



Time-aware sequence model for next-item recommendation

Dongjing Wang¹ · Dengwei Xu¹ · Dongjin Yu¹ · Guandong Xu²

Published online: 8 September 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract

The sequences of users' behaviors generally indicate their preferences, and they can be used to improve next-item prediction in sequential recommendation. Unfortunately, users' behaviors may change over time, making it difficult to capture users' dynamic preferences directly from recent sequences of behaviors. Traditional methods such as Markov Chains (MC), Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks only consider the relative order of items in a sequence and ignore 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 with Embedding layer and Coupled input and forget gate (IDLSTM-EC), which leverages time interval and duration information to accurately capture users' long-term and short-term preferences. In particular, the model incorporates global context information about sequences in the input layer to make better use of long-term memory. Furthermore, the model introduces the coupled input and forget gate and embedding layer to further improve efficiency and effectiveness. Experiments on real-world datasets show that the proposed approaches outperform the state-of-the-art baselines and can handle the problem of data sparsity effectively.

Keywords Recommendation · Sequence Modeling · Time-Aware · Long Short-Term Memory

1 Introduction

Nowadays, people are influenced to a large extent by the massive (and overwhelming) quantity of information that

has become available due to rapid development of the Internet and Information Technology (IT), which is known as the information overload problem [3]. Consequently, it is becoming increasingly difficult for users 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, 33], questions [4] and music [23, 24]. Existing recommendation methods include collaborative filtering-based recommendations [12, 30], content-based recommendations [16], social network-based recommendations [1, 13] and hybrid recommendation [7].

For many real-world applications, such as listening to music and game playing, users usually perform a series of actions within a period of time, forming behavior sequences. Such behavior sequences can be used to discover users' sequential patterns and to predict users' next new action (or item), which is called sequential recommendation, one of the typical applications of recommender systems. Traditional sequential recommendation models are mainly based on sequential pattern mining [32] and Markov Chain (MC) models [5]. The advent of deep learning has significantly boosted the performance of sequential recommendation [22, 31]. For example, Recurrent Neural Network (RNN) has

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 by Hangzhou Dianzi University under No. GK199900299012-017.

✉ Dongjin Yu
yudj@hdu.edu.cn

Dongjing Wang
dongjing.wang@hdu.edu.cn

Dengwei Xu
171050050@hdu.edu.cn

Guandong Xu
Guandong.xu@uts.edu.au

¹ School of Computer Science and Technology,
Hangzhou Dianzi University, Hangzhou, China

² Advanced Analytics Institute, University of Technology,
Sydney, Australia

been successfully applied in sequence modeling and next-item prediction/recommendation [9]. Furthermore, as a variant of RNN, Long Short-Term Memory (LSTM) network solves the problem of gradient disappearance in RNN and it provides better recommendation results. However, RNN and LSTM only focus on relative order information relating to items in sequence, and they ignore some important sequential information. In particular, items (or actions) that are close to one another in behavior sequence have strong correlations (although, in reality, this rule is not always true in sequential recommendation because users may have some haphazard behaviors which do not indicate their actual preferences).

In this paper, we propose to model the sequences of item information, interval information and duration information in users' behavior sequences in order to predict their next new items (or actions). Specifically, the next item refers to a new item that appears in other users' historical records, rather than the behavior sequence of target users. 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 more difficult) to recommend items that users may be interested in but which they have not yet interacted with, which are called new items for users. The scenario of this work is shown in Fig. 1. The length of time (time interval) between two adjacent items indicates the correlation between them. In other words, time information, in addition to order information, provides rich context for modeling users' dynamic preferences. Furthermore, 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 a very long time (duration).

