Train Once, Deploy Anywhere: Matryoshka Representation Learning for Multimodal Recommendation

Anonymous ACL submission

Abstract

Despite recent advancements in language and vision modeling, integrating rich multimodal knowledge into recommender systems continues to pose significant challenges. This is primarily due to the need for efficient recommendation, which requires adaptive and interactive responses. In this study, we focus on sequential recommendation and introduce a lightweight framework called full-scale Matryoshka representation learning for multimodal recommendation (fMRLRec). Our fMRLRec captures item features at different granularities, learning informative representations for efficient recommendation across multiple dimensions. To integrate item features from diverse modalities, fMRLRec employs a simple mapping to project multimodal item features into an aligned feature space. Additionally, we design an efficient linear transformation that embeds smaller features into larger ones, substantially reducing memory requirements for large-scale training on recommendation data. Combined with improved state space modeling techniques, fMRLRec scales to different dimensions and only requires one-time training to produce multiple models tailored to various granularities. We demonstrate the effectiveness and efficiency of fMRLRec on multiple benchmark datasets, which consistently achieves superior performance over state-of-the-art baseline methods.

1 Introduction

Recent advancements in language and multimodal modeling demonstrates significant potential for improving recommender systems (Touvron et al., 2023; Liu et al., 2023; OpenAI, 2023; Reid et al., 2024). Such progress can be largely attributed to: (1) language / vision features can provide additional descriptive information for understanding user preference and item characteristics (e.g. item descriptions); and (2) generic language capabilities acquired through language and vision pretraining tasks could be transferred for use in recommendation systems. Consequently, language and multimodal representations provide a robust foundation for enhancing the contextual relevance and accuracy of recommendations (Li et al., 2023a; Geng et al., 2023; Yue et al., 2023a; Wei et al., 2024b).

Despite performance improvements, different recommendation scenarios (e.g., centralized or federated recommender systems) often require varying granularities (i.e., model / dimension sizes) in item representations to achieve the balance between performance and efficiency (Han et al., 2021; Luo et al., 2022; Xia et al., 2023; Zeng et al., 2024). For instance, larger dimensions are typically required to encode language and vision features for fine-grained understanding and generation tasks, although marginally lower performance can often be achieved using considerably smaller feature sizes (Kusupati et al., 2022). To identify the optimal granularity for specific use cases in recommendation systems, methods like grid search or adaptive search heuristics are frequently utilized in training (Wang et al., 2024). However, such searches can lead to substantial training expenses or fail to identify the optimal model, particularly when given a large configuration space and constrained computational resources. Therefore, a train-once and deploy-anywhere solution is optimal for the efficient training of recommender systems, which should ideally meet the following criteria:

1. Training is only need once to yield multiple models of different sizes corresponding to various performance and memory requirements;

2. Training and inference should demand no more computational costs than training a single large model, allowing deployment of various model sizes at inference time.

Inspired by Matryoshka Representation Learning (MRL) (Kusupati et al., 2022), we introduce a lightweight multimodal recommendation framework named full-scale Matryoshka Representation Learning for Recommendation (fMRLRec). fMRLRec embeds smaller vector/matrix representations in larger ones like Matryoshka dolls and is only trained once without additional computation costs. Different from MRL that only embeds smaller final-layer activations into larger ones during training, fMRLRec pushes the efficiency of MRL training by introducing an efficient linear transformation that embeds both smaller weights and activations into larger ones, thereby reducing memory costs associated with both aspects. This approach is particularly effective for training recommender systems on large-scale data, offering a highly efficient framework for multi-granularity model training. Combined with further improvements in state-space modeling represented by (Yue et al., 2023b; Orvieto et al., 2023; Gu and Dao, 2023), the linear recurrence architecture in fMRLRec delivers both effectiveness and efficiency in recommendation performance across various benchmark datasets. We summarize our contributions below[1]:

1. We introduce a novel training framework for multimodal sequential recommendation (fMRLRec), which provides an efficient paradigm to learn models of varying granularities within a single training session.

2. fMRLRec introduces an efficient linear transformation that reduces memory costs by embedding smaller features into larger ones. Combined with improved state-space modeling, fMRLRec achieves both efficiency and effectiveness in multimodal recommendation.

3. We show the effectiveness and efficiency of our fMRLRec on benchmark datasets, where the proposed fMRLRec consistently outperforms state-of-the-art baselines with considerable improvements in training efficiency and recommendation performance.

2 Related Works

2.1 Multimodal Recommendation

Language and multimodal models are applied as recommender systems to understand user preferences and item properties (Hou et al., 2022; Li et al., 2023a; He and McAuley, 2016b; Wei et al., 2023). Current language-based approaches leverage pretrained models to improve item representations or re-rank retrieved items (Chen, 2023; Li et al., 2023b; Luo et al., 2023; Yue et al., 2023a; Xu et al., 2024). For example, VQ-Rec utilizes a language encoder and vector quantization to improve item features in cross-domain recommendation (Hou et al., 2023). To further incorporate visual data, existing methods focus on developing strategies that extracts informative user / item representations (Wei et al., 2019; Tao et al., 2020; Wang et al., 2023; Wei et al., 2024a,b). For instance, VIP5 leverages a pretrained transformer with additional vision encoder to learn user transition patters and improve recommendation performance (Geng et al., 2023). However, current models are not tailored to accommodate flexible item attributes or modalities, nor are they optimized for scalable model sizes and efficient inference. Moreover, multimodal approaches require substantial computational resources and separate training sessions for each model, rendering them largely ineffective for real-world applications. To address this, we introduce a lightweight multimodal recommendation framework in fMRLRec, offering multiple model sizes within a single training session and efficient inference capabilities across various scenarios.

2.2 Matryoshka Representation Learning

Matryoshka representation learning (MRL) constructs embeddings at different granularities using an identical model, thereby providing adaptability to varying computational resources without additional training (Kusupati et al., 2022). MRL proposes nested optimization of vectors in multiple dimensions using shared model parameters, demonstrating promising results on multiple downstream tasks and further applications (Cai et al., 2024; Hu et al., 2024; Li et al., 2024). Nevertheless, training MRL models demands additional memory for activations in its nested optimization, posing challenges for training recommender systems with large batches on extensive data. Furthermore, MRL remains unexplored for sequential modeling and efficient multimodal recommendation. As such, our fMRLRec aims to provide an adaptive framework for learning recommender systems using arbitrary modalities, delivering both efficacy and efficiency in multimodal sequential recommendation.

