SIGMA: Selective Gated Mamba for Sequential Recommendation

Ziwei Liu[1*], Qidong Liu[1,2*], Yejing Wang[1], Wanyu Wang[1†], Pengyue Jia[1], Maolin Wang[1], Zitao Liu[3], Yi Chang[4], Xiangyu Zhao[1]

[1]City University of Hong Kong
[2]School of Auto. Science & Engineering, MOEKLINNS Lab, Xi'an Jiaotong University
[3]Jinan University
[4]Jilin University
{ziwliu8-c, yejing.wang, wanyuwang4-c, jia.pengyue, morin.wang}@my.cityu.edu.hk, liuqidong@stu.xjtu.edu.cn, xianzhao@cityu.edu.hk, liuzitao@jnu.edu.cn, yichang@jlu.edu.cn

Abstract

Sequential Recommender Systems (SRS) have emerged as a promising technique across various domains, excelling at capturing complex user preferences. Current SRS have employed transformer-based models to give the next-item prediction. However, their quadratic computational complexity often leads to notable inefficiencies, posing a significant obstacle to real-time recommendation processes. Recently, Mamba has demonstrated its exceptional effectiveness in time series prediction, delivering substantial improvements in both efficiency and effectiveness. However, directly applying Mamba to SRS poses certain challenges. Its unidirectional structure may impede the ability to capture contextual information in user-item interactions, while its instability in state estimation may hinder the ability to capture short-term patterns in interaction sequences. To address these issues, we propose a novel framework called **S**elect**I**ve **G**ated **MA**mba for Sequential Recommendation (SIGMA). By introducing the Partially Flipped Mamba (PF-Mamba), we construct a special bi-directional structure to address the context modeling challenge. Then, to consolidate PF-Mamba's performance, we employ an input-dependent Dense Selective Gate (DS Gate) to allocate the weights of the two directions and further filter the sequential information. Moreover, for short sequence modeling, we devise a Feature Extract GRU (FE-GRU) to capture the short-term dependencies. Experimental results demonstrate that SIGMA significantly outperforms existing baselines across five real-world datasets.

Code — https://github.com/Applied-Machine-Learning-Lab/SIMGA

Introduction

Over the past decade, sequential recommender systems (SRS) have demonstrated promising potential across various domains, including content streaming platforms (Song et al. 2022; Zhao et al. 2023a), e-commerce (Wang et al. 2020) and other domains (Li et al. 2022). To harness this potential and meet the demand for accurate next-item predictions (Fang et al. 2020; Liu et al. 2023c), an increasing number of researchers are focusing on refining existing architectures and proposing novel approaches (Wang et al. 2019; Liu et al. 2024b; Wang et al. 2023).

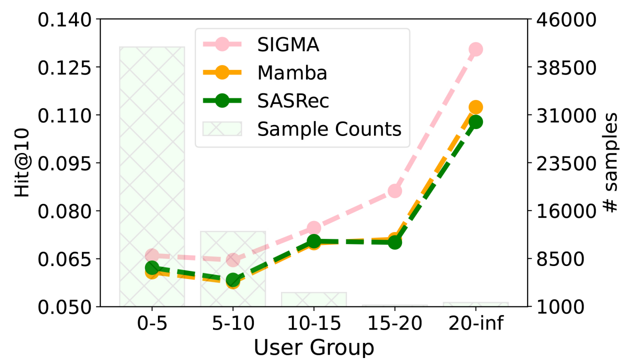

Figure 1: The illustration for long-tail user problem.

Recently, Transformer-based models have emerged as the leading approaches in sequential recommendation due to their outstanding performance (de Souza Pereira Moreira et al. 2021). By leveraging the powerful self-attention mechanism (Vaswani et al. 2017; Keles, Wijewardena, and Hegde 2023), these models have demonstrated a remarkable ability to deliver accurate predictions. However, despite their impressive performance, current transformer-based models are proven inefficient since the amount of computation grows quadratically as the length of the input sequence increases (Keles, Wijewardena, and Hegde 2023). Other approaches, such as RNN-based models (Jannach and Ludewig 2017) and MLP-based models (Li et al. 2023b; Gao et al. 2024; Liang et al. 2023), are proven to be efficient due to their linear complexity. Nevertheless, they have struggled with handling long and complex patterns (Yoon and Jang 2023). All these methods above seem to have suffered from a significant trade-off between effectiveness and efficiency. Consequently, a specially designed State Space Model (SSM) called Mamba (Gu and Dao 2023) has been proposed. By employing simple input-dependent selection on the original SSM (Liu et al. 2024a; Hamilton 1994), it has demonstrated remarkable efficiency and effectiveness.

*These authors contributed equally.
†Corresponding Author

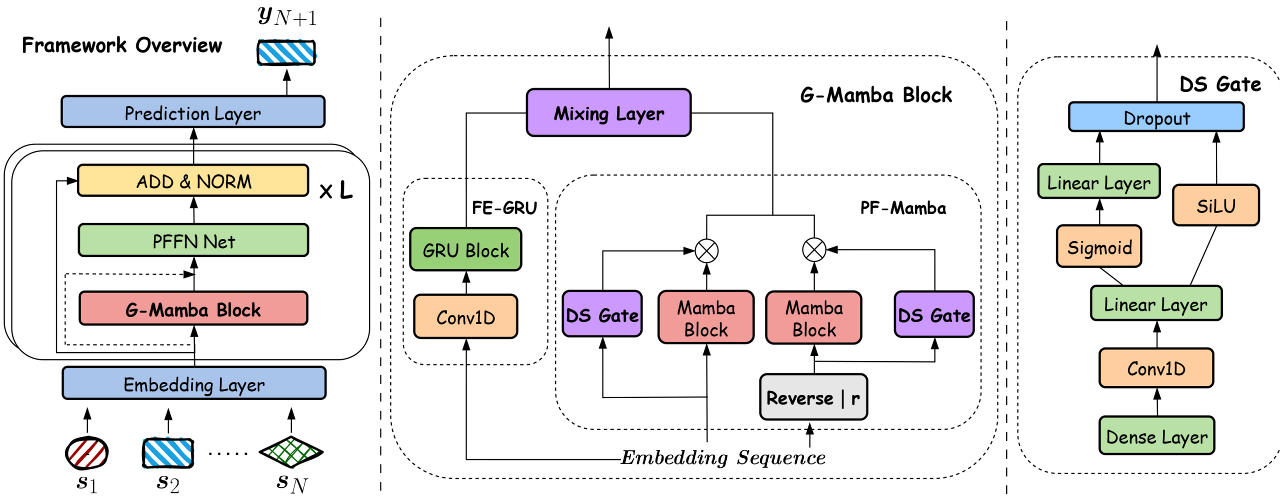

Figure 2: Framework of proposed SIGMA. The core part of this framework is the G-Mamba Block, which can directly tackle the context modeling and short sequence modeling challenges when introducing Mamba to SRS.

