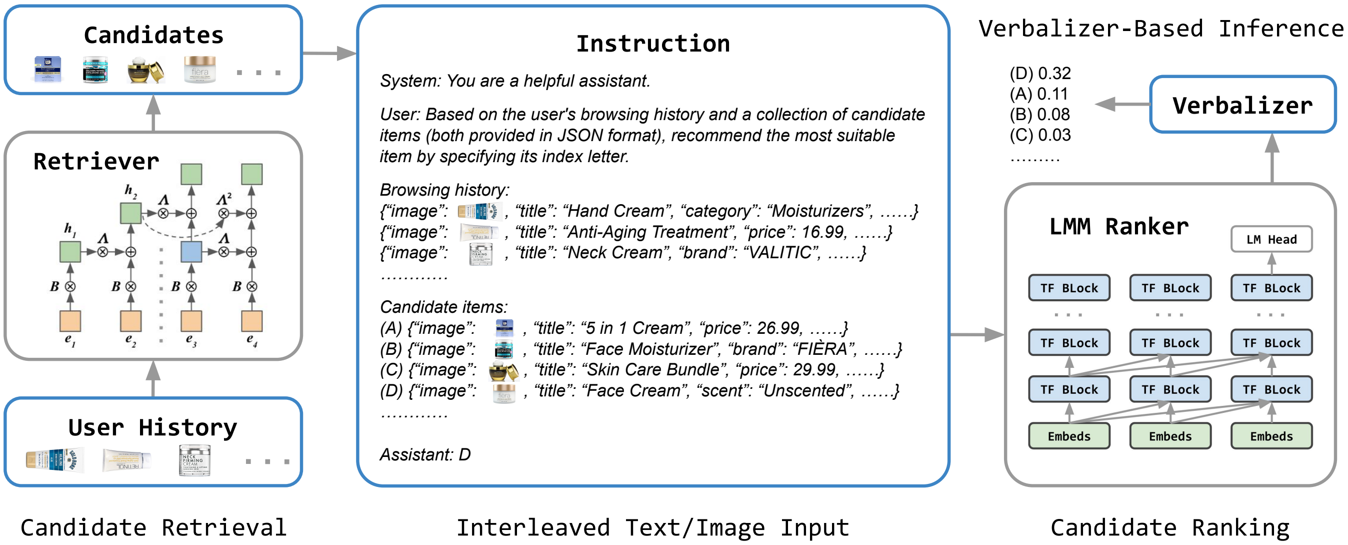
$$\boldsymbol{h}_t = \boldsymbol{A}\boldsymbol{h}_{t-1} + \boldsymbol{B}\boldsymbol{e}_t, \quad \boldsymbol{o}_t = \boldsymbol{C}\boldsymbol{h}_t + \boldsymbol{I}\boldsymbol{e}_t, \tag{2}$$

where $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ are learnable parameters of shape $\mathbb{R}^{H \times H}$ ($H$ is the hidden dimension), $\boldsymbol{I}$ is the identity matrix. $\boldsymbol{h}$, $\boldsymbol{e}$, $\boldsymbol{o}$ refer to the hidden state, input embedding (i.e., $\boldsymbol{e}_t$ is the learnt embedding for the item $x_t$) and output feature. Nevertheless, the repeated matrix multiplication can still be inefficient and may cause numerical instability as $t$ increases. To address this, we adopt matrix decomposition to reduce the frequent computation of $\boldsymbol{A}\boldsymbol{h}$ and enhance numerical stability of $\boldsymbol{h}$. Specifically, we decompose $\boldsymbol{A}$ with $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}^{-1}$, where $\boldsymbol{P}$ is invertible and $\boldsymbol{\Lambda}$ only consists of the diagonal elements $\lambda_1, \lambda_2, \ldots, \lambda_H$ (i.e., eignevalues). As such, computing $\boldsymbol{A}^n$ can be reduced to $\boldsymbol{P}\boldsymbol{\Lambda}^n\boldsymbol{P}^{-1}$, and thus substantially improving the efficiency on long interaction histories. Since the eigenvalues and eigenvectors of $\boldsymbol{A}$ may not always reside in $\mathbb{R}$, we extend $\boldsymbol{P}$, $\boldsymbol{\Lambda}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{h}$ to the complex space $\mathbb{C}$. We also initialize the matrix parameters $\boldsymbol{P}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ with truncated normal distribution [49, 80].

Based on the above decomposed linear recurrence, computing $\boldsymbol{A}\boldsymbol{h}$ can be replaced with matrix $\boldsymbol{P}$ and the diagonal $\boldsymbol{\Lambda}$, namely $\boldsymbol{P}\boldsymbol{\Lambda}^n\boldsymbol{P}^{-1}\boldsymbol{h}$. We then integrate $\boldsymbol{P}$ in the variables by assigning $\boldsymbol{P}^{-1}\boldsymbol{h} \rightarrow \boldsymbol{h}$, $\boldsymbol{P}^{-1}\boldsymbol{B} \rightarrow \boldsymbol{B}$ and $\boldsymbol{C}\boldsymbol{P}^{-1} \rightarrow \boldsymbol{C}$, which further reduces the formulation of $\boldsymbol{h}_t$ and output $\boldsymbol{o}_t$ in Equation (2) to:

$$\boldsymbol{h}_t = \boldsymbol{\Lambda}\boldsymbol{h}_{t-1} + \boldsymbol{B}\boldsymbol{e}_t, \quad \boldsymbol{o}_t = \Re(\boldsymbol{C}\boldsymbol{h}_t) + \boldsymbol{I}\boldsymbol{e}_t, \tag{3}$$

where the $\Re$ operation extracts the real part of the complex input $\boldsymbol{C}\boldsymbol{h}_t$. To improve the training stability and computation efficiency of linear recurrence, we represent the complex operations in polar form (i.e., $\boldsymbol{\Lambda} = \boldsymbol{r}e^{j\boldsymbol{\theta}} = \boldsymbol{r}(\cos(\boldsymbol{\theta}) + j\sin(\boldsymbol{\theta}))$) and adopt the ring initialization for the parameters within $\boldsymbol{\Lambda}$ [49, 80]. Aside from matrix decomposition and the above parameterization, computing Equation (3) is performed step-wise for each interaction in a sequence.



(D) 0.32
(A) 0.11
(B) 0.08
(C) 0.03
………

*System: You are a helpful assistant.*

*User: Based on the user's browsing history and a collection of candidate items (both provided in JSON format), recommend the most suitable item by specifying its index letter.*

*Browsing history:*
*{"image": , "title": "Hand Cream", "category": "Moisturizers", ……}*
*{"image": , "title": "Anti-Aging Treatment", "price": 16.99, ……}*
*{"image": , "title": "Neck Cream", "brand": "VALITIC", ……}*
*…………*

*Candidate items:*
*(A) {"image": , "title": "5 in 1 Cream", "price": 26.99, ……}*
*(B) {"image": , "title": "Face Moisturizer", "brand": "FIÈRA", ……}*
*(C) {"image": , "title": "Skin Care Bundle", "price": 29.99, ……}*
*(D) {"image": , "title": "Face Cream", "scent": "Unscented", ……}*
*…………*

*Assistant: D*

**Figure 2: The proposed PRIME. The left subfigure illustrates the retrieval stage with LRURec that generates candidate items. Using an instruction template with interleaved text and images, we transform user histories and candidates into multimodal input instruction for the LMM, followed by the efficient verbalizer-based ranking stage (right subfigure).**

To address this issue, we exploit the linearity of LRURec and implement parallel scanning to achieve $O(\log(t))$ time complexity w.r.t. input lengths [3, 32]. In sum, the adopted LRURec retriever is an efficient SSM model and learns item features to capture user transition patterns. Unlike recurrent neural networks, LRURec offers the benefits of both parallel training and incremental inference, and therefore facilitates efficient learning and rapid inference in the retrieval stage. In our PRIME, we collect the top-$k$ candidate items from LRURec for each input sequence, the top-$k$ items are stored and passed to the following ranking stage.