[1]We adopt publicly available datasets in our experiments and will release our implementation upon publication.

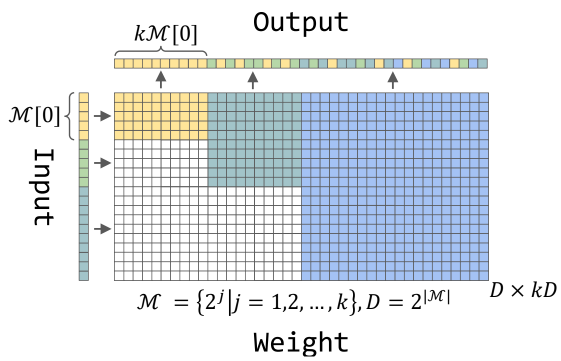


Figure 1: fMRLRec-based weight design, white cells indicate zeros and arrows show vector-matrix multiplication. Input slice $[0 : m]$ is only relevant to weight matrix slice $[0 : m, 0 : km]$ during training, convenient for variously-sized model weights extraction during inference time.

3 Methodologies

3.1 Problem Statement

We present fMRLRec with a research focus in multimodal sequential recommendation. Formally, given a user set $\mathcal{U} = \{u1, u2, ..., u_{|U|}\}$ and an item set $\mathcal{V} = \{v1, v2, ..., v_{|V|}\}$, user u's interacted item sequence in chronological order is denoted with $\mathcal{S}_u = [v_1^{(u)}, v_2^{(u)}, ..., v_n^{(u)}]$, where n is the sequence length. The sequential recommendation task is to predict the next item $v_{n+1}^{(u)}$ that user u will interact with. Mathematically, our objective can be formulated as the maximization of the probability of the next interacted item $v_{n+1}^{(u)}$ given $\mathcal{S}_u$:

$$p(v_{n+1}^{(u)} = v|\mathcal{S}_u) \tag{1}$$

3.2 Full-Scale Matryoshka Representation Learning for Recommendation

In this section, we elaborate on how we design the full-scale Matryoshka Representation Learning for multimodal sequential recommendation (fMRLRec). The majority of model parameters in neural networks can be represented with a set of 2-dimensional weights $\mathcal{W} = \{W_1, W_2, \ldots, W_n\}$ where $W_i \in \mathbb{R}^{d_1 \times d_2}, i \in \{1, 2, \ldots, n\}$, regardless of specific model architecture. Intuitively, fMRLRec aims to design the $W_i \in \mathcal{W}$ s.t. models of differently sizes $\mathcal{M} = [2, 4, 8, 16, \ldots, D]$ are trained only once at the same cost of only training size-D model. After training, any model sizes in $\mathcal{M}$ can be extracted from the size-D model to form *independent* small models for deployment. To achieve this goal, fMRLRec allows small models to be *embedded* in the largest model. Define sequential input as $X_i \in \mathbb{R}^{B \times L \times D}$ to be processed by $\mathcal{W}$, where B is batch size, L is item sequence length and D is the embedding size, there are three cases for the shape of $W_i \in \mathbb{R}^{d_1 \times d_2}$, denoted as $D(W_i)$,

$$\mathbf{D}(\mathbf{W}_i) = \begin{cases} \mathbf{D} \times \mathbf{kD} & \text{if } \mathbf{d}_1 < \mathbf{d}_2 \\ \mathbf{kD} \times \mathbf{D} & \text{if } \mathbf{d}_1 > \mathbf{d}_2 \\ \mathbf{D} \times \mathbf{D} & \text{if } \mathbf{d}_1 = \mathbf{d}_2 \end{cases} \tag{2}$$

Here, we assume $k \in \mathbb{Z}^+/\{1\}$ to ease the derivation since W_i often indicates linear up/down scaling by an integer k times (e.g., post-attention MLPs in transformer).

For case 1 where $D(W_i) = D \times kD$ and $X_i \in \mathbb{R}^{B \times L \times D}$, $X_i W_i$ indicates an up scale. We define the j's slice of X_i as $\mathbf{X}_i^{(j)} = \mathbf{X}_i[0 : \mathbf{M}[j]]$ and the j's slice of W_i as

$$\mathbf{W}_i^{(j)} = \begin{cases} \mathbf{W}_i[0 : \mathbf{M}[0], 0 : \mathbf{kM}[0]] & \text{if } j = 0 \\ \mathbf{W}_i[0 : \mathbf{M}[j], \mathbf{kM}[j-1] : \mathbf{kM}[j]] & \text{if } j > 0 \end{cases}$$

For case 2 where $D(w_i) = kD \times D$ and the corresponding input $X_i \in \mathbb{R}^{B \times L \times kD}$, $X_i W_i$ indicates a down scale. We define the j's slice of sequential input X_i as $\mathbf{X}_i^{(j)} = \mathbf{X}_i[0 : 2\mathbf{M}[j]]$ and the j's slice of W_i as

$$\mathbf{W}_i^{(j)} = \begin{cases} \mathbf{W}_i[0 : \mathbf{kM}[0], 0 : \mathbf{M}[0]] & \text{if } j = 0 \\ \mathbf{W}_i[0 : \mathbf{kM}[j], \mathbf{M}[j-1] : \mathbf{M}[j]] & \text{if } j > 0 \end{cases}$$

For case3 where $D(w_i) = D \times D$, assign $k = 1$ for any of above two cases yields $\mathbf{W}_i^{(j)}$.

Then, we perform matrix multiplication between $X_i^{(j)}$ and $W_i^{(j)}$ followed by concatenation along dimension j to form the output

$$\mathbf{Y}_i = [\mathbf{X}_i^{(0)}\mathbf{W}_i^{(0)}, \ldots, \mathbf{X}_i^{(z)}\mathbf{W}_i^{(z)}] \tag{3}$$

where $z = \log(D/2)$. Refer to figure 1 for case 1 of this process.