However, two significant challenges hinder the direct adoption of Mamba in SRS:

- **Context Modeling**: While previous researches has demonstrated Mamba's reliability in capturing sequential information (Gu and Dao 2023; Yang et al. 2024), its unidirectional architecture imposes significant limitations when applying to SRS. By only capturing users' past behaviors, Mamba can not leverage future contextual information, potentially leading to an incomplete understanding of users' preferences (Liu et al. 2024a; Sun et al. 2019). For instance, if a user consistently purchases household items but begins to show interest in sports equipment, a model that does not consider future context may struggle to recognize this shift, resulting in suboptimal next-item predictions (Jiang, Han, and Mesgarani 2024; Kweon, Kang, and Yu 2021).
- **Short Sequence Modeling**: This challenge is primarily driven by the long-tail user problem, a common issue in sequential recommendation. Long-tail users refer to such users who interact with only a few items but typically receive lower-quality recommendations compared to the normal ones (Kim et al. 2019a,b; Liu et al. 2024d). Furthermore, the instability in state estimation caused by limited data in short sequences (Gu and Dao 2023; Smith, Warrington, and Linderman 2022; Yu, Mahoney, and Erichson 2024) exacerbates this problem when Mamba is directly applied to SRS, highlighting the need for effectively modeling short sequences. For illustration, we compare two leading baselines, Mamba4Rec (Liu et al. 2024a) and SASRec (Kang and McAuley 2018), against our proposed framework on the Beauty dataset. As shown in Figure 1, the histogram depicts the number of users in each group, while the line represents recommendation performance in terms of Hit@10. SASRec outperforms Mamba4Rec in the first three groups, demonstrating Mamba4Rec's exacerbation of the long-tail user problem.

To address these challenges and better leverage Mamba's strengths, we propose an innovative framework called **S**elect**I**ve **G**ated **MA**mba for Sequential Recommendation (SIGMA). Our approach introduces the Partially Flipped Mamba (PF-Mamba), a specialized bidirectional structure that captures contextual information (Liu et al. 2024a; Jiang, Han, and Mesgarani 2024). We then introduce an input-dependent Dense Selective Gate (DS Gate) to allocate the weights of the two directions and further filter the information. Additionally, we develop a Feature Extract GRU (FE-GRU) to better model short-term patterns in interaction sequences (Hidasi et al. 2015), offering a possible solution to the long-tail user problem. Our contributions are summarized as follows:

- We identify the limitations of Mamba when applied to SRS, attributing them to its unidirectional structure and instability in state estimation for short sequences.
- We introduce SIGMA, a novel framework featuring a Partially Flipped Mamba with a Dense Selective Gate and a Feature Extract GRU, which respectively address the challenges of context modeling and short sequence modeling.
- We validate SIGMA's performance on five public real-world datasets, demonstrating its superiority.

Methodology

In this section, we will introduce a novel framework, SIGMA, which effectively addresses the aforementioned problems by adopting PF-Mamba with a Dense Selective Gate and a Feature Extract GRU. We will first present an overview of our proposed framework; then detail the important components of our architecture; and lastly introduce how we conduct our training and inference procedures.

Framework Overview

In this section, we present an overview of our proposed framework in Figure 2. Firstly, we employ an embedding layer to learn the representation for input items. After getting the high-dimensional interaction representation, we propose a G-Mamba block to selectively extract the information. Specifically, the G-Mamba block consists of a bidirectional Mamba path and a GRU path, which respectively address challenges in context modeling and short sequence modeling. Then, a Position-wise Feed-Forward Network (PFFN) is adopted to improve the modeling ability of users' actions in the hidden representation. Finally, processed by the prediction layer, we can get the accurate next-item predictions.

Embedding Layer

For existing SRS, It is necessary to map the sequential information in user-item interaction to a high-dimensional space (Zhao et al. 2023b) to effectively capture the temporal dependencies. In our framework, we choose a commonly used method for constructing the item embedding. Here, we denote the user set as $\mathcal{U} = \left\{u_1, u_2, \cdots, u_{|\mathcal{U}|}\right\}$ and the item set as $\mathcal{V} = \left\{v_1, v_2, \cdots, v_{|\mathcal{V}|}\right\}$. So for a chronologically ordered interaction sequence, it can be expressed as $\boldsymbol{S}_u = [s_1, s_2, \cdots, s_{n_u}]$, where n_u represents the length of the sequence for user $u \in \mathcal{U}$. For simplicity, we omit the mark (u) in the following sections. Regarding this interaction sequence as the input tensor, we denote D as the embedding dimension and use a learnable item embedding matrix $\boldsymbol{E} \in \mathbb{R}^{|\mathcal{V}| \times D}$ to adaptively projected s_i into the representation $\boldsymbol{h}_i$. The whole interaction sequence can be output as:

$$\boldsymbol{H}_0 = [\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_N] \tag{1}$$

where N denotes the length of user-item interactions.

G-Mamba Block

In this section, we will detail the design of our proposed G-Mamba Block. Starting with the input sequence processed by the Embedding Layer, this block introduces two paralleling paths *i.e.,* PF-Mamba and FE-GRU, which respectively address the context modeling challenge and short sequence modeling challenge. Specifically, for the contextual information loss caused by the unidirectional structure of Mamba (Gu and Dao 2023; Kweon, Kang, and Yu 2021), we introduce the Partially Flipped Mamba. It modifies the original unidirectional structure to a bi-directional one by employing a reverse block that retains the last r items while flipping the preceding items. Next, a Dense Selective Gate is proposed to properly allocate the weights of the two directions depending on the input sequence (Qin, Yang, and Zhong 2024; Zhang, Wang, and Zhao 2024). Additionally, for the long-tail user problem, we introduce the Feature Extract GRU to capture short-term preferences effectively (Hidasi et al. 2015; Kim et al. 2019b).

