Ada-Retrieval: An Adaptive Multi-Round Retrieval Paradigm for Sequential Recommendations

Lei Li[1], Jianxun Lian[2], Xiao Zhou[1*], Xing Xie[2]

[1]Gaoling School of Artificial Intelligence, Renmin University of China
[2]Microsoft Research Asia
leil@ruc.edu.cn, jianxun.lian@outlook.com, xiaozhou@ruc.edu.cn, xingx@microsoft.com

Abstract

Retrieval models aim at selecting a small set of item candidates which match the preference of a given user. They play a vital role in large-scale recommender systems since subsequent models such as rankers highly depend on the quality of item candidates. However, most existing retrieval models employ a single-round inference paradigm, which may not adequately capture the dynamic nature of user preferences and stuck in one area in the item space. In this paper, we propose Ada-Retrieval, an adaptive multi-round retrieval paradigm for recommender systems that iteratively refines user representations to better capture potential candidates in the full item space. Ada-Retrieval comprises two key modules: the item representation adapter and the user representation adapter, designed to inject context information into items' and users' representations. The framework maintains a model-agnostic design, allowing seamless integration with various backbone models such as RNNs or Transformers. We perform experiments on three widely used public datasets, incorporating five powerful sequential recommenders as backbone models. Our results demonstrate that Ada-Retrieval significantly enhances the performance of various base models, with consistent improvements observed across different datasets. Our code and data are publicly available at: https://github.com/ll0ruc/Ada-Retrieval.

Introduction

Recommender systems have become a crucial element in a wide range of online applications, encompassing e-commerce, social media, and entertainment platforms (Covington, Adams, and Sargin 2016; Ying et al. 2018). By providing personalized recommendations tailored to users' historical behavior and preferences, these systems enhance user experience and engagement. Among the diverse types of recommender systems, sequential recommender systems (Rendle 2010; Tang and Wang 2018) have attracted considerable interest due to their capacity to effectively capture temporal dynamics in user history and accurately forecast near-future user behaviors. In this domain, various backbone models have been proposed, including recurrent neural networks (RNNs) (Hidasi et al. 2015), convolutional neural networks (CNNs) (Tang and Wang 2018), transformers (Kang and
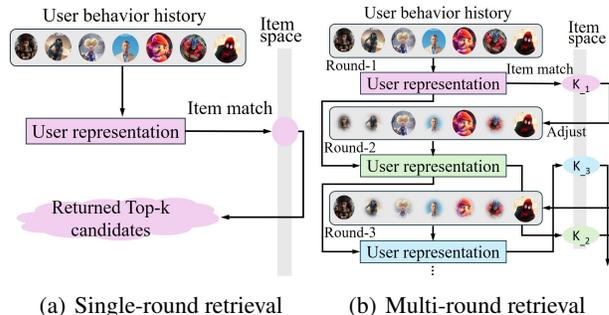
Figure 1: Illustrations of (a) the conventional single-round retrieval paradigm, and (b) our proposed adaptive multi-round retrieval paradigm, in which the final retrieval result is the union of each individual retrieval K_i.

McAuley 2018), and graph neural networks (GNNs) (Wu et al. 2019), each contributing to the ongoing advancement of sequential recommendation techniques.

This paper does not aim to propose a stronger backbone model. Instead, we observe that most existing models employ a **single-round inference** paradigm to retrieve the top-k item candidates (He and McAuley 2016; Tang and Wang 2018; Sun et al. 2019). Specifically, given users' profiles such as behavior histories, the model initiates the forward process and generates user representations, which are then used as queries to match the top-k most similar items in the database. However, this single-round inference paradigm may not adequately capture the dynamic nature of user preferences and adapt to the ever-changing diversity of the item space. As illustrated in Figure 1(a), once the model's forward pass is completed, the user representation remains fixed, resulting in a top-k search area in the item space that is confined to a static region. If the initial user representation is inaccurate or the user's future preferences are diverse, this paradigm may fail to deliver satisfactory performance.

We argue that a **multi-round inference** paradigm offers a more effective retrieval approach for recommender systems. As illustrated in Figure 1(b), the objective of retrieving k items is divided into n batches, with each batch representing a round of retrieving k/n items. The forward passes of user representation in different rounds are conducted indepen-

dently. If the previously retrieved items do not adequately match the user's preferences, the user representation will be adjusted in the next round, allowing the model to search for target items in a different region of the item space. Taking the search engine scenario as an analogy (Zhang and Nasraoui 2006), users may rewrite their queries if the currently retrieved information does not accurately address their questions. In this regard, previous rounds' retrieval can serve as feedback information (Lewandowski 2008), which helps refine the user representation if necessary. Thus, this multi-round paradigm presents a significant advantage, as it prevents user representations from being confined to a static area, enabling more dynamic and diverse recommendations.

As an embodiment of the new paradigm, we present Ada-Retrieval, an adaptive multi-round retrieval approach for recommender systems. Fundamentally, Ada-Retrieval alters the traditional training and inference process while maintaining a **model-agnostic design**, allowing seamless integration with various backbone models such as RNNs or Transformers. Ada-Retrieval comprises two key modules: the item representation adapter and the user representation adapter. Both modules aim to inject context information, which refers to previous user representations and retrieved items up to the current retrieval round, into items' and users' representations. The item representation adapter consists of a learnable filter (LFT) layer and a context-aware attention (CAT) layer, designed to adjust item embeddings in the user history according to the retrieval context. This enables the user model to potentially optimize for the next round of retrieval by considering the feedback of item candidate space. The user representation adapter, on the other hand, is composed of a Gated Recurrent Units (GRU) layer and a Multi-Layer Perceptron (MLP) layer. The GRU layer encodes all user representations generated in previous rounds as user context, while the MLP layer fuses this context with the current user representation to produce an adapted one. By incorporating these components, Ada-Retrieval can integrate contextual information during the retrieval process into traditional sequential recommendation models, generating progressively refined user representations for item retrieval while maintaining a lightweight and model-agnostic advantage.

