GLINT-RU: Gated Lightweight Intelligent Recurrent Units for Sequential Recommender Systems

Sheng Zhang*
City University of Hong Kong and High Magnetic Field Laboratory, Chinese Academy of Sciences
Hefei and Hong Kong, China
szhang844-c@my.cityu.edu.hk

Maolin Wang*
City University of Hong Kong
Hong Kong, China
morin.wang@my.cityu.edu.hk

Wanyu Wang†
City University of Hong Kong
Hong Kong, China
wanyu.wang@my.cityu.edu.hk

Jingtong Gao
City University of Hong Kong
Hong Kong, China
jingtong.gao@my.cityu.edu.hk

Xiangyu Zhao
City University of Hong Kong
Hong Kong, China
xianzhao@cityu.edu.hk

Yu Yang
City University of Hong Kong
Hong Kong, China
yu.yang@cityu.edu.hk

Xuetao Wei
Southern University of Science and Technology
Shenzhen, China
weixt@sustech.edu.cn

Zitao Liu
Jinan University
Guangzhou, China
liuzitao@jnu.edu.cn

Tong Xu
University of Science and Technology of China
Hefei, China
tongxu@ustc.edu.cn

Abstract

Transformer-based models have gained significant traction in sequential recommender systems (SRSs) for their ability to capture user-item interactions effectively. However, these models often suffer from high computational costs and slow inference. Meanwhile, existing efficient SRS approaches struggle to embed high-quality semantic and positional information into latent representations. To tackle these challenges, this paper introduces **GLINT-RU**, a lightweight and efficient SRS leveraging a single-layer dense selective Gated Recurrent Units (GRU) module to accelerate inference. By incorporating a dense selective gate, GLINT-RU adaptively captures temporal dependencies and fine-grained positional information, generating high-quality latent representations. Additionally, a parallel mixing block infuses fine-grained positional features into user-item interactions, enhancing both recommendation quality and efficiency. Extensive experiments on three datasets demonstrate that GLINT-RU achieves superior prediction accuracy and inference speed, outperforming baselines based on RNNs, Transformers, MLPs, and SSMs. These results establish GLINT-RU as a powerful and efficient solution for SRSs. The implementation code is publicly available for reproducibility. [1].

CCS Concepts

• **Information systems** → **Recommender systems**.

Keywords

Recommender Systems, Sequential Recommender Systems, Gated Recurrent Units, Efficient Model

ACM Reference Format:
Sheng Zhang, Maolin Wang, Wanyu Wang, Jingtong Gao, Xiangyu Zhao, Yu Yang, Xuetao Wei, Zitao Liu, and Tong Xu. 2025. GLINT-RU: Gated Lightweight Intelligent Recurrent Units for Sequential Recommender Systems. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1 (KDD '25), August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3690624.3709304

1 Introduction

In this era of data exploding, sequential recommender systems (SRSs) [10, 12, 16–18, 20, 33, 35, 37] have gained much attention in capturing users' preferences within a large amount of sequential interaction data. GRU4Rec [10] as one of the earliest session-based recommendation models, employs stacked Gated Recurrent Units (GRU) for item-to-item recommendations. However, the RNN-based methods [10, 15] are fading from the recommendation realm due to their relatively lower accuracy. In recent years, transformer-based SRSs [12, 26] have become increasingly popular for the powerful multi-head attention mechanism [28, 36]. They exhibit remarkable ability in capturing sequential interactions and delivering accurate predictions [14]. However, despite their effectiveness, current transformer-based models suffer from substantial computational demands and extended inference time, which is caused by the dot product operation in the attention mechanism [18, 29, 40].

To tackle the issue of the high computational cost of transformer-based SRSs, linear attention mechanisms are applied to reduce the computational complexity. For example, LinRec [18] changes the

*Equal contribution.
†Corresponding author.
[1]https://github.com/szhang-cityu/GLINT-RU

KDD '25, Toronto, ON, Canada

ACM ISBN 979-8-4007-1245-6/25/08
https://doi.org/10.1145/3690624.3709304

order of dot product between query and key matrices by designing a special mapping, dramatically reducing the inference time. LightSAN [6] projects historical interactions into interest representations with shorter lengths, thereby reducing the computational complexity of transformers to a linear scale. MLP-based frameworks [16, 21, 39] can also achieve fast inference speed and high performance by reshaping input sequence tensors [7]. SSM-based models [17, 31] outperforms the existing attention mechanism by utilizing the efficient selective SSM [9], within which a structured state tensor is used to address long-range dependencies.

However, to achieve high performance, the transformer-based SRSs, even with linear attention mechanism, require deeply stacked transformers, which decreases the model efficiency [34]. MLP-based SRSs with sequence mixing layers might suffer from extended inference time when dealing with long sequences. In addition, MLP-based models struggle to capture fine-grained positional dependencies. SSM-based models [17] may struggle to model effective semantic features into latent representations in long/short-term recommendation scenarios, as they might have difficulty learning important interactions. In this paper, we aim to further improve the efficiency and the accuracy of efficient models in various scenarios.

To further reduce resource consumption, accelerate inference speed, and enhance the model performance, we propose a novel efficient SRS framework named **G**ated **L**ightweight **I**ntellige**NT** **R**ecurrent **U**nits (GLINT-RU). Considering that stacking hybrid architectures may lead to a deeper model, which will cause a significant decrease in inference speed. and the traditional GRU module lacks the ability to adaptively adjust the output and filter out unimportant interactions. To tackle these challenges, we employ various gate mechanisms in appropriate positions to fully perceive the data environment and filter information automatically [1, 3, 34]. We introduce an expert mixing block that captures the item dependencies via GRU and utilizes linear attention to capture the global interaction information between users and items [11, 22]. This strategy not only improves the inference speed due to the linear computational complexity of paralleled GRU and linear attention mechanism but also enhances the context information. Additionally, we implement a dense selective GRU, which selects the output of the GRU adaptively and considers the connections among adjacent items. It leverages the gate mechanism to select crossed channels and extracts short-duration patterns to refine the model's understanding of user behavior dynamics. Moreover, a gated MLP block is utilized to select the outputs of the expert mixing block, deeply filtering the information based on the data environment.

We summarize the major contributions of our work as follows:

- In this paper, we introduce GLINT-RU, a novel and lightweight SRS that achieves remarkable inference speed only requiring a single layer. It is an advanced model that captures complex semantic features and fine-grained positional representations using an expert mixing mechanism, which substantially improves the performance of GLINT-RU over existing efficient SRS models.
- We introduce a dense selective GRU module, which not only incorporates connections between adjacent items but also empowers the model with the capability to selectively learn long-term dependencies. The integration of this advanced GRU module into the model markedly elevates its performance, establishing a new standard for efficient recommender systems.
- We conduct extensive experiments to verify the efficiency and effectiveness of GLINT-RU on various datasets and parameter settings. Our GLINT-RU improves the training and inference speed significantly and stably exhibits high performance.

2 Preliminaries

In this section we will briefly introduce our recommendation task, and then introduce the basic efficient GRU and linear attention modules used in our framework.

2.1 Problem Statement

For a sequential recommendation task, we have a set of users $\mathcal{U} = \{u_1, u_2, \ldots, u_{|\mathcal{U}|}\}$ who have historical interactions with a set of items $\mathcal{V} = \{v_1, v_2, \ldots, v_{|\mathcal{V}|}\}$. Among these users, the i-th user has a preferred item sequence denoted as $s_i = [v_1^{(i)}, v_2^{(i)}, \ldots, v_{n_i}^{(i)}]$, where n_i is the length of the item list that the i-th user interacts with. Our goal is to design an efficient framework and predict the rating score of the next item based on the historical interactions .

2.2 Gated Recurrent Units

As an essential part of GLINT-RU, the GRU [2] module contributes to the recommendation task by capturing the dependencies among the items and dynamically adjust its memory content [4, 24]. The update mechanism of a GRU cell is formulated as follows:

$$\begin{aligned} \boldsymbol{z}_t &= \sigma(\boldsymbol{W}_z \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_z), \\ \boldsymbol{r}_t &= \sigma(\boldsymbol{W}_r \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_r), \\ \tilde{\boldsymbol{h}}_t &= \tanh(\boldsymbol{W} \cdot [\boldsymbol{r}_t * \boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}), \\ \boldsymbol{h}_t &= \boldsymbol{z}_t * \boldsymbol{h}_{t-1} + (1 - \boldsymbol{z}_t) * \tilde{\boldsymbol{h}}_t, \end{aligned} \tag{1}$$

where $\sigma(\cdot)$ is the sigmoid activation function, $\boldsymbol{x}_t$ is the input of GRU module in t-th time step, $\boldsymbol{h}_t$ represents the t-th hidden states, $\boldsymbol{z}_t$ and $\boldsymbol{r}_t$ are the update gate and the reset gate, respectively. $\boldsymbol{b}_z, \boldsymbol{b}_r, \boldsymbol{b}$ are bias, $\boldsymbol{W}_z, \boldsymbol{W}_r, \boldsymbol{W}$ are trainable weight matrices. As is shown in Eq.(1), GRU uses the update gate to control the retained information volume from previous hidden states in the current time step, while the reset gate controls the information that should be forgotten.