The fMRLRec Operator Instead of computing equation 3, we would like the chunk/slice-wise multiplication of $X_i^{(j)} W_i^{(j)}$ for all $j = 1, 2, \ldots, \log(D/2)$ is computed by *one* forward pass to derive output Y_i. Specifically, we create a padding mask $P_i(\mathcal{M})$ of the same size as W_i that

$$\mathbf{P}_i(\mathcal{M}) = \{p_{rs} = 0 | w_{rs} \in \mathbf{W}_i, w_{rs} \notin \mathbf{W}_i^{(j)}\} \tag{4}$$

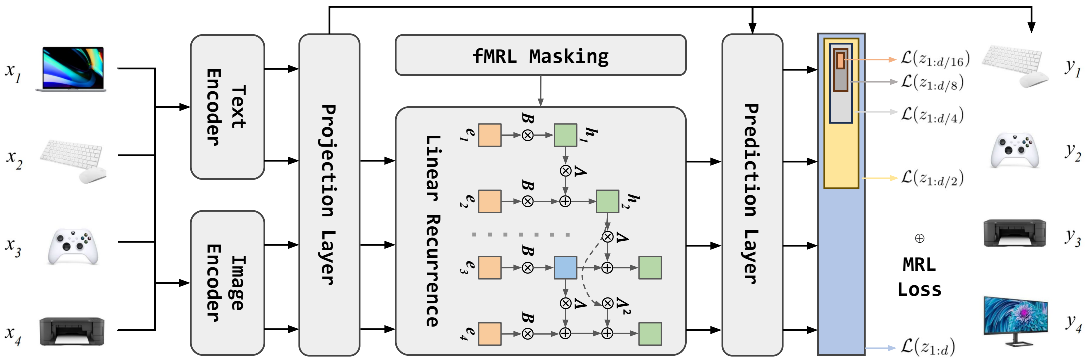


Figure 2: The overall architecture for fMRLRec.

Then we define the fMRLRec operator as:

$$\text{fMRLRec}(\mathbf{W}_i, \mathcal{M}) = \mathbf{P}_i(\mathcal{M}) \odot \mathbf{W}_i \quad (5)$$

Thus, $X_i \cdot \text{fMRLRec}(\mathbf{W}_i, P_i(\mathcal{M}))$ is equivalent to perform equation 3 but only with one time multiplication of X_i and masked W_i. See figure 1 for an illustration of the fMRLRec operator.

In summary, given a neural network represented by $\mathcal{W} = \{W_1, W_2, \dots W_n\}$ where $W_i \in \mathbb{R}^{d1 \times d2}$ and a set of sizes $\mathcal{M} = \{2, 4, 8, \dots, D\}$, we could find an fMRLRec-slicing of $\mathcal{W}$ such that the first $\mathcal{M}[j]$ elements of input X_i is only processed by corresponding chunks in W_i. After the model is trained, we take the first $[0 : \mathcal{M}[j], 0 : k\mathcal{M}[j]$ or $[0 : k\mathcal{M}[j], 0 : \mathcal{M}[j]]$ (depending on the cases in equation 2) slice for each W_i to form *independent* small models called fMRLRec-series models for inference. Also refer to the upper left of figure 1 for the slicing process. For $W_i \in \mathbb{R}^d$, one can leave it as is during training and naturally extract $[0 : \mathcal{M}[j]]$ of it during inference.

3.3 Overall Framework

The overall framework of fMRLRec is illustrated in fig. 2, including feature encodings, LRU-based recommendation module, fMRLRec weight masking, etc.

3.3.1 Language and Image Encoding

We adopt textural item description as the language input source and image as visual input. Given a metadata dictionary $\mathcal{M}$ containing attributes for each item i, we extract its attributes Title, Price, Brand and Categories and perform concatenation of attributes:

$$\text{Text}_i = \text{Title}_i + \text{Price}_i + \text{Brand}_i + \text{Categories}_i$$

We then encode these text attributes and image attributes using pretrained embedding models f. For each item i:

$$\mathbf{E}_{\text{lang},i} = f_{\text{lang}}(\text{Text}_i), \; \mathbf{E}_{\text{img},i} = f_{\text{img}}(\text{Img}_i) \quad (6)$$

We combine text and image embedding through concatenation followed by a simple yet effective linear projection:

$$\mathbf{E} = (\text{Concat}(\mathbf{E}_{\text{lang}}, \mathbf{E}_{\text{img}}))\mathbf{W}_{\text{proj}} + \mathbf{b}_{\text{proj}} \quad (7)$$

where $\mathbf{W}_{\text{proj}}$ and $\mathbf{b}_{\text{proj}}$ are the projection weights and $\mathbf{W}_{\text{proj}} \in \mathbb{R}^{(D_{\text{lang}} + D_{\text{img}}) \times D}$ and $\mathbf{b}_{\text{proj}} \in \mathbb{R}^D$.

3.3.2 Linear Recurrent Units

We adopt Linear Recurrent Units (LRU) for sequence processing for its (1) superior performance and (2) both low training and inference cost compared with RNN and Self-Attention-based models (Orvieto et al., 2023; Yue et al., 2023b). Intuitively, LRU is capable of parallel training like Self-Attention and inference like RNN, where inference complexity can be performed incrementally.