We perform experiments on three widely used public datasets, incorporating five powerful sequential recommenders as backbone models. Comprehensive results demonstrate that Ada-Retrieval can significantly enhance the performance of various base models, with consistent improvements observed across different datasets. For instance, on the Beauty dataset, Ada-Retrieval boosts SASRec's performance by 8.55% in terms of NDCG@50 and improves the best base model, FMLPRec, by 5.66%. The key contributions of this paper are summarized as follows:

- We propose Ada-Retrieval, a novel adaptive multi-round retrieval framework for sequential recommendations. Unlike traditional single-round retrieval, Ada-Retrieval iteratively refines user representations to better capture potential candidates across the entire item space.

- We design several key components, including LFT and CAT for the item representation adapter, and GRU and

MLP for the user representation adapter. These components enable the integration of contextual information in a model-agnostic manner.

- We conduct extensive experiments on real-world datasets to demonstrate the effectiveness of Ada-Retrieval, showing significant improvements over various sequential recommender systems.

Related Work

Deep Retrieval

In practical recommender systems, the retrieval stage (candidate generation) aims to efficiently retrieve a small subset of items, typically in the hundreds, from large corpora (Xie et al. 2020). With the rise of deep learning, there has been a surge in efforts to construct sophisticated retrieval models for recommender systems. Embedding-based methods often adopt a two-tower architecture, as seen in FM (Rendle 2010), YoutubeDNN (Covington, Adams, and Sargin 2016), and AFT (Hao et al. 2021), dividing the construction of user and item representations into two distinct branches. Innovations like TDM (Zhu et al. 2018) and JTM (Zhu et al. 2019) offer fresh perspectives on leveraging user-item dynamics through tree-based structures. Additionally, graph-based matching models (Xie et al. 2021) are proposed to learn user/item representations. Departing from the single-round inference paradigms of these methods, our model introduces a multi-round inference paradigm, providing a more effective retrieval approach for recommender systems.

Sequential Recommendation

Sequential recommendation, predicting future items to interact with based on correlations in item transitions within user activity sequences, has evolved from foundational Markov Chain models (He and McAuley 2016) to contemporary deep learning technologies. Caser (Tang and Wang 2018) employed CNNs to analyze sequences of item embeddings, while GRU4Rec (Hidasi et al. 2015) used Gated Recurrent Units (GRU) for session-driven recommendations. More recently, SASRec (Kang and McAuley 2018) incorporated self-attention mechanisms to selectively aggregate relevant items, refining user modeling. Inspired by the Cloze task, Bert4Rec (Sun et al. 2019) predicted masked items by jointly utilizing preceding and succeeding contexts. Training frameworks like CL4SRec (Xie et al. 2022) integrated contrastive approaches for diverse perspectives through data enhancement. Despite the success of these models, a challenge remains in generating diverse user feature representations. In addressing this, our model iteratively refines user representations, enhancing the capture of dynamic user preferences through inserted contextual information.

Preliminaries

Problem Formulation

Let us assume a set of users $\mathcal{U} = \{u_1, u_2, \ldots, u_{|\mathcal{U}|}\}$ and items $\mathcal{I} = \{i_1, i_2, \ldots, i_{|\mathcal{I}|}\}$, with $u \in \mathcal{U}$ representing a user and $i \in \mathcal{I}$ representing an item. The user behavior can
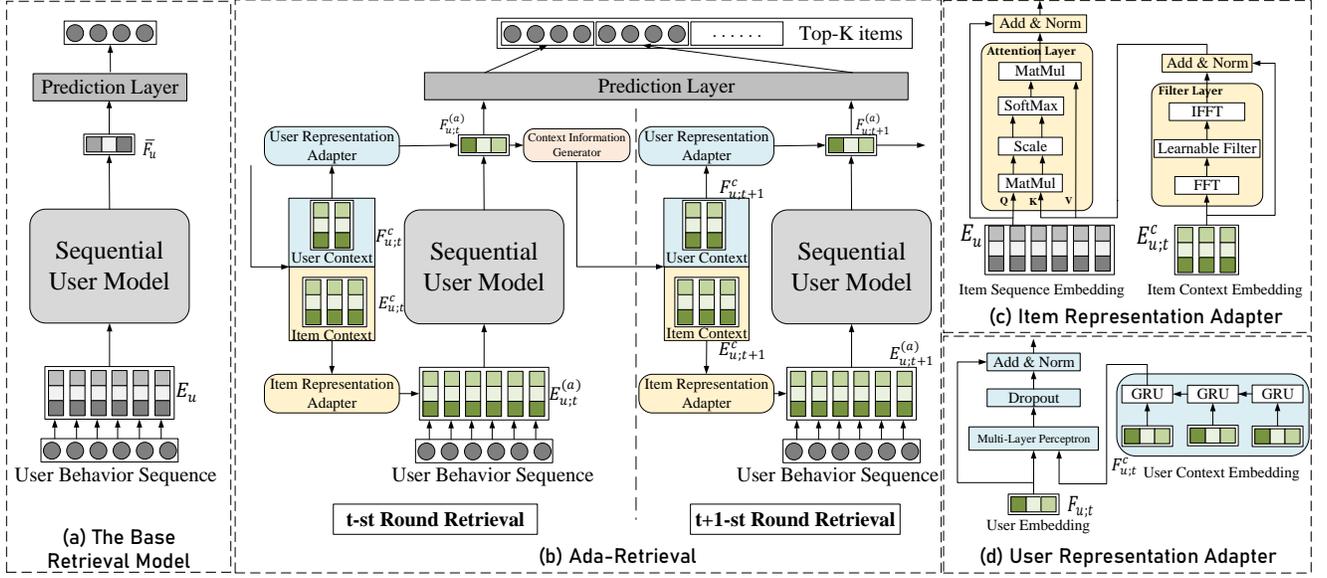
Figure 2: An overview of the traditional retrieval model (a) and our proposed Ada-Retrieval paradigm (b), which consists of two key parts: the item representation adapter (c) and the user representation adapter (d). We use colored elements to indicate the new components in Ada-Retrieval.