*3.2.2 LMM-based Ranker.* The primary goal of our LMM-based ranker is to incorporate diverse item attributes across modalities, enhancing the understanding of user-item interactions and thereby improving ranking performance. Therefore, the ranker model consists of a pretrained LLM for text understanding and an image encoder that extracts visual features based on input item images. Specifically, we leverage a pre-trained vision transformer (denoted with $f_{\text{vis}}$) as image encoder and choose Qwen2 as the base LLM, our model weights are initialized with Qwen2 VL 2B Instruct [67, 76].

In the ranking stage, the primary challenge for item representation involves understanding long context with multiple images from user history and candidate items. To this end, we utilize a JSON-like structure to describe each item with heterogeneous attributes across modalities, as we find the structured item template more beneficial than using separators alone to divide items and attributes. We illustrate an example prompt in Figure 2, with more details provided in Appendix A. To construct the text input, we design an instruction to describe the ranking task, followed by the user's history interactions and candidate items from the retriever. For each of the candidate items, we prepend a unique index letter to the item for identification. During training, the LMM ranker learns to respond with the index letter of the label item, whereas in inference, the response field is left empty for the model to predict.

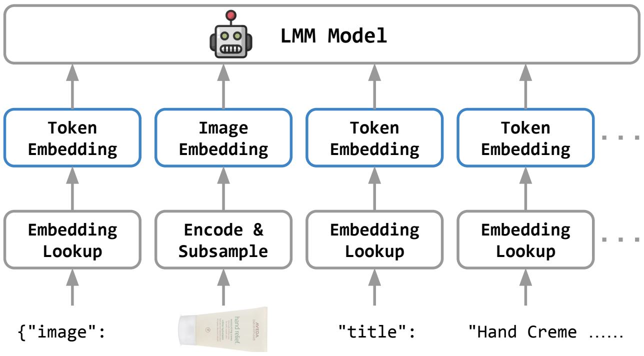


**Figure 3: Interleaved text / image input of PRIME, where vision token embeddings are extracted from the input image and projected to the token embedding space.**

For the text input, we tokenize the input string and convert it into token embeddings. For each interleaved image $x^{\text{v}}$ (<image> used as placeholder), the corresponding features can be extracted from the output of the image encoder. Here, we denote the extracted visual representation of input image $x^{\text{v}}$ with $z^{\text{v}} \in \mathbb{R}^{L \times H}$ (i.e., $z^{\text{v}} = f_{\text{vis}}(x^{\text{v}})$), with $L$ denoting the number of vision tokens for each image. Considering the extensive vision tokens (e.g., 576 for CLIP vision encoder using 14 as patch size [51]) and presence of multiple images in each example, we adopt image token subsampling to reduce the number of visual tokens. Specifically, we first adjust the image resolution from 56×56 to 168×168, preserving the most image details while optimizing the size for processing. Next, we apply a MLP layer to compress the size of adjacent 2×2 patches into a single vision token, followed by applying 2D rotational position embeddings to encode image position information [60, 67]. As such, the reduced number of vision tokens ranges from 4 to 36, significantly lowering the image token count compared to existing LMMs. Consequently, the interleaved input can be efficiently integrated regardless of the item attributes and modalities. Our interleaved input processing is illustrated in Figure 3.

### 3.2.3 *Verbalizer-Based Inference.*

Despite being guided with instructions, the LMM does not efficiently output ranking scores for all candidate items. To solve this problem, existing works prompt language models to generate a ranked list of candidate items [5, 25, 75]. Alternatively, point-wise, pair-wise or set-wise ranking approaches have been introduced to compute the similarity between input queries and candidates [37, 45, 87]. However, both approaches are computationally expensive for an increased number of candidate items and often require further post-processing to obtain the item ranks. Unlike such methods, we leverage a simple parameter-free verbalizer that efficiently maps the LM head output (i.e., logits over all tokens) to ranking scores of the candidate items. Recall that we use index letters to differentiate candidate items (e.g., (A) hand cream, (B) facial cleanser etc.) and map the ground truth item to its corresponding index letter. Consequently, we instruct the model to output an index letter, and candidate scores are derived by extracting the logits linked to these letters from the LM head. In other words, the retrieved scores correspond to the next-token logits of the candidate item indices. As such, the candidate item scores can be obtained within a single forward pass (see Figure 4). The advantages of our verbalizer-based inference are two-fold: (1) PRIME enables training without requiring a ranked list as ground truth; and (2) PRIME skips token-by-token generation, producing ranking scores within a single forward pass. Therefore, our approach not only simplifies the ranking process, but also enhances efficiency by avoiding the decoding process and further post-processing, making it highly suitable for real-time recommendation.

During training, the label item is mapped to the corresponding index letter, and our goal is to maximize the probability of the label given user history and candidate items (i.e., $P_{\theta_{\text{rank}}}(y|\boldsymbol{x})$). That is, we compute the loss on the LLM's response, which includes the index letter and EOS tokens. This setup enables the LLM to understand input queries and appropriately rank candidate items. Notably, the verbalizer-based inference aligns with the next-token prediction task, and thus PRIME can be seamlessly integrated with various causal language modeling tasks to facilitate multi-tasking capabilities. By employing index letters and a simple verbalizer, PRIME learns to rank items based on user history while maintaining its generative capabilities. Furthermore, the ranking stage requires just a single forward pass to compute scores for all candidate items, significantly enhancing inference speed. To further improve PRIME's efficiency in training, we implement low-rank adaptation (LoRA) for instruction-tuning the LMM, we also adopt flash-attention and gradient accumulation for long context efficiency. Additionally, we quantize the pretrained weights and learn limited parameters ($\sim$ 0.5% of the LMM parameters) to conserve memory, balancing between resource usage and processing speed [8, 9, 26].

### 3.2.4 *Online Preference Optimization.*

While PRIME provides a multimodal retrieval and ranking framework, these two stages are trained independently without accounting for their interactions. Yet neglecting the interactions between these stages may result in a suboptimal solution for two-stage recommendation [21, 46]. To address this limitation, we propose an online, on-policy preference optimization framework that leverages feedback from the LMM ranker. By sampling candidate sets and maximizing the probability of preferred over dispreferred ones, this framework iteratively refines the retriever for optimal retrieval–ranking collaboration [16, 53].

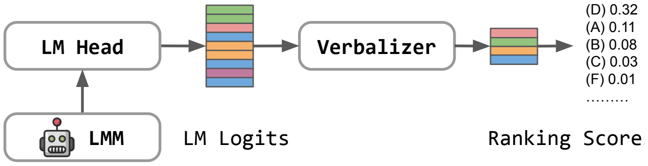


**Figure 4: The proposed verbalizer-based inference in our PRIME, which transforms LM head logits into ranking scores over the index letters of the candidate items.**

| Name | #User | #Item | #Image | #Inter. | Density |
|---|---|---|---|---|---|
| **Beauty** | 22,363 | 12,101 | 12,023 | 198k | 0.073 |
| **Clothing** | 39,387 | 23,033 | 22,299 | 278k | 0.031 |
| **Sports** | 35,598 | 18,357 | 17,943 | 296k | 0.045 |
| **Toys** | 19,412 | 11,924 | 11,895 | 167k | 0.072 |