Partially Flipped Mamba. This module is proposed to address the context modeling challenge by leveraging the bi-directional structure. Current bi-directional methods like Dual-path Mamba (Jiang, Han, and Mesgarani 2024) or Vision Mamba (Zhu et al. 2024) usually just flip the whole input sentence to enable the global capturing capability. Although it allows the model to have a better understanding of the context, it significantly reduces the influence of short-term patterns in interaction sequences, leading to the loss of important interest dependencies. To address this issue, we introduce a partial flip method and integrate it with the Mamba block to construct a bi-directional structure. Followed by embedding sequence $\boldsymbol{H}_0$ in Equation (1), the partially flip function adaptively reverses the first n items while remaining the last r items in the input tensor from $\boldsymbol{H}_0 = [\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_n, \boldsymbol{h}_{n+1} \cdots, \boldsymbol{h}_N]$ to $\boldsymbol{H}_0^f = [\boldsymbol{h}_n, \cdots, \boldsymbol{h}_2, \boldsymbol{h}_1, \boldsymbol{h}_{n+1}, \cdots, \boldsymbol{h}_N]$. r is a pre-defined hyperparameter that equals $N - n$, which determines the range of the remaining items, *i.e.,* what extent we focus on the short-term preferences. After processing the input sequence, we utilize two Mamba blocks to construct a bi-directional architecture and process these two sequences as follows:

$$\begin{aligned} \boldsymbol{M}_0 &= \mathrm{Mamba}\left(\boldsymbol{H}_0\right) \in \mathbb{R}^{L \times D} \\ \boldsymbol{M}_0^f &= \mathrm{Mamba}\left(\boldsymbol{H}_0^f\right) \in \mathbb{R}^{L \times D} \end{aligned} \tag{2}$$

where L and D respectively represent the sequence length and hidden dimension. These two feature representations will then get a dot product with an input-dependent DS Gate to further learn the user preferences.

$$\hat{\boldsymbol{M}}_0 = \mathcal{G}_1\left(\boldsymbol{H}_0\right) \cdot \boldsymbol{M}_0 + \mathcal{G}_1\left(\boldsymbol{H}_0^f\right) \cdot \boldsymbol{M}_0^f \tag{3}$$

where $\mathcal{G}_1$ represents the designed DS Gate and $\hat{\boldsymbol{M}}_0 = [\boldsymbol{m}_1, \boldsymbol{m}_2, \cdots, \boldsymbol{m}_N]^T$ denotes the output from PF-Mamba.

Dense Selective Gate. To allocate the weights of two Mamba blocks and further filter the information according to the input sequence, we design an input-dependent Dense Selective Gate. It starts with a dense layer and a Conv1d layer to extract the original sequential information from the context, which can be formalized as follows:

$$\boldsymbol{G}_0 = \mathrm{Conv1d}\left(\boldsymbol{H}_0 \boldsymbol{W}_\sigma^{(1)} + b_\sigma^{(1)}\right) \tag{4}$$

where $\boldsymbol{H}_0$ is denoted as the output of embedding layer followed by Equation (1). Then, we introduce a forget gate and a SiLU gate (Qin, Yang, and Zhong 2024) to generate the weights from the interaction sequence:

$$\begin{aligned} \boldsymbol{\delta}_1\left(\boldsymbol{G}_0\right) &= \boldsymbol{G}_0 \boldsymbol{W}_\delta^{(1)} + b_\delta^{(1)} \\ \mathcal{G}_0(\boldsymbol{G}_0) &= \sigma\left(\boldsymbol{\delta}_1\left(\boldsymbol{G}_0\right)\right) \end{aligned} \tag{5}$$

where $W_\delta^{(1)} \in \mathbb{R}^{D \times D}$ is the weight, $\boldsymbol{b}_\delta^{(1)} \in \mathbb{R}^D$ is bias; $\mathcal{G}_0$ is denoted as the symbol of forget gate; $\sigma(\cdot)$ represents the Sigmoid activation function (He et al. 2018). By employing this $\mathcal{G}_0$, we can control the information flow in $\boldsymbol{G}_0$ to selectively retain or suppress certain information (De et al. 2024). Apart from the $\mathcal{G}_0$, we also employ a SiLU function to further improve the capability for capturing more complex patterns and features (Nwankpa et al. 2018). Therefore, We can conclude our DS Gate as follows:

$$\mathcal{G}_1(\boldsymbol{H}_0) = \mathrm{SiLU}\left(\boldsymbol{\delta}_1(\boldsymbol{G}_0)\right) + \mathcal{G}_0(\boldsymbol{G}_0) \tag{6}$$

This method allows the PF-Mamba to balance two directions of the input sequence and produce a global representation.

Feature Extract GRU. To handle Mamba's undesirable performance on short sequence modeling, we introduce one more GRU path called Feature Extract GRU in our SIGMA framework. Considering efficiency and effectiveness, we only introduce one convolution function before the GRU cell to extract and mix the features (Yuan et al. 2019). By employing this one-dimensional convolution with a well-designed kernel size, we can aggregate and extract information from the short-term pattern of the input embedding sequence. Then, we can extract the hidden representation by utilizing GRU's impressive capability to capture short-term dependencies. The whole processing procedure can then be formalized as follows:

$$\begin{aligned} \boldsymbol{C} &= \mathrm{Conv1d}\left(\boldsymbol{H}_0\right) = \left[\boldsymbol{c}_1, \boldsymbol{c}_2, \cdots, \boldsymbol{c}_n\right] \\ \boldsymbol{z}_t &= \sigma\left(\boldsymbol{W}_z \cdot \left[\boldsymbol{f}_{t-1}, \boldsymbol{c}_t\right] + \boldsymbol{b}_z\right) \\ \boldsymbol{r}_t &= \sigma\left(\boldsymbol{W}_r \cdot \left[\boldsymbol{f}_{t-1}, \boldsymbol{c}_t\right] + \boldsymbol{b}_r\right) \\ \tilde{\boldsymbol{f}}_t &= \tanh\left(\boldsymbol{W} \cdot \left[\boldsymbol{r}_t \odot \boldsymbol{f}_{t-1}, \boldsymbol{c}_t\right] + \boldsymbol{b}\right) \\ \boldsymbol{f}_t &= \boldsymbol{z}_t \odot \boldsymbol{f}_{t-1} + (1 - \boldsymbol{z}_t) \odot \tilde{\boldsymbol{f}}_t \end{aligned} \tag{7}$$

where $\sigma(\cdot)$ is the sigmoid activation function, $\boldsymbol{c}_t$ is the input of GRU module in t^{th} time step, $\boldsymbol{f}_t$ represents the t^{th} hidden states, $\boldsymbol{z}_t$ and $\boldsymbol{r}_t$ are the update gate and the reset gate, respectively. $\boldsymbol{b}_z$, $\boldsymbol{b}_{\boldsymbol{r}}$, $\boldsymbol{b}$ are bias, $\boldsymbol{W}_z$, $\boldsymbol{W}_r$, $\boldsymbol{W}$ are trainable weight matrices. The final output of FE-GRU can be denoted as $\boldsymbol{F}_0 = [\boldsymbol{f}_1, \boldsymbol{f}_2, \cdots, \boldsymbol{f}_N] \in \mathbb{R}^{L \times D}$.