be denoted as $\mathcal{S} = \{s_1, s_2, \ldots, s_{|\mathcal{U}|}\}$. In sequential recommendation, a user's behavior sequence is typically ordered chronologically: $s_u = \{i_1, i_2, \ldots, i_n\}$, where $s_u \in \mathcal{S}$ and $u \in \mathcal{U}$. The objective of sequential recommendation is to predict the next item the user is likely to interact with, denoted as $p(i_{n+1}|i_{1:n})$.

Base Sequential Model

A common architecture for a sequential recommender system typically consists of three key components: an embedding lookup layer $\mathrm{EMB}(\cdot)$, a sequential encoding layer $\mathrm{SEL}(\cdot)$, and a prediction layer $\mathrm{PRED}(\cdot)$, as illustrated in Figure 2(a). When provided with a user behavior sequence $s_u = \{i_1, i_2, \ldots, i_n\}$, the sequence initially passes through the embedding lookup layer $\mathrm{EMB}(\cdot)$, resulting in a sequence of corresponding item embeddings:

$$\mathbf{E_u} = \mathrm{EMB(s_u)} = \{\mathbf{e_1}, \mathbf{e_2}, \ldots, \mathbf{e_n}\} \qquad (1)$$

Subsequently, the embeddings of this item sequence are processed through a sequential user encoder, represented as $\mathrm{SEL}(\cdot)$, which can be implemented using appropriate backbones such as RNNs or Transformers:

$$\mathbf{F_u} = \mathrm{SEL}(\mathbf{E_u}) \qquad (2)$$

Here, $\mathbf{F_u}$ represents the embedding vector serving as the user representation. Then, $\mathbf{F_u}$ is combined with a target item vector $\mathbf{E_i}$ as the input to the prediction layer $\mathrm{PRED}(\cdot)$:

$$\hat{y}_{ui} = \mathrm{PRED}(\mathbf{F_u}, \mathbf{E_i}) \qquad (3)$$

The prediction layer is commonly implemented using either a dot product or cosine similarity, particularly for retrieval purposes.

Methodology

Our model, Ada-Retrieval, introduces an adaptive multi-round retrieval approach for recommender systems. The overall framework is depicted in Figure 2(b). At the core of the adaptive retrieval paradigm are two meticulously crafted adaptation modules: the Item Representation Adapter (IRA) and the User Representation Adapter (URA). These modules seamlessly integrate contextual information into user preference modeling. In contrast to traditional sequential recommendation models, our modifications primarily focus on the input and output of the sequential encoding layer $\mathrm{SEL}(\cdot)$:

$$\mathbf{E_u^{(a)}} = \mathrm{IRA}(\mathbf{E_u}; \mathbf{E_u^c}) \qquad (4)$$

$$\mathbf{F_u^{(a)}} = \mathrm{URA}(\mathbf{F_u}; \mathbf{F_u^c}) \qquad (5)$$

where $\mathbf{E_u^c}$, $\mathbf{F_u^c}$ are the feature representation of item context and user context. $\mathbf{E_u^{(a)}}$ and $\mathbf{F_u^{(a)}}$ represent the adjusted feature representation of item/user. In the next section, we will introduce the details of each proposed component.

Item Representation Adapter

The item representation adapter is designed to recalibrate item embeddings within users' historical data based on the prevailing retrieval item context. Further details are illustrated in Figure 2(c).

Learnable Filter Layer Considering potential noise in item context information from previous rounds, we use a single learnable filter block for refining item features efficiently. This approach draws inspiration from the filter-enhanced MLP (Zhou et al. 2022) used in recommendation systems, which typically employs multiple stacked blocks to enhance item feature representations by removing noise.

Upon receiving the item context information for the current round, denoted as $\mathcal{C}_u^t = \{i_1, i_2, \cdots, i_{\mathcal{C}_u^t}\}$, where i represents items retrieved in previous rounds, our initial step involves passing it through an encoder layer to extract its features, $\mathbf{E_u^c} = \text{EMB}(\mathcal{C}_u^t)$. Subsequently, we apply the Fast Fourier Transform (FFT), denoted as $\mathcal{F}(\cdot)$, along the item dimension. This operation transforms the item context representation matrix $\mathbf{E_u^c}$ into the frequency domain:

$$\mathbf{X_u^c} = \mathcal{F}(\mathbf{E_u^c}) \tag{6}$$

Note that $\mathbf{X_u^c}$ is a complex tensor representing the spectrum of $\mathbf{E_u^c}$. We can then proceed by multiplying it with a learnable filter, denoted as W:

$$\widetilde{\mathbf{X}}_{\mathbf{u}}^{\mathbf{c}} = W \odot \mathbf{X_u^c} \tag{7}$$