**Table 1: Statistics of the datasets.**

Specifically, we focus on post-training the retriever $f_{\text{retr}}$, as it is smaller in size and can be optimized to provide an improved candidate set. As a result, optimizing the retriever model offers greater potential for improvement compared to refining the ranking stage [39]. For a set of candidates $\boldsymbol{c} = \{c_i\}_{i=1}^{k}$ sampled from the retriever, we define the joint likelihood of $\boldsymbol{c}$ given input $\boldsymbol{x}$ as:

$$P_{\theta_{\text{retr}}}(\boldsymbol{c}|\boldsymbol{x}) = \prod_{i=1}^{k} P_{\theta_{\text{retr}}}(c_i|\boldsymbol{x}), \tag{4}$$

where $P_{\theta_{\text{retr}}}(c_i|\boldsymbol{x})$ denotes the probability of item $c_i$ conditioned on $\boldsymbol{x}$. The odds of retrieving $\boldsymbol{c}$ given $\boldsymbol{x}$ can be defined by:

$$\mathsf{odds}_{\theta_{\text{retr}}}(\boldsymbol{c}|\boldsymbol{x}) = \frac{P_{\theta_{\text{retr}}}(\boldsymbol{c}|\boldsymbol{x})}{1 - P_{\theta_{\text{retr}}}(\boldsymbol{c}|\boldsymbol{x})}. \tag{5}$$

Similar to [22], for input $\boldsymbol{x}$, we define the odds ratio OR for two candidate sets $\boldsymbol{c}_w$ and $\boldsymbol{c}_l$ as:

$$\mathsf{OR}_{\theta_{\text{retr}}}(\boldsymbol{c}_w, \boldsymbol{c}_l, \boldsymbol{x}) = \frac{\mathsf{odds}_{\theta_{\text{retr}}}(\boldsymbol{c}_w|\boldsymbol{x})}{\mathsf{odds}_{\theta_{\text{retr}}}(\boldsymbol{c}_l|\boldsymbol{x})}, \tag{6}$$

which quantifies how much more likely $\boldsymbol{c}_w$ is to be retrieved compared to $\boldsymbol{c}_l$ for the input sequence $\boldsymbol{x}$. By using the learnt ranker model $f_{\text{rank}}$ as annotator, our objective is to maximize the probability of retrieving $\boldsymbol{c}_w$ over $\boldsymbol{c}_l$, thereby further optimizing the retriever $f_{\text{retr}}$ to improve the synergy between retrieval and ranking.

To construct the candidate sets $\boldsymbol{c}_w$ and $\boldsymbol{c}_l$, we sample two sets of size $k$ from the retriever model $f_{\text{retr}}$. Both sets are evaluated by the ranker, and the set with the higher ranking score is designated as $\boldsymbol{c}_w$, while the other is assigned as $\boldsymbol{c}_l$. In other words, the candidate set

| **Dataset** | **Metric** | **ID-Based** | | | | **Text and Multimodal** | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | SAS | BERT | FMLP | LRU | UniS. | P5 | VQRec | RecF. | LLMR. | MMSSL | VIP5 | PRIME |
| **Beauty** | N@5 | 0.0274 | 0.0275 | 0.0318 | 0.0339 | 0.0274 | 0.0358 | 0.0303 | 0.0258 | 0.0474 | 0.0189 | 0.0415 | **0.0525** |
| | R@5 | 0.0456 | 0.0420 | 0.0539 | 0.0565 | 0.0484 | 0.0490 | 0.0514 | 0.0428 | 0.0669 | 0.0308 | 0.0517 | **0.0736** |
| | N@10 | 0.0364 | 0.0350 | 0.0416 | 0.0438 | 0.0375 | 0.0409 | 0.0411 | 0.0341 | 0.0554 | 0.0252 | 0.0464 | **0.0609** |
| | R@10 | 0.0734 | 0.0653 | 0.0846 | 0.0871 | 0.0799 | 0.0646 | 0.0849 | 0.0686 | 0.0940 | 0.0506 | 0.0667 | **0.0998** |
| **Clothing** | N@5 | 0.0075 | 0.0062 | 0.0091 | 0.0104 | 0.0127 | 0.0140 | 0.0104 | 0.0137 | 0.0201 | 0.0089 | 0.0155 | **0.0213** |
| | R@5 | 0.0134 | 0.0100 | 0.0167 | 0.0192 | 0.0221 | 0.0253 | 0.0197 | 0.0234 | 0.0282 | 0.0146 | 0.0269 | **0.0309** |
| | N@10 | 0.0104 | 0.0084 | 0.0123 | 0.0140 | 0.0175 | 0.0199 | 0.0149 | 0.0192 | 0.0239 | 0.0122 | 0.0218 | **0.0246** |
| | R@10 | 0.0227 | 0.0169 | 0.0266 | 0.0304 | 0.0372 | 0.0347 | 0.0336 | 0.0405 | 0.0396 | 0.0249 | **0.0462** | 0.0410 |
| **Sports** | N@5 | 0.0143 | 0.0137 | 0.0194 | 0.0204 | 0.0141 | 0.0138 | 0.0173 | 0.0127 | 0.0257 | 0.0123 | 0.0159 | **0.0281** |
| | R@5 | 0.0267 | 0.0215 | 0.0329 | 0.0344 | 0.0237 | 0.0215 | 0.0304 | 0.0211 | 0.0383 | 0.0198 | 0.0253 | **0.0413** |
| | N@10 | 0.0210 | 0.0181 | 0.0252 | 0.0266 | 0.0195 | 0.0167 | 0.0235 | 0.0173 | 0.0326 | 0.0163 | 0.0188 | **0.0342** |
| | R@10 | 0.0474 | 0.0355 | 0.0508 | 0.0536 | 0.0408 | 0.0310 | 0.0497 | 0.0350 | 0.0590 | 0.0321 | 0.0342 | **0.0603** |
| **Toys** | N@5 | 0.0291 | 0.0241 | 0.0308 | 0.0366 | 0.0254 | 0.0547 | 0.0314 | 0.0292 | 0.0577 | 0.0173 | 0.0556 | **0.0599** |
| | R@5 | 0.0534 | 0.0355 | 0.0534 | 0.0601 | 0.0477 | 0.0631 | 0.0577 | 0.0501 | 0.0773 | 0.0286 | 0.0651 | **0.0831** |
| | N@10 | 0.0380 | 0.0299 | 0.0408 | 0.0463 | 0.0362 | 0.0569 | 0.0423 | 0.0398 | 0.0647 | 0.0224 | 0.0580 | **0.0695** |
| | R@10 | 0.0807 | 0.0535 | 0.0845 | 0.0901 | 0.0811 | 0.0701 | 0.0915 | 0.0832 | 0.1088 | 0.0445 | 0.0726 | **0.1128** |
| **Average** | N@5 | 0.0196 | 0.0179 | 0.0228 | 0.0253 | 0.0199 | 0.0296 | 0.0224 | 0.0204 | 0.0377 | 0.0144 | 0.0321 | **0.0405** |
| | R@5 | 0.0348 | 0.0273 | 0.0392 | 0.0426 | 0.0355 | 0.0397 | 0.0398 | 0.0344 | 0.0527 | 0.0235 | 0.0423 | **0.0572** |
| | N@10 | 0.0265 | 0.0229 | 0.0300 | 0.0327 | 0.0277 | 0.0336 | 0.0305 | 0.0276 | 0.0442 | 0.0190 | 0.0363 | **0.0473** |
| | R@10 | 0.0561 | 0.0428 | 0.0616 | 0.0653 | 0.0598 | 0.0501 | 0.0649 | 0.0568 | 0.0754 | 0.0380 | 0.0549 | **0.0785** |