Mixing Layer. To capture user-item interactions globally and get the comprehensive hidden representation, we introduce another layer to mix the outputs of the FE-GRU and PF-Mamba for the next-item prediction. The procedure can be formalized as follows:

$$\boldsymbol{Z}_0 = a_1 \boldsymbol{M} + a_2 \boldsymbol{F}_0 \in \mathbb{R}^{L \times D} \tag{8}$$

where a_1, a_2 are all trainable parameters. Then, we employ a linear layer to capture complex relationships:

$$\hat{\boldsymbol{Z}}_0 = \boldsymbol{Z}_0 \boldsymbol{W}_\delta^{(2)} + \boldsymbol{b}_\delta^{(2)} \tag{9}$$

where $\boldsymbol{W}_\delta^{(2)} \in \mathbb{R}^{D \times D}$ is the weight, $\boldsymbol{b}_\delta^{(2)} \in \mathbb{R}^D$ is bias.

PFFN Network

To capture the complex features, we further leverage a position-wise feed-forward network (PFFN Net) (Liu et al. 2024a; Kang and McAuley 2018):

$$\boldsymbol{R}_0 = \mathrm{GELU}\left(\hat{\boldsymbol{Z}}_0 \boldsymbol{W}_\delta^{(3)} + b_\delta^{(3)}\right) \boldsymbol{W}_\delta^{(4)} + b_\delta^{(4)} \tag{10}$$

where $\boldsymbol{W}_\delta^{(3)} \in \mathbb{R}^{D \times 4D}, \boldsymbol{W}_\delta^{(4)} \in \mathbb{R}^{4D \times D}, b_\delta^{(3)} \in \mathbb{R}^{4D}, b_\delta^{(4)} \in \mathbb{R}^D$ are parameters of two dense layers, $\boldsymbol{R}_0$ represents the user representation. After that, we employ a layer normalization and a residual path to stabilize the training process and ensure that the gradients flow more effectively through the network. To maintain generality, the subscript (0) here only denotes that the final user representation is obtained by 1 SIGMA layer. Actually, we can stack more such layers to better capture complex user preferences.

Train and Inference

In this subsection, we will present some details about the training and inference progress in our framework. As mentioned in Equation (10), we get the mixed hidden state representation $\boldsymbol{R}_0$, which involves the sequential information

Dataset	# Users	# Items	Sparsity	Avg.length
Yelp	82,900	64,210	99.98%	9.68
Sports	75,185	48,567	99.98%	8.07
Beauty	22,364	12,102	99.93%	8.88
ML-1M	6,041	3,417	95.53%	165.60
Games	55,145	17,287	99.94%	9.01

Table 1: The statistics of datasets

for the first N items. Assuming the embedding for items as $\mathbf{H}^{item} = \left[\mathbf{h}_1^{item}, \mathbf{h}_2^{item}, \cdots, \mathbf{h}_K^{item}\right] \in \mathbb{R}^{K \times D}$, where K denotes the total number of items. The details for the next-item prediction can be formalized as follows:

$$\begin{aligned} \mathrm{logits}_{ik} &= \sum_{j=1}^{d} \mathbf{R}_{ij} \cdot \mathbf{H}_{kj}^{item} \\ P_{ik} &= \frac{\exp(\mathrm{logits}_{ik})}{\sum_{l=1}^{M} \exp(\mathrm{logits}_{il})} \end{aligned} \tag{11}$$

Where logits_{ik} and P_{ik} respectively represent the prediction score and corresponding probability of the i-th sample for the k-th item. Correspondingly, we can formulate our Cross Entropy Loss (CE) (Zhang and Sabuncu 2018) and minimize it as:

$$\mathcal{L}_{CE} = -\frac{1}{B} \sum_{i=1}^{B} \log P_{i, y_i} \tag{12}$$

Where y_i represents the actual positive sample for i-th sample and B represents the batch size. By constantly updating the loss in each epoch, we can obtain the optimal weighting parameters and correspondingly get an accurate next-item prediction.

Experiment

In this section, we first introduce the experiment setting. Then, we present extensive experiments to evaluate the effectiveness of SIGMA.

Experiment Setting

Dataset. We conduct comprehensive experiments on five representative real-world datasets *i.e.,* Yelp[1], Amazon series[2] (Beauty, Sports and Games) and MovieLens-1M[3].The statistics of datasets after preprocessing are shown in Table 1. For the grouped user analysis, all datasets are categorized into three subsets based on user interaction length: "Short" $(0-5)$, "Medium" $(5-20)$, and "Long" $(20+)$. Additionally, we arrange user interactions sequentially by time across all datasets.

Evaluation Metrics. To assess performance, we use Top-10 Hit Rate (HR@10), Top-10 Normalized Discounted Cumulative Gain (NDCG@10), and Top-10 Mean Reciprocal Rank (MRR@10) as evaluation metrics, all of which are