In order to make better use of sequential patterns as well as time information, we present a novel recommendation model, namely the Interval- and Duration-aware LSTM

with Embedding layer and Coupled input and forget gate (IDLSTM-EC). Specifically, an interval gate and a duration gate are firstly introduced to preserve users' short-term and long-term preferences. Time information from the two gates is then seamlessly combined to improve next-new-item recommendation. An embedding layer is used after the input layer to incorporate more important information, such as the global context and long-term memory. The main contributions of this paper are listed as follows:

- We propose a novel next-item recommendation model, which can make better use of important time information (such as interval and duration) in time sequences;
- We further improve the model's effectiveness and efficiency by adding the embedding layer and coupled input and forget gate;
- Experimental results on real-world datasets show that the proposed model outperforms the state-of-the-art baselines and can handle the problem of data sparsity effectively.

The rest of this paper is structured as follows: related works are introduced in Section 2. We then illustrate the motivation for 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 future work.

2 Related works

2.1 Traditional sequence models

Traditional sequential models can be further divided into sequential pattern mining [32] and Markov Chain (MC) models [5]. Recommender systems based on sequential pattern mining firstly mine frequent patterns of 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

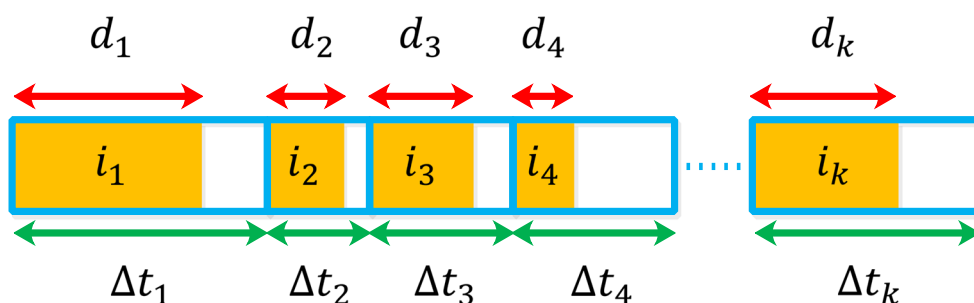


Fig. 1 Sequential behaviors with intervals and durations. i_k represents the k^{th} item in the sequence; Δt_k represents the interval between i_k and i_{k+1} ; and d_k represents the duration of i_k

performance, especially in terms of 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 MC model assumes that the current user's interaction depends on one or more recent interaction events. Therefore, it can only capture local information on the sequence, and it ignores global information relating to the sequence. Rendle et al. proposed a 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 MC 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 representations of users or items, which contain 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 the 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 network 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 users' 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 the factorization machine to construct a recommendation model, which effectively reduces model parameters and improves the recommendation performance. Yu et al. [33] used information based on users' context behavior semantics with the Point-of-Interest (POI) recommendation model to solve the data sparsity problem. However, sequential recommendation methods based on factorization are easily affected by 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 space through a new coding method. The embedding model is used in many fields, such as word2vec and GloVe (global vectors) [17]. Among these, the vector obtained by embedding the model is usually used for the input of 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. 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 has been that based on Recurrent Neural Networks (RNN). These are well suited for modeling complex dynamics in sequences due to their special structure [11, 21, 35]. Zhang et al. [35] proposed a novel framework based on RNN 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 Network (MCPRN) and improved the accuracy and diversity of recommendations. Yu et al. [34] proposed a new sequential recommendation model, namely SLi-Rec, by combining the traditional RNN structure and matrix factorization techniques. However, SLi-Rec does not incorporate the long-term interests of users in the neural network model. Wu et al. [28] proposed a long- and short-term preference learning model (LSPL) that considers both long-term and short-term interests. Specifically, LSPL uses LSTM to capture sequential patterns and learn sequence context information. We have compared traditional methods in details, and their strengths and weaknesses are listed in Table 1.

3 Data analysis and motivation

Some studies [28, 36] play an important role in user preferences modeling and can effectively improve the recommendation performance. In this section, we will introduce the time information in the experimental data and further analyze the time information. We will present a case using a game dataset to explain the role of time interval and duration information, and we will then present some samples of game-playing data in Table 2. For example, **Record 1** in Table 2 indicates that user ID No.386576 played "League of Legends" on 2016-09-01 at 01:00:47 (timestamp) for 8700 seconds (duration).