Given input $x_k \in \mathbb{R}^{B \times H_{\text{in}}}$ at time step k, hidden state $h_{k-1} \in \mathbb{R}^{B \times H_{\text{in}}}$, learnable matrices $A \in \mathbb{R}^{H \times H_{\text{in}}}$, $B \in \mathbb{R}^{H \times H_{\text{in}}}$, $C \in \mathbb{R}^{H_{\text{out}} \times H_{\text{in}}}$ and $D \in \mathbb{R}^{H_{\text{out}} \times H_{\text{in}}}$:

$$\mathrm{h_k} = \mathbf{A}\mathrm{h_{k-1}} + \mathbf{B}\mathrm{x_k}, \quad \mathrm{y_k} = \mathbf{C}\mathrm{h_k} + \mathbf{D}\mathrm{x_k}, \quad (8)$$

The input and output dimensions are denoted with H_{in} and H_{out} (i.e., embedding size), and the hidden dimension size with H. Different from RNN models (i.e., $h_k = \sigma(Ah_{k-1} + Bx_k)$), we discard the non-linearity σ to enable parallelization:

$$\begin{aligned} \mathrm{h_k} &= \mathbf{A}\mathrm{h_{k-1}} + \mathbf{B}\mathrm{x_k} \\ &= \mathbf{A}^2\mathrm{h_{k-2}} + \mathbf{AB}\mathrm{x_{k-1}} + \mathbf{B}\mathrm{x_k} = \dots \\ &= \sum_{\mathrm{i=1}}^{\mathrm{k}} \mathbf{A}^{\mathrm{k-i}}\mathbf{B}\mathrm{x_i} \quad \text{with} \quad \mathrm{h_1} = \mathbf{B}\mathrm{x_1}. \end{aligned} \quad (9)$$

Therefore, LRU can be trained in parallel (via parallel scan) as Self-Attention (equation 9) and enable fast inference as RNN models (equation 8).

3.3.3 Overall LRU-Based Recommendation Framework

We first pad for the combined embeddings $\mathbf{E}_i$ output by equation 7 to maximum length of all sequences. Then, the padded embeddings $\mathbf{E}_i$ are processed through N blocks. For each block $i \in \{1, \ldots, N\}$, we first perform layer normalization to the input followed by a LRU layer:

$$\mathrm{LayerNorm}(\mathbf{X}) = \alpha \odot \frac{\mathbf{X} - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta \tag{10}$$

$$\mathrm{LRUNorm}(\mathbf{X}) = \mathrm{LRU}(\mathrm{LayerNorm}(\mathbf{X})) \tag{11}$$

Due to the lack of non-linearity for LRU, we further process the output of LRU layer by a gated non-linear feed-forward network (FFN) to improve training dynamics and model performance. Specifically, our FFN is defined as:

$$\mathrm{Gate} = \mathrm{SiLU}(\mathbf{X}\mathbf{W}^{(g)} + \mathbf{b}^{(g)})$$

$$\mathrm{FFN} = (\mathrm{Gate} \odot (\mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)}))\mathbf{W}^{(2)} + \mathbf{b}^{(2)}$$

As the network gets deeper, some signal of the input from the earlier layers might be forgotten. Thus, we add sub-layer connections in FFN by adding pre-layer normalization and residual connection:

$$\mathrm{SubLayer}(\mathrm{FFN}, \mathbf{X}) = \mathrm{FFN}(\mathrm{LayerNorm}(\mathbf{X})) + \mathbf{X}$$

3.3.4 fMRLRec Plugin to Overall Framework

Next, we apply fMRLRec-based weight design. Given a set of sizes $\mathcal{M} = \{2, 4, 8, \ldots, D\}$, any $W_i \in \mathbb{R}_d$, we leave it as is. For $W_i \in \mathbb{R}^{d_1 \times d_2}$, we apply the fMRLRec operator defined in section 3.2 to W_i as:

$$\mathbf{W'}_i = \mathrm{fMRLRec}(\mathbf{W}_i, \mathcal{M}) \tag{12}$$

During inference time, *independent* models $\mathcal{Q} = \{\mathcal{W'}^{(1)}, \mathcal{W'}^{(2)}, \ldots, \mathcal{W'}^{(|\mathcal{M}|)}\}$ could be extracted as described in the last paragraph of section 3.2.

Prediction Layer After the final layer N, we extract the activation at the last time step t of the final layer as $z_t^{(N)} \in \mathbb{R}^D$, and use it to compute the relevance $r_{i,t} \in \mathbb{R}$ for all items in the pool $v_i \in \mathcal{V}$. Specifically, we perform dot product between $z_t^{(N)}$ with the input/shared embedding layer weight $E_W \in \mathbb{R}^{|\mathcal{V}| \times D}$:

$$r_{i,t} = \left(\mathbf{z}_t^{(N)} \mathbf{E}_w^T\right)_i \tag{13}$$

The higher $r_{i,t}$, the more likely a user is to consider item v_i for the next time step. This way we could generate recommendations by ranking the relevance score $r_{i,t}$.

3.3.5 Network Training

As we derive the relevance score of item i as $r_{i,t}(\theta)$ where θ stands for all parameters used to compute r, we treat the relevance score as logits to compute Cross-Entropy (CE) loss for entire network optimization. While LRURec can be trained with CE loss, it is not enough to yield performant models of sizes $\mathcal{M} = \{2, 4, 8, \ldots, D\}$ as traditional CE loss only explicitly optimizes the largest model of size D. We solve this issue by introducing explicit loss terms as introduced in (Kusupati et al., 2022) to pair with our fMRLRec-style weight matrix for best performance:

$$\mathcal{L}_{\text{fMRLRec}} = \min_{\theta} \frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} \sum_{m \in \mathcal{M}} \mathcal{L}\left(\mathbf{r}_i(\theta[:m]), \mathbf{y}_i\right) \tag{14}$$

where $\mathcal{L}$ is a multi-class softmax cross-entropy loss function based on ranking scores and the label item.

4 fMRLRec Memory Efficiency

In this subsection, we analyze fMRLRec model-series memory efficiency by driving the number of parameters plus activations needed to train model sizes of $\mathcal{M} = \{2, 4, 8, \ldots, D\}$ or $\mathcal{M} = \{2^j | j = 1, 2, \ldots, k\}$ as (1) A train-once fMRLRec model-series and (2) Independent models. Define $W^{(j)} = \{w_1^{(j)}, w_2^{(j)}, \cdot, w_n^{(j)}\}$ as the layer weights of model size j and $X_i \in \mathbb{R}^{B \times L \times D}$ as sequential input data for w_i, where B is the batch size, L is the sequence length and $D = 2^j$ is the model size. We assume every weight has the same scaling factor γ to simplify notations. Thus, $\gamma \cdot (2^j)^2$ and $\gamma \cdot 2^j$ are number of parameters for 2d and 1d weight. Here, we only consider 2d weights saves the most parameters.