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

[2]https://cseweb.ucsd.edu/jmcauley/datasets.html\#amazon_reviews

[3]https://grouplens.org/datasets/movielens/

Datasets	Eval Metrics	GRU4Rec	BERT4Rec	SASRec	LinRec	FEARec	Mamba	ECHO	**SIGMA**	Improv.
Yelp	HR@10	0.0441	0.0489	0.0551	<u>0.0579</u>	0.0554	0.0552	0.0578	**0.0629***	8.82%
	NDCG@10	0.0296	0.0317	0.0354	0.0382	<u>0.0391</u>	0.0344	0.0389	**0.0412***	5.37%
	MRR@10	0.0218	0.0243	0.0297	<u>0.0322</u>	0.0321	0.0290	0.0302	**0.0346***	7.45%
Sports	HR@10	0.0523	0.0579	0.0721	0.0709	**0.0746**	0.0676	0.0689	<u>0.0735</u>	-1.47%
	NDCG@10	0.0486	0.0501	0.0546	0.0541	<u>0.0575</u>	0.0563	0.0569	**0.0590***	2.62%
	MRR@10	0.0453	0.0477	0.0513	0.0501	0.0521	0.0527	<u>0.0534</u>	**0.0556***	4.12%
Beauty	HR@10	0.0612	0.0764	0.0852	0.0837	<u>0.0967</u>	0.0880	0.0903	**0.0986***	1.96%
	NDCG@10	0.0334	0.0395	0.0532	0.0519	0.0530	0.0540	<u>0.0567</u>	**0.0604***	6.53%
	MRR@10	0.0242	0.0285	0.0392	0.0371	0.0410	0.0436	<u>0.0447</u>	**0.0488***	7.83%
ML-1M	HR@10	0.2944	0.2977	0.2998	0.3102	<u>0.3283</u>	0.3253	0.3239	**0.3308***	0.76%
	NDCG@10	0.1652	0.1687	0.1692	0.1764	0.1843	<u>0.1868</u>	0.1848	**0.1906***	2.03%
	MRR@10	0.1252	0.1294	0.1279	0.1357	<u>0.1459</u>	0.1413	0.1429	**0.1479***	1.37%
Games	HR@10	0.1484	0.1502	0.1592	0.1604	<u>0.1616</u>	0.1564	0.1578	**0.1627***	0.68%
	NDCG@10	0.0964	0.0978	0.1002	0.1021	0.1032	<u>0.1050</u>	0.1044	**0.1088***	3.62%
	MRR@10	0.0735	0.0728	0.0794	0.0824	0.0843	<u>0.0894</u>	0.0887	**0.0924***	3.36%

Table 2: Overall performance comparison between SIGMA and other baselines. The best results are bold, and the second-best are underlined. "*" indicates the improvements are statistically significant (i.e., one-sided t-test with $p < 0.05$) over baselines.

widely used in related studies (Gu and Dao 2023; Jiang, Han, and Mesgarani 2024; De et al. 2024). These metrics offer a comprehensive evaluation of the SRS's performance. All experimental results reported are averages from five independent runs of the framework.

Implementation Details. In this section, we provide a detailed description of our framework's implementation. For GPU selection, all experiments are conducted on a single NVIDIA L4 GPU. The Adam optimizer (Kingma and Ba 2014) is used with a learning rate of 0.001. For a fair comparison, the embedding dimension for all tested models is set to 64. Other implementation details are the same as original papers (Liu et al. 2024a; Wang, He, and Zhu 2024; Kang and McAuley 2018).

Baselines. To demonstrate the effectiveness and efficiency of our proposed framework, we compare SIGMA with state-of-the-art transformer-based models (**BERT4Rec** (Sun et al. 2019), **SASRec** (Kang and McAuley 2018), **LinRec** (Liu et al. 2023a), **FEARec** (Du et al. 2023)), RNN-based models (**GRU4Rec** (Jannach and Ludewig 2017)), and SSM-based models (**Mamba4Rec** (Liu et al. 2024a), denoted as Mamba, **ECHOMamba4Rec** (Wang, He, and Zhu 2024), denoted as ECHO).

Dataset	Model	Inf. Time	GPU Mem.
Beauty	SASRec	123ms	7.58G
	FEARec	129ms	8.11G
	LinRec	<u>72ms</u>	3.08G
	Mamba	<u>72ms</u>	**2.58G**
	ECHO	78ms	3.01G
	SIGMA	**68ms**	<u>2.89G</u>
Games	SASRec	189ms	7.23G
	FEARec	260ms	7.98G
	LinRec	173ms	3.68G
	Mamba	<u>174ms</u>	3.40G
	ECHO	178ms	<u>3.19G</u>
	SIGMA	**171ms**	**3.11G**
Yelp	SASRec	443ms	9.28G
	FEARec	483ms	10.01G
	LinRec	<u>353ms</u>	7.46G
	Mamba	361ms	**7.32G**
	ECHO	368ms	<u>8.46G</u>
	SIGMA	**352ms**	8.27G

Table 3: Efficiency comparison of inference time per batch (ms) and GPU memory usage (GB).

Overall Performance Comparison

As shown in Table 2, we present a performance comparison on five datasets. The results show that our SIGMA framework outperforms all competing transformer-based, RNN-based, and SSM-based baselines, with significant improvements ranging from 0.76% to 8.82%. Such a comparison highlights the effectiveness of our unique design for combining Mamba with the sequential recommendation.

From the results, RNN-based models struggle with complex dependencies, resulting in relatively inferior performance. Besides, transformer-based models often show comparable performance, suggesting their powerful capacities in sequence modeling by self-attention. However, they still slightly lag behind our SIGMA because of the short sequence modeling problem they are facing and Mamba's more powerful abilities in capturing long-term dependency (Yang et al. 2024).

In terms of the SSM-based models, we find that they also underperform our SIGMA consistently, because of the context modeling and short sequence modeling problems mentioned before. Specifically, Mamba4Rec and

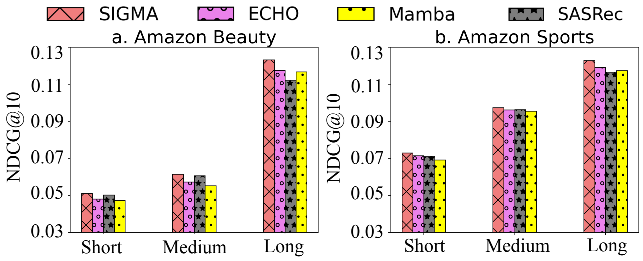

Figure 3: User group analysis on Beauty and Sports.

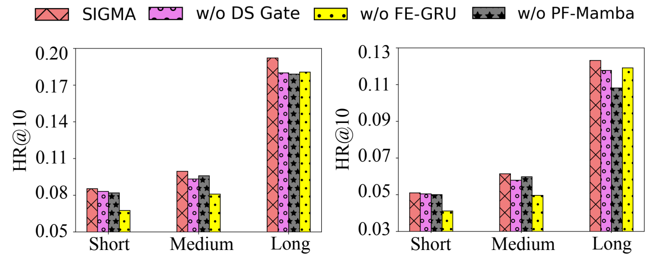

Figure 4: Ablation analysis on Beauty.

ECHOMamba4Rec show inferior performance in the Sports and Beauty datasets, whose average lengths are relatively shorter. Such a phenomenon emphasizes their weaknesses in long-tail users by direct adaptation of Mamba for the sequential recommendation.

Efficiency Comparison

In this section, we analyze the efficiency of SIGMA compared to other baselines by examining the inference time per batch and GPU memory usage during inference. The results, presented in Table 3, offer several valuable insights. First, we can find that the Mamba-based methods, including our SIGMA, can achieve higher efficiency remarkably compared with the transformer-based methods, except for LinRec. The reason lies in the simple input-dependent selection mechanism of Mamba. Then, though the efficiency-specified LinRec also owns comparable efficiency, it slightly downgrades the effectiveness of the transformer. By comparison, our SIGMA can achieve a better efficiency-effectiveness trade-off.