**Table 2: Sequential recommendation results, where each row signifies a dataset and each column a model. We use SAS, BERT, FMLP, LRU, UniS., RecF. and LLMR. to abbreviate SASRec, BERT4Rec, FMLP-Rec, LRURec, UniSRec, RecFormer and LlamaRec. N and R are NDCG and Recall, we mark the best results in bold and underline the second best results.**

where the target $y$ achieves a higher rank (determined by $f_{\text{rank}}$) is preferred. Using the input sequence $\boldsymbol{x}$ and the constructed candidate sets $\boldsymbol{c}_w$ and $\boldsymbol{c}_l$, we compute the odds ratio preference loss $\mathcal{L}_{\text{or}}$ with:

$$\mathcal{L}_{\text{or}} = -\log \sigma(\log \frac{\mathsf{odds}_{\theta_{\text{retr}}}(\boldsymbol{c}_w|\boldsymbol{x})}{\mathsf{odds}_{\theta_{\text{retr}}}(\boldsymbol{c}_l|\boldsymbol{x})}), \tag{7}$$

which maximizes the likelihood of $\boldsymbol{c}_w$ over $\boldsymbol{c}_l$. To maintain retriever performance without relying a reference model, we also incorporate the cross entropy loss into the preference optimization process. Consequently, the overall objective can be defined as:

$$\mathcal{L} = \mathcal{L}_{\text{ce}} + \beta\mathcal{L}_{\text{or}}, \tag{8}$$

where $\mathcal{L}_{\text{ce}}$ is the cross entropy loss for training the retriever model and $\beta$ is a hyperparameter that controls the influence of preference learning. In summary, we introduce an online, on-policy preference optimization phase to bridge the gap between the retrieval and ranking stages. By sampling from the retriever and using feedback from the LMM ranker, this phase optimizes $f_{\text{retr}}$ and directly improves the collaboration between the retrieval and re-ranking stages. The optimization integrates a preference term with a supervised training loss, ensuring that the retriever produces higher-quality candidates and achieves improved alignment for re-ranking.

## 3.3 Overall Framework

Overall, PRIME introduces a novel two-stage retrieval and ranking framework that enhances both efficiency and alignment in multimodal recommendation. It comprises two key components: (1) a lightweight retriever LRURec that leverages linear recurrence to process long user histories and identify candidate items; and (2) an instruction-tuned LMM ranker that captures fine-grained user preferences by integrating multimodal item attributes with user history. Unlike existing methods [5, 15, 34, 73], PRIME introduces verbalizer-based inference, which eliminates the need for autoregressive generation during test-time, allowing ranking scores to be computed efficiently in a single forward pass. Additionally, PRIME incorporates an online, on-policy preference optimization phase, where the ranker provides feedback to iteratively refine the retriever and thereby improving retrieval-ranking alignment. These design choices make PRIME a highly adaptable and effective framework for multimodal recommendation.

# 4 Experiments

## 4.1 Experiment Settings

*4.1.1 Datasets.* Our model is evaluated on four datasets: *Beauty*, *Clothing, Shoes & Jewelry* (Clothing), *Sports & Outdoors* (Sports) and *Toys & Games* (Toys) [18, 47]. For preprocessing, we follow [5, 15] to construct input sequences in chronological order and iteratively filter users and items with fewer than 5 interactions (i.e., 5-core). We include item attributes including *title*, *image*, *price*, *brand* and *categories*. We report the dataset statistics with number of users (#User), items (#Item), images (#Image), interactions (#Inter.) and dataset density (Density, in %), with details presented in Table 1.

| **PRIME Variants** | **Beauty** | | **Clothing** | | **Sports** | | **Toys** | | **Average** | |
|---|---|---|---|---|---|---|---|---|---|---|
| | N@10 | R@10 | N@10 | R@10 | N@10 | R@10 | N@10 | R@10 | N@10 | R@10 |
| **Original PRIME** | **0.0609** | 0.0998 | **0.0246** | **0.0410** | **0.0342** | **0.0603** | **0.0695** | 0.1128 | **0.0473** | **0.0785** |
| **PRIME w/o Preference Optim.** | 0.0593 | 0.0977 | 0.0240 | 0.0400 | 0.0326 | 0.0571 | 0.0693 | 0.1127 | 0.0463 | 0.0769 |
| **PRIME w/ SASRec Retriever** | 0.0497 | 0.0942 | 0.0214 | 0.0373 | 0.0325 | 0.0563 | 0.0603 | 0.1017 | 0.0410 | 0.0724 |
| **PRIME w/ FMLP-Rec Retriever** | 0.0588 | **0.1017** | 0.0239 | 0.0403 | 0.0333 | 0.0586 | 0.0685 | 0.1036 | 0.0461 | 0.0785 |
| **PRIME w/ Llava Ranker (7B)** | 0.0590 | 0.0943 | 0.0238 | 0.0390 | 0.0329 | 0.0576 | 0.0679 | **0.1132** | 0.0459 | 0.0760 |
| **PRIME w/ Mantis Ranker (8B)** | 0.0599 | 0.0973 | 0.0242 | 0.0406 | 0.0321 | 0.0589 | 0.0680 | 0.1128 | 0.0460 | 0.0774 |

**Table 3: Ablation results of PRIME, where each row represents a variant of PRIME and each column denotes a dataset. N and R are NDCG and Recall, we also mark the best results in bold and underline second best results.**

4.1.2 *Baseline Methods.* We adopt multiple state-of-the-art recommender models for baseline comparison, including *ID-based* and *text & multimodal* methods. In particular, we adopt *ID-based* SASRec, BERT4Rec, FMLP-Rec and LRURec [30, 61, 80, 86]. Text and multimodal methods include UniSRec, P5, VQRec, RecFormer, LlamaRec, MMSSL and VIP5 [14, 15, 23, 24, 34, 72, 79]. The baseline models are configured and trained according to the methodologies described in the original works, with unspecified hyperparameters searched / used as recommended. All baseline methods and PRIME are evaluated under identical conditions.

4.1.3 *Evaluation.* In our evaluation, we adopt the leave-one-out approach where for each data example, we use the last item for testing, the second last item for validation, and the remaining items for training [15, 20, 24, 78]. We use 50 as maximum item length for all methods and datasets. The evaluation metrics are normalized discounted cumulative gain (NDCG@$k$) and recall (Recall@$k$, equivalent to hit rate) with $k \in [5, 10]$. For all involved methods, we save the model with the best validation NDCG@10 scores for testing, where we compute the metric values by ranking the ground-truth item against all other items (including history items).

## 4.2 Experiment Results

4.2.1 *Main Results.* We first evaluate the performance of PRIME along with various baselines, as reported in Table 2. In this table, each row corresponds to a dataset, while each column denotes a different recommender model. From the reported results we observe: (1) the proposed PRIME consistently achieves high performance across all metrics and selected datasets, with the only exception of Recall@10 score on the Clothing dataset. The subpar score of Recall@10 on Clothing may be attributed to the limitations of the retriever, where LRURec scores 0.304 on Recall@10 and falls short compared to 0.462 of VIP5. (2) PRIME significantly outperforms ID-based, text-based and multimodal baseline methods. For instance, in the Beauty dataset, PRIME shows a substantial improvement in Recall@5, increasing from 0.0669 to 0.0736, indicating a 10.01% improvement over the best-performing baseline. (3) Compared to the best baseline LlamaRec (7B), PRIME consistently achieves superior performance despite being significantly smaller (2B), highlighting the effectiveness of our multimodal recommendation approach and online preference optimization. (4) PRIME performs well averaging across all datasets, suggesting robustness and generalizability on different data categories and recommendation contexts. For example, PRIME outperforms the second best baseline on Recall@5 by a significant margin of 8.54%. (5) PRIME particularly excels in ranking the retrieved candidate items. For example, PRIME achieves average @5 improvements of 7.98% compared to 5.56% gains on @10 metrics, indicating enhanced user preference estimation and precise ranking performance. Overall, PRIME demonstrates significant advantages in handling complex multimodal data and can effectively rank candidate items, and thereby improving retrieval and ranking performance while maintaining inference efficiency.