Case 1: For fMRLRec-based training, number of parameters needed $N(W) = \sum_{i=1}^{n} \gamma(\cdot(2^k)^2)$, which is $n \cdot \gamma \cdot 2^{(2k)}$; The number of activation generated $N(A) = \sum_{i=1}^{n} \gamma \cdot B \cdot L \cdot D$. Empirically, $B \in \{32, 64, 128\}$ and $L = 50$, thus $B \cdot L = \delta \cdot 2^k$, $\delta > 1$. Then, we have $N(A) = n \cdot \gamma \cdot \delta \cdot 2^{(2k)}$.

Case 2: For Independent training, the number of parameters needed $N(W) = \sum_{j=1}^{k} \sum_{i=1}^{n} \gamma \cdot (2^j)^2$, by summation of the geometric series, $N(W) = n \cdot \gamma \cdot \frac{4^{k+1} - 4}{3}$, the number of activation generated $N(A) = \sum_{j=1}^{k} \sum_{i=1}^{n} \gamma \cdot B \cdot L \cdot D$. Empirically,

Table 1: Statistics of the datasets.

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

$B \in \{32, 64, 128\}$ and $L = 50$, thus $B \cdot L = \delta \cdot 2^j$, $\delta > 1$. Then, we have $N(A) = \sum_{j=1}^{k} n \cdot \gamma \cdot \delta \cdot 2^{(2j)}$, which is equivalent to $N(A) = n \cdot \gamma \cdot \delta \cdot \frac{4^{k+1}-4}{3}$.

In summary, the ratio of parameters and activations between fMRLRec-based training and Independent training is $R = (n \cdot \gamma \cdot \frac{4^{k+1}-4}{3})/(n \cdot \gamma \cdot 2^{(2k)})$ or $(n \cdot \gamma \cdot \delta \cdot \frac{4^{k+1}-4}{3})/(n \cdot \gamma \cdot \delta \cdot 2^{(2k)}) \approx 1.33$. This indicates a parameter saving rate R_s of ≈ 0.33 against the fMRLRec model. Empirically, for a common setting $n = 4$ linear layers with scaling factor $\gamma = 2$ and $D = 512$, the weights saved are approximately $4(n) \times 0.33(R) \times 512(D) \times 1024(2D) \approx 700K$, the number of activation saved for four layer is approximately $4(n) \times 0.33(R) \times 32(B) \times 50(L) \times 1024(2D) \approx 2M$. This is to a great extent saving memory usage if independent training is executed in parallel or saving training time if executed sequentially.

5 Experimental Setup

5.1 Datasets

For evaluating our models, we select four commonly used benchmarks from *Amazon.com* known for real-word sparsity, namely *Beauty*, *Clothing, Shoes & Jewelry* (Clothing), *Sports & Outdoors* (Sports) and *Toys & Games* (Toys) (McAuley et al., 2015; He and McAuley, 2016a). For preprocessing, we follow (Yue et al., 2022; Chen, 2023; Geng et al., 2023) to construct the input sequence in chronological order and apply 5-core filtering to exclude users and items with less than five-time appearances. For textural feature selection, we choose *title*, *price*, *brand* and *categories*; For visual features, we use *photos* of the items. We also filter out items without above metadata. Detailed statistics of the datasets are reported in table 1 including users (#User), items (#Item), images (#Image), interactions (#Inter.) and dataset density in percentages.

5.2 Baseline Methods

For baseline models, we select a series of state-of-the-art recommendation models grouped as *ID-based*, *Text-based* and *Multimodal*. *ID-based* models include SASRec, BERT4Rec, FMLP-Rec and LRURec (Kang and McAuley, 2018; Sun et al., 2019; Zhou et al., 2022; Yue et al., 2023b). Text-based methods include UniSRec, VQRec and RecFormer (Hou et al., 2022, 2023; Li et al., 2023a). We also include multimodal baselines MMSSL, VIP5 (Wei et al., 2023; Geng et al., 2023), More details about baselines is discussed in Appendix A.1.

5.3 Implementations

For training fMRLRec and all baseline models, we utilize AdamW optimizer with learning rate of 1e-3/1e-4 with maximum epochs of 500. Validation is performed per epoch and the training is stopped once validation performance does not improve for 10 epochs. The model with best validation performance is saved for testing and metrics report. For hyperparameters, we find (1) embedding/model size, the number of fMRLRec-LRU layers, dropout rate and weight decay be the most sensitive ones for model performance. Specifically, we grid-search the embedding/model size in [64, 128, 256, 512, 1024, 2048], the number of fMRLRec-LRU layers in [1,2,4,8], dropout rate from [0.1,0,2,...,0.8] on a 0.1-stride and weight decay from [1e-6, 1e-4, 1e-2]. The best hyper-parameters for each datasets are reported in Appendix A.2; We follow (Geng et al., 2023) and set maximum length of input sequence as 50. For validation and test, we adopt two metrics NDCG@k and Recall@k, $k \in \{5, 10\}$ typical for recommendation algorithm evaluation.

6 Experimental Results

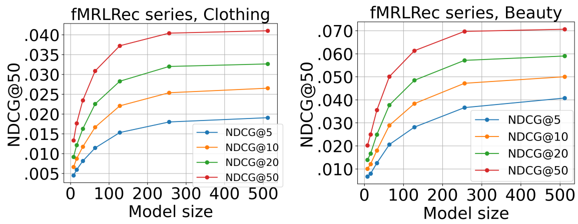
6.1 Main Performance Analysis

Here, we compare the performance of fMRLRec with state-of-the-art baseline models in table 2. We use SAS, BERT, FMLP, LRU, UniS., RecF., fMRLRec to abbreviate SASRec BERT4Rec, FMLP-Rec LRURec, UniSRec, RecFormer and fMRL-Rec. The best metrics are marked in bold and the second best metrics are underlined. Overall, fMRLRec outperforms all baseline models in almost all cases with exceptions of Recall@10 for Toys. Specifically, We observe that: (1) fMRLRec on average outperforms the second-best model by 19.46% across all datasets and metrics (2) fMRL-Rec shows superior ranking performance by having a more significant gain of NDCG which is ranking sensitive than Recall. For example, fMRLRec achieves NDCG@5 improvement of 25.08% over the second best model, which is greater than the

Table 2: Main performance results of fMRLRec and baselines.