where $\odot$ is the element-wise multiplication. Finally, we employ the inverse FFT to revert the modulated spectrum, $\widetilde{\mathbf{X}}_{\mathbf{u}}^{\mathbf{c}}$, back into the time domain, subsequently updating the sequence representations:

$$\widetilde{\mathbf{E}}_{\mathbf{u}}^{\mathbf{c}} = \mathcal{F}^{-1}(\widetilde{\mathbf{X}}_{\mathbf{u}}^{\mathbf{c}}) \tag{8}$$

where $\mathcal{F}^{-1}(\cdot)$ denotes the inverse 1D FFT, which converts the complex tensor into a real number tensor. To avoid overfitting, dropout layer (Srivastava et al. 2014), residual connection structure (He et al. 2016), and layer normalization operations (Ba, Kiros, and Hinton 2016) are applied on the obtained output $\mathbf{H_u^c}$:

$$\mathbf{H_u^c} = \text{LayerNorm}(\mathbf{E_u^c} + \text{Dropout}(\widetilde{\mathbf{E}}_{\mathbf{u}}^{\mathbf{c}})) \tag{9}$$

Context-aware Attention Layer Attention mechanisms have proven effective in recommender systems (Kang and McAuley 2018; Tan et al. 2021a). They empower the model to selectively focus on different segments of the sequence based on their relevance to the immediate prediction task. The following is the standard dot-product attention:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^{\text{T}}}{\sqrt{\text{d}}})\mathbf{V} \tag{10}$$

Here, $\mathbf{Q}$ denotes queries, $\mathbf{K}$ stands for keys, and $\mathbf{V}$ represents values. The embedding size is denoted by d, and the scale factor $\sqrt{d}$ is introduced to prevent excessively large values in the inner product.

In the typical self-attention mechanism, $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are derived from the same input vector but are produced using distinct weight matrices. However, in our scenario, $\mathbf{Q}$ corresponds to the item sequence features, while $\mathbf{K}$ embodies the item context features. Therefore, the context-aware attention mechanism can be articulated as:

$$\widetilde{\mathbf{H}}_{\mathbf{u}}^{\mathbf{c}} = \text{Attention}(\mathbf{E_u}, \mathbf{H_u^c}, \mathbf{H_u^c}) \tag{11}$$

Certain pieces of item context information wield more influence in determining the subsequent item with which the user might engage. The attention mechanism enables the model to autonomously pinpoint these pivotal items, bestowing upon them greater weights. Following this, we incorporate the layer normalization and dropout operations to alleviate the gradient vanishing and unstable training problems as:

$$\mathbf{E_u^{(a)}} = \text{LayerNorm}(\mathbf{E_u} + \text{Dropout}(\widetilde{\mathbf{H}}_{\mathbf{u}}^{\mathbf{c}})) \tag{12}$$

A Case with Sequential User Model Following the acquisition of the adjusted item sequence feature representation, it seamlessly integrates into a conventional sequential recommendation model. Taking SASRec as an illustration, the user feature representation is denoted as $\mathbf{F_u} = \text{TRFM}(\mathbf{E_u})$, where $\text{TRFM}(\cdot)$ signifies the Transformer architecture within SASRec. In the context of Ada-Retrieval, the user feature representation is expressed as $\mathbf{F_u} = \text{TRFM}(\mathbf{E_u^{(a)}})$, representing a modification to the input. It is noteworthy that Ada-Retrieval, with its model-agnostic nature, refrains from altering the intrinsic parameters of SAS-Rec. Instead, it dynamically adjusts the current item sequence input features based on the item context information.

User Representation Adapter

Utilizing the available user context information, we formulate the design of the user representation adapter to produce adaptive user representations, as illustrated in Figure 2(d).

Gated Recurrent Unit Layer Recurrent Neural Networks (RNNs) have been developed to model variable-length sequence data (Sherstinsky 2020), showcasing promising advancements in recommendation systems (Li et al. 2017; Guo et al. 2020). Their efficacy stems from their capacity to capture a user's sequential behavior. Gated Recurrent Units (GRUs) (Cho et al. 2014) represent a more sophisticated variant of RNNs designed to address the vanishing gradient challenge. Essentially, user features derived from earlier rounds exert influence on the current one, with this influence diminishing as the round distance grows. Therefore, we leverage the capabilities of the GRU module to encode user representations accumulated from previous rounds.

$$\widetilde{\mathbf{F}}_{\mathbf{u}}^{\mathbf{c}} = \text{GRU}(\mathbf{F_u^c}) \tag{13}$$

$\mathbf{F_u^c}$ serves as the feature representation encapsulating user context, consisting of adjusted user feature representations generated in previous rounds.

With the trivial feature extractor of the user context, we essentially use the final hidden state as the representation of the user's context representation $\widetilde{\mathbf{F}}_{\mathbf{u}}^{\mathbf{c}}$.

Multi-Layer Perceptron Layer After deriving the user's context feature representation $\widetilde{\mathbf{F}}_{\mathbf{u}}^{\mathbf{c}}$, we concatenate it with the currently generated user feature representation $\mathbf{F_u}$. This combined representation then undergoes processing through a two-layer Multilayer Perceptron (MLP) with a ReLU activation function. The process is defined as follows:

$$\mathbf{F_u^{(a)}} = W_2 \, \text{ReLU}(W_1[\widetilde{\mathbf{F}}_{\mathbf{u}}^{\mathbf{c}}; \mathbf{F_u}] + b_1) + b_2 \tag{14}$$

where W_1, b_1, W_2, b_2 are trainable parameters.

Subsequently, we incorporate skip connection and layer normalization operations, as detailed in Eq. (12), to produce the final user representation $\mathbf{F_u^{(a)}}$.

