**Noname manuscript No.**
(will be inserted by the editor)

# Time-aware Sequence Model for Next Item Recommendation

**Dongjing Wang · Dengwei Xu · Dongjin Yu ✉ · Guandong Xu**

Received: date / Accepted: date

**Abstract** The sequences of users' behaviors generally indicate their preferences, which can be used to improve next item prediction in sequential recommendation. Unfortunately, the users' behaviors may change over time, and it remains a great challenge to capture the user's dynamic preference directly from her/his recent behaviors sequence. Traditional methods such as Markov Chains, Recurrent Neural Networks, and Long Short-Term Memory (LSTM) Networks only consider the relative order of items in sequence, but ignore some important time information, such as the time interval and duration in the sequence. In this paper, we propose a novel sequential recommendation model, named Interval and Duration aware LSTM (IDLSTM), which leverages the interval and duration information to accurately model users' long-term and short-term preferences. In particular, the IDLSTM model incorporates the global context information of the sequence in the input layer to make better use of long-term memory. Furthermore, we also present an improved version of IDLSTM, namely IDLSTM with Embedding layer and Coupling input and output gates (IDLSTM-EC), which introduces the coupled input and forget

This research was supported by Zhejiang Provincial Natural Science Foundation of China under No. LQ20F020015, and the Fundamental Research Funds for the Provincial University of Zhejiang under No. GK199900299012-017.

D. Wang
School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, China
E-mail: dongjing.wang@hdu.edu.cn

D. Xu
School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, China
E-mail: 171050050@hdu.edu.cn

✉D. Yu
School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, China
E-mail: yudj@hdu.edu.cn

G. Xu
Advanced Analytics Institute, University of Technology Sydney, Australia
E-mail: Guandong.xu@uts.edu.au

gates and embedding layer to improve the efficiency and effectiveness. Experiments on real world datasets show that the proposed approaches outperform the state-of-the-art baselines, and is able to handle the problem of data sparsity effectively.

**Keywords** Recommendation · Sequence Modeling · Time-aware · Long Short-Term Memory

## 1 Introduction

Nowadays, people are severely influenced by the overwhelming massive information due to the rapid development of the Internet and Information Technology (IT), which is known as information overload problem [3]. Consequently, it becomes increasingly difficult to find the information they really need. Therefore, recommender systems have been proposed to help users find the contents that they want, such as research articles [8], Point-of-Interest [20,32], question [4] and music [23,24]. Existing recommendation methods include collaborative filtering-based recommendations [12,29], content-based recommendations [16], social network-based recommendations [13,1], and hybrid recommendation [7].

For many real-world applications, such as music listening and game playing, users usually make a series of actions within a period of time, which form behavior sequences. Such behavior sequence can be used to discover users' sequential patterns and help to predict users' next new action (or item), which is called sequential recommendation, one of the typical applications of recommender system. The traditional sequential recommendation models are mainly based on sequential pattern mining [31] and Markov Chain models [5]. The advent of deep learning has significantly boosted the performance of sequential recommendation [30,22]. For example, Recurrent Neural Networks (RNN) has been successfully applied in sequence modeling and next item prediction/recommendation [9]. Furthermore, as a variant of RNN, Long Short-Term Memory (LSTM) Networks solves the problem of gradient disappearance in RNN and provides better recommendation results. However, both RNN and LSTM focus only on relative order information of items in sequence, and ignore some important sequential information. Especially, items (or actions) that are close to each other in behavior sequence have strong correlations. In fact, this rule is not always true in sequential recommendation, because users may have some haphazard behaviors which do not indicate their preferences.

In this paper, we propose to model the sequences of items information, interval information and duration information in the users' behavior sequences to predict their next new items (or actions). Specifically, the next item refers to a new item that the user appears in other users' historical records instead of the target user's behavior sequence. For example, for a user's event sequence $\{A, B, A, C\}$, we will use $\{A, B, A\}$ to predict "$C$" instead of $\{A, B\}$ to predict "$A$". This is because "$A$" has already appeared in the sequence and repeated predictions are meaningless. In this sequence, "$C$" is a new item. Obviously, in sequential recommender systems, it is more meaningful (and also difficult) to

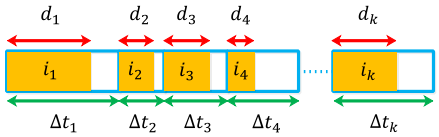



Fig. 1: Sequential behaviors with intervals and durations. $i_k$ represents the $k^{th}$ item in the sequence, $\Delta t_k$ represents the interval between $i_k$ and $i_{k+1}$, and $d_k$ represents duration of $i_k$.

recommend items that users may be interested in but they haven't interacted with, which are called new items for users. The scenario of this work is shown in Figure 1. The length of time (time interval) between two adjacent items indicates the correlation between them. In other words, the time information, in addition to the order information, reveals rich context for users' dynamic preferences modeling. Besides, the duration of users' actions or behaviors is also related to their preferences for corresponding items. For example, a user may be very interested in a game (or similar games) if he/she plays it for very long time (duration).

In order to make better use of items as well as time information, we present a novel recommendation model, namely Interval and Duration LSTM (IDLSTM). Specifically, an interval gate and a duration gate are firstly introduced to preserve users' short-term and long-term preferences, respectively. Then, the information of the two gates is seamlessly combined to improve next new item recommendation. Afterwards, an embedding layer is used after the input layer to incorporate the global context information and long-term memory. The main contributions of this paper are listed as follows:

- We propose a novel next new item recommendation model, which can make better use of important time information in sequences, such as interval and duration.
- We further improve the model's effectiveness and efficiency by adding an embedding layer and using coupled input and forget gates.
- Experiment results on real world datasets show that the proposed models outperform the state-of-the-art baselines, and is able to handle the problem of data sparsity effectively.

The rest of this paper is structured as follows. The related works are introduced in Section 2. Then we illustrate the motivation of this work with data statics and analysis in Section 3. Section 4 discusses the proposed methods in detail, and Section 5 demonstrates the experimental results and analysis. Finally, Section 6 concludes the paper and outlines the future work.

## 2 Related Works

### 2.1 Traditional Sequence Models

The traditional sequential models can be further divided into sequential pattern mining [31] and Markov Chain models [5]. Recommender systems based on sequential pattern mining first mine frequent patterns of data on sequence data, and then recommend via sequential pattern matching. For the sake of efficiency, these models may filter some infrequent but important patterns, which limits the recommendation performance, especially in coverage. Markov Chains (MC) methods are also used for sequence modeling. The main idea of such sequential recommendation models is to use the MC to model the probability of users' interaction events in the sequence and then predict the next event based on probability. Specifically, the Markov Chain model assumes that the current user interaction depends on one or more recent interaction events. Therefore, it can only capture local information of the sequence and ignore the global information of the sequence. Rendle et al. proposed Factorized Personalized Markov Chains (FPMC) model [19] and introduced an adaption of the Bayesian Personalized Ranking (BPR) [18] framework for sequential data modeling and recommendation. However, the Markov Chain model mainly focuses on the relationships between items in the short term, and it is not able to incorporate important information in long sequences.