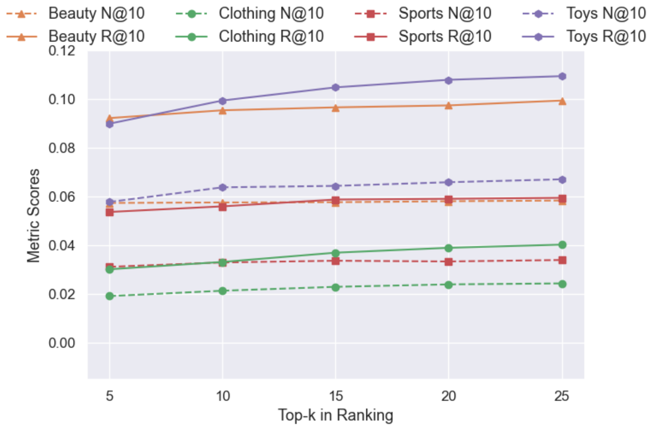


**Figure 5: Top-$k$ sensitivity in ranking, the x-axis denotes the candidate number and y-axis indicates the ranking scores.**

4.2.2 *Ablation Study.* We evaluate the proposed method by ablating modules in PRIME. In particular, we study different variants to assess the effectiveness of individual components. The included variants are: (1) PRIME without online preference optimization (PRIME w/o preference optim.); (2) PRIME with varying retriever, where we select SASRec and FMLP-Rec while keeping the LMM ranker fixed [30, 86]; (3) PRIME with alternative LMM ranker, replacing the LMM model with Llava 1.5 (Vicuna 7B) and Mantis (Llama3 8B) [28, 41]. From the ablation results in Table 3, we observe: (1) *PRIME consistently outperforms its variants on both NDCG@10 and Recall@10 across all datasets.* This indicates that

| PRIME Variants | Beauty | | Clothing | | Sports | | Toys | | Average | |
|---|---|---|---|---|---|---|---|---|---|---|
| | N@10 | R@10 | N@10 | R@10 | N@10 | R@10 | N@10 | R@10 | N@10 | R@10 |
| **Original PRIME** | **0.0609** | **0.0998** | **0.0246** | **0.0410** | **0.0342** | **0.0603** | **0.0695** | **0.1128** | **0.0473** | **0.0785** |
| **PRIME w/ CLS Pooling** | 0.0601 | 0.0993 | 0.0241 | 0.0406 | 0.0336 | 0.0601 | 0.0680 | 0.1103 | 0.0464 | 0.0776 |
| **PRIME w/ Title Only** | 0.0548 | 0.0934 | 0.0231 | 0.0383 | 0.0318 | 0.0568 | 0.0650 | 0.1080 | 0.0437 | 0.0741 |
| **PRIME w/o Image** | 0.0600 | 0.0965 | 0.0235 | 0.0400 | 0.0329 | 0.0587 | 0.0664 | 0.1102 | 0.0457 | 0.0763 |
| **PRIME w/o Title** | 0.0526 | 0.0897 | 0.0190 | 0.0350 | 0.0301 | 0.0542 | 0.0614 | 0.1049 | 0.0408 | 0.0709 |

**Table 4: Item attribute analysis of PRIME, where each row represents a variant of item representation and each column denotes a dataset. N and R are NDCG and Recall, we mark the best results in bold and underline second best results.**

the original configuration delivers optimal recommendation performance. (2) *The proposed online preference optimization enhances the dynamics between retrieval and ranking.* This can further improve the NDCG@10 and Recall@10 scores by 2.16% and 2.08% respectively. (3) *PRIME shows better performance compared to variants with different retrievers (SASRec and FMLP-Rec).* On average, PRIME achieves 0.0473 in NDCG@10 compared to 0.0461 of the second best retriever setup. This suggests that the LRURec retriever used in PRIME is both effective and efficient in retrieving candidate items. (4) *Variants with alternative LMM rankers, such as Llava and Mantis, generally underperform compared to PRIME despite their larger sizes.* In few cases, Llava and Mantis slightly outperform PRIME due to increased model capacity and enhanced language understanding. In summary, the results imply that the original PRIME configuration offers the best balance of performance and efficiency.

*4.2.3 Top-k Sensitivity.* In addition to the ablation, we also study the sensitivity of the number of candidates, aiming to investigate the optimal $k$ at which recommendation performance peaks. To achieve this, we evaluate how NDCG@10 and Recall@10 scores change with different number of candidates. We present the results visually in Figure 5, where the x-axis represents the number of candidates and the y-axis represents the metric values. Based on the ranking performance with varying $k$, the optimal number of candidates for re-ranking lies between 20 and 25. For lower values of $k$, expanding the candidate set consistently leads to performance improvements. For example, incrementally increasing the number of $k$ in Toys yields improvements of 10.48%, 5.35%, 2.83% and 1.42% in Recall@10, indicating diminishing gains for further increases. In contrast, using 20 as the candidate size preserves 98.18% of the average Recall@10 performance, while also reducing the inference cost (due to reduced context length). Furthermore, we observe that performance varies across datasets, with greater gains achieved on Clothing and Toys compared to Beauty and Sports. Overall, setting the number of candidates $k$ between 20 and 25 optimizes the balance between computational efficiency and recommendation accuracy.

*4.2.4 Attribute Analysis.* To assess the significance of various item attributes for recommendation performance, we evaluate PRIME with different item attributes and modalities. Specifically, we evaluate PRIME under different attribute settings: using CLS token for image pooling (i.e., $|z^{\mathrm{V}}| = 1$), title-only, without the image and without title. The results are presented in Table 4, with rows representing different variants of PRIME and the columns representing the datasets. We observe that removing image features, item title or further attributes leads to consistent performance drops. Notably, using CLS pooled image features can largely preserve the performance of the LMM re-ranker, offering an efficient alternative for multimodal recommendation. Among further variants of PRIME, removing the item title results in the most significant performance decline (with a 13.74% drop in NDCG@10), followed by the title-only variant. In summary, PRIME demonstrates both generalizability and effectiveness by incorporating a full spectrum of features with both text and image attributes, highlighting the effectiveness of our multimodal and cross-attribute approach for delivering enhanced recommendation performance.

*4.2.5 Cross-Domain Analysis.* We also analyze the robustness and domain generalization of the proposed PRIME. This evaluation combines the target-domain retriever and the LMM ranker trained on different source domains, with the results visually presented in Figure 6. Each subfigure represents a target dataset, and the groups within each subfigure corresponding to different source domains. The bars illustrate the performance of PRIME *relative to in-domain results* in percentages. From the results, we note several key observations: (1) PRIME demonstrates exceptional generalization capabilities, maintaining high performance across domains. Notably, for all metrics across source-target combinations, PRIME achieves over 90% relative performance compared to in-domain results, even for challenging source-target pairs. (2) While the relative performance occasionally lowers (e.g., NDCG@5 in Clothing → Beauty at 90.8%), these drops remain moderate and still outperform most baseline methods in Table 2, further highlighting PRIME's efficacy in handling diverse and heterogeneous domains. (3) Surprisingly, cross-domain results could exceed in-domain performance. For example, in Beauty → Toys, NDCG@5 and Recall@5 scores reach 101.7% and 101.9% respectively. This suggests that certain source features enhance target-domain recommendations, potentially due to similar user behavior and item characteristics. (4) Consistency is evident across all scenarios, with Recall@5 and Recall@10 showing similar trends to NDCG@5 and NDCG@10. This indicates that PRIME performs well and is consistently reliable for both retrieval and ranking stages. In summary, the cross-domain evaluation results show that PRIME achieves robust and reliable performance across diverse dataset combinations, effectively bridging domain gaps and preserving high quality in multimodal recommendation. These results underline the adaptability and practical applicability of PRIME in dynamic, cross-domain environments.