Context Information Generator

To collect the user context information generated in each round, we utilize a stacking methodology to assemble an array of context vectors:

$$\mathbf{F_{u;t}^c} = \text{STACK}(\{\mathbf{F_{u;1}^{(a)}}, \mathbf{F_{u;2}^{(a)}}, \cdots, \mathbf{F_{u;t-1}^{(a)}}\}) \tag{15}$$

Concurrently, from the entire pool of candidate items, we retrieve the top-k items with the highest scores aligned with the user representation of the current round, $\mathbf{F}_{\mathbf{u}}^{(\mathbf{a})}$. The corresponding item IDs are then added to the item context pools:

$$\mathcal{C}_u^t = \mathcal{C}_u^{t-1} + \text{top-}k \underset{i \in \mathcal{I}}{\arg\max} \, \text{Sim}(\mathbf{F}_{\mathbf{u};\mathbf{t}-\mathbf{1}}^{(\mathbf{a})}; \mathbf{E_i}) \quad (16)$$

where Sim is a function to measure the feature similarity between users and items. Here, we utilize the dot product for this purpose. It is crucial to highlight that the item context, determined through this similarity measure, will subsequently be input into the embedding look-up layer to retrieve their corresponding feature representations.

Model Prediction and Optimization

After T iterative rounds, Ada-Retrieval yields T user representations $\{\mathbf{F}_{\mathbf{u};\mathbf{1}}^{(\mathbf{a})}, \mathbf{F}_{\mathbf{u};\mathbf{2}}^{(\mathbf{a})}, \cdots, \mathbf{F}_{\mathbf{u};\mathbf{T}}^{(\mathbf{a})}\}$. Then we multiply it by the item embedding matrix $\mathbf{E}$ to predict the relevance of the candidate item:

$$\hat{y}_{ui;t} = \mathbf{E_i^T} \mathbf{F}_{\mathbf{u};\mathbf{t}}^{(\mathbf{a})} \quad (17)$$

Each of these representations is employed to retrieve k/T items, which are then sequentially concatenated to form the final set of top-k items.

We anticipate that the actual item i chosen by user u should result in a higher score $\hat{y}_{ui}$. Hence, we utilize the cross-entropy loss to optimize the model parameters. The objective function for the t-th round is formulated as:

$$\mathcal{L}_t = - \sum_{u \in \mathcal{U}, i \in \mathcal{I}} y_{ui;t} \log(\sigma(\hat{y}_{ui;t})) + (1 - y_{ui;t}) \log(1 - \sigma(\hat{y}_{ui;t}))$$
$$(18)$$

To emphasize early and accurate predictions of the positive item, we introduce a decay factor λ to weigh each round's contribution to the overall training loss as:

$$\mathcal{L} = \sum_{t=1}^{T} \lambda^t \mathcal{L}_t \quad (19)$$

To optimize training efficiency, we employ a two-phase training approach. Initially, we pre-train a foundational sequential model. Subsequently, in the second phase, we fine-tune Ada-Retrieval, utilizing the pre-trained model as a starting point. In this phase, we concurrently update both the parameters Θ of Ada-Retrieval and the Φ of the base model.

Experiments

Experimental Settings

Datasets. To validate our proposed method across diverse data types, we assess the model using three publicly available benchmark datasets. Beauty and Sports represent subsets of the Amazon Product dataset (McAuley et al. 2015), capturing user reviews of Amazon.com products. The Yelp dataset[1] is a sizable collection of extended item sequences derived from business recommendations, using only transaction records post-January 1st, 2019. For uniformity, we categorize interaction records by users or sessions and sequence

---

[1]https://www.yelp.com/dataset

Dataset	Beauty	Sports	Yelp
# Sequences	22,363	25,598	30,431
# Items	12,101	18,357	20,033
# Actions	198,502	296,337	316,354
# Sparsity	99.93%	99.95%	99.95%

Table 1: Statistics of datasets after preprocessing.

them chronologically based on timestamps. Following (Sun et al. 2019; Li, Wang, and McAuley 2020), we filter out users/items with fewer than 5 interactions. Detailed statistics for each dataset are summarized in Table 1.

Evaluation Settings. To facilitate comprehensive model evaluation, we employ the leave-one-out strategy (Kang and McAuley 2018; Zhou et al. 2020) for partitioning each user's item sequence into training, validation, and test sets. Diverging from conventional sampling practices, our approach considers all items not previously engaged with by the user as candidate items (Krichene and Rendle 2020). The evaluation metrics adopted for assessing model performance encompass top-k Hit Ratio (HR@k) and top-k Normalized Discounted Cumulative Gain (NDCG@k).

Implementation Details. When comparing with existing models, we adopt the optimal parameter settings from their original papers and conduct a meticulous grid search around these configurations for baseline models. Ada-Retrieval is implemented using Python 3.8 and PyTorch 1.12.1, executed on NVIDIA V100 GPUs with 32GB memory. Training parameters include an Adam optimizer with a learning rate of 0.001 and a batch size of 1024. Across all datasets, we set the maximum sequence length to 50, embedding dimension to 64, and training epochs to a maximum of 200. For Ada-Retrieval, we varied hyperparameters T and λ within the ranges [3, 8] and [0.1, 0.9], respectively, with step sizes of 1 and 0.2. The experiments were conducted five times, and results, reported as averages with standard deviations, reflect the model's performance. We also employed an early-stopping strategy, halting training if HR@50 performance on the validation set continuously decreased over 10 consecutive epochs.