### 2.2 Latent Representation based Sequential Models

Latent representation models learn the potential representation of each user or item, which contains some latent dependencies and features. The main categories in the latent representation model are the factorization machine [10] and the embedding model [26]. Sequential recommendation methods based on factorization machine usually use matrix factorization to factorize the observed user-item interaction matrix into potential vectors of users and items. Nisha et al. [14] used networks representation learning methods to capture implicit semantic social information and improve the performance of recommender systems. Wang et al. [25] proposed a Hierarchical Representation Model (HRM) based on the user's overall interests and final behaviors. Pan et al. [15] combined factorization and neighborhood-based methods, and proposed a novel method called matrix factorization with multiclass preference context (MF-MPC). Shi et al. [20] used factorization machine to construct a recommendation model, which effectively reduces the model parameters and improves the performance of the recommendation. Yu et al. [32] used the information based on users' context behaviors semantics on the POI recommendation model to solve the data sparse problem. However, sequential recommendation methods based on factorization are easily affected by the sparse observation data. The sequential recommendation model based on the embedded model usually maps all user interactions in the sequence into a potential low-dimensional

Table 1: Strengths and weaknesses of traditional methods

| **Methods** | **Strengths** | **Weaknesses** |
|---|---|---|
| Sequential pattern mining | 1. effective for fixed sequential patterns;<br>2. very efficient | 1. losing some infrequent but important modes<br>2. low coverage or diversity of recommendation results |
| Markov Chain models | capturing local information of the sequence very well | ignoring the global information of the sequence |
| Factorization machine | good representation of the inherent characteristics of the event | easily affected by data sparsity |
| Embedding model | reducing the dimension of event representation and suitable for many tasks | cannot modeling the features in sequence adaptively |
| Recurrent Neural Networks | well suited for modeling complex dynamics in sequences | 1. relying on the sequence information<br>2. cannot preserve the user's interest for a long time |

space through a new coding method. The embedding model is used in many fields, such as word2vec, and GloVe [17]. Among them, the vector obtained by embedding the model is usually used for the input of the neural networks. It should be noted that the representation vector is obtained by the order of interaction between users or items, which is completely different from the vector in collaborative filtering. The embedding models make the models tend to use global information rather than local information.

### 2.3 Deep Learning based Sequential Models

In recent years, the most commonly used deep learning method in sequential recommendation is the Recurrent Neural Networks (RNN). Recurrent Neural Networks is well suited for modeling complex dynamics in sequences due to its special structure [11,21,33] Zhang et al. [33] proposed a novel framework based on Recurrent Neural Networks which can model user sequence information through click events. Twardowski et al. [21] combined context information to propose a recommender that can handle the long-term and short-term interests of users in the news domain. Hu et al. [11] proposed a neural networks model using item context to better model the purchasing behaviors of users. In order to improve the session-based recommender system, Wang et al. [27] designed effective Mixture-Channel Purpose Routing Networks (MCPRNs) and improved the accuracy and diversity of recommendations. The RNN-based recommendation methods rely too much on the sequence information, and at the same time cannot save the user's interest for a long time.

## 3 Data Analysis and Motivation

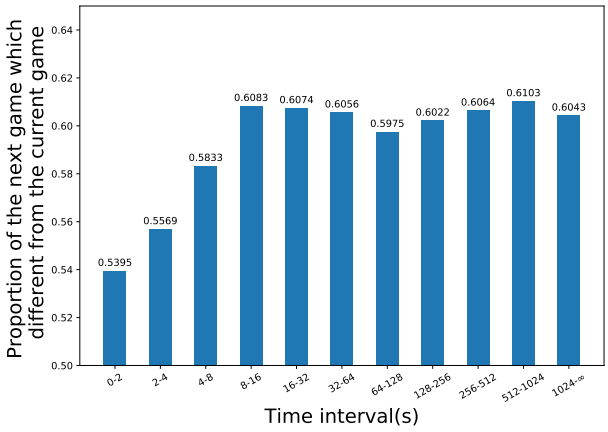
Table 2: Examples of game playing data

| Record# | Timestamp | UserId | Game | Duration(s) |
|---|---|---|---|---|
| 1 | 2016-09-01 01:00:47 | 386576 | League of Legends | 8700 |
| 2 | 2016-09-01 18:45:25 | 386576 | QQGame | 600 |
| 3 | 2016-09-03 17:31:28 | 386576 | League of Legends | 1500 |
| 4 | 2016-09-03 20:22:21 | 386576 | CrossFire | 1200 |
| 5 | 2016-09-16 21:57:25 | 386576 | QQ Speed | 600 |
| 6 | 2016-09-16 22:52:33 | 386576 | CrossFire | 2400 |
| 7 | 2016-09-17 00:27:34 | 386576 | League of Legends | 12300 |
| 8 | 2016-09-17 18:37:48 | 386576 | League of Legends | 1740 |
| 9 | 2016-09-18 02:08:41 | 386576 | CrossFire | 6780 |
| 10 | 2016-09-19 01:17:36 | 386576 | Call of Duty Online | 900 |
| 11 | 2016-07-17 10:47:08 | 635033 | DNF | 3000 |
| 12 | 2016-07-18 09:57:07 | 635033 | DNF | 1800 |
| 13 | 2016-07-18 11:22:07 | 635033 | DNF | 4500 |
| 14 | 2016-07-18 14:47:08 | 635033 | QQ Speed | 3000 |
| 15 | 2016-07-19 10:37:07 | 635033 | QQ Speed | 2700 |

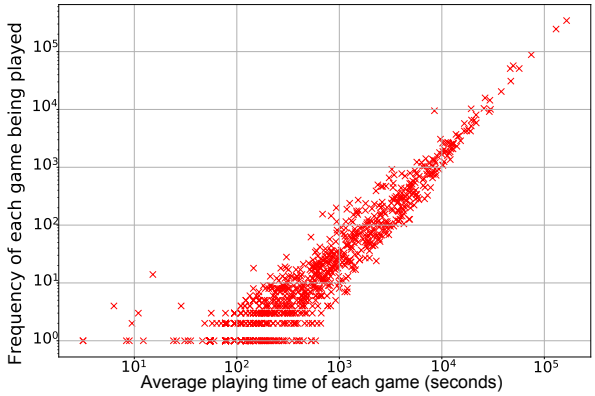
In this section, we will introduce the motivation of this works, which is further explained via data analysis. We use the game dataset as a case to explain the role of time interval and duration, and then present some samples of game-playing data in Table 2. For example, **Record 1** in Table 2 indicates that the user with the user ID of 386576 played "League of Legends" at 2016-09-01 01:00:47 (timestamp) for 8700 seconds (duration).

The interval represents the difference between two adjacent records' timestamps of the same user. For example, the interval between **Record 2** and **Record 1** in Table 2 is 63878 seconds. Zhu et al. [34] showed that the shorter the interval is, the greater the impact of the current item has on the next item. One reason is that users may repeatedly play similar games in a short period of time, which represents their short-term preferences. For example, the user with the user ID of 386576 frequently plays "League of Legends" in a short period of time. Besides, the information in Figure 2a shows that the proportion of adjacent games are different in the sequence increases overall when the time interval is longer. In other words, longer interval indicates low correlation between adjacent items, which also influence the modeling of users' preferences.