The GRU (Gated Recurrent Unit) module, equipped with update and reset gates in sequential GRU Cells, is adept at capturing the relationships among the items throughout the sequence while maintaining a relatively low computational complexity. However, the sequential information in GRU cannot be interacted with, and each hidden state is primarily encoded from preceding elements, which restricts the representational capacity of GRU-based recommendation models, particularly in capturing complex item dependencies across the entire sequence.

2.3 Linear Attention Mechanism

The attention mechanism as a powerful core of the transformer structure, exhibits performance in learning sequence interactions in recommendation tasks. However, the high computational cost of the dot product between query matrix $\boldsymbol{Q}$ and key matrix $\boldsymbol{K}$ substantially lower the inference speed of transformer-based SRSs especially when the sequence length N is much larger than hidden size d [25, 29]. To tackle this issue, the linear attention mechanism [18] designs a special mapping function to change the order of the dot product

and reduce the computational complexity to $O(Nd^2)$. The linear attention mechanism can be written as:

$$A'(Q,K,V) = \mathcal{X}_1\left(elu(Q)\right)\left(\mathcal{X}_2\left(elu(K)\right)^{\mathrm{T}}V\right), \tag{2}$$

where $\mathcal{X}_1$ and $\mathcal{X}_2$ are row-wise and column-wise L_2 normalization mappings, respectively, Q, K, V are learnable query, key and value matrices and A' is the output attention score. This approach mitigates the issue that the softmax layer concentrates on the scores of merely a few positions, enlarging the information capacity of the attention mechanism [18]. By implementing linear attention, our GLINT-RU framework is capable of learning interactions between items in long sequences.

3 METHODOLOGY

In this section, we will introduce the overall framework and its components that effectively capture semantic features and positional information, followed by the complexity analysis of GLINT-RU.

3.1 Framework Overview

Many existing recommendation frameworks depend on transformer structure [12, 18, 26], which incurs substantial computational overhead and low inference speed. Restricted by the large computational complexity of stacked transformers, linear attention-based models have approached a plateau in terms of minimizing inference time and resource consumption. Uniquely, we propose an advanced recommendation framework that integrates the linear attention mechanism and efficient dense selective GRU module, which further reduces the computational cost compared with stacked linear transformers and SSM-based models. Additionally, this dense selective GRU module also enables our framework to understand both semantic features and dependencies from item sequences, and substantially reduce the computational cost and inference time.

Figure 1.(a) shows the structure of our GLINT-RU. As is shown in Figure 1.(b)-(d), GLINT-RU integrates an **expert mixing block** for mixing sequential information from the **dense selective GRU** expert and the linear attention expert, and a **gated MLP block** for further learning and filtering complex user behaviors.

In the **expert mixing block**, the **dense selective GRU** module is employed to capture the long/short-term item-wise dependencies, and selectively learn the sequential information. In addition, the linear attention expert is responsible for modeling item interactions from the user. By combining these two powerful expert modules, our GLINT-RU is capable of adaptively learning both temporal and semantic item features from the sequence, which performs fine-grained modeling of complex user behaviors.

After the user-item interactions are selectively learned by mixing block, the item scores are conveyed to the **gated MLP block**, where the information is filtered according to the data environment. The framework employs various gates in appropriate positions to deeply filter information, improving the model's flexibility and the ability to perceive and select information.

3.2 Item Embedding Layer

For sequential recommendation tasks, information on items interacted by users should be encoded to tensors through the embedding layer [38]. We denote the length of input user-item interactions as N, and embedding size as d. For a interaction sequence $s_i = [v_1, v_2, \ldots, v_n, \ldots v_{n_i}]$, the n-th item $v_n \in \mathbb{R}^{D_n}$ can be projected into the representation $\boldsymbol{e}_n$ by the following formulation:

$$\boldsymbol{e}_n = \boldsymbol{W}_n v_n, \tag{3}$$

where $\boldsymbol{W}_n \in \mathbb{R}^{d\times D_n}$ is trainable weighted matrix. The embedding layer outputs the encoded item sequence in a tensor:

$$\boldsymbol{E} = [\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_N]^{\mathrm{T}}. \tag{4}$$

In traditional transformer-based models, positional embeddings are typically necessary because the attention mechanism lacks the inherent capability to encode temporal information [12]. Uniquely, in this paper we employ the GRU module to model temporal item dependencies, which generates fine-grained representations with positional information for the items. Therefore, we decide not to add the positional embedding layer into the framework.

3.3 Dense Selective GRU

Existing GRU cell learns sequential data by conveying information from preceding cells. Although this mechanism has the superiority of capturing the long-term dependencies in the sequence, it predominantly focuses on information from previous items, while potentially overlooking the valuable context information provided by adjacent items, which are often closely related in real-world applications. To address these challenges, we introduce dense selective GRU shown in Figure 1.(b) as the core component of GLINT-TU. This innovation extracts local temporal features and generates fine-grained positional information using the update mechanism in GRU module. By implementing dense selective GRU, the computational complexity can be further reduced, and the recommendation accuracy of the GRU-based framework can be substantially improved.

3.3.1 ***Dense GRU module.*** Therefore, to enable each GRU cell to aggregate local temporal features of user behavior, we introduce a temporal convolution layer , where adjacent item information is adaptively fused before being fed into the GRU module:

$$\boldsymbol{C} = \mathrm{TemporalConv1d}(\boldsymbol{X}\boldsymbol{W_0} + \boldsymbol{b_0}) \tag{5}$$

where $\boldsymbol{X} = [x_1, x_2, \ldots, \boldsymbol{x}_N]^{\mathrm{T}}$ is the input tensor with d feature dimensions, and $\boldsymbol{C} = [\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_N]^{\mathrm{T}}$ is the output of the convolution operation $\mathrm{TemporalConv1d}(\cdot)$ with N steps. $\boldsymbol{W}_0$ and $\boldsymbol{b}_0$ are weight matrix and bias, respectively. The size of the convolution kernel is set as k. According to Eq.(1), the output of the $\mathrm{GRUCell}(\cdot)$ can be divided into a latent item representation $\hat{\boldsymbol{h}}_n$ and a fine-grained positional representation $\boldsymbol{p}_n$ learned by historical hidden states:

$$\begin{aligned} &\boldsymbol{h}_n = \mathrm{GRUCell}(\boldsymbol{c}_n, \boldsymbol{h}_{n-1}) = \hat{\boldsymbol{h}}_n + \boldsymbol{p}_n \\ &\hat{\boldsymbol{h}}_n = (1-z_t)\tilde{\boldsymbol{h}}_n + \prod_{i=1}^{n} z_i \boldsymbol{h}_0, \quad \boldsymbol{p}_n = \sum_{k=1}^{n-2}\prod_{i=k+1}^{n} z_i(1-z_k)\boldsymbol{h}_k \end{aligned} \tag{6}$$

where z_k is the reset gate at the k-th time step. It is noteworthy that each positional representation $\boldsymbol{p}_n$ is generated by aggregated historical hidden states with varied update intensities. Then we