Main Results with Various Backbone Models

Backbones. As Ada-Retrieval is model-agnostic, we evaluate its performance with representative sequential recommenders, including GRU4Rec (Hidasi et al. 2015), SASRec (Kang and McAuley 2018), NextItNet (Yuan et al. 2019), SRGNN (Wu et al. 2019), and FMLPRec (Zhou et al. 2022). These models employ diverse architectures, encompassing RNN, CNN, GNN, and MLP.

Results. We train the base sequential models and their corresponding Ada-Retrieval counterparts using three datasets. The top 50 recommendation results are shown in Table 2.

Here, we consistently observe that Ada-Retrieval consistently and significantly outperforms all base sequential models across all datasets and metrics. Notably, Ada-Retrieval

| Datasets | Models | GRU4Rec | | SASRec | | NextItNet | | SRGNN | | FMLPRec | |
		HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG
Beauty	Base	13.126	4.574	17.110	6.506	12.539	4.064	12.411	4.242	17.935	6.876
	Ada.	**14.175**	**4.915**	**17.741**	**7.062**	**12.948**	**4.288**	**13.274**	**4.375**	**18.531**	**7.265**
	Improv.	7.99%	7.46%	3.69%	8.55%	3.27%	5.50%	6.96%	3.12%	3.32%	5.66%
Sports	Base	7.644	2.447	10.924	4.046	7.939	2.563	7.704	2.504	11.607	4.238
	Ada.	**8.226**	**2.683**	**11.352**	**4.234**	**8.425**	**2.663**	**8.551**	**2.731**	**11.903**	**4.444**
	Improv.	7.61%	9.67%	3.92%	4.65%	6.12%	3.90%	11.00%	9.07%	2.55%	4.85%
Yelp	Base	9.252	2.765	12.062	3.770	9.828	2.971	10.501	3.141	13.013	4.029
	Ada.	**10.415**	**2.985**	**12.637**	**3.852**	**11.224**	**3.316**	**11.688**	**3.454**	**13.430**	**4.157**
	Improv.	12.57%	7.98%	4.77%	2.19%	14.20%	11.60%	11.31%	9.96%	3.20%	3.18%

Table 2: Top-50 performance comparison of five backbone models and Ada-Retrieval (Ada.) on three datasets. All the metrics in the table are percentage numbers with '%' omitted.

(GRU4Rec) exhibits an average improvement of 8.37% in terms of NDCG@50 over GRU4Rec on the three datasets, while Ada-Retrieval (SASRec) demonstrates an improvement of 5.57% over SASRec.

Whether it is RNN-based (GRU4Rec), Transformer-based (SASRec), CNN-based (NextItNet), GNN-based (SRGNN), or MLP-based (FMLPRec), Ada-Retrieval seamlessly integrates and consistently enhances performance. Notably, Ada-Retrieval shares the same embedding layer, user model architectures, and prediction layers with its respective base models. This shared architecture underscores its effectiveness in adapting a base model according to varying contextual information for the specific task. In essence, Ada-Retrieval embodies a plug-and-play property, allowing the augmentation of any given base model with adaptation modules while preserving the original architecture's integrity.

Comparison with Multi-Interest Models

Baselines. Our method employs multi-round adaptive learning to progressively generate multiple user representations. Although it fundamentally differs from another research direction, multi-interest-aware user modeling, there are several similarities between the two approaches, such as producing multiple user representations during inference. In this regard, we compared Ada-Retrieval with several multi-interest retrieval models as baselines: **DNN** (Covington, Adams, and Sargin 2016) (also known as *YouTube DNN*), **MIND** (Li et al. 2019), **ComiRec** (Cen et al. 2020), and **SINE** (Tan et al. 2021b)

Results. The overall results are presented in Table 3. It is evident that approaches utilizing multiple user representation vectors (such as MIND, ComiRec, and Ada-Retrieval) exhibit superior performance compared to those employing a single representation (DNN). This finding highlights the effectiveness of multiple user representation vectors in capturing diverse user interests and, consequently, elevating recommendation accuracy.

Generally, Ada-Retrieval consistently outperforms other models across all metrics on the three datasets, underscoring its effectiveness. This success can be attributed to two key factors: 1) Ada-Retrieval's multi-round retrieval paradigm,

| Methods | Beauty | | Sports | | Yelp | |
	HR	NDCG	HR	NDCG	HR	NDCG
DNN	13.705	4.726	8.798	2.890	11.241	3.317
MIND	14.045	5.002	8.888	2.918	11.320	3.443
ComiRec	<u>14.394</u>	<u>5.232</u>	<u>9.270</u>	<u>3.250</u>	11.479	3.523
SINE	13.191	4.325	9.087	2.978	<u>12.091</u>	<u>3.724</u>
Ada.	**17.741**	**7.062**	**11.352**	**4.234**	**12.637**	**3.852**

Table 3: Top-50 performance comparison of several baselines and Ada-Retrieval (SASRec) on three datasets.

| Methods | Beauty | | Sports | | Yelp | |
	HR	NDCG	HR	NDCG	HR	NDCG
Base	17.110	6.506	10.924	4.046	12.062	3.770
w/o LFT	17.549	6.945	11.287	4.186	12.474	3.834
w/o CAT	17.536	6.948	11.141	4.182	12.566	3.826
w/o IRA	17.209	6.670	10.980	4.101	12.331	3.815
w/o GRU	17.348	6.845	11.020	4.137	12.464	3.817
w/o MLP	17.227	6.716	10.692	3.963	12.026	3.691
w/o URA	17.308	6.793	10.782	4.017	12.147	3.720
w/o PT	17.200	6.553	10.739	3.969	12.008	3.692
Ada.	**17.741**	**7.062**	**11.352**	**4.234**	**12.637**	**3.852**

Table 4: Results of Ablation Study.

which refines user representations iteratively based on contextual information, enabling more precise identification of potential candidates across the entire item space. 2) Unlike multi-interest methods that rely on heuristic rules to determine the number of interests, Ada-Retrieval autonomously discerns the depth and range of users' preferences.