Besides, duration is also an important feature in sequence modeling and sequential prediction/recommendation. As shown in Table 2, the duration indicates how long a user plays a game. Generally, duration can reflect the degree of the users' preferences for corresponding items. When the user plays a game for a longer duration, he/she will play the game more frequently, and in other words, he/she is more interested in the corresponding game. For example, in Table 2, the frequently played games, "League of Legends" and "CrossFire",




(a) The relationship between time interval and adjacent game types.




(b) The relationship between duration and frequency of the game playing.

Fig. 2: Statistics of game data sets

have longer durations. As shown in Figure 2b, we further illustrate the relationship between the average duration of the game and the frequency of game being played with data analysis. The results show that longer duration indicates that users are interested in corresponding games and tend to play them more frequently and spend more time (duration). In other words, the users' preferences for the game are reflected in the duration.

In general, users have both long-term and short-term preferences [28]. Specifically, long-term interest refers to the users' long-term and static interest. For example, some users only like role-playing games, so they may be playing this type of game most of the time. But in fact, the users' preferences may change over time, and the next item or action is more likely to depend on users' recent behaviors, which is called short-term interest. For example, some

users only like role-playing games, but they may try some popular strategy games. In order to better capture users' long-term and short-term interests, we need to make better use of both the time interval information and duration information. Therefore, we propose a novel recommendation model to incorporate the interval and duration information into next new item recommendation.

## 4 The Proposed Method (IDLSTM-EC)

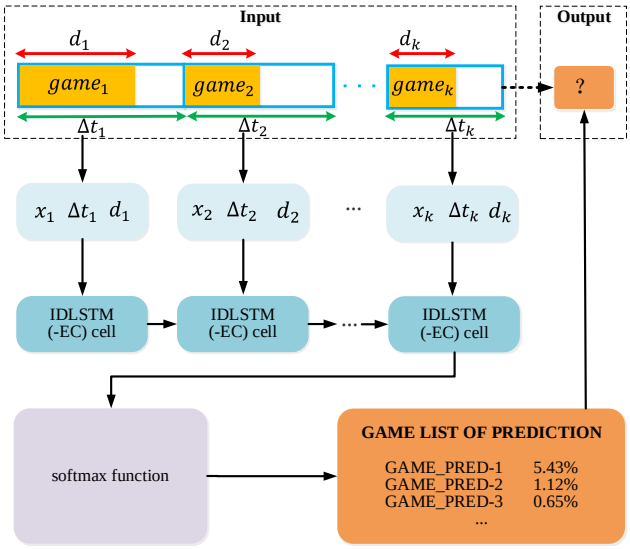



Fig. 3: The architecture of the proposed model in sequential recommendation. In architecture, a game sequence is taken as an example, where $\{game_1, game_2, \cdots, game_k\}$ is the sequence, $x$ is the one-hot vector of the $game$, $\Delta t$ is the time interval, $d$ is the duration. IDLSTM(-EC) is the proposed model in this paper.

The time information in this work includes both time interval and duration. Specifically, interval indicates correlation between current item and next item in the sequence and the duration affects the user's preferences for corresponding events (similar to the rating). Inspired by the analysis and motivation in Section 3, we propose a novel next new item recommendation method namely Interval and Duration aware LSTM (IDLSTM), as well as its improved version, namely IDLSTM with Embedding layer and Coupling input and output gates (IDLSTM-EC).

Figure 3 shows how the proposed model perform prediction and recommendation based on users' sequences. In the process, we firstly extract the interval

$\Delta t$, duration $d$, and $x$ from the sequence $\{game_1, game_2, \cdots, game_k\}$, where $x$ is the one-hot vector of the $game$. Then, we feed the obtained information into the IDLSTM(-EC) cell. Finally, the result output by the IDLSTM(-EC) cell is passed through the softmax function to obtain the probability of each game to be played next. Compared with Recurrent Neural Networks (RNN) or Long Short-Term Memory (LSTM) Networks, the proposed model can incorporate three kinds of inputs (item, time interval and duration) into sequence modeling and recommendation in a unified way.

The architectures of the proposed models as well as LSTM are shown in Figure 4. Specifically, in order to illustrate the improvement of the proposed model, we use different colored lines to highlight the improvement. In addition, parameters that appear in Section 4 are explained in Table 3. Next, we will introduce IDLSTM as well as the improved model IDLSTM-EC in details. The following mainly introduces the models used in the IDLSTM(-EC) cell part.

### 4.1 IDLSTM

As shown in Figure 4b, IDLSTM introduces an interval gate $I$ and a duration gate $D$ into the LSTM model. As shown in the figure, we use purple lines to highlight processing of time interval and duration with the time interval gate and duration gate. Specifically, the interval gate models the impact of the current event on the next event based on time interval information, while the duration gate is used to model users' long-term interest in various items based on duration information.

The equation for the interval gate $I_k$ and the duration gate $D_k$ are formally defined as follows:

$$I_k = \sigma_{t_i}\left(W_{t_i} x_k + \sigma'_{t_i}\left(S_{t_i} \Delta t_k\right) + b_{t_i}\right), \tag{1}$$

$$D_k = \sigma_{t_d}\left(W_{t_d} x_k + \sigma'_{t_d}\left(S_{t_d} d_k\right) + b_{t_d}\right). \tag{2}$$

Further, the interval gate $I_k$ and the duration gate $D_k$ are added to IDLSTM, which is defined as follows:

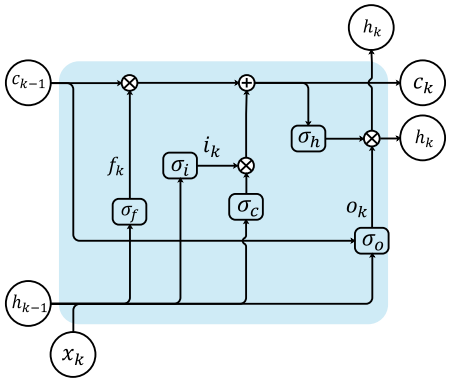
$$i_k = \sigma_i(W_i x_k + U_i h_{k-1} + P_i \circ \hat{c}_{k-1} + b_i), \tag{3}$$

$$f_k = \sigma_f(W_f x_k + U_f h_{k-1} + P_f \circ \hat{c}_{k-1} + b_f), \tag{4}$$

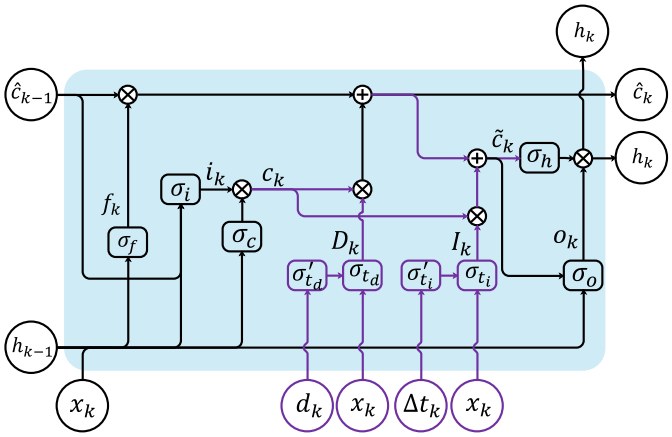
$$c_k = i_k \circ \sigma_c(W_c x_k + U_c h_{k-1} + b_c), \tag{5}$$