Dataset	Metric	ID-Based				Text-Based			Multimodal		
		SAS	BERT	FMLP	LRU	UniS.	VQRec	RecF.	MMSSL	VIP5	fMRLRec
Beauty	N@5	0.0274	0.0275	0.0318	0.0339	0.0274	0.0303	0.0258	0.0189	0.0339	**0.0413**
	R@5	0.0456	0.0420	0.0539	0.0565	0.0484	0.0514	0.0428	0.0308	0.0417	**0.0624**
	N@10	0.0364	0.0350	0.0416	0.0438	0.0375	0.0411	0.0341	0.0252	0.0367	**0.0507**
	R@10	0.0734	0.0653	0.0846	0.0871	0.0799	0.0849	0.0686	0.0506	0.0603	**0.0914**
Cloth.	N@5	0.0075	0.0062	0.0091	0.0104	0.0127	0.0104	0.0137	0.0089	0.0122	**0.0191**
	R@5	0.0134	0.0100	0.0167	0.0192	0.0221	0.0197	0.0234	0.0146	0.0152	**0.0359**
	N@10	0.0104	0.0084	0.0123	0.0140	0.0175	0.0149	0.0192	0.0122	0.0183	**0.0265**
	R@10	0.0227	0.0169	0.0266	0.0304	0.0372	0.0336	0.0405	0.0249	0.0298	**0.0590**
Sports	N@5	0.0143	0.0137	0.0194	0.0204	0.0141	0.0173	0.0127	0.0123	0.0136	**0.0229**
	R@5	0.0267	0.0215	0.0329	0.0344	0.0237	0.0304	0.0211	0.0198	0.0264	**0.0363**
	N@10	0.0210	0.0181	0.0252	0.0266	0.0195	0.0235	0.0173	0.0163	0.0213	**0.0290**
	R@10	0.0474	0.0355	0.0508	0.0536	0.0408	0.0497	0.0350	0.0321	0.0315	**0.0553**
Toys	N@5	0.0291	0.0241	0.0308	0.0366	0.0254	0.0314	0.0292	0.0173	0.0334	**0.0465**
	R@5	0.0534	0.0355	0.0534	0.0601	0.0477	0.0577	0.0501	0.0286	0.0474	**0.0659**
	N@10	0.0380	0.0299	0.0408	0.0463	0.0362	0.0423	0.0398	0.0224	0.0374	**0.0542**
	R@10	0.0807	0.0535	0.0845	0.0901	0.0811	**0.0915**	0.0832	0.0445	0.0642	0.0900
Avg.	N@5	0.0196	0.0179	0.0228	0.0253	0.0199	0.0224	0.0204	0.0144	0.0233	**0.0283**
	R@5	0.0348	0.0273	0.0392	0.0426	0.0355	0.0398	0.0344	0.0235	0.0327	**0.0493**
	N@10	0.0265	0.0229	0.0300	0.0327	0.0277	0.0305	0.0276	0.0191	0.0284	**0.0369**
	R@10	0.0561	0.0428	0.0616	0.0653	0.0598	0.0649	0.0568	0.0381	0.0465	**0.0758**

Recall@5 gains of 19.73%. This is also true for NDCG@10 gains of 19.99% compared with recall gains of 13.04%. (3) fMRLRec demonstrates significant benefits for sparse datasets, Clothing and Sports, by averaging 25.77% improvements. In contrast, the average gains is lower as 13.15% for relatively denser datasets as Beauty and Toys. (4) The subpar score of Recall@10 on Toys for fMRLRec might be attributed to the limitations of the retrieval model, LRU. The pure item-ID-based LRURec scores 0.0901 that falls short than the best score 0.0915 for VQRec. In summary, our results suggest fMRLRec can effectively leverage multimodal item representation to rank items of user preference and improve recommendation performance.



(a) NDCG for Clothing (b) NDCG for Beauty

Figure 3: fMRLRec-model series performance curve against model size. fMRLRec features a significantly slower performance drop rate from 4.31% to 29.62% compared to the model compression rate of 50%.

6.2 fMRLRec Model-Series Performance

In this subsection, we analyze the performance of our full scale Matryoshka Representation Learning (fMRLRec) by extracting from trained models the differently-sized sub-models of $\mathcal{M} = \{8, 16, 32, \dots, D\}$, where $D = 512$ here for best performance. Specific sub-model performance is shown in figure 3. Using NDCG for Clothing as an example, we observe that: (1) NDCG decrease rate for Clothing ranges from 4.31% to 29.62% which is significantly lower than the exponential model compressed by a rate of 50%. This is consistent with the *Scaling Law* (Kaplan et al., 2020) that doubling the model size usually does not mean doubling performance. Despite statement of the *Scaling Law*, the specific performance retained varies for datasets/tasks and are expensive to tune. Tackling this pain point, fMRLRec curve in figure 3 provides flexible options of how much metric scores to retain for developers with limited computational resources. And obtaining fMRLRec such patterns only requires a one-time training of the largest model as introduced in section 3.2.

Table 3: Ablation performance for fMRLRec by removing either of language (lang.) or visual features or both.