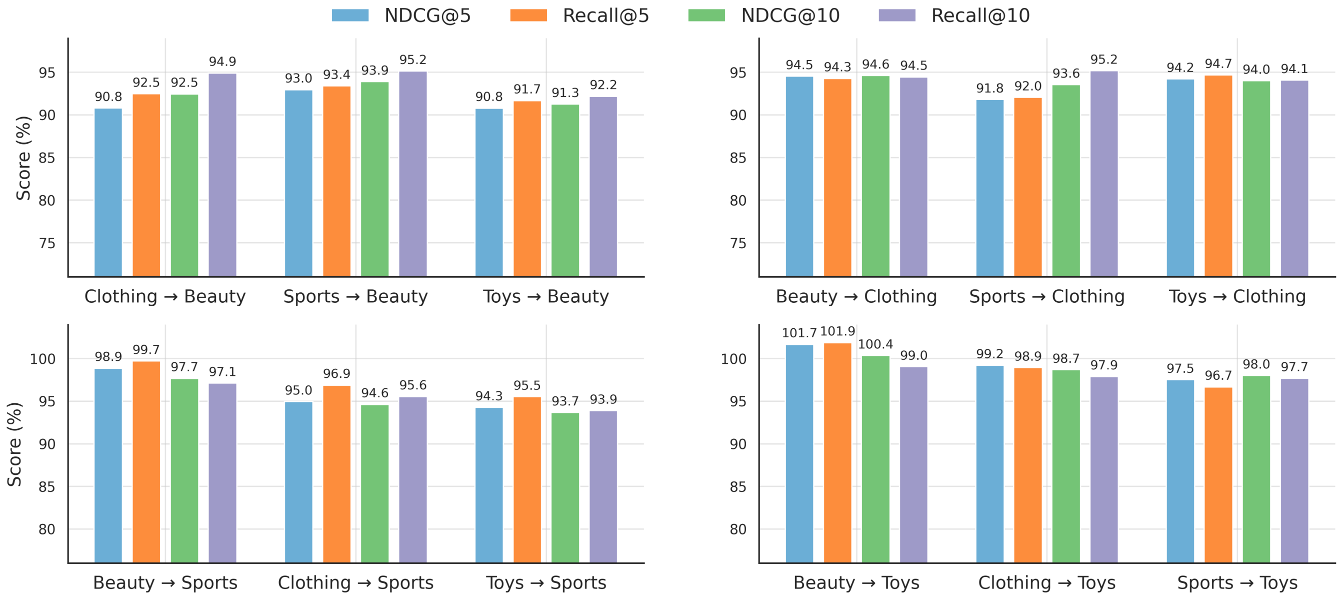


**Figure 6: Cross-domain evaluation results for all source-target dataset combinations. Each subfigure corresponds to a target dataset, and each group within the subfigure representing a source dataset. The bars depict relative metric performance as a percentage compared to the in-domain evaluation results (see Table 2).**

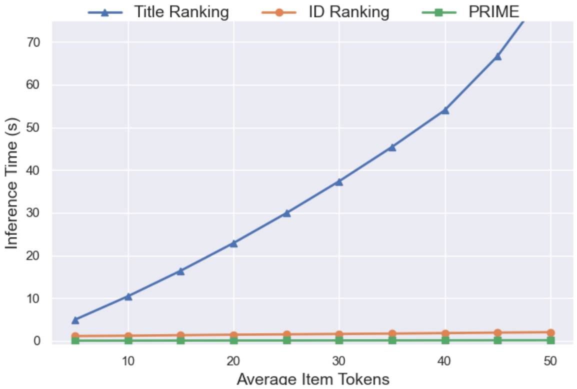


**Figure 7: Inference efficiency of PRIME compared to list-wise generation of item titles and IDs. X-axis represents average item token length and y-axis represents inference time.**

*4.2.6 Inference Efficiency.* Finally, we assess the efficiency of PRIME by comparing our verbalizer-based inference to the autoregressive generation baselines. We rank the top-25 candidates from retriever and measure the inference speed. As baselines, we adopt the identical LMM backbone and consider list-wise title generation (Title Ranking) and item ID generation (ID Ranking). We vary the average token length of candidate items and report the batched inference time in Figure 7. As expected, we observe substantially better efficiency in batched inference compared to generative alternatives that rely on stepwise decoding. The PRIME approach surpasses baseline ranking methods by a wide margin, making it particularly well-suited for real-world multimodal recommendation.

## 5 Conclusion

In this work, we introduce a two-stage multimodal framework PRIME for sequential recommendation. In particular, we leverage a lightweight retriever model to identify potential candidate items upon user interaction history. Next, our LMM-based ranker is employed to re-rank the candidates, leveraging both text and image attributes in items to enhance the understanding of user preference transition and item characteristics. To enhance the collaboration between retrieval and ranking, we propose an online, on-policy preference optimization phase, exploiting the LMM feedback to refine the retriever model performance. Furthermore, the proposed verbalizer-based approach is designed to achieve efficient inference, where all candidate item scores can be obtained within a single forward pass. To validate the efficacy and efficiency of PRIME, we conducted extensive experiments that consistently demonstrate the superiority of PRIME over state-of-the-art baseline methods.

## 6 Acknowledgments

This research is supported in part by the National Science Foundation under Grant No. CNS-2427070, IIS-2331069, IIS-2202481, IIS-2130263, CNS-2131622. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

## References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems* 35 (2022), 23716–23736.

[2] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. *arXiv preprint arXiv:2305.00447* (2023).

[3] Guy E Blelloch. 1989. Scans as primitive parallel operations. *IEEE Transactions on computers* 38, 11 (1989), 1526–1538.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[5] Zheng Chen. 2023. PALR: Personalization Aware LLMs for Recommendation. *arXiv preprint arXiv:2305.07622* (2023).

[6] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311* (2022).

[7] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. 2024. Instructblip: Towards general-purpose vision-language models with instruction tuning. *Advances in Neural Information Processing Systems* 36 (2024).

[8] Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691* (2023).

[9] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314* (2023).

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota.

[11] Sumanth Doddapaneni, Krishna Sayana, Ambarish Jash, Sukhdeep Sodhi, and Dima Kuzmin. 2024. User Embedding Model for Personalized Language Prompting. *arXiv preprint arXiv:2401.04858* (2024).

[12] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783* (2024).

[13] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. 2023. Chat-rec: Towards interactive and explainable llms-augmented recommender system. *arXiv preprint arXiv:2303.14524* (2023).

[14] 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 *Proceedings of the 16th ACM Conference on Recommender Systems*. 299–315.

[15] Shijie Geng, Juntao Tan, Shuchang Liu, Zuohui Fu, and Yongfeng Zhang. 2023. VIP5: Towards Multimodal Foundation Models for Recommendation. *arXiv preprint arXiv:2305.14302* (2023).

[16] Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, et al. 2024. Direct language model alignment from online ai feedback. *arXiv preprint arXiv:2402.04792* (2024).

[17] Jesse Harte, Wouter Zorgdrager, Panos Louridas, Asterios Katsifodimos, Dietmar Jannach, and Marios Fragkoulis. 2023. Leveraging large language models for sequential recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*. 1096–1102.

[18] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*. 507–517.

[19] 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.

[20] Zhankui He, Handong Zhao, Zhe Lin, Zhaowen Wang, Ajinkya Kale, and Julian McAuley. 2021. Locker: Locally constrained self-attentive sequential recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 3088–3092.

[21] Karl Higley, Even Oldridge, Ronay Ak, Sara Rabhi, and Gabriel de Souza Pereira Moreira. 2022. Building and Deploying a Multi-Stage Recommender System with Merlin. In *Proceedings of the 16th ACM Conference on Recommender Systems*. 632–635.

[22] Jiwoo Hong, Noah Lee, and James Thorne. 2024. Reference-free monolithic preference optimization with odds ratio. *arXiv e-prints* (2024), arXiv–2403.

[23] Yupeng Hou, Zhankui He, Julian McAuley, and Wayne Xin Zhao. 2023. Learning vector-quantized item representation for transferable sequential recommenders. In *Proceedings of the ACM Web Conference 2023*. 1162–1171.

[24] Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2022. Towards universal sequence representation learning for recommender systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 585–593.