$$\hat{c}_k = f_k \circ \hat{c}_{k-1} + D_k \circ c_k, \tag{6}$$

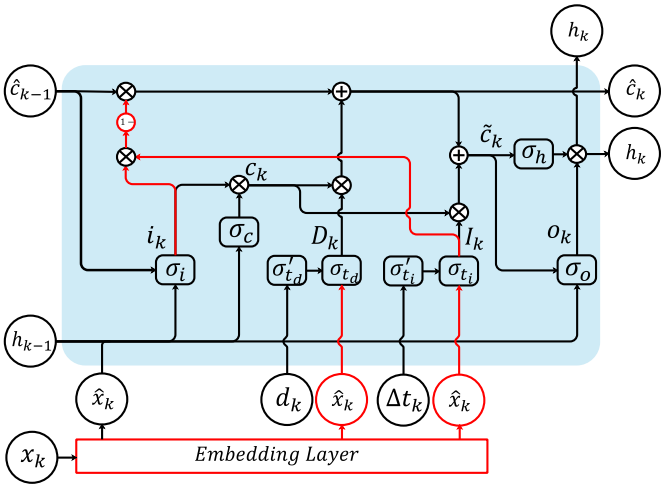
$$\widetilde{c}_k = \hat{c}_k + I_k \circ c_k, \tag{7}$$




(a) LSTM




(b) IDLSTM (the proposed model)




(c) IDLSTM-EC (the proposed model)

Fig. 4: Architectures of models (a) LSTM, (b) IDLSTM and (c) IDLSTM-EC. IDLSTM has a time gate $I_k$ and a duration gate $D_k$. $I_k$ is designed to model time interval $\Delta t$ for the impact of the current event on the next event and $D_k$ is designed to model duration $d$ for users' interests. IDLSTM-EC is a variant of IDLSTM using coupled input and forget gates and input $x$ has been converted to $\hat{x}$ by embedding layer. To illustrate our improvement on the LSTM model, we use purple lines to highlight the improvements in IDLSTM. Besides, to clarify the improvement of IDLSTM-EC, we use red lines to mark the difference from IDLSTM.

Table 3: Parameter Description

| **Parameter** | **Description** |
|---|---|
| $x_k$ | one-hot vector of $game_k$ |
| $\hat{x}_k$ | embedding of $game_k$ |
| $\Delta t_k$ | time interval between $game_k$ and $game_{k+1}$ |
| $d_k$ | duration of $game_k$ |
| $I_k$ | time interval gate |
| $W_{t_i}$ | weight of input $x_k$ in $I_k$ |
| $S_{t_i}$ | weight of interval $\Delta t_k$ |
| $b_{t_i}$ | bias of interval gate $I_k$ |
| $D_k$ | duration gate |
| $W_{t_d}$ | weight of input $x_k$ in $D_k$ |
| $S_{t_d}$ | weight of duration $d_k$ |
| $b_{t_d}$ | bias of interval gate $D_k$ |
| $\sigma_{t_i}, \sigma_{t_d}$ | sigmoid function |
| $\sigma'_{t_i}, \sigma'_{t_d}$ | hyperbolic tangent (tanh) function |
| $i_k$ | input gate vector |
| $f_k$ | forget gate vector |
| $o_k$ | output gate vector |
| $c_k$ | cell state vector |
| $\hat{c}_k$ | new cell with duration gate information based on $c_k$ |
| $\widetilde{c}_k$ | new cell with interval gate and duration gate information based on $\hat{c}_k$ |
| $h_k$ | hidden state vector |
| $W_i$ | weight of input gate |
| $W_f$ | weight of forget gate |
| $W_c$ | weight of cell |
| $W_o$ | weight of output gate |
| $U_i$ | weight of hidden unit of input gate |
| $U_f$ | weight of hidden unit of forget gate |
| $U_o$ | weight of hidden unit of output gate |
| $b_i$ | bias of input gate |
| $b_f$ | bias of forget gate |
| $b_o$ | bias of output gate |
| $b_c$ | bias of cell |
| $\sigma_i, \sigma_f, \sigma_o$ | sigmoid function |
| $\sigma_c, \sigma_h$ | hyperbolic tangent (tanh) function |
| $\circ$ | operator denotes element-wise product (Hadamard product) |

$$o_k = \sigma_o(W_o x_k + V_o \Delta t_k + U_o h_{k-1} + P_o \circ \widetilde{c}_k + b_o), \tag{8}$$

$$h_k = o_k \circ \sigma_h(\widetilde{c}_k). \tag{9}$$

The input of the duration gate $D_k$ is added to $\hat{c}_k$ to associate the input vector $x_k$ with both the input gate and the duration gate. Besides, we add the interval gate $I_k$ to $\widetilde{c}_k$ so that $\widetilde{c}_k$ incorporates both the information of the interval gate $I_k$ and the duration gate $D_k$.

The cell $\hat{c}_k$ is used to further model the user's interest by adding the information of the duration gate $D_k$. We also add cell $\widetilde{c}_k$ to combine duration and interval information for recommendation.

Specifically, a small interval (large time interval gate $I_k$) means that the current item has a great influence on the next item. Correspondingly, $\hat{c}_{k-1}$ will be relatively small, and the next item is more influenced. In this way, IDLSTM can combine the duration and interval information to perform a more precise recommendation. On the other hand, $\widetilde{c}_k$ is directly connected to the output gate and is used to control the output together with the output gate. In addition, $\Delta t$ is added to the output gate to control the output better with other parameters in the output gate, and $V_o$ is the weight coefficient of the input gate. IDLSTM combines the information of time interval and duration well, and the use of interval gate and duration gate enables the two kinds of time information to be preserved for a longer period of time.

### 4.2 IDLSTM-EC

In order to further improve the effectiveness and efficiency of the proposed model, we propose IDLSTM-EC, which introduces an embedding layer to utilize more sequence information and uses coupled input and forget gates. The structure of IDLSTM-EC is shown in Figure 4c and we use red lines to mark main improvements.

- **Adding embedding layer:** In the IDLSTM model, all inputs are converted into one-hot vectors, which may lose some important information, such as the correlation between different items. In fact, the co-occurrence and context relationships between the inputs play important roles in sequential recommendation. However, IDLSTM only employs part of the context information but fail to utilize the global context. In order to incorporate more context information, an embedding layer is added after the input to transform the original one-hot vectors into low dimensional real-valued vectors (embeddings), which can effectively capture important features of items and their relationships in the training data. Specifically, the Global Vectors (GloVe) [17] method is used to train the embedding vector. GloVe model is a popular embedding method, which obtains vectors through unsupervised learning. Unlike other embedding methods, GloVe model incorporates global information and contextto capture more important information.
- **Using coupled input and forget gates:** The parameters of the proposed model are reduced by using coupled input and forget gates. Thus, Eq. 4 will be removed and Eq. 6 is modified as follows:

$$\hat{c}_k = (1 - I_k \circ i_k) \circ \hat{c}_{k-1} + D_k \circ c_k. \tag{10}$$

  Specifically, $\hat{c}_k$ is the main cell and the input is affected by both $I_k$ and the input gate, $f_k$ is replaced with $(1 - I_k \circ i_k)$ in $\hat{c}_k$. The IDLSTM-EC model coupling input and forget gate increase the efficiency by reducing model parameters. At the same time, the reduction of parameters prevents the model from overfitting to some extent.