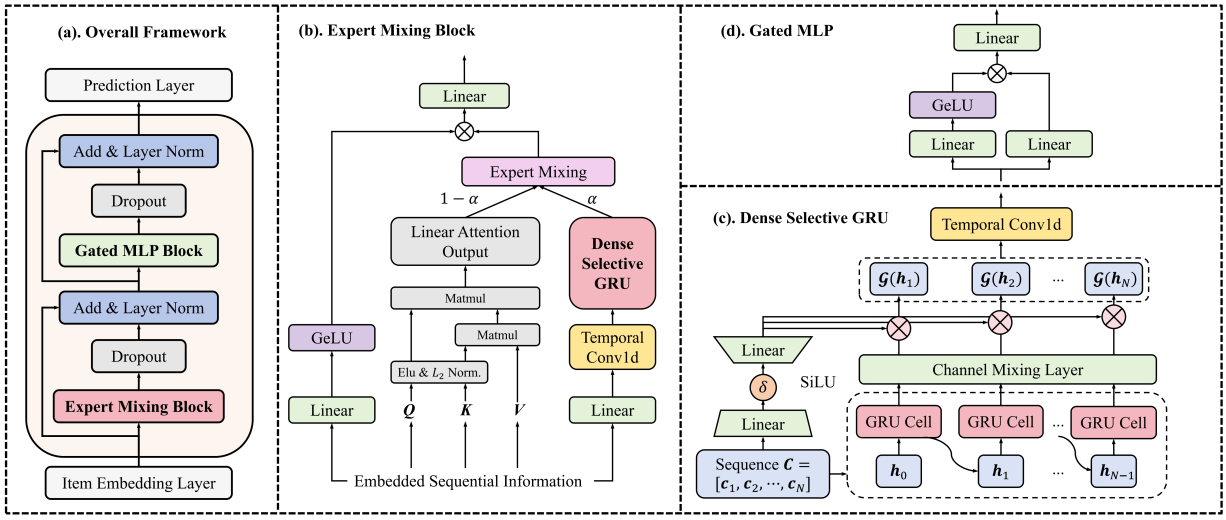

Figure 1: (a). Framework of proposed GLINT-RU. (b). Expert mixing block employs paralleled attention and GRU to effectively learn semantic features and fine-grained positional information. (c). Dense Selective GRU as the core part of the framework, deeply selects and aggregates the hidden states. (d). Gated MLP block is utilized to deeply filter the feed forward network.

project the hidden states into a latent space using the channel crossing layer, which can be written as:

$$\Phi(\boldsymbol{H}) = \boldsymbol{H}\boldsymbol{W}_H + \boldsymbol{b}_H, \tag{7}$$

where $\boldsymbol{H} = [\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_N]^{\mathrm{T}}$ is the output of GRU, $\boldsymbol{W}_H$ is the learnable weight matrix and b_H are bias. Although each input state $\boldsymbol{c}_t$ incorporates information from both preceding and subsequent items, each output hidden state is still determined by the hidden state of the preceding time step. Therefore, to capture the context information of the output sequential hidden states, we implement a temporal convolution on the crossed features. This convolution layer extracts local temporal features to understand user behavior dynamics, and enhance the predictive accuracy of our model:

$$Y = \mathrm{TemporalConv1d}(\mathcal{G}) \tag{8}$$

where $\mathcal{G}$ is the Selective Gate function, and $Y = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N]^{\mathrm{T}}$ is the output matrix from the dense selective GRU module. The two convolution layers together with GRU cells improve the density of the sequential information, enabling each hidden state to be learned from behaviors of more input time steps.

3.3.2 ***Selective Gate.*** To filter the hidden information of the GRU module, we design a selective gate where outputs of the feature crossing layer are selected based on the input state of the GRU. The selective gate weights are generated by two tiny linear layers with SiLU activation function [5], and we use them to select useful hidden states and filter information:

$$\begin{aligned} \boldsymbol{\delta}_1(\boldsymbol{C}) &= \boldsymbol{C}\boldsymbol{W}_\delta^{(1)} + b_\delta^{(1)}, \\ \mathcal{G}(\boldsymbol{H}) = \Omega(\boldsymbol{\delta}_1(\boldsymbol{C}), \boldsymbol{H}) &= (\mathrm{SiLU}(\boldsymbol{\delta}_1(\boldsymbol{C}))\boldsymbol{W}_\Omega^{(1)} + \boldsymbol{b}_\Omega^{(1)}) \otimes \Phi(\boldsymbol{H}), \end{aligned} \tag{9}$$

where $\boldsymbol{C} = [\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_N]^{\mathrm{T}}$ also serves as the input of the GRU module, $\boldsymbol{W}_\delta^{(1)}$, $\boldsymbol{W}_\Omega^{(1)}$ are weight matrices, $\boldsymbol{b}_\delta^{(1)}, \boldsymbol{b}_\Omega^{(1)}$ are bias. With this gate mechanism, our GRU-based model could become more flexible with the ability to perceive the data environment.

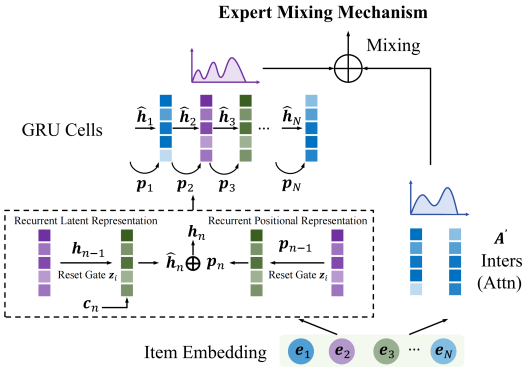

Figure 2: General process of expert mixing mechanism. GRU captures long-term dependencies with recurrent latent and fine-grained positional representations. Attention layer learns semantic features fro important item interactions.

3.4 Expert Mixing Block

Existing efficient SRSs lack the ability to exploit semantic features, fine-grained positional information, and dependencies simultaneously. As is shown in Figure 1.(c), by introducing the linear attention mechanism and mixing these two powerful experts, the computational complexity will be substantially reduced compared with transformer-based models, and more high-quality item representations can be generated. Moreover, the two employed experts are parallel in our framework, which further improves the model's efficiency. The expert mixing mechanism is shown in Figure 2.

In real applications, the conditions of the data vary a lot. The GRU is naturally suited for sequential data, demonstrating effectiveness in sequential recommendation tasks that exhibit strong temporal dependencies, while attention focuses on relevant items in the sequence dynamically. To adapt to complex data conditions, we

attribute appropriate weights to the two experts by a mixing gate:

$$\begin{aligned} \boldsymbol{M} &= \alpha_1^{(t)} \boldsymbol{A}'(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) + \alpha_2^{(t)} \boldsymbol{Y}, \\ \alpha_i^{(t)} &= \text{softmax}(\alpha_i^{(t-1)}) = \frac{\alpha_i^{(t-1)}}{e^{\alpha_1^{(t-1)}} + e^{\alpha_2^{(t-1)}}}, \quad i = 1, 2, \end{aligned} \tag{10}$$

where $\alpha_1^{(t)}, \alpha_2^{(t)}$ are trainable mixing parameters at t-th training iteration. Then we filter this output by introducing another data-aware gate which selects the outputs based on the original input data batch:

$$\begin{aligned} \boldsymbol{\delta}_2(\boldsymbol{X}) &= \boldsymbol{X}\boldsymbol{W}_\delta^{(2)} + b_\delta^{(2)}, \\ \boldsymbol{Z} = \Omega_2(\boldsymbol{\delta}_2(\boldsymbol{X}), \boldsymbol{M}) &= (\text{GeLU}(\boldsymbol{\delta}_2(\boldsymbol{X}))) \otimes \boldsymbol{M}, \end{aligned} \tag{11}$$

where $\boldsymbol{X}$ is the input of expert mixing block, $\boldsymbol{W}_\delta^{(2)}$ is the weight matrix of linear layer, $\boldsymbol{b}_\delta^{(2)}$ are bias.

3.5 Gated MLP Block

Most existing efficient SRSs, utilize two-layer feed forward networks to capture the nonlinear relationships among features before giving predictions. To further enhance the performance of the model and augment useful features from the expert mixing block, we introduce the gated MLP block shown in Figure 1.(d), which employs a gate mechanism again to deeply filter the information and generate item representations for predictions.

$$\begin{aligned} \boldsymbol{\delta}_3(\boldsymbol{Z}) &= \boldsymbol{Z}\boldsymbol{W}_\delta^{(3)} + b_\delta^{(3)}, \\ \boldsymbol{P} = \Omega_2(\boldsymbol{\delta}_3(\boldsymbol{Z}), \boldsymbol{Z}) &= (\text{GeLU}(\boldsymbol{\delta}_3(\boldsymbol{Z}))) \otimes (\boldsymbol{Z}\boldsymbol{W} + \boldsymbol{b}), \\ \boldsymbol{R} &= \boldsymbol{P}\boldsymbol{W}_o + \boldsymbol{b}_o \end{aligned} \tag{12}$$

where $\boldsymbol{Z}$ is the output of expert mixing block, $\boldsymbol{P}$ denotes the output of gated linear layer, $\boldsymbol{R}$ represents the item representation, and $\boldsymbol{W}_\delta^{(3)}, \boldsymbol{W}_o, \boldsymbol{W}$ are weight matrices, $\boldsymbol{b}_\delta^{(3)}, \boldsymbol{b}_o, \boldsymbol{b}$ are bias. The recommendation scores are generated by item representations and embeddings, followed by the prediction score $\hat{y}_i$ of the i-th item:

$$\hat{y}_i = \text{softmax}(\boldsymbol{R}_i(\boldsymbol{e}_i)^{\mathrm{T}}), \tag{13}$$

where $\boldsymbol{R}_i$ is the representation of i-th item. Loss function, model training methods and the algorithm are displayed in **Appendix A**.