The interval represents the difference between two adjacent timestamp records for the same user. For example, the interval between **Record 2** and **Record 1** in Table 2 is 63878 seconds. Zhu et al. [37] showed that the shorter the interval, the greater the impact of the current item on the next item. One reason for this is that users may

Table 1 Strengths and weaknesses of traditional methods

Methods	Strengths	Weaknesses
Sequential pattern mining	1. effective for fixed sequential patterns 2. very efficient;	1. losing some infrequent but important modes 2. low coverage or diversity of recommendation results
MC models	Capturing local information on the sequence very well	ignoring global information on the sequence
Factorization machine	Good representation of inherent characteristics of the event	Easily affected by data sparsity
Embedding model	Reducing the dimension of event representation and suitable for many tasks	Cannot model features of the sequence adaptively
RNN	Well suited for modeling complex dynamics in sequences	1. relies on the sequence information 2. cannot preserve the user's interest for long

repeatedly play similar games for a short period of time, which represents their short-term preferences. For example, user ID No. 386576 frequently plays “League of Legends” for a short period of time. Furthermore, the information in Fig. 2a shows that the proportion of adjacent games are different in the sequence increases overall when the time interval is longer. In other words, a longer interval indicates low correlation between adjacent items, which also influences the modeling of users’ preferences.

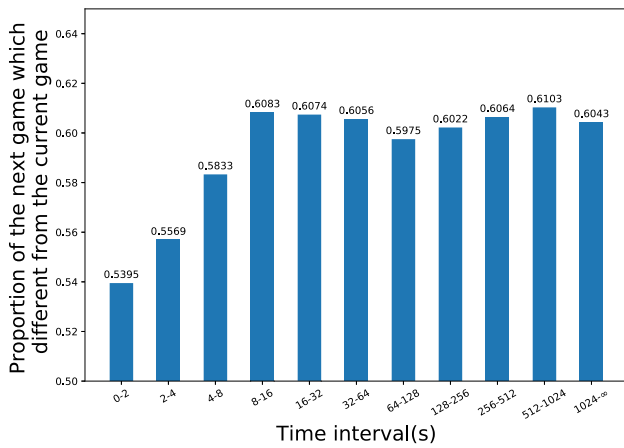
Furthermore, duration is also an important feature in sequence modeling and sequential prediction/recommendation. As shown in Table 2, the duration indicates how long users play a game for. Generally, duration can reflect the degree of users’ preferences for corresponding items. When a user plays a game for a longer duration, he/she will play the game more frequently; 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”) have longer

durations. As shown in Fig. 2b, our data analysis further illustrates the relationship between the average duration of the game and the frequency with which the game is played. The results show that a longer duration indicates that users are interested in corresponding games and tend to play them more frequently and spend more time playing them (duration). In other words, users’ preferences for the game are reflected in the duration.

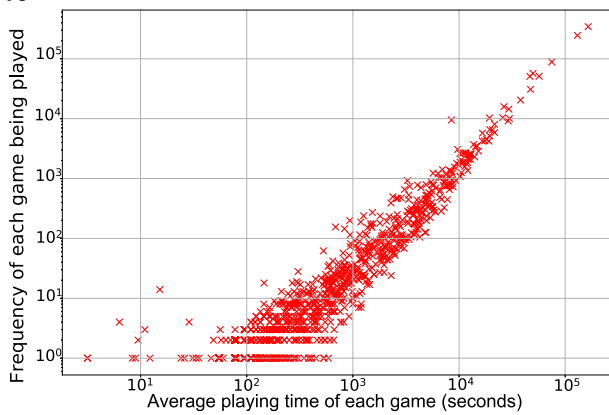
In general, users have both long-term and short-term preferences [29]. Specifically, long-term interest refers to users’ long-term and static interest. For example, some users only like role-playing games, so they may play this type of game most of the time. However, users’ preferences can change over time, and the next item or action is more likely to depend on users’ recent behaviors (called short-term interest). For example, although some users predominantly like role-playing games, they may also try popular strategy games. In order to better capture users’ long-term and short-term interests, we need to make better use of

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



(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

both time interval information and duration information. Therefore, we propose a novel recommendation model which incorporates interval and duration information into next-new-item recommendation.