## 5 Experiments

In this section, we will evaluate the proposed models as well as state-of-the-art baselines on two real-world datasets. The First dataset is the game-playing records collected from the word-leading Internet bar which has the largest number of game players in China. The second one is a public music listening dataset LastFM-1K[1], which include all the music listening sequences and timestamps of nearly 1,000 listeners up to May 5, 2009. We have made preprocess on the two datasets and delete users and items with only a few records. The statistics information of the final datasets is shown in Table 4, where #(∗) indicates the number of ∗, and Average (Item) indicates the average number of interactions by all users.

Table 4: Statistics of Two Datasets

| | **#(Records)** | **#(Users)** | **#(Item)** | **Average(Item)** |
|---|---|---|---|---|
| **Game dataset** | 955,377 | 2,153 | 1,003 | 952.5 |
| **LastFM** | 769,674 | 967 | 5,000 | 795.9 |

### 5.1 Compared Methods

The proposed models are compared with the state-of-the-art recommendation methods, including traditional recommendation methods and the variants of LSTM mentioned above.

#### *5.1.1 Baselines*

We adopt two kinds of recommendation methods as baselines, including general recommendation models and sequence-based recommendation models. Specifically, general models mainly perform traditional, non-sequential recommendation, while sequence-based models can perform next item recommendation via machine learning or neural networks. Besides, we also compare different versions of the proposed models to show the effectiveness of each improved component.

General recommendation models:

- **POP**: Popularity predictor which recommends the most popular items to users.
- **UBCF**: User-Based Collaborative Filtering.
- **BPR**: Bayesian personalized ranking [18].

Sequence-based next item recommendation models:

[1] http://ocelma.net/MusicRecommendationDataset/lastfm-1K.html

- **FPMC**: Factorizing Personalized Markov Chains [19].
- **Session-RNN**: A variant of traditional RNN which can capture the user's short-term interest.
- **Peephole-LSTM**: A variant of LSTM which adds "peephole connection" to allow all gates to accept input from the state [6].
- **Peephole-LSTM with time**: This model adds the time information to Peephole-LSTM for a fair comparison.
- **Time-LSTM**: A variant of LSTM that adds two time gates to the traditional LSTM [34].

#### *5.1.2 The proposed methods*[1]

- **IDLSTM**: The proposed model which has the interval gate and the duration gate.
- **IDLSTM-C**: An improved model of IDLSTM, which employs the coupled input and forget gates.
- **IDLSTM-E**: An improved model of IDLSTM, which adds an embedding layer.
- **IDLSTM-EC**: An improved model of IDLSTM, which adds an embedding layer, and couples the input and forget gates.

### 5.2 Experiment Setup

In the experiment, the task is to predict the next new item users will be most likely to interact with according to the existing behavior sequences. During the training phase, an improved stochastic gradient descent method called Adagrad [2] is used, which is able to adapt the learning rate to the parameters. Specifically, Adagrad can improve the convergence ability of the model by increasing the learning rate of sparse parameters. In addition, the cross-entropy is chosen as the loss function, which is defined as follows:

$$Loss = -\frac{1}{M}\sum(pos_i \times y_i \log \hat{y}_i), \tag{11}$$

where $M$ is the number of training samples, $y_i$ is the value of the real item, $\hat{y}_i$ is the value of the predicted item, and $pos_i$ is the new-item indicator. When $y_i$ corresponds to a new item, $pos_i = 1$. Otherwise $pos_i = 0$.

All experiments are conducted on the PC with Intel(R) Core(TM) i9-7900X @ 3.30GHz and GeForce GTX 1080 Ti, 64GB memory, and Ubuntu 16.04.

### 5.3 Evaluations Metrics

The proposed models are evaluated with two metrics, including Recall and MRR.

[1] https://github.com/vallzey/IDLSTM

– **Recall:** Recall (aka sensitivity) is defined as follows:

$$Recall@n = \frac{\#(n,hit)}{\#(all)}, \tag{12}$$

where $\#(n, hit)$ is the number of predicted results in the top-$n$ of the recommended list, and $\#(all)$ is the number of all test samples. Recall is a common evaluation criterion and is usually used to evaluate if the recommendation lists contain the target item.

– **MRR:** MRR (Mean Reciprocal Rank) is a ranking evaluation metrics which indicates the average of the reciprocal ranks of the target items in recommendation list. Formally, it is defined as follows:

$$MRR@n = \frac{1}{\#(all)} \times \sum \frac{1}{rank_i}, \tag{13}$$

where $rank_i$ denotes the ranking of the $i$-th test target item in the recommendation list. If $rank_i > n$, $\frac{1}{rank_i} = 0$. The MRR is the average of the reciprocal levels of the target items. When the $Recall@n$ of several models are similar (different models have similar proportion of target items appearing recommendation list), we can use MRR to further evaluate them. Especially, $MRR@n$ is the same with $Recall@n$ when $n = 1$.

## 5.4 Results and Analysis

Table 5: Recall of the proposed methods and baselines

| ***Game dataset*** | ***Recall@1*** | ***Recall@5*** | ***Recall@10*** | ***MRR@1*** | ***MRR@5*** | ***MRR@10*** |
|---|---|---|---|---|---|---|
| POP | 0.0170 | 0.1246 | 0.1756 | 0.0170 | 0.0526 | 0.0594 |
| UBCF | 0.0170 | 0.0652 | 0.1160 | 0.0170 | 0.0323 | 0.0387 |
| BPR | 0.0595 | 0.1501 | 0.3116 | 0.0595 | 0.0916 | 0.1121 |
| FPMC | 0.0142 | 0.1246 | 0.2040 | 0.0142 | 0.0493 | 0.0597 |
| Session-RNN | 0.0312 | 0.1983 | 0.3683 | 0.0312 | 0.0873 | 0.1298 |
| Peephole-LSTM | 0.1246 | 0.3654 | 0.5127 | 0.1246 | 0.2085 | 0.2278 |
| Peephole-LSTM with time | 0.1417 | 0.3768 | 0.5270 | 0.1416 | 0.2266 | 0.2463 |
| Time-LSTM | 0.1671 | 0.4165 | 0.5581 | 0.1671 | 0.2634 | 0.2831 |
| IDLSTM | 0.2323 | 0.4844 | 0.6034 | 0.2323 | 0.3232 | 0.3484 |
| IDLSTM-C | 0.2351 | 0.4844 | 0.5836 | 0.2351 | 0.3291 | 0.3427 |
| IDLSTM-E | **0.2720** | **0.5099** | 0.6176 | **0.2720** | 0.3576 | 0.3720 |
| IDLSTM-EC | 0.2663 | 0.5071 | **0.6204** | 0.2663 | **0.3604** | **0.3752** |
| ***LastFM*** | ***Recall@1*** | ***Recall@5*** | ***Recall@10*** | ***MRR@1*** | ***MRR@5*** | ***MRR@10*** |
| POP | 0.0103 | 0.0182 | 0.0213 | 0.0103 | 0.0117 | 0.0193 |
| UBCF | 0.0121 | 0.0187 | 0.0226 | 0.0121 | 0.0149 | 0.0216 |
| BPR | 0.0297 | 0.0452 | 0.0554 | 0.0297 | 0.0335 | 0.0489 |
| FPMC | 0.0153 | 0.0228 | 0.0282 | 0.0153 | 0.0182 | 0.0254 |
| Session-RNN | 0.0292 | 0.0448 | 0.0529 | 0.0292 | 0.0322 | 0.0482 |
| Peephole-LSTM | 0.0680 | 0.0831 | 0.0871 | 0.0680 | 0.0781 | 0.0802 |
| Peephole-LSTM with time | 0.0691 | 0.0769 | 0.0926 | 0.0691 | 0.0702 | 0.0809 |
| Time-LSTM | 0.0810 | 0.1040 | 0.1259 | 0.0810 | 0.1161 | 0.1209 |
| IDLSTM | 0.0943 | 0.1635 | 0.1642 | 0.0943 | 0.1290 | 0.1327 |
| IDLSTM-C | 0.0931 | 0.1622 | 0.1658 | 0.0931 | 0.1210 | 0.1389 |
| IDLSTM-E | 0.1099 | 0.1648 | 0.1978 | 0.1099 | 0.1328 | 0.1428 |
| IDLSTM-EC | **0.1124** | **0.1758** | **0.2088** | **0.1124** | **0.1350** | **0.1431** |