3.6 Complexity Analysis

In this subsection, we will explain why GLINT-RU has inherent superiority over other popular SRS models in model efficiency.

Given that the sequence length is N, the embedding size is d and the kernel size for GLINT-RU is k, the time complexity of GLINT-RU is $O((2k + 12)Nd^2)$. The complexity is calculated throughout the network, from the embedding layer to the prediction layer.

Discussion. Our GLINT-RU shows significantly low and linear time complexity, as GLINT-RU is a highly paralleled mixed network with only one layer to achieve high performance. Our framework utilizes paralleled expert networks and employs an efficient GRU module to capture long-term dependencies. GLINT-RU is more efficient than other models in the following aspects:

- **GLINT-RU v.s. Transformer-based SRSs**: Firstly, traditional transformer-based model [12] suffers from large computational complexity, especially when the sequence length N is large. Linear attention mechanism [18] can be applied to substantially reduce the computational cost. However, they still require stacked transformer structures to achieve high performance, while GLINT-RU achieves outstanding performance with only one layer.
- **GLINT-RU v.s. MLP-based SRSs**: Secondly, The inference speed of the MLP-based models [7] may decrease when faced with long sequence length, as the sequence mixing layer has quadratic time complexity. In contrast, the paralleled expert module of GLINT-RU is more suitable for processing long sequential data.
- **GLINT-RU v.s. SSM-based SRSs**: Thirdly, state-space models [9] may require complex matrix operations and recursive calculations, which may be difficult to parallelize efficiently in practical calculations. In contrast, GLINT-RU's paralleled expert module has a simpler update mechanism and higher efficiency.

4 Experiments

In this section, we conduct extensive experiments to show the effectiveness and efficiency of our GLINT-RU Framework. After we introduce our implementation details, the experiment results will be analyzed in detail. The experiments in this section are set to answer the following research questions:

- **RQ1:** How does GLINT-RU framework perform compared with other state-of-the-art SRS baseline models?
- **RQ2:** To what extent does GLINT-RU improve model efficiency compared with other state-of-the-art SRS frameworks?
- **RQ3:** How do the dense selective GRU, the linear attention mechanism, and the gated MLP contribute to GLINT-RU?
- **RQ4:** How does the hyperparameter setting affect the performance of GLINT-RU?

4.1 Datasets and Evaluation Metrics

We evaluate GLINT-RU based on three benchmark datasets ML-1M [2], Amazon-Beauty and Amazon video Games [3]. Below is the basic introduction of MovieLens-1M, Amazon Beau-ty, and Amazon Video Games datasets.

- **MovieLens 1M:** comprises user ratings of movies. It includes about 1 million anonymous ratings from users who joined MovieLens. The dataset provides information about user IDs, movie IDs, ratings, and timestamps. It is widely used for research in collaborative filtering and recommendation systems.
- **Amazon Beauty:** is a subset of the Amazon product data, focusing specifically on beauty products. It includes user reviews and ratings of various beauty products available on Amazon. The dataset contains metadata such as product descriptions, categories, prices, and brand information. It is valuable for research in sentiment analysis, product recommendations, and consumer behavior analysis within the beauty industry.
- **Amazon Video Games:** is a subset of the Amazon product data, focusing specifically on video game products. It includes user reviews and ratings of various video game products available on Amazon. The dataset contains metadata such as product descriptions, categories, price, and platform information. It is also valuable for research in sentiment analysis, product recommendation, and consumer behavior analysis in the video games domain.

[2] https://grouplens.org/datasets/movielens/
[3] https://cseweb.ucsd.edu/ jmcauley/datasets.html#amazon_reviews

Table 1: Statistical Information of Adopted Datasets.

Datasets	# Users	# Items	# Interactions	Avg.Length	Sparsity
ML-1M	6,041	3,707	1,000,209	165.60	95.53%
Beauty	22,364	12,102	198,502	8.88	99.93%
Video Games	24,304	10,673	231,780	9.54	99.91%

Statistical information of the three datasets is shown in Table 1, where all the three datasets are sparse. The two Amazon datasets have relatively small data volumes, about 200,000, but they have a large number of users and items, and the average length of interaction sequences is short. In contrast, ML-1M has a larger data volume, reaching 1 million, but the number of users and items is small, with much longer interaction sequence lengths.

We adopt Recall, Mean Reciprocal Rank (MRR), and Normalized Discounted Cumulative Gain (NDCG) as the evaluation metrics for our experiments. The interactions are grouped by users chronologically, and the datasets are split by the leave-one-out strategy [18]. To be more specific, the penultimate item of the interaction sequence is used for validation. Therefore, the size of validation set is determined by the number of users.

4.2 Baselines

In this paper, we compare our GLINT-RU with two types of baseline models, i.e., traditional SRS models and efficient SRS models. The traditional SRS model includes various SRS benchmarks, while the efficient SRS models improve existing model's structure and computational methods, thus significantly enhancing the model efficiency. Adopted baselines are listed as follows:

Traditonal SRS models

(1) **GRU4Rec** [10]: utilizes GRUs to capture sequential dependencies within user interactions for session-based recommendations.
(2) **BERT4Rec** [26]: adapts the Bidirectional Encoder Representations from Transformers (BERT) architecture to model user behaviors for personalized recommendation.
(3) **SASRec** [12]: captures long-term and short-term user preferences by applying a multi-head attention mechanism.

Efficient SRS Models

(1) **LinRec** [18]: reduces the computational costs substantially by changing the dot product of attention mechanism in the transformer-based models. We select SASRec as the backbone of LinRec.
(2) **SMLP4Rec** [7] uses a tri-directional fusion scheme to learn correlations on sequence, channel, and feature dimensions efficiently.
(3) **LightSAN** [6] projects the initial interactions into representations with shorter lengths, which is also an efficient approach for transformer-based models.
(4) **Mamba4Rec** [17]: explores the potential of selective SSMs for efficient sequential recommendation, which substantially improve the efficiency of SRS models.

4.3 Implementation

In this subsection, we introduce the implementation details of the GLINT-RU. We use Adam optimizer [13] with the learning rate of 0.001 for our training process. Both the train and evaluation batch size are set as 2048. The hidden size is set as 128 for ML-1M and 64 for Amazon Beauty and Video Games. As is shown in Table 1, the average length of ML-1M, Beauty, and Video Games are 165, 8.88, and 9.54, so we set the maximum sequence length as 200 for ML-1M, and 100 for the two amazon datasets. We adopt the dropout rate of 0.5 for Amazon datasets considering their high level of sparsity, compared with 0.2 for ML-1M. We construct the attention-based baselines with 2 transformer layers so that they can achieve high performance. Other implementation details follow the settings of original papers [7, 17, 18]. Please find the complete hyperparameter settings and more implementation details in **Appendix B**.

4.4 Overall Performance (RQ1)

In this subsection, we compare the performance of GLINT-RU with both traditional recommendation frameworks and state-of-the-art efficient models. The results, as shown in Table 2, demonstrate the effectiveness of GLINT-RU on metrics Recall@10, MRR@10, and NDCG@10 in bold. According to the above table, it is evident that GLINT-RU defeats all the selected transformer-, RNN-, MLP-, and SSM-based baselines. We improve the performance upper bound of efficient recommendation models by $0.34\% \sim 3.74\%$.

Traditional RNN-based models, like GRU4Rec, might have difficulty in dealing with complex user behaviors. Although it can capture the long-term dependencies from long sequences, it struggles to learn effectively from extremely sparse datasets. Traditional attention-based methods like BERT4Rec and SASRec have great performance on the three datasets, but it is quite inefficient due to the high computational complexity of the attention mechanism.