Variants / Dataset		Beauty		Clothing		Sports		Toys	
Metric		NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall
fMRLRec	@5	**0.0413**	**0.0624**	**0.0191**	**0.0359**	**0.0229**	**0.0363**	**0.0465**	**0.0659**
	@10	**0.0507**	**0.0914**	**0.0265**	**0.0590**	**0.0290**	**0.0553**	**0.0542**	0.0900
fMRLRec w/ Lang. only	@5	0.0370	0.0557	0.0162	0.0289	0.0223	0.0350	0.0436	0.0642
	@10	0.0468	0.0860	0.0220	0.0468	0.0276	0.0515	0.0525	**0.0919**
fMRLRec w/ Image only	@5	0.0373	0.0555	0.0162	0.0298	0.0197	0.0299	0.0439	0.0630
	@10	0.0465	0.0840	0.0225	0.0493	0.0251	0.0468	0.0522	0.0889
fMRLRec w/o Lang. & Image	@5	0.0386	0.0517	0.0122	0.0171	0.0222	0.0298	0.0457	0.0607
	@10	0.0446	0.0700	0.0143	0.0235	0.0256	0.0405	0.0508	0.0764

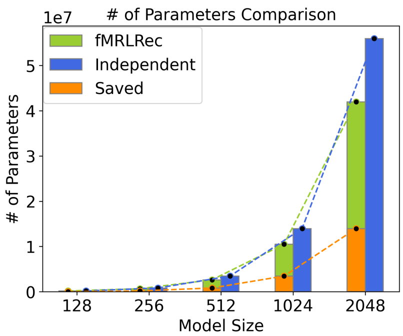


Figure 4: fMRL features a one-time training of model sizes $\mathcal{M} = \{2, 4, \ldots, 2^n\}$ that saves $\approx 33\%$ parameters compared to training every size independently.

6.3 Parameter Saving of fMRLRec

Discussed in Section 4, the model parameter saving rate R_s between fMRLRec-model series and independently trained models is theoretically around $1/3$ of the former. We demonstrate in figure 4 this behavior given model sizes of $\mathcal{M} = \{2^7, 2^8, \ldots, 2^{11}\}$. The green, blue and orange bar represents the number of parameters of fMRLRec-series, independently trained models and ones saved, respectively. Empirically, $R_s = [0, 25.16\%, 31.39\%, 32.90\%, 33.25\%]$ for $\mathcal{M}[j] \in \mathcal{M}$, which converges to ≈ 0.33 as j gets larger and is consistent with our theoretical analysis in Section 4.

6.4 Ablation Study

In this section, we further evaluate the designs of features and modules of fMRLRec by a series of ablation studies in table 3. Specifically, we construct different variants of fMRLRec as: (1) fMRLRec w/ Language only: the fMRLRec model with only the text-based attributes of items such as Title, brand, etc. and their corresponding embeddings. (2) fMRLRec w/ Image only: the fMRLRec model only with the image processor and embeddings. (3) fMRLRec w/o Language & Image: fMRLRec removing all the language and image related feature processing and embeddings. A randomly initialized embedding table is used as item representations. We monitor the change of NDCG and Recall of above variants. In particular, (1) Language features shows a major contribution for the performance as the performance drop is the smallest as 8.41% as only language features are removed. (2) Images also benefit performance of fMRLRec as removing image features incurs a performance drop of 10.97%; (3) Losing both image and language features induces the largest performance drop of 21.52% which justifies contributions of both modalities; In summary, our ablation results show that both language and image feature processing and fusion are effective towards improving the recommendation performance of fMRLRec.

7 Conclusions

In this work, we introduce a lightweight framework fMRLRec for efficient multimodal recommendation across multiple granularities. In particular, we adopt Matryoshka representation learning and design an efficient linear transformation to embeds smaller features into larger ones. Moreover, we incorporate cross-modal features and further improves the state-space modeling for sequential recommendation. Consequently, fMRLRec can yield multiple model sizes with competitive performance within a single training session. To validate the effectiveness and efficiency of fMRLRec, we conducted extensive experiments, where fMRLRec consistently demonstrate the superior performance over state-of-the-art baseline models.

8 Limitations

We have discussed the the ability of fMRLRec to perform one-time training and yield models in multiples sizes ready for deployment. However, we have not experimented on other recommendation tasks such as click rate prediction and multi-basket recommendation, etc. Even though we adopted LRU, a state-of-the-art recommendation module for fMRLRec, other types of sequential/non-sequential models needs to be tested for a more compete performance pattern. More broadly, The idea of full-Scale Matryoshka Representation Learning (fMRL) can be applied to other ML domains that utilize neural network weights; We have yet to explore behaviors of fMRL in those fields where the scale of models and data varies significantly. We plan to conduct more theoretical analysis and experiments for above mentioned aspects in future works.

References

Mu Cai, Jianwei Yang, Jianfeng Gao, and Yong Jae Lee. 2024. Matryoshka multimodal models. *arXiv preprint arXiv:2405.17430*.

Zheng Chen. 2023. Palr: Personalization aware llms for recommendation. *arXiv preprint arXiv:2305.07622*.

Shijie Geng, Juntao Tan, Shuchang Liu, Zuohui Fu, and Yongfeng Zhang. 2023. Vip5: Towards multimodal foundation models for recommendation. *arXiv preprint arXiv:2305.14302*.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Jialiang Han, Yun Ma, Qiaozhu Mei, and Xuanzhe Liu. 2021. Deeprec: On-device deep learning for privacy-preserving sequential recommendation in mobile commerce. In *Proceedings of the Web Conference 2021*, pages 900–911.

Ruining He and Julian McAuley. 2016a. 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*, pages 507–517.

Ruining He and Julian McAuley. 2016b. Vbpr: visual bayesian personalized ranking from implicit feedback. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

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*, pages 1162–1171.

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*, pages 585–593.

Wenbo Hu, Zi-Yi Dou, Liunian Harold Li, Amita Kamath, Nanyun Peng, and Kai-Wei Chang. 2024. Matryoshka query transformer for large vision-language models. *arXiv preprint arXiv:2405.19315*.

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

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, et al. 2022. Matryoshka representation learning. *Advances in Neural Information Processing Systems*, 35:30233–30249.

Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. 2023a. Text is all you need: Learning language representations for sequential recommendation. *arXiv preprint arXiv:2305.13731*.

Jinming Li, Wentao Zhang, Tian Wang, Guanglei Xiong, Alan Lu, and Gerard Medioni. 2023b. Gpt4rec: A generative framework for personalized recommendation and user interests interpretation. *arXiv preprint arXiv:2304.03879*.

Xianming Li, Zongxi Li, Jing Li, Haoran Xie, and Qing Li. 2024. 2d matryoshka sentence embeddings. *arXiv preprint arXiv:2402.14776*.

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

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*.