The comparisons between the proposed model and baseline models are presented in Table 5. The results show that the proposed model has the best performance in sequential recommendation. Besides, models with neural network structure have better performance than models without considering sequence factors. Specifically, IDLSTM-EC achieves 11.1% and 32.5% improvements than the best baseline method in terms of Recall@10 and MRR@10, respectively on game datasets. Besides, the improvements of IDLSTM-EC on LastFM datasets are 65.8% (Recall@10) and 18.3% (MRR@10), separately. Next, we will analysis the performance of each model in details.

- **POP, UBCF, BPR and FPMC**: POP only recommends items with high popularity, which results in low coverage. UBCF and BPR ignore the dependence of items in sequences and they cannot model users' short-term preferences. However, in the sequential recommendation, the recent items usually have large in decision making. Meanwhile, FPMC method achieves better performance than POP and UBCF because it combines matrix factorization and Markov chains to model users' behavior sequences. However, FPMC have difficulty retaining information in the sequence for a long time and it does not fit well with long sequences.
- **Session-RNN**: Session-RNN mainly captures the user's short-term interests, but does not consider the user's long-term interests, which limits its performance. However, in sequential recommendation, both users' long and short-term preferences play important roles in sequential recommendation.
- **Peephole-LSTM and Peephole-LSTM with time**: The Peephole-LSTM does not work well due to the lack of time information. Compared to Peephole-LSTM, the Peephole-LSTM with time incorporates time information into the input, an achieves slightly better performance in most cases. However, it is not quite effective to add time information directly to the input. In addition, and those two approaches cannot capture or preserve users' long-term preferences accurately.
- **Time-LSTM**: Time-LSTM incorporate time information into the sequences modeling process in a more effective way, so it achieves better performance than Peephole-LSTM and Peephole-LSTM with time. Especially, the lack of duration information in Time-LSTM decreases its ability to fully utilize time information or capture the users' preferences accurately, which limits its performance.
- **IDLSTM and IDLSTM-E**: IDLSTM and IDLSTM-E perform better than all baselines in Recall and MRR. This shows that it is better to use the duration gate and the interval gate at the same time to perform recommendation. Besides, the performance of IDLSTM-E is much better than that of IDLSTM. The reason is that the proposed methods can utilize both time interval and duration with gate mechanisms effectively to perform better recommendation. Besides, the results also show both interval and duration information is important in sequence modeling as well as capturing users' long and short-term preferences. The performance of IDLSTM-E is better than IDLSTM. The reason is that the embedding layer

based on GloVe can effectively capture the global information in users' behavior sequences, which enables the proposed methods to achieve better performance in sequential recommendation.

– **IDLSTM, IDLSTM-C and IDLSTM-EC**: Extensive experiments have shown that IDLSTM using coupled input and forget gates (IDLSTM-C) has no significant improvement than IDLSTM in accuracy evaluation. The efficency comparisons of each model to run an epoch are listed in Table 6, which shows that IDLSTM-C and IDLSTM-EC are approximately 6% faster than IDLSTM and IDLSTM-E. Therefore, the efficiency of the proposed models is improved via using coupled gates to reduce the parameters that need to be trained. Besides, traditional method cannot achieve accurate results although they require much less time due to their concise structure. In addition, in actual applications, the recommendation models are generally pre-trained offline, so the comparison test time is more meaningful. Specifically, all recommendation methods can perform recommendation in test phase within close and reasonable time.

In conclusion, traditional recommendation methods (such as UBCF and BPR) do not consider the dynamic changes in user interests, which results in poor results. The sequence-based methods (such as Session-RNN, Peephole-LSTM, Peephole- LSTM with time, Time-LSTM, and IDLSTM(-EC)) achieves better performance than traditional recommendation methods due to the effectiveness of Recurrent Neural Network in modeling users' behavior sequences. Especially, the proposed models can make better use of the time interval and duration information, which is very important for sequence modeling and sequential prediction/recommendation. In addition, the improvement of IDLSTM-EC over IDLSTM shows that the global information (related to users' long-term preference) captured by IDLSTM-EC is quite important in sequential recommendation.

## 5.5 Effect of units' number

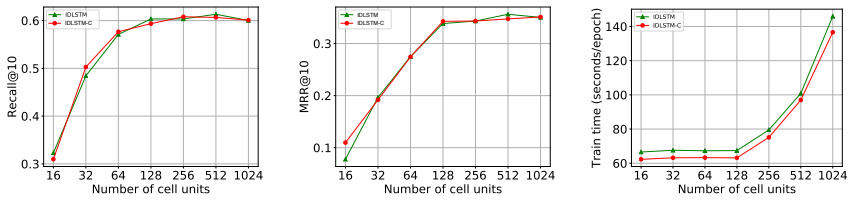
In this subsection, we evaluate the influence of the number of cell units and the number of embedding layer units with two experiments. In the first experiment, the effect of the number of cells is evaluated. Then, the best number of units in the first experiment is used in the second experiment to evaluate the effect of the number of embedding layer units.