4 The proposed method (IDLSTM-EC)

The time information in this work includes both time interval and duration. Specifically, the interval indicates the correlation between the current item and next item in the sequence, and the duration indicates 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-item recommendation method, namely Interval- and Duration-aware LSTM with Embedding layer and Coupled input and output gate (IDLSTM-EC).

Figure 3 shows how the proposed model performs prediction and recommendation based on users’ sequences. In the process, we firstly extract the interval Δt , duration d and

x from the sequence $\{game_1, game_2, \dots, game_k\}$, where x is the one-hot vector of the $game$. We then 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 RNN or LSTM, 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 model and LSTM are shown in Fig. 4. Specifically, in order to illustrate the advantages of the proposed model, we use different colored lines to highlight the improvements made. In addition, parameters that appear in Section 4 are explained in Table 3. Next, we will describe the IDLSTM-EC model in detail.

As shown in Fig. 4b, the IDLSTM-EC 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 on the basis of time interval information, while the duration gate is used to model users’ long-term interest in various items on the basis of duration information.

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

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

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

Furthermore, the interval gate I_k and the duration gate D_k are added to the LSTM in Fig. 4a, which is defined as follows:

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

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

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

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

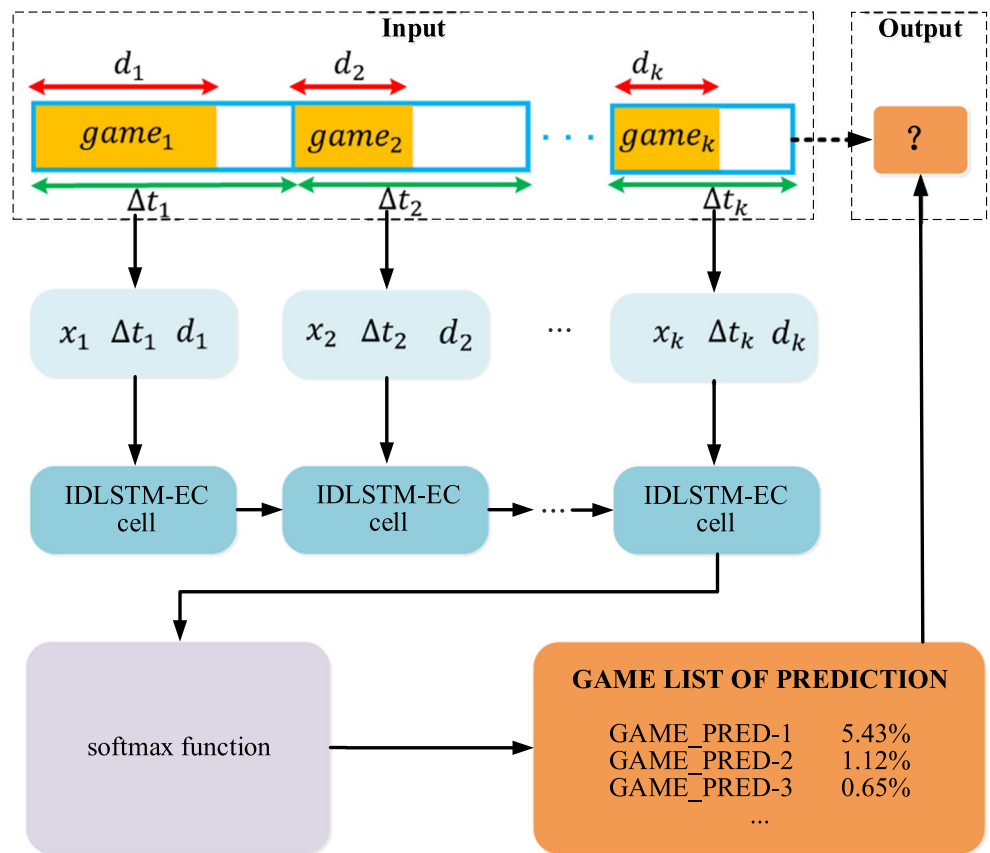
$$\tilde{c}_k = \hat{c}_k + I_k \circ c_k, \tag{7}$$