Ablation Study

Our proposed Ada-Retrieval includes a learnable filter layer (LFT), a context-aware attention layer (CAT) within the item representation adapter (IRA), and a GRU layer and MLP layer in the user representation adapter (URA). We conduct
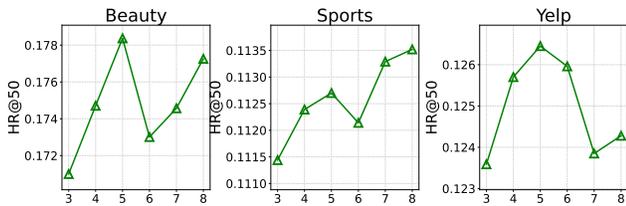
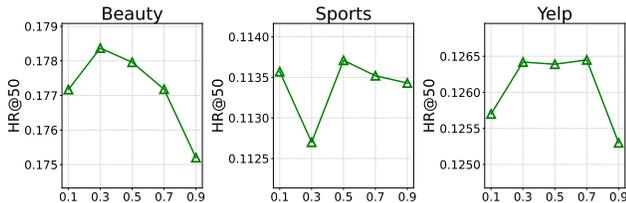Figure 3: Effect of the number of recommendation batches.

Figure 4: Sensitivity analysis of parameter λ.

Figure 5: Improvement between Ada-Retrieval and SASRec on each round.

an ablation study comparing Ada-Retrieval (SASRec) with SASRec on three datasets to analyze the contribution of each part. Additionally, we explore different training strategies, such as without pre-training (w/o PT) in Ada-Retrieval. The results are reported in Table 4.

Upon omitting the filter layer, there is a discernible drop in performance, suggesting that learnable filters play a pivotal role in mitigating the effects of noisy data within the item context. When we replace the attention layer with the average of item embeddings in context, the decline in performance suggests that the attention mechanism allows the model to automatically identify key items by assigning higher weights to them. The most pronounced degradation in w/o IRA highlights the vital role of item context data in Ada-Retrieval's user simulation process.

In terms of user context, removing either the GRU layer or the MLP module results in a significant performance drop compared to Ada-Retrieval, highlighting the effectiveness of our user representation adapter in integrating context information. Notably, omitting the MLP causes a more pronounced decline in performance than using the model without the URA. This suggests that directly incorporating the user context vector into the current user representation introduces noise, emphasizing the importance of carefully designing a fusion module to effectively leverage user context.

Additionally, jointly training Φ and Θ from scratch results in inferior performance compared to Ada-Retrieval in three datasets, highlighting the significance of the two-stage training procedure. This can be attributed to the pre-trained base model's ability to generate more accurate and robust context information, which facilitates the training of the model.

Hyper-parameter Analysis

We further investigate the impact of our model's hyperparameters, specifically T and λ, on three datasets.

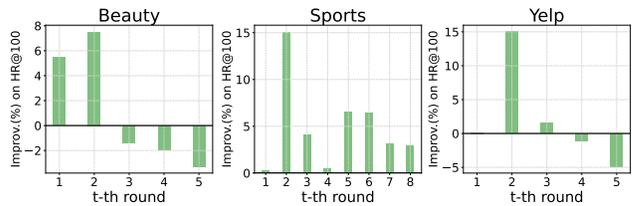In Figure 3, the optimal performance is observed when T is set to 5 for the Beauty dataset, 8 for Sports, and 6 for

Yelp. The model's performance exhibits a monotonically increasing trend as T rises from 1 to the optimal T^*. However, exceeding T^* introduces unpredictability due to excessive inference rounds.

Figure 4 reveals that the performance of Ada-Retrieval initially increases with the rise of λ. It gradually reaches its peak when λ is 0.3 for Beauty, 0.5 for Sports, and 0.7 for Yelp. Subsequently, the performance begins to decline. When the factor λ is too low or high, it fails to provide useful supervisory information for training. Therefore, choosing an appropriate value for λ with a validation set is crucial.

Analysis of Each Round

One core aspect of the Ada-Retrieval model is its cascading multi-round preference modeling of users. Thus, we compare the performance difference between Ada-Retrieval and the base model in each turn, assessing the improvement in top-k recommendations made by Ada-Retrieval (SASRec) over SASRec, as depicted in Figure 5.

Ada-Retrieval consistently exhibits substantial improvements in the early rounds, achieving a 7.5% enhancement on Beauty, 15% on Sports, and Yelp in terms of HR@100. However, as the rounds progress, the performance advantage narrows to a modest 3% on Sports and experiences a slight dip of -2.5% on Beauty. Nevertheless, when considering the overall performance, Ada-Retrieval consistently outperforms. This suggests that Ada-Retrieval excels at rapidly and precisely elevating items—those that the base model either misses or ranks lower—during the preliminary rounds.

Conclusion

In this paper, we introduce Ada-Retrieval, a novel adaptive multi-round retrieval paradigm for sequential recommendations, which provides a more dynamic and diverse approach compared to the traditional single-round inference paradigm. This model-agnostic framework incorporates key components, such as the item representation adapter and user representation adapter, to effectively refine the retrieval process in a progressive manner. Extensive experiments on publicly available datasets demonstrate the effectiveness of Ada-Retrieval, emphasizing its potential to enhance the performance of various sequential recommender systems. Future work may include investigating the theoretical foundations of the benefits provided by the multi-round retrieval paradigm and expanding its application to large-language models, such as augmenting task-planning abilities.