Grouped Users Analysis

This section presents the recommendation quality for users with varying lengths of interaction histories, aiming to provide a deeper insight into SIGMA's effectiveness in enhancing the experience of long-tail users. We illustrate the results on Beauty and Sports in Figure 3 and find that:

- Mamba4Rec that adopts the vanilla Mamba structures for SRS presents poor performance for 'short' and 'medium' users. While ECHO, which designs a bi-directional modeling module for SRS, achieves slightly better results while is still worse than SASRec.
- Our SIGMA defeats all baselines on all groups, where FE-GRU contributes to the short-sequence modeling and PF-Mamba boosts the overall performance.

Ablation Study

In this section, we analyze the efficacy of three key components within SIGMA, including PF-Mamba (partial flipping and DS gate), and FE-GRU. We design three variants: (1) *w/o partial flipping*: this variant uses the original interaction sequence without partial flipping; (2) *w/o DS gate*: the second variant linearly combines the output of two Mamba blocks; (3) *w/o FE-GRU*: this variant drops the Feature Extract GRU. We test these variants on Beauty and present results in Table 4 and Figure 4. We can conclude that:

Model Components	HR@10	NDCG@10	MRR@10
Default	**0.0986**	**0.0604**	**0.0488**
w/o partial flipping	0.0953	0.0586	0.0473
w/o DS gate	0.0954	0.0571	0.0455
w/o FE-GRU	0.0750	0.0470	0.0382

Table 4: Ablation study on Beauty.

- With the bi-directional interaction sequences, partial flipping contributes to improving the recommendation performance for all users.
- DS gate significantly boosts the SIGMA by balancing the information from two directions.
- FE-GRU is crucial for enhancing the experience of users with few interactions with strong short sequence modeling ability. And it has a huge impact on the overall performance, highlighting the importance of tackling the long-tail user problem.

Hyperparameter Analysis

In this section, we conduct experiments on Beauty to analyze the influence of two significant hyperparameters: (i) r, the remaining range in the partial flipping method; (ii) L, the number of stacked SIGMA layers. The results are respectively visualized in Figure 5 and Table 5.

From Figure 5, we can find that our proposed SIGMA framework achieves the best results when $r = 5$, offering two valuable insights as follows: (i) when r is relatively large ($r = N$ represents "w/o flipping"), it is challenging for SIGMA to leverage the limited bi-directional information ($N - r$ items are flipped); (ii) when r is relatively small ($r = 0$ represents "whole flipping"), users may lose the short-term preference due to the exceeding flipping range, which is reflected as a varied Hit@10 and NDCG@10 performance in Figure 5. These phenomenons justify the significance of partial flipping with a proper r, defending the effectiveness of SIGMA.

From Table 5, we observe that increasing the number of SIGMA layers does not guarantee the improvement of recommendation performance, but significantly impairs the inference efficiency, which can be attributed to the overfitting problem of multiple SIGMA layers. In addition, it is noteworthy that the performance of a single SIGMA layer is very close to the optimal one, indicating the strong modeling ability and superior efficiency of SIGMA.

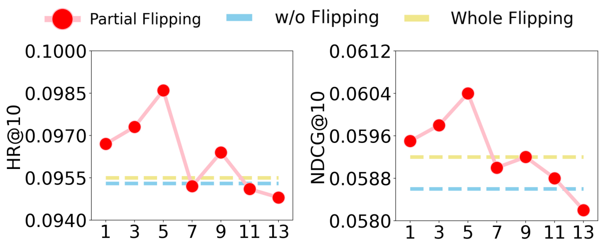

Figure 5: Parameter study for r on Beauty.

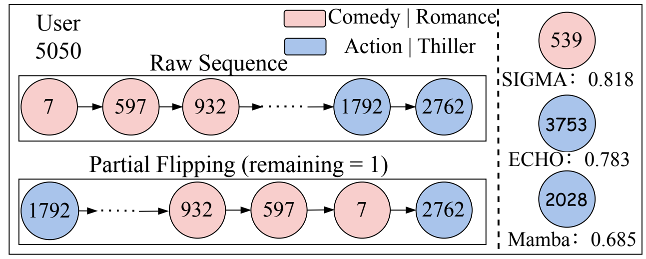

Figure 6: Case study for User-5050 in ML-1M.

#layers	HR@10	NDCG@10	Inf Time	GPU Mem
1	0.0986	0.0604	66ms	3.03G
2	0.0994	0.0611	122ms	4.30G
4	0.0963	0.0589	227ms	6.83G

Table 5: Parameter study for L on Beauty.

Methods	HR@10	NDCG@10	MRR@10
Mamba	0.2998	0.1776	0.1391
ECHO	0.3041	0.1799	0.1403
SIGMA	**0.3139**	**0.1825**	**0.1423**
Improv.	3.22%	1.44%	1.42%

Table 6: Performance comparison on User-5050.

Case Study

In this section, we leverage a specific example in ML-1M to illustrate the effectiveness of partial flipping in SIGMA. Specifically, we choose a user (ID: 5050) and present the interaction sequence before and after the partial flipping in the left part of Figure 6. With $r = 1$, only the last item 2762 remained at the original position, and other items are flipped. From this example, we can find that this user prefers comedy and romance movies (pink balls), as well as action and thriller movies (blue balls). Without the flipping, baselines focus on the most recent interactions on action and thriller movies and provide incorrect recommendations of the same genres (movie 3753 and 2028). While our SIGMA, with PF-Mamba, notices the previous preference for comedy and romance movies, makes the accurate recommendation of movie 539. Furthermore, we also present the overall performance for user-5050 in Table 6, where SIGMA significantly defeats baselines.

Related Work

Sequential Recommendation

Advancements in deep learning have enhanced recommendation systems, improving next-item prediction accuracy (Liu et al. 2023b; Wang et al. 2024; Liu et al. 2024c). Early frameworks using CNNs and RNNs struggled with long-term dependencies due to catastrophic forgetting (de Souza Pereira Moreira et al. 2021; Kim et al. 2019a). Transformer-based models, with their self-attention mechanism, significantly improved performance by capturing complex user-item interactions (Li et al. 2023a), but they suffer from inefficiency due to quadratic computational complexity (Keles, Wijewardena, and Hegde 2023). To address this trade-off, we propose SIGMA, a novel framework that combines effectiveness and efficiency.