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

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

Input from 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. We then add the interval gate I_k to \tilde{c}_k so that \tilde{c}_k incorporates information from both the interval gate I_k and the duration gate D_k .

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



The cell \hat{c}_k is used to further model the user's interest by adding information from the duration gate D_k . We also add cell \tilde{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 significant influence on the next item. Correspondingly, \hat{c}_{k-1} will be relatively small, and the next item is influenced to an even greater extent. In this way, the IDLSTM-EC can combine duration and interval information to perform a more precise recommendation. On the other hand, \tilde{c}_k is directly connected to the output gate and is used to control the output, together with the output gate. In addition, Δt 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. The IDLSTM-EC combines information on time interval and duration well, and use of the interval gate and duration gate enables the two kinds of time information to be preserved for a longer period of time.

In order to further improve the effectiveness and efficiency of the proposed model, the model introduces the embedding layer to utilize more sequence information and the coupled input and forget gate.

- **Adding the embedding layer:** In the IDLSTM-EC model, all inputs are converted into one-hot vectors,

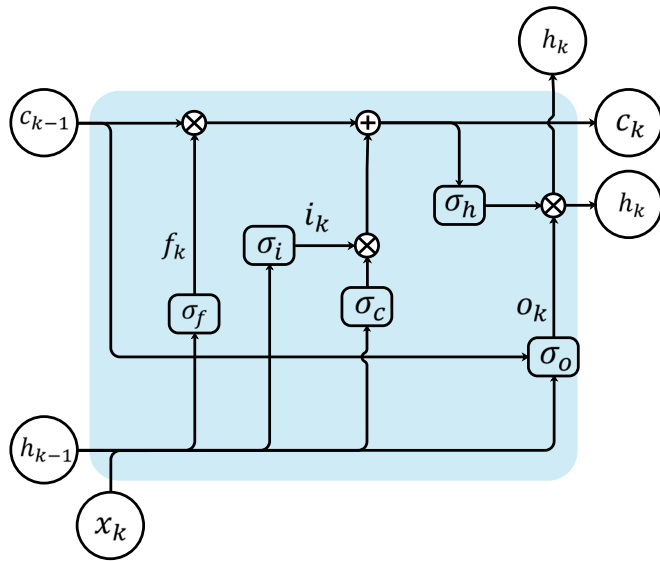
which may result in some important information being lost, such as the correlation between different items. In fact, the co-occurrence and context relationships between the inputs play important roles in sequential recommendation. However, the IDLSTM-EC only employs part of the context information but fails to utilize the global context. In order to incorporate more contextual 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 GloVe [17] method is used to train the embedding vector. The GloVe model is a popular embedding method, which obtains vectors through unsupervised learning. Unlike other embedding methods, the GloVe model incorporates global information and context to capture more important information.

- **coupling input and forget gates:** The parameters of the proposed model are reduced by the coupled input and forget gate. Thus, (4) will be removed and (6) is modified as follows:

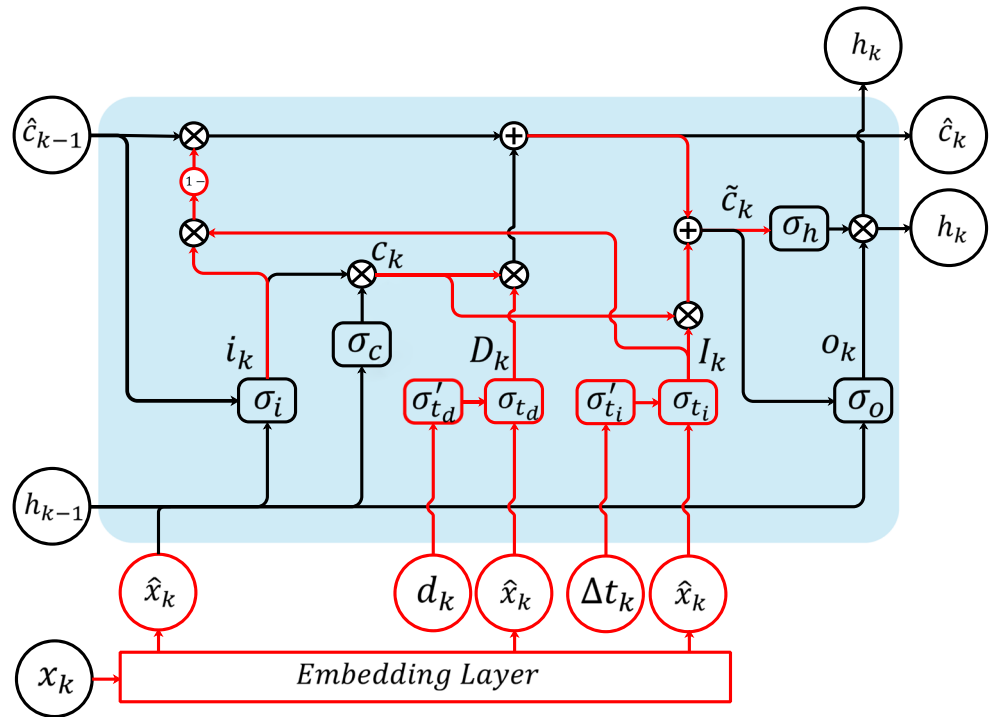
$$\hat{c}_k = (1 - I_k \circ i_k) \circ \hat{c}_{k-1} + D_k \circ c_k. \tag{10}$$

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

Fig. 4 Architectures of models: **a** LSTM and **b** IDLSTM-EC. The IDLSTM-EC has a time gate I_k and a duration gate D_k . Specifically, I_k is designed to model time interval Δt in order to gauge the impact of the current event on the next event, and D_k is designed to model duration d to indicate users' interests. Furthermore, the IDLSTM-EC uses the coupled input and forget gate, and input x has been converted to \hat{x} by the embedding layer. We use red lines to highlight the improved features of the IDLSTM-EC in comparison with traditional LSTM



(a) LSTM



(b) IDLSTM-EC (the proposed model)

$(1 - I_k \circ i_k)$ in \hat{c}_k . The IDLSTM-EC with coupled input and forget gate increases the model's efficiency by reducing the model parameters. At the same time, reduction of the parameters prevents the model from overfitting to some extent.

5 Experiments

In this section, we will evaluate the proposed model as well as state-of-the-art baselines on two real-world datasets. The first dataset comprises game-playing records collected from

Table 3 Parameter description

Parameter	Description
x_k	one-hot vector of $game_k$
\hat{x}_k	embedding of $game_k$
Δt_k	time interval between $game_k$ and $game_{k+1}$
d_k	duration of $game_k$
I_k	time interval gate which incorporates time interval information
W_{t_i}	weight of input x_k in I_k
S_{t_i}	weight of interval Δt_k
b_{t_i}	bias of interval gate I_k
D_k	duration gate which incorporates time duration information
W_{t_d}	weight of input x_k in D_k
S_{t_d}	weight of duration gate 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 that controls the extent to which new information flows into c_k
f_k	forget gate vector that controls the extent to which old information remains in c_k
o_k	output gate vector which controls the output information from \tilde{c}_k to h_k
c_k	cell state vector with history information
\hat{c}_k	new cell state vector which captures time duration information
\tilde{c}_k	new cell state vector which captures time interval information
h_k	hidden state vector, which is also the output vector of the IDLSTM-EC unit
W_i	weight of input gate i_k
W_f	weight of forget gate f_k
W_c	weight of cell state vector c_k
W_o	weight of output gate o_k
U_i	weight of hidden state vector h_k for input gate i_k
U_f	weight of hidden state vector h_k for forget gate f_k
U_o	weight of hidden state vector h_k for output gate o_k
b_i	bias of input gate i_k
b_f	bias of forget gate f_k
b_o	bias of output gate o_k
b_c	bias of cell state vector c_k
$\sigma_i, \sigma_f, \sigma_o$	sigmoid function
σ_c, σ_h	hyperbolic tangent (tanh) function
\circ	element-wise product (Hadamard product) operator