Efficient model LinRec changes the softmax operation, and takes attention scores from more positions into consideration, improving the performance on long-term sequential recommendation tasks compared with its backbone SASRec. SMLP4Rec and Mamba4Rec achieve impressive performance on the three datasets. However, Mamba4Rec shows enhanced proficiency in modeling long sequences (ML-1M) but exhibits low performance in relatively short sequences (Beauty and Video Games). Conversely, the SMLP4Rec shows superior performance in tasks with short sequences while being less effective with longer sequences. Additionally, SMLP4Rec requires features from the items to enhance its performance. Uniquely, GLINT-RU integrates the advantages of a linear attention mechanism and dense selective GRU module, adaptively extracting dependencies from recurrent latent item representations, fine-grained positional representations, and important semantic features. The gate mechanism employed in GLINT-RU substantially enhances its ability to filter the information based on the dynamic data environment and mix the experts according to the data adaptively.

In summary, GLINT-RU as a novel efficient framework, shows its superiority over state-of-the-art baselines. This underscores the potential of dense selective GRU and models with hybrid modules as more powerful tools for recommendation tasks.

4.5 Efficiency Comparison (RQ2)

In this subsection, we analyze the efficiency of GLINT-RU and state-of-the-art sequential recommendation models. We evaluate the model efficiency according to the inference time of each mini-batch, training time, and GPU memory occupation.

The results, shown in Table 3, provide several valuable insights: Firstly, by utilizing the efficient Dense GRU module and linear attention module, GLINT-RU dramatically reduces the training

Table 2: Overall performance comparison between GLINT-RU and baselines.

Models	ML-1M			Amazon Beauty			Amazon Video Games		
	Recall@10	MRR@10	NDCG@10	Recall@10	MRR@10	NDCG@10	Recall@10	MRR@10	NDCG@10
GRU4Rec	0.6954	0.4055	0.4748	0.3851	0.1891	0.2351	0.6028	0.2929	0.3660
BERT4Rec	0.7119	0.4041	0.4776	0.3478	0.1584	0.2027	0.5490	0.2541	0.2916
SASRec	0.7205	0.4251	0.4958	0.4332	0.2325	0.2798	0.6459	0.3404	0.4128
LinRec	0.7184	0.4316	0.5002	0.4270	0.2314	0.2775	0.6384	0.3355	0.4073
LightSANs	0.7195	0.4314	0.5003	0.4406	0.2358	0.2840	0.6488	0.3415	0.4142
SMLP4Rec	0.6753	0.3870	0.4558	0.4457	0.2408	0.2891	0.6480	0.3484	0.4195
Mamba4Rec	0.7238	0.4368	0.5054	0.4233	0.2213	0.2689	0.6488	0.3389	0.4123
GLINT-RU	**0.7379***	**0.4517***	**0.5202***	**0.4472***	**0.2498***	**0.2964***	**0.6573***	**0.3549***	**0.4266***
Improv.	1.95%	3.30%	2.93%	0.34%	3.74%	2.53%	1.31%	1.87%	1.69%

Recommendation performance of GLINT-RU and existing state-of-the-art benchmark SRS models have been shown. " $*$ " indicates the improvements are **statistically significant** (i.e., two-sided t-test with $p < 0.05$) over baselines). The best results are bolded, and the second-best are underlined.

Table 3: Efficiency comparison.

Datasets	Model	Infer.	Training	GPU Memory
ML-1M	BERT4Rec	88ms	285s/epoch	21.51GB
	SASRec	55ms	172s/epoch	21.51GB
	LinRec	37ms	101s/epoch	11.67GB
	LightSANs	43ms	130s/epoch	16.99GB
	SMLP4Rec	51ms	151s/epoch	16.13GB
	Mamba4Rec	41ms	108s/epoch	**7.72G**
	GLINT-RU	**31ms**	**86s/epoch**	8.81G
	Improv.	16.22%	17.44%	-
Beauty	BERT4Rec	1372ms	13s/epoch	11.69GB
	SASRec	444ms	8.1s/epoch	7.67GB
	LinRec	340ms	5.6s/epoch	4.14GB
	LightSANs	427ms	7.2s/epoch	4.57GB
	SMLP4Rec	361ms	5.3s/epoch	2.95GB
	Mamba4Rec	351ms	4.5s/epoch	**2.32G**
	GLINT-RU	**278ms**	**3.8s/epoch**	2.62G
	Improv.	18.24%	15.56%	-
Video Games	BERT4Rec	1290ms	15s/epoch	10.98GB
	SASRec	406ms	9.6s/epoch	7.65GB
	LinRec	327ms	6.6s/epoch	4.13GB
	LightSANs	369ms	8.3s/epoch	4.55GB
	SMLP4Rec	389ms	9.1s/epoch	3.38GB
	Mamba4Rec	309ms	5.6s/epoch	**2.28G**
	GLINT-RU	**247ms**	**4.5s/epoch**	2.49G
	Improv.	20.06%	19.64%	-

Inference time of each mini-batch (batch size = 2048), training time and GPU memory of GLINT-RU and other baseline models. The best results are bolded, and the second best results are underlined.

time and inference time, improving the training inference time by 15% ∼ 20% compared with most efficient recommendation baseline models. In addition, due to the low computational cost of GRU and linear attention mechanism, GLINT-RU exhibits minimal GPU memory consumption, which is comparable to the state-of-the-art SSM-based efficient model Mamba4Rec. In Section 3.6, we demonstrate that the GLINT-RU exhibits low theoretical computational complexity, as we employ the parallel networks and efficient GRU as core components of our recommender system. The theoretical analysis has been verified by the results in Table 3.

Table 4: Ablation study for Components of GLINT-RU.

Model Components	Recall@10	MRR@10	NDCG@10
Default	**0.7379***	**0.4517***	**0.5202***
w/o Gated MLP (Light GLINT-RU)	0.7260	0.4369	0.5060
w/o Attention	0.7195	0.4312	0.5001
w/o GRU	0.6762	0.3913	0.4593
w/o Temporal Conv1d	0.7232	0.4322	0.5019

" $*$ " indicates the improvements are **statistically significant** (i.e., two-sided t-test with $p < 0.05$) over baselines)

Traditional transformer-based recommendation models like SASRec and BERT4Rec suffer from extended inference time and high GPU memory occupation. When processing long sequential data, the conventional attention mechanism falls behind novel efficient models due to its high computational cost. Among all the baseline models, the SSM-based Mamba4Rec framework exhibits impressive efficiency, but Mamba requires complex mathematical computation, which slows down its inference and training speed. Additionally, LinRec suffers from the inherent shortage of its backbone SASRec that requires stacked transformer layers to enhance the model performance. Such large transformer structures extend both the inference and training time. Although SMLP4Rec achieves high performance in the model accuracy, it struggles to train and inference efficiently, especially when processing long-term sequential data.

4.6 Ablation Study (RQ3)

In the ablation study, we remove the gated MLP block, linear attention expert, dense selective GRU expert, and the two temporal convolution layers individually to verify the efficacy of each component. We conduct the ablation study on the ML-1M dataset, and the results are outlined in Table 4, providing insightful observations.

The results verify the essential role of the dense selective GRU module, as the performance of the model will dramatically decrease

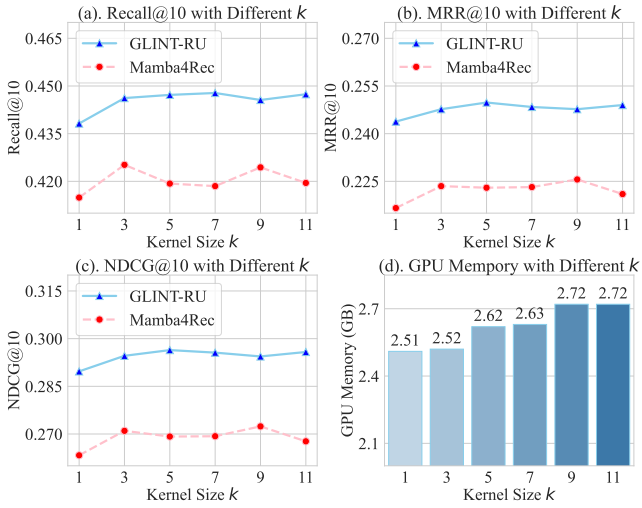

Figure 3: Impacts of kernel size k of the temporal convolution on the performance of GLINT-RU and Mamba4Rec and the GPU Memory occupation of GLINT-RU.

without the GRU module. This reveals the insights that the gated GRU module effectively captures the dependencies of interactions with fine-grained positional representations. The linear attention mechanism understands the interactions of relevant items in the sequence. As is shown in Table 4, it improves the performance of GLINT-RU to some extent. Adding a temporal convolution layer incorporates context information from adjacent items, resulting in an enhancement in model performance. In addition, the gated MLP block plays a similar role as the feed-forward network in our framework, which filters complex information from the expert mixing block. It is noteworthy that even without the gated MLP block our framework still outperforms all the state-of-the art efficient models, demonstrating its inherent remarkable superiority for sequential recommendation tasks. After we remove the gated MLP, the GPU memory occupation of GLINT-RU becomes **7.63GB**, less than Mamba4Rec shown in Table 3, and the inference time will be reduced to **241ms**. We name this framework "Light GLINT-RU", which is more applicable to resource-constrained scenarios. We further discuss the ablation study on activation functions in **Appendix C**, where we highlight their impact on performance. Additionally, we provide a detailed analysis of the ablation study conducted on the Amazon Beauty and Amazon Video Games datasets in **Appendix E**, emphasizing the contributions of each component to the overall performance of GLINT-RU.