[25] Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2023. Large language models are zero-shot rankers for recommender systems. *arXiv preprint arXiv:2305.08845* (2023).

[26] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685* (2021).

[27] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276* (2024).

[28] Dongfu Jiang, Xuan He, Huaye Zeng, Cong Wei, Max Ku, Qian Liu, and Wenhu Chen. 2024. MANTIS: Interleaved Multi-Image Instruction Tuning. *arXiv preprint arXiv:2405.01483* (2024).

[29] Bowen Jin, Hansi Zeng, Guoyin Wang, Xiusi Chen, Tianxin Wei, Ruirui Li, Zhengyang Wang, Zheng Li, Yang Li, Hanqing Lu, et al. 2023. Language Models As Semantic Indexers. *arXiv preprint arXiv:2310.07815* (2023).

[30] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*. IEEE, 197–206.

[31] Wang-Cheng Kang, Jianmo Ni, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi, and Derek Zhiyuan Cheng. 2023. Do LLMs Understand User Preferences? Evaluating LLMs On User Rating Prediction. *arXiv preprint arXiv:2305.06474* (2023).

[32] Richard E Ladner and Michael J Fischer. 1980. Parallel prefix computation. *Journal of the ACM (JACM)* 27, 4 (1980), 831–838.

[33] Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. 2024. What matters when building vision-language models? *arXiv preprint arXiv:2405.02246* (2024).

[34] Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. 2023. Text Is All You Need: Learning Language Representations for Sequential Recommendation. *arXiv preprint arXiv:2305.13731* (2023).

[35] Jinming Li, Wentao Zhang, Tian Wang, Guanglei Xiong, Alan Lu, and Gerard Medioni. 2023. GPT4Rec: A generative framework for personalized recommendation and user interests interpretation. *arXiv preprint arXiv:2304.03879* (2023).

[36] Lei Li, Yongfeng Zhang, and Li Chen. 2023. Prompt distillation for efficient llm-based recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 1348–1357.

[37] Xinhang Li, Chong Chen, Xiangyu Zhao, Yong Zhang, and Chunxiao Xing. 2023. E4SRec: An Elegant Effective Efficient Extensible Solution of Large Language Models for Sequential Recommendation. *arXiv preprint arXiv:2312.02443* (2023).

[38] Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2024. Llara: Large language-recommendation assistant. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1785–1795.

[39] Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Rich James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, et al. 2023. Ra-dit: Retrieval-augmented dual instruction tuning. *arXiv preprint arXiv:2310.01352* (2023).

[40] Dugang Liu, Shenxian Xian, Xiaolin Lin, Xiaolian Zhang, Hong Zhu, Yuan Fang, Zhen Chen, and Zhong Ming. 2024. A Practice-Friendly Two-Stage LLM-Enhanced Paradigm in Sequential Recommendation. *arXiv preprint arXiv:2406.00333* (2024).

[41] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *arXiv preprint arXiv:2304.08485* (2023).

[42] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[43] Yuqing Liu, Yu Wang, Lichao Sun, and Philip S Yu. 2024. Rec-GPT4V: Multimodal Recommendation with Large Vision-Language Models. *arXiv preprint arXiv:2402.08670* (2024).

[44] Zhuang Liu, Yunpu Ma, Matthias Schubert, Yuanxin Ouyang, and Zhang Xiong. 2022. Multi-Modal Contrastive Pre-training for Recommendation. In *Proceedings of the 2022 International Conference on Multimedia Retrieval*. 99–108.

[45] Sichun Luo, Bowei He, Haohan Zhao, Yinya Huang, Aojun Zhou, Zongpeng Li, Yuanzhang Xiao, Mingjie Zhan, and Linqi Song. 2023. RecRanker: Instruction Tuning Large Language Model as Ranker for Top-k Recommendation. *arXiv preprint arXiv:2312.16018* (2023).

[46] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Ji Yang, Minmin Chen, Jiaxi Tang, Lichan Hong, and Ed H Chi. 2020. Off-policy learning in two-stage recommender systems. In *Proceedings of The Web Conference 2020*. 463–473.

[47] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. 43–52.

[48] OpenAI. 2023. GPT-4 Technical Report. *ArXiv* abs/2303.08774 (2023).

[49] Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. 2023. Resurrecting recurrent neural networks for long sequences. *arXiv preprint arXiv:2303.06349* (2023).

[50] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

[51] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 8748–8763.

[52] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. (2018).

[53] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems* 36 (2024).

[54] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, et al. 2023. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems* 36 (2023), 10299–10315.

[55] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. In *International Conference on Machine Learning*. PMLR, 8821–8831.

[56] Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530* (2024).

[57] Xubin Ren, Wei Wei, Lianghao Xia, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2023. Representation learning with large language models for recommendation. *arXiv preprint arXiv:2310.15950* (2023).

[58] Adam Roberts, Colin Raffel, Katherine Lee, Michael Matena, Noam Shazeer, Peter J Liu, Sharan Narang, Wei Li, and Yanqi Zhou. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. (2019).

[59] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100* (2022).

[60] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. RoFormer: enhanced transformer with rotary position embedding. CoRR abs/2104.09864 (2021). *arXiv preprint arXiv:2104.09864* (2021).

[61] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1441–1450.

[62] Rui Sun, Xuezhi Cao, Yan Zhao, Junchen Wan, Kun Zhou, Fuzheng Zhang, Zhongyuan Wang, and Kai Zheng. 2020. Multi-modal knowledge graphs for recommender systems. In *Proceedings of the 29th ACM international conference on information & knowledge management*. 1405–1414.

[63] Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agent. *arXiv preprint arXiv:2304.09542* (2023).

[64] Yi Tay, Mostafa Dehghani, Vinh Q Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, et al. 2022. Ul2: Unifying language learning paradigms. *arXiv preprint arXiv:2205.05131* (2022).

[65] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).

[66] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[67] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. 2024. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191* (2024).

[68] Yancheng Wang, Ziyan Jiang, Zheng Chen, Fan Yang, Yingxue Zhou, Eunah Cho, Xing Fan, Xiaojiang Huang, Yanbin Lu, and Yingzhen Yang. 2023. Recmind: Large language model powered agent for recommendation. *arXiv preprint arXiv:2308.14296* (2023).

[69] Yueqi Wang, Zhenrui Yue, Huimin Zeng, Dong Wang, and Julian McAuley. 2024. Train Once, Deploy Anywhere: Matryoshka Representation Learning for Multimodal Recommendation. *arXiv preprint arXiv:2409.16627* (2024).

[70] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. 2021. Simvlm: Simple visual language model pretraining with weak supervision. *arXiv preprint arXiv:2108.10904* (2021).

[71] Tianxin Wei, Bowen Jin, Ruirui Li, Hansi Zeng, Zhengyang Wang, Jianhui Sun, Qingyu Yin, Hanqing Lu, Suhang Wang, Jingrui He, et al. 2024. Towards unified multi-modal personalization: Large vision-language models for generative recommendation and beyond. *arXiv preprint arXiv:2403.10667* (2024).

[72] Wei Wei, Chao Huang, Lianghao Xia, and Chuxu Zhang. 2023. Multi-Modal Self-Supervised Learning for Recommendation. In *Proceedings of the ACM Web Conference 2023*. 790–800.

[73] Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Llmrec: Large language models with graph augmentation for recommendation. (2024), 806–815.

[74] Junda Wu, Cheng-Chun Chang, Tong Yu, Zhankui He, Jianing Wang, Yupeng Hou, and Julian McAuley. 2024. Coral: Collaborative retrieval-augmented large language models improve long-tail recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3391–3401.

[75] Lanling Xu, Junjie Zhang, Bingqian Li, Jinpeng Wang, Mingchen Cai, Wayne Xin Zhao, and Ji-Rong Wen. 2024. Prompting Large Language Models for Recommender Systems: A Comprehensive Framework and Empirical Analysis. *arXiv preprint arXiv:2401.04997* (2024).