Selective State Space Model

Currently, SSM-based models have been proven effective in time-series prediction due to their ability to capture the hidden dynamics (Smith, Warrington, and Linderman 2022; Hamilton 1994). To further address the issues of catastrophic forgetting and long-term dependency in sequential processing, a special SSM called Mamba was introduced. Attributing to its unique selectivity (Gu and Dao 2023), Mamba shows remarkable performance without leveraging any sequence denoising methods (Zhang et al. 2023, 2022; Lin et al. 2023) or feature selecting methods (Lin et al. 2022), even when addressing long sequences (Yang et al. 2024). However, it still suffers from some challenges when applying to SRS *i.e.,* context and short sequence modeling, which are mainly caused by Mamba's original structure and the inflexibility in hidden state transferring. Correspondingly, we introduce a bi-directional module called Partially Flipped Mamba and a Feature Extract GRU in our SIGMA framework, which somewhat address these problems and explores a novel way to leverage Mamba in SRS.

Conclusion

In this paper, we analyze the challenges of applying Mamba to SRS and propose a novel framework, SIGMA, to address these challenges. We introduce a bidirectional PF-Mamba, featuring a well-designed DS gate, to allocate the weights of each direction and address the context modeling challenge, enabling our framework to leverage information from both past and future user-item interactions. Furthermore, to address the challenge of short sequence modeling, we propose FE-GRU to enhance the hidden representations for interaction sequences, mitigating the impact of long-tail users to some extent. Finally, we conduct extensive experiments on five real-world datasets, verifying SIGMA's superiority and validating the effectiveness of each module.

Acknowledgements

This research was partially supported by Collaborative Research Fund No.C1043-24GF, Research Impact Fund (No.R1015-23), APRC - CityU New Research Initiatives (No.9610565, Start-up Grant for New Faculty of CityU), CityU - HKIDS Early Career Research Grant (No.9360163), Hong Kong ITC Innovation and Technology Fund Midstream Research Programme for Universities Project (No.ITS/034/22MS), Hong Kong Environmental and Conservation Fund (No. 88/2022), and SIRG - CityU Strategic Interdisciplinary Research Grant (No.7020046), Huawei (Huawei Innovation Research Program), Tencent (CCF-Tencent Open Fund, Tencent Rhino-Bird Focused Research Program), Ant Group (CCF-Ant Research Fund, Ant Group Research Fund), Alibaba (CCF-Alimama Tech Kangaroo Fund No. 2024002), CCF-BaiChuan-Ebtech Foundation Model Fund, and Kuaishou.

References

De, S.; Smith, S. L.; Fernando, A.; Botev, A.; Cristian-Muraru, G.; Gu, A.; Haroun, R.; Berrada, L.; Chen, Y.; Srinivasan, S.; et al. 2024. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *arXiv preprint arXiv:2402.19427*.

de Souza Pereira Moreira, G.; Rabhi, S.; Lee, J. M.; Ak, R.; and Oldridge, E. 2021. Transformers4rec: Bridging the gap between nlp and sequential/session-based recommendation. In *Proceedings of the 15th ACM conference on recommender systems*, 143–153.

Du, X.; Yuan, H.; Zhao, P.; Qu, J.; Zhuang, F.; Liu, G.; Liu, Y.; and Sheng, V. S. 2023. Frequency enhanced hybrid attention network for sequential recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 78–88.

Fang, H.; Zhang, D.; Shu, Y.; and Guo, G. 2020. Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations. *ACM Transactions on Information Systems (TOIS)*, 39(1): 1–42.

Gao, J.; Zhao, X.; Li, M.; Zhao, M.; Wu, R.; Guo, R.; Liu, Y.; and Yin, D. 2024. SMLP4Rec: An Efficient all-MLP Architecture for Sequential Recommendations. *ACM Transactions on Information Systems*, 42(3): 1–23.

Gu, A.; and Dao, T. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Hamilton, J. D. 1994. State-space models. *Handbook of econometrics*, 4: 3039–3080.

He, J.; Li, L.; Xu, J.; and Zheng, C. 2018. ReLU deep neural networks and linear finite elements. *arXiv preprint arXiv:1807.03973*.

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

Jannach, D.; and Ludewig, M. 2017. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the eleventh ACM conference on recommender systems*, 306–310.

Jiang, X.; Han, C.; and Mesgarani, N. 2024. Dual-path mamba: Short and long-term bidirectional selective structured state space models for speech separation. *arXiv preprint arXiv:2403.18257*.

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

Keles, F. D.; Wijewardena, P. M.; and Hegde, C. 2023. On the computational complexity of self-attention. In *International Conference on Algorithmic Learning Theory*, 597–619. PMLR.

Kim, Y.; Kim, K.; Park, C.; and Yu, H. 2019a. Sequential and Diverse Recommendation with Long Tail. In *IJCAI*, volume 19, 2740–2746.

Kim, Y.; Kim, K.; Park, C.; and Yu, H. 2019b. Sequential and Diverse Recommendation with Long Tail. In *IJCAI*, volume 19, 2740–2746.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kweon, W.; Kang, S.; and Yu, H. 2021. Bidirectional distillation for top-K recommender system. In *Proceedings of the Web Conference 2021*, 3861–3871.

Li, C.; Wang, Y.; Liu, Q.; Zhao, X.; Wang, W.; Wang, Y.; Zou, L.; Fan, W.; and Li, Q. 2023a. STRec: Sparse transformer for sequential recommendations. In *Proceedings of the 17th ACM Conference on Recommender Systems*, 101–111.

Li, M.; Zhang, Z.; Zhao, X.; Wang, W.; Zhao, M.; Wu, R.; and Guo, R. 2023b. Automlp: Automated mlp for sequential recommendations. In *Proceedings of the ACM Web Conference 2023*, 1190–1198.

Li, X.; Qiu, Z.; Zhao, X.; Wang, Z.; Zhang, Y.; Xing, C.; and Wu, X. 2022. Gromov-wasserstein guided representation learning for cross-domain recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 1199–1208.

Liang, J.; Zhao, X.; Li, M.; Zhang, Z.; Wang, W.; Liu, H.; and Liu, Z. 2023. Mmmlp: Multi-modal multilayer perceptron for sequential recommendations. In *Proceedings of the ACM Web Conference 2023*, 1109–1117.

Lin, W.; Zhao, X.; Wang, Y.; Xu, T.; and Wu, X. 2022. AdaFS: Adaptive feature selection in deep recommender system. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 3309–3317.