4.7 Parameter Analysis

We conduct parameter analysis on the dataset Amazon Beauty. We will first analyze the impact of the crucial hyperparameter kernel size k in GLINT-RU, and then we will analyze the model performance as the hidden size d and number of GLINT-RU layers L changes. The discussion on these parameters provides valuable insights into the superiority of GLINT-RU.

Kernel size k. The impacts of the parameter k on the model performance are shown in Figure 3 and 4. Figure 3.(a)-(c) displays the model performance of GLINT-RU with different kernel sizes.

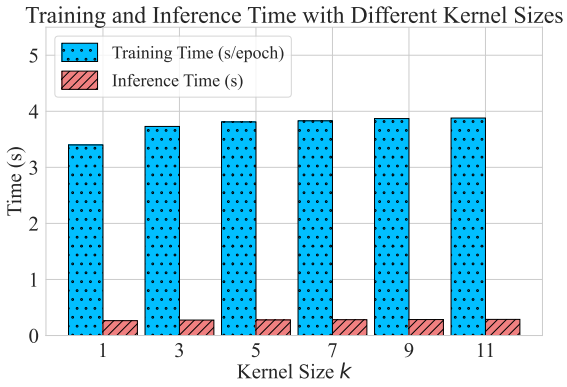

Figure 4: Impacts of kernel size k of the temporal convolution on the training and inference time.

Our model exhibits stable and high performance, providing a wide range of kernel sizes. This finding indicates the robustness of the GLINT-RU framework. This enhancement in performance can be attributed to the fact that a larger kernel size aggregates information from more items, thereby learning a more extensive context into the hidden state. However, as the kernel size continues to expand, dense selective GRU might incorporate irrelevant data into the output state, which might lead to a marginal decline in the accuracy. Mamba4Rec as a novel efficient SSM-based model, also employs a temporal convolution layer in the model structure. As the kernel size changes, the performance of Mamba4Rec becomes quite unstable compared with our GLINT-RU. Our GLINT-RU consistently outperforms Mamba4Rec across all kernel sizes, further demonstrating the superiority of this novel GLINT-RU framework.

As can be seen in the hist plots in Figure 3.(d) and Figure 4, increasing the kernel size has slight impacts on the training/inference time of each mini-batch and the GPU memory occupation, which further verifies the efficiency and stability of our model.

Hidden size d**.** We change the model size by using different hidden sizes d and compare the performance of GLINT-RU with state-of-the-art baselines. The results shown in Figure 5.(a)-(c) indicate that GLINT-RU achieves state-of-the-art performance at small hidden sizes. When d is set as 64, the predictive efficacy of the GLINT-RU substantially surpasses the upper bounds of performance achieved by other baseline models. Achieving excellent results with a relatively small model dimension illustrates the superior expressiveness and significant advantages of the GLINT-RU. SMLP4Rec requires additional features to enhance the model accuracy, and its inference speed is significantly affected by the variation in the hidden size, demonstrating relatively low efficiency. The prediction accuracy of Mamba4Rec is inferior and decreases dramatically as the hidden size d gets larger, indicating that it struggles to learn effective information from short behavior sequences. Attention-based models SASRec, LightSAN, and LinRec show more stable results, but they can only capture item interactions to predict the ratings and require stacked transformers to achieve relatively high performance, which has negative impacts on model efficiency in Figure 5.(d).

Number of Layers L**.** We increase the number of GLINT-RU layers from 1 to 4 and observe the model performance and efficiency. The results and the experimental details are illustrated in **Appendix D**.

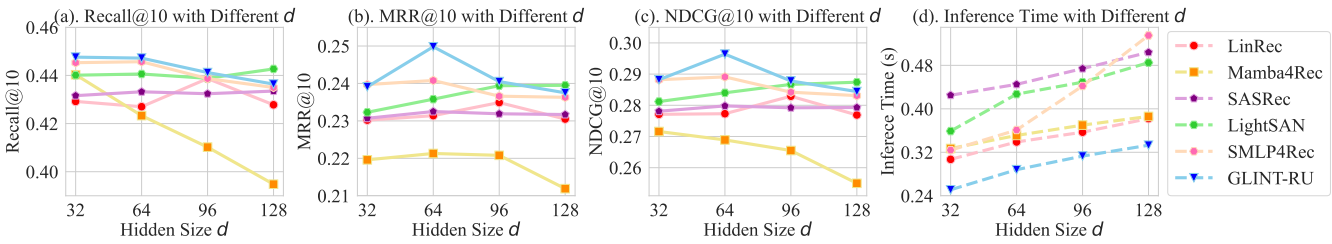

Figure 5: Impacts of hidden size d on the performance of GLINT-RU and state-of-the-art baselines.

The results indicate that our single-layer GLINT-RU can achieve high model efficiency and accuracy simultaneously.

In summary, GLINT-RU effectively combines the long/short-term dependencies with effective positional representations and important interactions which enables it to be an accurate and efficient SRS under a broad range of parameter choices.

5 Related Works

Traditional Sequential Recommendation Models Transformers and RNNs are widely recognized as popular and effective frameworks for sequential recommendation tasks. Numerous studies have been conducted to explore the potential of transformers in enhancing recommendation performance. The transformer-based SASRec [12] predicts user preferences by leveraging multi-head attention to model both long- and short-term interactions. SSE-PT [30] further improves recommendation accuracy by incorporating user embeddings into the transformer architecture. Similarly, MB-STR [32], a transformer-based variant, captures diverse user behavior dynamics and effectively mitigates the challenges posed by data sparsity. On the other hand, RNN-based methods like GRU4Rec [10] model dependencies among items through gated recurrent units, enabling them to learn sequential patterns. HRNN [23], an advanced RNN-based framework, integrates an additional GRU layer to extract session-level information and dynamically track user preferences over time. In comparison to these models, GLINT-RU demonstrates superior capability in capturing high-quality semantic features and fine-grained positional representations, leading to improved model accuracy and performance.

Efficient Recommendation Models To tackle the high computational complexity of existing SRSs, and accelerate the inference speed for real-world applications, researchers endeavor to invent increasingly efficient models. AutoSeqRec [19] is constructed based on Autoencoder, which is an innovative work that captures long-term preferences through collaborative filtering. To further improve the model efficiency, DMAN [27] combines the long-term attention net and recurrent attention net to memorize users' interests dynamically and support efficient inference. LinRec [18] cuts down the computational cost of transformer-based backbones by changing the dot product in the attention mechanism. SSM-based models like Mamba4Rec [17] and RecMamba [31] utilize selective state space models to achieve high performances and high efficiency, becoming emerging powerful tools for sequential recommendation tasks. FMLP-Rec [39] with learnable filters and SMLP4Rec [7] with diverse mixers are representative pure MLP-based efficient SRSs. LRURec [33] is constructed based on linear recurrent units and achieves fast inference by employing recursive parallelization. These models often require stacked network structures to achieve better results, and their performance is not stable enough when faced with different sequence lengths. However, GLINT-RU only needs one layer to achieve stable high performance.

6 Conclusion

In this paper, we have presented an innovative dense selective GRU framework, GLINT-RU for sequential recommendation tasks. Due to the paralleled network design and implementation of efficient dense selective GRU with linear complexity, the computational cost can be substantially reduced, resulting in state-of-the-art inference speed. Additionally, our GLINT-RU models improve the quality of semantic features and fine-grained positional information for recommendation tasks. Gate mechanisms are widely applied in GLINT-RU, deeply filtering and selecting information. It uses a dense selective GRU that aggregates information from adjacent items and generates high-quality latent item representations based on dependencies with fine-grained positional information to learn context information. By mixing dense selective GRU with linear attention, GLINT-RU can capture important interactions and item dependencies simultaneously. Our extensive experiments demonstrate that GLINT-RU achieves outstanding performance, not only improving model accuracy but also accelerating training and inference speed dramatically. These results underscore GLINT-RU's potential to become a novel, stable, and efficient framework in various scenarios. As an efficient recommender system that defeats state-of-the-art models, we believe our novel framework will become a valuable foundation.