### *5.5.1 Effect of cell units' number*

We first set the number of cell units to (16, 32, 64, 128, 256, 1024) and evaluate the impact of the number of cell units of IDLSTM and IDLSTM-C in terms of Recall and MRR. As shown in Figure 5, two models have similar performance in Recall@10 and MRR@10. In addition, IDLSTM-C takes less time than IDLSTM with the increase of number of cell units. Meanwhile, the promotion

Table 6: Time for each model to run an epoch

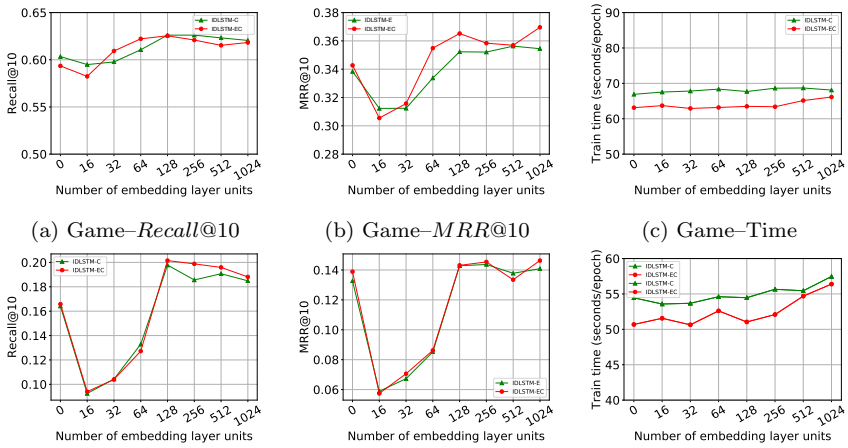
| **Game dateset** | **Training time (s/epoch)** | **Testing time (s)** |
|---|---|---|
| POP | Memory-based | 0.01 |
| UBCF | Memory-based | 0.31 |
| BPR | 0.10 | 0.97 |
| FPMC | 0.43 | 0.40 |
| Session -RNN | 45.23 | 1.91 |
| Peephole-LSTM | 51.65 | 1.93 |
| Peephole-LSTM with time | 53.72 | 1.94 |
| Time-LSTM | 65.82 | 1.95 |
| IDLSTM | 65.37 | 1.98 |
| IDLSTM-E | 65.29 | 1.97 |
| IDLSTM-C | 56.84 | 1.96 |
| IDLSTM-EC | 59.90 | 1.97 |
| **LastFM** | **Training time (s/epoch)** | **Testing time (s)** |
| POP | Memory-based | 0.01 |
| UBCF | Memory-based | 0.15 |
| BPR | 0.08 | 0.58 |
| FPMC | 0.34 | 0.23 |
| Session -RNN | 42.24 | 0.93 |
| Peephole-LSTM | 44.95 | 1.10 |
| Peephole-LSTM with time | 45.04 | 1.11 |
| Time-LSTM | 53.22 | 1.15 |
| IDLSTM | 54.45 | 1.11 |
| IDLSTM-E | 51.47 | 1.03 |
| IDLSTM-C | 54.47 | 1.16 |
| IDLSTM-EC | 51.54 | 1.09 |




(a) Game–*Recall*@10 (b) Game–$MRR$@10 (c) Game–Time

(d) LastFM–*Recall*@10 (e) LastFM–$MRR$@10 (f) LastFM–Time

Fig. 5: The effect of different numbers of cell units on *Recall*@10, $MRR$@10 and time of an epoch.




(a) Game–*Recall*@10 (b) Game–*MRR*@10 (c) Game–Time

(d) LastFM–*Recall*@10 (e) LastFM–*MRR*@10 (f) LastFM–Time

Fig. 6: The effect of different numbers of embedding units on *Recall*@10, *MRR*@10 and time of an epoch.

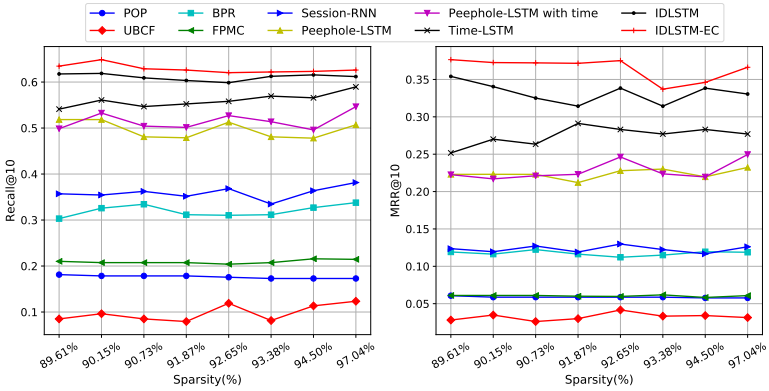
of Recall@10 and MRR@10 becomes gradually stable. In particular, after the number of cell units is larger than 128, the performance of Recall@10 and MRR@10 are not much improved. Thus, the best number of cell units is set as 128, which enable the proposed model to capture most important information.

### *5.5.2 Effect of embedding units' number*

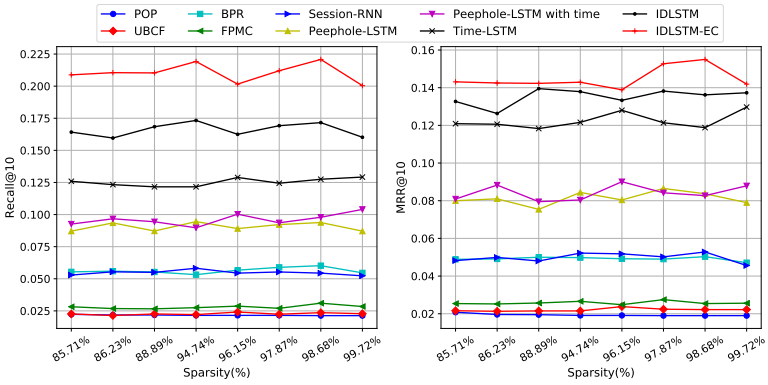
The effect of different numbers of embedding units is further investigated where the number of cell units is set to 128, and the results are shown in Figure 6. Especially, the results without embedding layer are also added at 0-abscissa for comparison. When the number of cells in the embedding layer is less than 32, the performance of the proposed models is lower than that without the embedding layer model. Therefore, it is necessary to have enough units in embedding layer to ensure that the sequence information is well preserved in recommendation model.

As shown in Figures 6c and 6f, although the time varies, the overall fluctuation is not large, because the number of embedding layer units' parameters only accounts for a small part of the model. Therefore, different numbers of embedding layer units do not have much impact on the efficiency of the proposed approach.

Besides, as shown in Figures 6a, 6b, 6d and 6e, when the number of embedding layer units increases from 16 to 128, Recall@10 and MRR@10 are also improved. However, when the number of embedding units becomes larger than 128, Recall@10 and MRR@10 have no significant increase, even result in a downward trend. The reason is that excessive unit may cause overfitting. Therefore, the number of embedding layer units is set as 128.




(a) Game dataset




(b) LastFM

Fig. 7: Comparison of different models on data with different sparsity.

## 5.6 Impact of data sparsity

We further evaluate the proposed methods against baselines on datasets with different sparsity to verify their ability of dealing with sparse data. Specifically, the items with frequency less than d are removed in the dataset, where d is set to (0, 5, 10, 15, 20, 30, 40, 50) respectively, and the sparsity of corresponding datasets are $(97.04\%, 94.50\%, 93.38\%, 92.65\%, 91.87\%, 90.73\%, 90.15\%, 89.61\%)$ for Game dataset and $(99.72\%, 98.68\%, 97.87\%, 96.15\%, 94.74\%, 88.89\%, 86.23\%, 85.71\%)$ for LastFM. As shown in Figure 7, the performance of the various methods is not significantly decreased as the sparsity increases, because our task is to recommend next new items that users may be interested in but haven't interacted with. Especially, some items with low frequency are excluded to change the sparsity of dataset, which may also remove some key items or correlations. For example, if "$C$" is removed from sequence, next new item recommender system (our work) performs one predictions, $\{A, A\} - >$"$B$".

The performance will decrease if "$C$" is key item for prediction of "$B$". But even so, the proposed method achieves better performance than baselines in terms of Recall@10 and MRR@10. In conclusion, our methods can deal with data with different sparsity effectively.