Acknowledgements

References

Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Cen, Y.; Zhang, J.; Zou, X.; Zhou, C.; Yang, H.; and Tang, J. 2020. Controllable multi-interest framework for recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2942–2951.

Cho, K.; Van Merriënboer, B.; Bahdanau, D.; and Bengio, Y. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Covington, P.; Adams, J.; and Sargin, E. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, 191–198.

Guo, Q.; Sun, Z.; Zhang, J.; and Theng, Y.-L. 2020. An attentional recurrent neural network for personalized next location recommendation. In *Proceedings of the AAAI Conference on artificial intelligence*, 83–90.

Hao, X.; Liu, Y.; Xie, R.; Ge, K.; Tang, L.; Zhang, X.; and Lin, L. 2021. Adversarial feature translation for multi-domain recommendation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2964–2973.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

He, R.; and McAuley, J. 2016. Fusing similarity models with markov chains for sparse sequential recommendation. In *2016 IEEE 16th international conference on data mining (ICDM)*, 191–200. IEEE.

Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.

Kang, W.-C.; and McAuley, J. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, 197–206. IEEE.

Krichene, W.; and Rendle, S. 2020. On sampled metrics for item recommendation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 1748–1757.

Lewandowski, D. 2008. The retrieval effectiveness of web search engines: considering results descriptions. *Journal of documentation*, 64(6): 915–937.

Li, C.; Liu, Z.; Wu, M.; Xu, Y.; Zhao, H.; Huang, P.; Kang, G.; Chen, Q.; Li, W.; and Lee, D. L. 2019. Multi-interest network with dynamic routing for recommendation at Tmall. In *Proceedings of the 28th ACM international conference on information and knowledge management*, 2615–2623.

Li, J.; Ren, P.; Chen, Z.; Ren, Z.; Lian, T.; and Ma, J. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 1419–1428.

Li, J.; Wang, Y.; and McAuley, J. 2020. Time interval aware self-attention for sequential recommendation. In *Proceedings of the 13th international conference on web search and data mining*, 322–330.

McAuley, J.; Targett, C.; Shi, Q.; and Van Den Hengel, A. 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.

Rendle, S. 2010. Factorization machines. In *2010 IEEE International conference on data mining*, 995–1000. IEEE.

Sherstinsky, A. 2020. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404: 132306.

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1): 1929–1958.

Sun, F.; Liu, J.; Wu, J.; Pei, C.; Lin, X.; Ou, W.; and Jiang, P. 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.

Tan, Q.; Zhang, J.; Liu, N.; Huang, X.; Yang, H.; Zhou, J.; and Hu, X. 2021a. Dynamic memory based attention network for sequential recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, 4384–4392.

Tan, Q.; Zhang, J.; Yao, J.; Liu, N.; Zhou, J.; Yang, H.; and Hu, X. 2021b. Sparse-interest network for sequential recommendation. In *Proceedings of the 14th ACM international conference on web search and data mining*, 598–606.

Tang, J.; and Wang, K. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining*, 565–573.

Wu, S.; Tang, Y.; Zhu, Y.; Wang, L.; Xie, X.; and Tan, T. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, 346–353.

Xie, R.; Liu, Q.; Liu, S.; Zhang, Z.; Cui, P.; Zhang, B.; and Lin, L. 2021. Improving accuracy and diversity in matching of recommendation with diversified preference network. *IEEE Transactions on Big Data*, 8(4): 955–967.

Xie, R.; Qiu, Z.; Rao, J.; Liu, Y.; Zhang, B.; and Lin, L. 2020. Internal and Contextual Attention Network for Cold-start Multi-channel Matching in Recommendation. In *IJCAI*, 2732–2738.

Xie, X.; Sun, F.; Liu, Z.; Wu, S.; Gao, J.; Zhang, J.; Ding, B.; and Cui, B. 2022. Contrastive learning for sequential recommendation. In *2022 IEEE 38th international conference on data engineering (ICDE)*, 1259–1273. IEEE.

Ying, R.; He, R.; Chen, K.; Eksombatchai, P.; Hamilton, W. L.; and Leskovec, J. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 974–983.

Yuan, F.; Karatzoglou, A.; Arapakis, I.; Jose, J. M.; and He, X. 2019. A simple convolutional generative network for next item recommendation. In *Proceedings of the twelfth ACM international conference on web search and data mining*, 582–590.

Zhang, Z.; and Nasraoui, O. 2006. Mining search engine query logs for query recommendation. In *Proceedings of the 15th international conference on World Wide Web*, 1039–1040.

Zhou, K.; Wang, H.; Zhao, W. X.; Zhu, Y.; Wang, S.; Zhang, F.; Wang, Z.; and Wen, J.-R. 2020. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM international conference on information & knowledge management*, 1893–1902.

Zhou, K.; Yu, H.; Zhao, W. X.; and Wen, J.-R. 2022. Filter-enhanced MLP is all you need for sequential recommendation. In *Proceedings of the ACM web conference 2022*, 2388–2399.

Zhu, H.; Chang, D.; Xu, Z.; Zhang, P.; Li, X.; He, J.; Li, H.; Xu, J.; and Gai, K. 2019. Joint optimization of tree-based index and deep model for recommender systems. *Advances in Neural Information Processing Systems*, 32.

Zhu, H.; Li, X.; Zhang, P.; Li, G.; He, J.; Li, H.; and Gai, K. 2018. Learning tree-based deep model for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1079–1088.