Acknoledgement

This research was partially supported by 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), Collaborative Research Fund (No.C1043-24GF), 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

[1] Jianxin Chang, Chenbin Zhang, Yiqun Hui, Dewei Leng, Yanan Niu, Yang Song, and Kun Gai. 2023. Pepnet: Parameter and embedding personalized network for infusing with personalized prior information. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3795–3804.

[2] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[3] Soham De, Samuel L Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, et al. 2024. Griffin: Mixing Gated Linear Recurrences with Local Attention for Efficient Language Models. *arXiv preprint arXiv:2402.19427* (2024).

[4] Rahul Dey and Fathi M Salem. 2017. Gate-variants of gated recurrent unit (GRU) neural networks. In *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*. IEEE, 1597–1600.

[5] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. 2018. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks* 107 (2018), 3–11.

[6] Xinyan Fan, Zheng Liu, Jianxun Lian, Wayne Xin Zhao, Xing Xie, and Ji-Rong Wen. 2021. Lighter and better: low-rank decomposed self-attention networks for next-item recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*. 1733–1737.

[7] Jingtong Gao, Xiangyu Zhao, Muyang Li, Minghao Zhao, Runze Wu, Ruocheng Guo, Yiding Liu, and Dawei Yin. 2024. SMLP4Rec: An Efficient all-MLP Architecture for Sequential Recommendations. *ACM Transactions on Information Systems* 42, 3 (2024), 1–23.

[8] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 249–256.

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

[10] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[11] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation* 3, 1 (1991), 79–87.

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

[13] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[14] Chengxi Li, Yejing Wang, Qidong Liu, Xiangyu Zhao, Wanyu Wang, Yiqi Wang, Lixin Zou, Wenqi Fan, and Qing Li. 2023. STRec: Sparse Transformer for Sequential Recommendations. In *Proceedings of the 17th ACM Conference on Recommender Systems*. 101–111.

[15] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1419–1428.

[16] Muyang Li, Xiangyu Zhao, Chuan Lyu, Minghao Zhao, Runze Wu, and Ruocheng Guo. 2022. MLP4Rec: A pure MLP architecture for sequential recommendations. *arXiv preprint arXiv:2204.11510* (2022).

[17] Chengkai Liu, Jianghao Lin, Jianling Wang, Hanzhou Liu, and James Caverlee. 2024. Mamba4Rec: Towards Efficient Sequential Recommendation with Selective State Space Models. *arXiv preprint arXiv:2403.03900* (2024).

[18] Langming Liu, Liu Cai, Chi Zhang, Xiangyu Zhao, Jingtong Gao, Wanyu Wang, Yifu Lv, Wenqi Fan, Yiqi Wang, Ming He, et al. 2023. 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.

[19] Sijia Liu, Jiahao Liu, Hansu Gu, Dongsheng Li, Tun Lu, Peng Zhang, and Ning Gu. 2023. Autoseqrec: Autoencoder for efficient sequential recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 1493–1502.

[20] Ziwei Liu, Qidong Liu, Yejing Wang, Wanyu Wang, Pengyue Jia, Maolin Wang, Zitao Liu, Yi Chang, and Xiangyu Zhao. 2024. Bidirectional gated mamba for sequential recommendation. *arXiv preprint arXiv:2408.11451* (2024).

[21] Chao Long, Huanhuan Yuan, Junhua Fang, Xuefeng Xian, Guanfeng Liu, Victor S Sheng, and Pengpeng Zhao. 2024. Learning Global and Multi-granularity Local Representation with MLP for Sequential Recommendation. *ACM Transactions on Knowledge Discovery from Data* 18, 4 (2024), 1–15.

[22] Saeed Masoudnia and Reza Ebrahimpour. 2014. Mixture of experts: a literature survey. *Artificial Intelligence Review* 42 (2014), 275–293.

[23] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *proceedings of the Eleventh ACM Conference on Recommender Systems*. 130–137.

[24] Guizhu Shen, Qingping Tan, Haoyu Zhang, Ping Zeng, and Jianjun Xu. 2018. Deep learning with gated recurrent unit networks for financial sequence predictions. *Procedia computer science* 131 (2018), 895–903.

[25] Zhuoran Shen, Mingyuan Zhang, Haiyu Zhao, Shuai Yi, and Hongsheng Li. 2021. Efficient attention: Attention with linear complexities. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 3531–3539.

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

[27] Qiaoyu Tan, Jianwei Zhang, Ninghao Liu, Xiao Huang, Hongxia Yang, Jingren Zhou, and Xia Hu. 2021. Dynamic memory based attention network for sequential recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 4384–4392.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* 30 (2017).

[29] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768* (2020).

[30] Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. 2020. SSE-PT: Sequential recommendation via personalized transformer. In *Proceedings of the 14th ACM conference on recommender systems*. 328–337.

[31] Jiyuan Yang, Yuanzi Li, Jingyu Zhao, Hanbing Wang, Muyang Ma, Jun Ma, Zhaochun Ren, Mengqi Zhang, Xin Xin, Zhumin Chen, et al. 2024. Uncovering Selective State Space Model's Capabilities in Lifelong Sequential Recommendation. *arXiv preprint arXiv:2403.16371* (2024).

[32] Enming Yuan, Wei Guo, Zhicheng He, Huifeng Guo, Chengkai Liu, and Ruiming Tang. 2022. Multi-behavior sequential transformer recommender. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*. 1642–1652.

[33] Zhenrui Yue, Yueqi Wang, Zhankui He, Huimin Zeng, Julian McAuley, and Dong Wang. 2024. Linear recurrent units for sequential recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*. 930–938.

[34] Sheng Zhang, Maolin Wang, Yao Zhao, Chenyi Zhuang, Jinjie Gu, Ruocheng Guo, Xiangyu Zhao, Zijian Zhang, and Hongzhi Yin. 2024. EASRec: Elastic Architecture Search for Efficient Long-term Sequential Recommender Systems. *arXiv preprint arXiv:2402.00390* (2024).

[35] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM computing surveys (CSUR)* 52, 1 (2019), 1–38.

[36] Kesen Zhao, Lixin Zou, Xiangyu Zhao, Maolin Wang, and Dawei Yin. 2023. User retention-oriented recommendation with decision transformer. In *Proceedings of the ACM Web Conference 2023*. 1141–1149.

[37] Xiangyu Zhao, Maolin Wang, Xinjian Zhao, Jiansheng Li, Shucheng Zhou, Dawei Yin, Qing Li, Jiliang Tang, and Ruocheng Guo. 2023. Embedding in recommender systems: A survey. *arXiv preprint arXiv:2310.18608* (2023).

[38] Xiangyu Zhao, Maolin Wang, Xinjian Zhao, Jiansheng Li, Shucheng Zhou, Dawei Yin, Qing Li, Jiliang Tang, and Ruocheng Guo. 2023. Embedding in Recommender Systems: A Survey. *arXiv preprint arXiv:2310.18608* (2023).

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

[40] You Zhou, Xiujing Lin, Xiang Zhang, Maolin Wang, Gangwei Jiang, Huakang Lu, Yupeng Wu, Kai Zhang, Zhe Yang, Kehang Wang, et al. 2023. On the opportunities of green computing: A survey. *arXiv preprint arXiv:2311.00447* (2023).

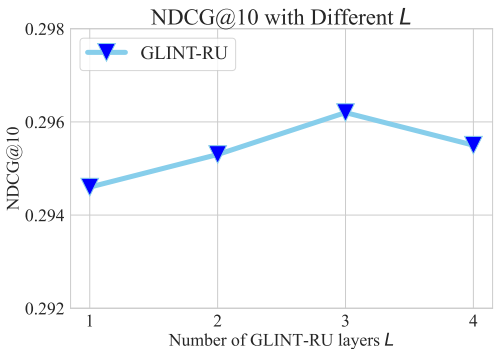

Figure 6: Impacts of L on the model performance.