[76] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671* (2024).

[77] Yuyang Ye, Zhi Zheng, Yishan Shen, Tianshu Wang, Hengruo Zhang, Peijun Zhu, Runlong Yu, Kai Zhang, and Hui Xiong. 2024. Harnessing multimodal large language models for multimodal sequential recommendation. *arXiv preprint arXiv:2408.09698* (2024).

[78] Zhenrui Yue, Zhankui He, Huimin Zeng, and Julian McAuley. 2021. Black-box attacks on sequential recommenders via data-free model extraction. In *Proceedings of the 15th ACM Conference on Recommender Systems*. 44–54.

[79] Zhenrui Yue, Sara Rabhi, Gabriel de Souza Pereira Moreira, Dong Wang, and Even Oldridge. 2023. LlamaRec: Two-stage recommendation using large language models for ranking. *arXiv preprint arXiv:2311.02089* (2023).

[80] Zhenrui Yue, Yueqi Wang, Zhankui He, Huimin Zeng, Julian McAuley, and Dong Wang. 2023. Linear Recurrent Units for Sequential Recommendation. *arXiv preprint arXiv:2310.02367* (2023).

[81] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. 2023. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 11975–11986.

[82] 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).

[83] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068* (2022).

[84] Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D Manning, and Curtis P Langlotz. 2022. Contrastive learning of medical visual representations from paired images and text. In *Machine Learning for Healthcare Conference*. PMLR, 2–25.

[85] 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).

[86] Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Filter-enhanced MLP is all you need for sequential recommendation. In *Proceedings of the ACM web conference 2022*. 2388–2399.

[87] Shengyao Zhuang, Honglei Zhuang, Bevan Koopman, and Guido Zuccon. 2023. A setwise approach for effective and highly efficient zero-shot ranking with large language models. *arXiv preprint arXiv:2310.09497* (2023).

# A Appendix

## A.1 Baselines

We select multiple state-of-the-art baselines to compare with PRIME. In particular, we adopt *ID-based* SASRec, BERT4Rec, FMLP-Rec and LRURec [30, 61, 80, 86], text-based UniSRec, VQRec and RecFormer [23, 24, 34], and multimodal baselines MMSSL, VIP5 [15, 72]. We report the details of baseline methods:

- *Self-Attentive Sequential Recommendation (SASRec)* is a transformer sequential recommender. SASRec uses unidirectional self-attention to capture transition patterns [30].
- *Bidirectional Encoder Transformers for Sequential Recommendation (BERT4Rec)* is similar to SASRec but utilizes bidirectional attention. BERT4Rec learns via masked training [61].
- *Filter-enhanced MLP for Recommendation (FMLP-Rec)* adopts an all-MLP architecture with filter-enhanced layers. FMLP-Rec also applies Fast Fourier Transform (FFT) to improve user-item representation learning [86].
- *Linear Recurrence Units for Sequential Recommendation (LRURec)* is based on linear recurrence and is optimized for parallelized training and incremental inference. LRURec thus provides both efficient training and inference speed [80].
- *Universal Sequence Representation for Recommender Systems (UniSRec)* is a text-based transformer recommender system. UniSRec leverage pretrained language models to generate item features for next-item prediction [24].
- *Pretrain, Personalized Prompt & Predict Paradigm for Recommendation (P5)* is a encoder-decoder model pretrained on multiple tasks using item IDs. P5 predicts the next-item by performing conditional generation on item IDs [14].
- *Vector-Quantized Item Representation for Sequential Recommenders (VQRec)* is also text-based sequential recommender. VQRec quantizes language model-based item features to improve recommendation performance [23].
- *Language Representations for Sequential Recommendation (RecFormer)* is language model-based architecture for recommendation. RecFormer adopts two-stage contrastive learning to improve item representation learning [34].
- *Two-Stage Recommendation using Large Language Models for Ranking (LlamaRec)* is a two-stage sequential recommendation framework that integrates efficient retrieval and large language models (LLMs) for ranking [79].
- *Multi-Modal Self-Supervised Learning for Recommendation (MMSSL)* is a multimodal recommender using graphs and multimodal item features for recommendation. MMSSL is trained in a self-supervised fashion [72].
- *Multimodal Foundation Models for Recommendation (VIP5)* is a multimodal encoder-decoder recommender using item IDs and multimodal attributes for multi-taks recommendation. VIP5 is trained via conditional generation [15].

All models are trained according to the methodologies described in the original works, with unspecified hyperparameters used as recommended. For both P5 and VIP5 baselines, item IDs are randomized to prevent information leakage in inference (i.e., patterns like $a, a+1, a+2, \ldots$) [54]. Baseline methods and PRIME are evaluated under identical conditions, we introduce the implementation details of both PRIME and baseline methods in the following.

## A.2 Implementation Details

For the baseline methods, we refer to the original works for implementation and the selection of hyperparameters [15, 23, 24, 30, 34, 61, 72, 79, 80, 86]. Baseline models and the retriever model in PRIME are trained with AdamW optimizer using a learning rate of 1e-3 / 1e-4 and maximum epochs of 500. Validation is performed every epoch and early stopping is triggered if validation NDCG@10 does not improve in 10 epochs. 50 is used as maximum item length for all methods and datasets. We perform grid search in training with weight decay from [0, 1e-2] and dropout rate from [0.4, 0.6]. For language and multimodal baseline methods, pretrained checkpoints are used for initialization if available, and all models undergo further training on the selected datasets using suggested configuration and training strategies. Similarly, we conduct hyperparameter tuning within the ranges reported in the original studies and select models based on the highest NDCG@10 scores achieved in validation. Our retrieval model LRURec is trained under similar conditions to ID-based methods, with the exception of using Recall@20 as the validation metric. In our LMM ranker, we use the top-25 candidates from the retriever model and initialize the model with Qwen2 VL (`Qwen2-VL-2B-Instruct`). The item attribute length is truncated if it exceeds 32 tokens, and the maximum token length is set as 16k. We also adopt flash-attention to improve long context efficiency. We adopt 4-bit quantization in LoRA and use 8 as dimension $r$, 16 as $\alpha$ as well as 0.05 dropout rate. The learning rate is 1e-4 with target modules being all linear layers except the LM head and image encoder, resulting in trainable parameters about 0.5% of the total 2B parameters. The model is tuned for 1 epoch and validated every 100 iterations with early stopping patience of 5. For online preference tuning, we train the retriever model for 2 epochs with 5e-5 learning rate, we search dropout rate from $[0.1, 0.2]$ and $\beta$ from $[0.01, 0.1, 0.2]$ [22]. Similarly, we validate every 100 iterations and perform early stopping with patience of 10 [8, 9, 26, 67].

As for the adopted item attributes and multimodal prompt, we provide an example of our item representation is as follows:

```
{
    "image": <image>,
    "title": title,
    "price": price,
    "brand": brand,
    "categories": categories
},
```

where `<image>` is a special token that needs to be replaced by image tokens from the vision encoder. For particular items, `title`, `price`, `brand` and `categories` are populated using the item's metadata. We construct prompts by including multiple items from both user history and retrieved items, with the template available in our implementation. For constructing the candidate sets $\boldsymbol{c}_w$ and $\boldsymbol{c}_l$, we sample two sets of size $k$ from the retriever model $f_{\text{retr}}$. We then label the sampled sets $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ for each of the following cases:
(1) Both contain the ground truth $y$: use $f_{\text{rank}}$ to see which set ranks $y$ higher, the set with higher rank is used as $\boldsymbol{c}_w$, the other as $\boldsymbol{c}_l$.
(2) Only one contains $y$: that set becomes $\boldsymbol{c}_w$, the other becomes $\boldsymbol{c}_l$.
(3) Neither contains $y$: resample until one of the above applies.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009