Lin, W.; Zhao, X.; Wang, Y.; Zhu, Y.; and Wang, W. 2023. Autodenoise: Automatic data instance denoising for recommendations. In *Proceedings of the ACM Web Conference 2023*, 1003–1011.

Liu, C.; Lin, J.; Wang, J.; Liu, H.; and Caverlee, J. 2024a. Mamba4rec: Towards efficient sequential recommendation with selective state space models. *arXiv preprint arXiv:2403.03900*.

Liu, L.; Cai, L.; Zhang, C.; Zhao, X.; Gao, J.; Wang, W.; Lv, Y.; Fan, W.; Wang, Y.; He, M.; et al. 2023a. Linrec: Linear attention mechanism for long-term sequential recommender systems. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 289–299.

Liu, Q.; Hu, J.; Xiao, Y.; Zhao, X.; Gao, J.; Wang, W.; Li, Q.; and Tang, J. 2024b. Multimodal recommender systems: A survey. *ACM Computing Surveys*, 57(2): 1–17.

Liu, Q.; Wu, X.; Wang, W.; Wang, Y.; Zhu, Y.; Zhao, X.; Tian, F.; and Zheng, Y. 2024c. Large language model empowered embedding generator for sequential recommendation. *arXiv preprint arXiv:2409.19925*.

Liu, Q.; Wu, X.; Zhao, X.; Wang, Y.; Zhang, Z.; Tian, F.; and Zheng, Y. 2024d. Large Language Models Enhanced Sequential Recommendation for Long-tail User and Item. *arXiv preprint arXiv:2405.20646*.

Liu, S.; Cai, Q.; Sun, B.; Wang, Y.; Jiang, J.; Zheng, D.; Jiang, P.; Gai, K.; Zhao, X.; and Zhang, Y. 2023b. Exploration and regularization of the latent action space in recommendation. In *Proceedings of the ACM Web Conference 2023*, 833–844.

Liu, Z.; Tian, J.; Cai, Q.; Zhao, X.; Gao, J.; Liu, S.; Chen, D.; He, T.; Zheng, D.; Jiang, P.; et al. 2023c. Multi-task recommendations with reinforcement learning. In *Proceedings of the ACM Web Conference 2023*, 1273–1282.

Nwankpa, C.; Ijomah, W.; Gachagan, A.; and Marshall, S. 2018. Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv:1811.03378*.

Qin, Z.; Yang, S.; and Zhong, Y. 2024. Hierarchically gated recurrent neural network for sequence modeling. *Advances in Neural Information Processing Systems*, 36.

Smith, J. T.; Warrington, A.; and Linderman, S. W. 2022. Simplified state space layers for sequence modeling. *arXiv preprint arXiv:2208.04933*.

Song, F.; Chen, B.; Zhao, X.; Guo, H.; and Tang, R. 2022. Autoassign: Automatic shared embedding assignment in streaming recommendation. In *2022 IEEE International Conference on Data Mining (ICDM)*, 458–467. IEEE.

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.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, H.; Liu, X.; Fan, W.; Zhao, X.; Kini, V.; Yadav, D.; Wang, F.; Wen, Z.; Tang, J.; and Liu, H. 2024. Rethinking large language model architectures for sequential recommendations. *arXiv preprint arXiv:2402.09543*.

Wang, J.; Louca, R.; Hu, D.; Cellier, C.; Caverlee, J.; and Hong, L. 2020. Time to Shop for Valentine's Day: Shopping Occasions and Sequential Recommendation in E-commerce. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, 645–653.

Wang, S.; Hu, L.; Wang, Y.; Cao, L.; Sheng, Q. Z.; and Orgun, M. 2019. Sequential recommender systems: challenges, progress and prospects. *arXiv preprint arXiv:2001.04830*.

Wang, Y.; He, X.; and Zhu, S. 2024. EchoMamba4Rec: Harmonizing Bidirectional State Space Models with Spectral Filtering for Advanced Sequential Recommendation. *arXiv preprint arXiv:2406.02638*.

Wang, Y.; Lam, H. T.; Wong, Y.; Liu, Z.; Zhao, X.; Wang, Y.; Chen, B.; Guo, H.; and Tang, R. 2023. Multi-task deep recommender systems: A survey. *arXiv preprint arXiv:2302.03525*.

Yang, J.; Li, Y.; Zhao, J.; Wang, H.; Ma, M.; Ma, J.; Ren, Z.; Zhang, M.; Xin, X.; Chen, Z.; et al. 2024. Uncovering Selective State Space Model's Capabilities in Lifelong Sequential Recommendation. *arXiv preprint arXiv:2403.16371*.

Yoon, J. H.; and Jang, B. 2023. Evolution of deep learning-based sequential recommender systems: from current trends to new perspectives. *IEEE Access*, 11: 54265–54279.

Yu, A.; Mahoney, M. W.; and Erichson, N. B. 2024. There is HOPE to Avoid HiPPOs for Long-memory State Space Models. *arXiv preprint arXiv:2405.13975*.

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, C.; Chen, R.; Zhao, X.; Han, Q.; and Li, L. 2023. Denoising and prompt-tuning for multi-behavior recommendation. In *Proceedings of the ACM Web Conference 2023*, 1355–1363.

Zhang, C.; Du, Y.; Zhao, X.; Han, Q.; Chen, R.; and Li, L. 2022. Hierarchical item inconsistency signal learning for sequence denoising in sequential recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2508–2518.

Zhang, S.; Wang, M.; and Zhao, X. 2024. GLINT-RU: Gated Lightweight Intelligent Recurrent Units for Sequential Recommender Systems. *arXiv preprint arXiv:2406.10244*.

Zhang, Z.; and Sabuncu, M. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31.

Zhao, K.; Liu, S.; Cai, Q.; Zhao, X.; Liu, Z.; Zheng, D.; Jiang, P.; and Gai, K. 2023a. KuaiSim: A comprehensive simulator for recommender systems. *Advances in Neural Information Processing Systems*, 36: 44880–44897.

Zhao, X.; Wang, M.; Zhao, X.; Li, J.; Zhou, S.; Yin, D.; Li, Q.; Tang, J.; and Guo, R. 2023b. Embedding in recommender systems: A survey. *arXiv preprint arXiv:2310.18608*.

Zhu, L.; Liao, B.; Zhang, Q.; Wang, X.; Liu, W.; and Wang, X. 2024. Vision mamba: Efficient visual representation learning with bidirectional state space model. *arXiv preprint arXiv:2401.09417*.