Sichun Luo, Yuanzhang Xiao, and Linqi Song. 2022. Personalized federated recommendation via joint representation learning, user clustering, and model adaptation. In *Proceedings of the 31st ACM international conference on information & knowledge management*, pages 4289–4293.

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, pages 43–52.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

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*.

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*.

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*, pages 1441–1450.

Zhulin Tao, Yinwei Wei, Xiang Wang, Xiangnan He, Xianglin Huang, and Tat-Seng Chua. 2020. Mgat: Multimodal graph attention network for recommendation. *Information Processing & Management*, 57(5):102277.

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*.

Jinpeng Wang, Ziyun Zeng, Yunxiao Wang, Yuting Wang, Xingyu Lu, Tianxiang Li, Jun Yuan, Rui Zhang, Hai-Tao Zheng, and Shu-Tao Xia. 2023. Missrec: Pre-training and transferring multi-modal interest-aware sequence representation for recommendation. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 6548–6557.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *ArXiv*, abs/2212.03533.

Yueqi Wang, Zhankui He, Zhenrui Yue, Julian McAuley, and Dong Wang. 2024. Auto-encoding or auto-regression? a reality check on causality of self-attention-based sequential recommenders. *arXiv preprint arXiv:2406.02048*.

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

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*, pages 790–800.

Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024b. Llmrec: Large language models with graph augmentation for recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 806–815.

Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. 2019. Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video. In *Proceedings of the 27th ACM international conference on multimedia*, pages 1437–1445.

Xin Xia, Junliang Yu, Qinyong Wang, Chaoqun Yang, Nguyen Quoc Viet Hung, and Hongzhi Yin. 2023. Efficient on-device session-based recommendation. *ACM Transactions on Information Systems*, 41(4):1–24.

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*.

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

Zhenrui Yue, Yueqi Wang, Zhankui He, Huimin Zeng, Julian McAuley, and Dong Wang. 2023b. Linear recurrent units for sequential recommendation. *arXiv preprint arXiv:2310.02367*.

Zhenrui Yue, Huimin Zeng, Ziyi Kou, Lanyu Shang, and Dong Wang. 2022. Defending substitution-based profile pollution attacks on sequential recommenders. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 59–70.

Huimin Zeng, Zhenrui Yue, Qian Jiang, and Dong Wang. 2024. Federated recommendation via hybrid retrieval augmented generation. *arXiv preprint arXiv:2403.04256*.

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*, pages 11975–11986.

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*, pages 2388–2399.

A Appendix

A.1 Baselines

We select multiple state-of-the-art baselines to compare with fMRLRec. In particular, we adopt *ID-based* SASRec, BERT4Rec, FMLP-Rec and LRURec (Kang and McAuley, 2018; Sun et al., 2019; Zhou et al., 2022; Yue et al., 2023b), text-based UniSRec, VQRec and RecFormer (Hou et al., 2022, 2023; Li et al., 2023a), and multimodal baselines MMSSL, VIP5 (Wei et al., 2023; Geng et al., 2023). We report the details of baseline methods:

- *Self-Attentive Sequential Recommendation (SASRec)* is the first transformer-based sequential recommender. SASRec uses unidirectional self-attention to capture transition patterns (Kang and McAuley, 2018).

- *Bidirectional Encoder Representations from Transformers for Sequential Recommendation (BERT4Rec)* is similar to SASRec but utilizes bidirectional self-attention. BERT4Rec learns via masked training (Sun et al., 2019).

- *Filter-enhanced MLP for Recommendation (FMLP-Rec)* also adopts an all-MLP architecture with filter-enhanced layers. FMLP-Rec also applies Fast Fourier Transform (FFT) to improve representation learning (Zhou et al., 2022).

- *Linear Recurrence Units for Sequential Recommendation (LRURec)* is based on linear recurrence and is optimized for parallelized training. LRURec thus provides both efficient training and inference speed (Yue et al., 2023b).

- *Universal Sequence Representation for Recommender Systems (UniSRec)* is a text-based recommender system. UniSRec leverage pretrained language models to generate item features for next-item prediction (Hou et al., 2022).

- *Vector-Quantized Item Representation for Sequential Recommenders (VQRec)* is also text-based sequential recommender. VQRec quantizes language model-based item features to improve performance (Hou et al., 2023).

- *Language Representations for Sequential Recommendation (RecFormer)* is language model-based architecture for recommendation. RecFormer adopts contrastive learning to improve item representation (Li et al., 2023a).

- *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 (Wei et al., 2023).

- *Multimodal Foundation Models for Recommendation (VIP5)* is a multimodal recommender using item IDs and multimodal attributes for multitaks recommendation. VIP5 is trained via conditional generation (Geng et al., 2023).

All models are trained according to the methodologies described in the original works, with unspecified hyperparameters used as recommended. All baseline methods and fMRLRec are evaluated under identical conditions.

A.2 Implementations

We discuss further implementation details other than data processing, evaluation metrics, early stopping, etc., as already reported in section 5. We adopt pretrained E5 (Wang et al., 2022) and SigLip (Zhai et al., 2023) for language and image encoding; The tuning phase basically lasts for 5-6 hours on a single NVIDIA-A100 (40GB) GPU. For hyperparameters, we find the most sensitive ones towards performance as follows and report the best hyper-parameters found:

- Embedding/model size: We grid-search Embedding/model size among [64, 128, 256, 512, 1024, 2048], the best values for datasets, Beauty, Clothing, Sport and Toys are 512, 512, 256 and 512. This indicates a slightly-compressed space is required for our recommendation task than the pretrained language/image feature space of dimension 1024 in our case.

- The number of fMRLRec-based LRU layers: We grid-search the number of layers among [1,2,4,8]. The best performing value is 2 for all datasets.

- Dropout rate: We grid-search the dropout rate among [0.1,0,2, ..., 0.8] on a 0.1-stride. We find small dropout rates as 0.1 and 0.2 (best) are typically optimal for all dimensions from 128 to 1024 for multimodal-recommendation.

- Weight decay: We grid search the weight decay among [1e-6, 1e-4, 1e-2] and finds 1e-2 to be the best performing value.