Algorithm 1 Optimization Algorithm of GLINT-RU

Input: User-item interactions $\mathcal{S} = [s_1, \cdots, s_{|\mathcal{U}|}]$, corresponding ground-truth labels y
Output: Well-trained weights $\mathcal{W}^*$ of model f

```
Randomly initialize the model parameters W*.
t = 0 (t represents the number of iteration)
for Epoch in 1, ..., max epoch do
    for Batch in 1, ..., batch number do
        Sample training batch data S' from S.
        Generate predictions from f(S').
        Calculate Loss according to Eq.(14).
        Update the expert mixing weights α_i using Eq.(10).
        Update parameters W via minimizing the loss L.
        if Converged then
            Return f and parameters W*.
        end if
    end for
end for
Return f and parameters W*.
```

A Model Training Methods

In this section, we introduce the details of the training methods of GLINT-RU, including the loss function and the optimization algorithm of GLINT-RU displayed through pseudo-code.

Based on the prediction score in Eq.(13), we adopt the cross entropy loss function defined below for training GLINT-RU:

$$\mathcal{L} = \sum_{i=1}^{n_i} \left(y \log(\hat{y}_i) + (1 - y) \log(1 - \hat{y}_i) \right), \tag{14}$$

where y are the ground-truth labels of the user-item interactions. Now we can train GLINT-RU by following the guidance in Algorithm 1. As can be seen in the optimization algorithm, the training process of GLINT-RU can be easily implemented. The paralleled expert linear attention and dense selective GRU can be balanced by updating mixing weights. Additionally, to help GLINT-RU find a better local optimal solution, we follow the suggestion of [8] to initialize the model parameters $\mathcal{W}$, which helps prevents the vanishing/exploding gradient problem.

B Hyperparameter Settings

In this section, we will introduce the hyperparameter settings of our GLINT-RU and the baselines, and offer implementation details for reproducibility.

The parameter settings of all baselines are displayed in Table 5. In addition to the hyperparameter settings in Table 5, we also set the number of interests in LightSANs to 5, as the the average sequence length is less than 10, which is shown in Table 1. We set the expansion factor in Mamba4Rec as 2, and set the expansion factor in SMLP4Rec as 1 to avoid the GPU memory occupation error. The feature selection in SMLP4Rec is consistent with the settings in the original paper. [7]. For ML-1M dataset, we choose ['genre', 'movie title', 'release year'] as the input features in the SMLP structure. And for Amazon Beauty and Games, we select ['categories', 'brand'] as the input features. These features are employed to enhance the prediction performance from the item ids in SMLP4Rec.

C Ablation Study: Activation Functions

Unlike ReLU, which has a sharp kink at zero, SiLU is smooth and continuously differentiable. This smoothness facilitates better gradient flow during backpropagation, leading to faster and more stable training. Additionally, SiLU mitigates the vanishing gradient problem more effectively than Sigmoid and Tanh, as its gradient does not saturate as quickly, allowing for more efficient training of deeper networks. Incorporating SiLU as the activation function in the gate introduces non-linearity while retaining important item information and filtering out irrelevant item information, which is advantageous for complex tasks. Studies have demonstrated that SiLU enhances training stability and performance in deep learning models. In practical applications, user-item interactions often contain noise, as some selected items may be irrelevant. Under such conditions, employing SiLU typically achieves higher accuracy compared to traditional activation functions like ReLU and ELU. To further verify the effectiveness of the SiLU activation function, we conducted additional experiments where SiLU was replaced by ELU, Sigmoid, and ReLU in the dense selective gate. The results are presented in Table 6.

D Parameter Analysis: Layer Number

In this section, we conduct the parameter analysis on the number of layers L to observe the model's accuracy and efficiency in stacked GLINT-RU. We change the number of GLINT-RU layers from 1 to 4, and evaluate the model performance with NDCG@10 using the dataset Amazon Beauty. The results are shown in Figure 6.

From Figure 6, it is evident that as the number of layers increases, the accuracy of the GLINT-RU model exhibits a relatively gradual upward trend. This indicates that stacking GLINT-RU layers contributes positively to improving accuracy. However, this improvement is not pronounced, and the model's performance may even slightly decline when the number of layers becomes excessively large. Additionally, we analyze the model efficiency of GLINT-RU in Table 7: The results in Table 7 indicate that as the number of layers increases, the efficiency of GLINT-RU decreases rapidly. Combined with the previous model performance analysis, we can conclude that under resource constraints, using only one layer of efficient GLINT-RU can achieve high model efficiency and accuracy simultaneously.

Table 5: Hyperparameter settings of GLINT-RU and the baselines.

Model	Dataset	hidden size	weight decay	dropout	layers	heads	max length	kernel size	train&eval batch size
GRU4Rec	ML-1M	128	0	0.2	2	-	200	-	[2048, 2048]
	Beauty	64	0	0.5	2	-	100	-	[2048, 2048]
	Games	64	0	0.5	2	-	100	-	[2048, 2048]
BERT4Rec	ML-1M	128	0	0.2	2	8	200	-	[2048, 2048]
	Beauty	64	0	0.5	2	8	100	-	[2048, 2048]
	Games	64	0	0.5	2	8	100	-	[2048, 2048]
SASRec	ML-1M	128	0	0.2	2	8	200	-	[2048, 2048]
	Beauty	64	0	0.5	2	8	100	-	[2048, 2048]
	Games	64	0	0.5	2	8	100	-	[2048, 2048]
LinRec	ML-1M	128	0	0.2	2	8	200	-	[2048, 2048]
	Beauty	64	0	0.5	2	8	100	-	[2048, 2048]
	Games	64	0	0.5	2	8	100	-	[2048, 2048]
LightSANs	ML-1M	128	0	0.2	2	8	200	-	[2048, 2048]
	Beauty	64	0	0.5	2	8	100	-	[2048, 2048]
	Games	64	0	0.5	2	8	100	-	[2048, 2048]
SMLP4Rec	ML-1M	128	0	0.2	2	-	200	-	[2048, 2048]
	Beauty	64	0	0.5	2	-	100	-	[2048, 2048]
	Games	64	0	0.5	2	-	100	-	[2048, 2048]
Mamba4Rec	ML-1M	128	0	0.2	2	-	200	3	[2048, 2048]
	Beauty	64	0	0.5	2	-	100	3	[2048, 2048]
	Games	64	0	0.5	2	-	100	3	[2048, 2048]
GLINT-RU	ML-1M	128	0	0.2	2	8	200	3	[2048, 2048]
	Beauty	64	0	0.5	2	8	100	3	[2048, 2048]
	Games	64	0	0.5	2	8	100	3	[2048, 2048]

Table 6: Ablation study on activation functions in the dense selective gate.

Method	Recall@10	MRR@10	NDCG@10
Default (SiLU)	0.4472	0.2498	0.2964
ReLU	0.4293	0.2300	0.2770
Sigmoid	0.4328	0.2319	0.2793
ELU	0.4308	0.2321	0.2789

Table 7: Efficiency analysis on the parameter L

L	Inference Time	Training Speed	GPU Memory
1	278ms	3.8s/epoch	2.62GB
2	397ms	6.2s/epoch	5.17GB
3	501ms	8.9s/epoch	6.97GB
4	610ms	11.8s/epoch	8.86GB

E Ablation Study: Amazon Beauty Dataset and Video Games Dataset

To further verify the effectiveness of the components in GLINT-RU, we conducted more ablation studies on the Amazon Beauty and Amazon Video Games datasets. The results are shown in Tables 9 and 8. The results demonstrate that the gated GRU module plays a critical role in capturing dependencies among interactions and fine-grained positional representations. The linear attention mechanism enhances the model's ability to capture interactions between relevant items within the sequence. Adding a temporal convolution layer improves performance by incorporating contextual information from adjacent items. Finally, the gated MLP block filters complex information from the expert mixing block, further contributing to the overall performance. These findings indicate that all components of GLINT-RU are effective and collectively contribute to its high performance on both the Amazon Beauty and Amazon Video Games datasets.

Table 8: Ablation study results on Amazon Video Games.

Method	Recall@10	MRR@10	NDCG@10
Default (GLINT-RU)	0.6573	0.3549	0.4266
w/o GRU	0.6379	0.3125	0.4023
w/o Linear Attention	0.6459	0.3375	0.4106
w/o Temporal Conv1d	0.6443	0.3351	0.4084
w/o Gated MLP	0.6416	0.3339	0.4068

Table 9: Ablation study results on Amazon Beauty.

Method	Recall@10	MRR@10	NDCG@10
Default (GLINT-RU)	0.4472	0.2498	0.2964
w/o GRU	0.4239	0.2298	0.2755
w/o Linear Attention	0.4296	0.2314	0.2781
w/o Temporal Conv1d	0.4246	0.2290	0.2751
w/o Gated MLP	0.4265	0.2308	0.2768