## 6 Conclusions

In the paper, we propose a novel time-aware sequence modeling method and apply it to next new item recommendation. Specifically, the proposed method introduced two gates, i.e., a duration gate for modeling users' preferences and an interval gate for modeling the impact of the current item on the next item in sequences. In addition, we adopt GloVe to take advantage of global context information and further improve its efficiency by using coupled input and forgot gate. Experiments on real world datasets show that the proposed models outperform state-of-the-art baselines, including LSTM and its variants. Besides, the experimental results also demonstrate the effectiveness of the proposed methods when handing sparse data.

In the future, we will try utilizing attention mechanism to extract key features and their relevance from sequences. Besides, it is general agreed that users' personalized interests play an important role in recommendations. Therefore, we will try enhancing the model's ability of adapting to users with different preferences. In addition, we will also consider incorporating content information such as text and description to further improve the performance of sequential recommendation.

## References

1. Dakhel, A.M., Malazi, H.T., Mahdavi, M.: A social recommender system using item asymmetric correlation. Applied Intelligence **48**(3), 527–540 (2018)
2. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. The Journal of Machine Learning Research **12**, 2121–2159 (2011)
3. Eppler, M.J., Mengis, J.: The concept of information overload: A review of literature from organization science, accounting, marketing, mis, and related disciplines. The information society **20**(5), 325–344 (2004)
4. Fu, C.: User correlation model for question recommendation in community question answering. Applied Intelligence **50**(2), 634–645 (2020)
5. Garcin, F.F., Dimitrakakis, C., Faltings, B.: Personalized news recommendation with context trees. In: 7th ACM Recommender Systems Conference (Recsys 2013), CONF (2013)
6. Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R., Schmidhuber, J.: Lstm: A search space odyssey. IEEE transactions on neural networks and learning systems **28**(10), 2222–2232 (2016)
7. Guan, Y., Wei, Q., Chen, G.: Deep learning based personalized recommendation with multi-view information integration. Decision Support Systems **118**, 58–69 (2019)
8. Gupta, S., Varma, V.: Scientific article recommendation by using distributed representations of text and graph. In: Proceedings of the 26th International Conference on World Wide Web Companion, pp. 1267–1268. International World Wide Web Conferences Steering Committee (2017)

9. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. In: 4th International Conference on Learning Representations, ICLR 2016 (2016)
10. Hidasi, B., Tikk, D.: General factorization framework for context-aware recommendations. Data Mining and Knowledge Discovery **30**(2), 342–371 (2016)
11. Hu, L., Chen, Q., Zhao, H., Jian, S., Cao, L., Cao, J.: Neural cross-session filtering: Next-item prediction under intra-and inter-session context. IEEE Intelligent Systems **33**(6), 57–67 (2018)
12. Linden, G., Smith, B., York, J.: Amazon. com recommendations: Item-to-item collaborative filtering. IEEE Internet computing **7**(1), 76–80 (2003)
13. Ma, H., Zhou, D., Liu, C., Lyu, M.R., King, I.: Recommender systems with social regularization. In: Proceedings of the fourth ACM international conference on Web search and data mining, pp. 287–296. ACM (2011)
14. Nisha, C., Mohan, A.: A social recommender system using deep architecture and network embedding. Applied Intelligence **49**(5), 1937–1953 (2019)
15. Pan, W., Ming, Z.: Collaborative recommendation with multiclass preference context. IEEE Intelligent Systems **32**(2), 45–51 (2017)
16. Pazzani, M.J., Billsus, D.: Content-based recommendation systems. In: The Adaptive Web: Methods and Strategies of Web Personalization, pp. 325–341. Springer Berlin Heidelberg (2007)
17. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532–1543 (2014)
18. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: Bpr: Bayesian personalized ranking from implicit feedback. In: Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence, pp. 452–461. AUAI Press (2009)
19. Rendle, S., Freudenthaler, C., Schmidt-Thieme, L.: Factorizing personalized markov chains for next-basket recommendation. In: Proceedings of the 19th international conference on World wide web, pp. 811–820. ACM (2010)
20. Shi, H., Chen, L., Xu, Z., Lyu, D.: Personalized location recommendation using mobile phone usage information. Applied Intelligence **49**(10), 3694–3707 (2019)
21. Twardowski, B.: Modelling contextual information in session-aware recommender systems with neural networks. In: Proceedings of the 10th ACM Conference on Recommender Systems, pp. 273–276. ACM (2016)
22. Wang, D., Deng, S., Xu, G.: Sequence-based context-aware music recommendation. Information Retrieval Journal **21**(2-3), 230–252 (2018)
23. Wang, D., Deng, S., Zhang, X., Xu, G.: Learning to embed music and metadata for context-aware music recommendation. World Wide Web **21**(5), 1399–1423 (2018)
24. Wang, D., Zhang, X., Yu, D., Xu, G., Deng, S.: Came: Content-and context-aware music embedding for recommendation. IEEE Transactions on Neural Networks and Learning Systems (2020)
25. Wang, P., Guo, J., Lan, Y., Xu, J., Wan, S., Cheng, X.: Learning hierarchical representation model for nextbasket recommendation. In: Proceedings of the 38th International ACM SIGIR conference on Research and Development in Information Retrieval, pp. 403–412. ACM (2015)
26. Wang, S., Hu, L., Cao, L., Huang, X., Lian, D., Liu, W.: Attention-based transactional context embedding for next-item recommendation. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
27. Wang, S., Hu, L., Wang, Y., Sheng, Q.Z., Orgun, M., Cao, L.: Modeling multi-purpose sessions for nextitem recommendations via mixture-channel purpose routing networks. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence, pp. 1–7. AAAI Press (2019)
28. Xiang, L., Yuan, Q., Zhao, S., Chen, L., Zhang, X., Yang, Q., Sun, J.: Temporal recommendation on graphs via long-and short-term preference fusion. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 723–732 (2010)
29. Xiao, T., Shen, H.: Neural variational matrix factorization for collaborative filtering in recommendation systems. Applied Intelligence **49**(10), 3558–3569 (2019)

30. Xing, S., Wang, Q., Zhao, X., Li, T., et al.: Content-aware point-of-interest recommendation based on convolutional neural network. Applied Intelligence **49**(3), 858–871 (2019)
31. Yap, G.E., Li, X.L., Philip, S.Y.: Effective next-items recommendation via personalized sequential pattern mining. In: International conference on database systems for advanced applications, pp. 48–64. Springer (2012)
32. Yu, D., Xu, K., Wang, D., Yu, T., Li, W.: Point-of-interest recommendation based on user contextual behavior semantics. International Journal of Software Engineering and Knowledge Engineering **29**(11n12), 1781–1799 (2019)
33. Zhang, Y., Dai, H., Xu, C., Feng, J., Wang, T., Bian, J., Wang, B., Liu, T.Y.: Sequential click prediction for sponsored search with recurrent neural networks. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, AAAI'14, pp. 1369–1375. AAAI Press (2014)
34. Zhu, Y., Li, H., Liao, Y., Wang, B., Guan, Z., Liu, H., Cai, D.: What to do next: modeling user behaviors by time-lstm. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence, pp. 3602–3608. AAAI Press (2017)