the world-leading internet bar which has the largest number of game players in China. The second one is a public music listening dataset: LastFM-1K¹, which includes all the music listening sequences and timestamps of nearly 1,000 listeners up to May 5, 2009. We preprocessed the two datasets and deleted users and items with only a few records. Statistical information on the final datasets is shown in Table 4, where #(*) indicates the number of *, and Average (Item) indicates the average number of interactions for all users.

5.1 Compared methods

In this section, the proposed model is compared with state-of-the-art recommendation methods, including traditional

Table 4 Statistics for the 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

¹<http://ocelma.net/MusicRecommendationDataset/lastfm-1K.html>

recommendation methods and the variants of LSTM mentioned above.

5.1.1 Baselines

Two kinds of recommendation methods have been adopted 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. We also compare different versions of the proposed model 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:

- **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 a "peephole connection" to allow all gates to accept input from the state [6].
- **Peephole-LSTM with time**: This model adds time information to the Peephole-LSTM for a fair comparison.
- **Time-LSTM**: A variant of LSTM that adds two time gates to the traditional LSTM [37].

5.1.2 The proposed methods²

As three variants of the proposed IDLSTM-EC model, IDLSTM-E, IDLSTM-C and IDLSTM are included in the ablation experiments. In particular, they are used as baselines to show the effectiveness of two key components in IDLSTM-EC (i.e., the embedding layer and the coupled input and forget gate). All methods are described as follows:

- **IDLSTM-EC**: Interval- and Duration-aware LSTM with Embedding layer and Coupled input and forget gate.
- **IDLSTM-E**: IDLSTM-EC model with only the embedding layer.
- **IDLSTM-C**: IDLSTM-EC model with only the coupled input and forget gate.
- **IDLSTM**: IDLSTM-EC model without the embedding layer and the coupled input and forget gate.

Specifically, IDLSTM-C and IDLSTM-E are used as baselines to evaluate the effectiveness of the embedding layer component and the coupled input and forget gate component in IDLSTM-EC, respectively. Besides, IDLSTM is used as a baseline to evaluate the effectiveness of combining the two key components in IDLSTM-EC.

5.2 Experimental setup

In the experiment, the task is to predict the next new item that users will be most likely to interact with according to existing behavior sequences. During the training phase, an improved stochastic gradient descent method called Adagrad [2] was used, which can 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, cross-entropy was chosen as the loss function, defined as follows:

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

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

5.3 Evaluation metrics

The proposed model was evaluated with two metrics, including Recall and MRR.

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

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

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 metric which indicates the average of the reciprocal ranks of the target items in a recommendation list. Formally, it is defined as follows:

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

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 a similar proportion of target items appearing in the recommendation list),

²<https://github.com/vallzey/IDLSTM>

we can use MRR to further evaluate them. In particular, $MRR@n$ is the same with $Recall@n$ when $n = 1$.

5.4 Comparison with baselines

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. Furthermore, models with a neural network structure perform better than models without considering sequence factors. Specifically, the performance of the IDLSTM-EC is 11.1% and 32.5% better than the best baseline method in terms of Recall@10 and MRR@10, respectively, on game datasets. Furthermore, the level of improvement achieved by the IDLSTM-EC on LastFM datasets is 65.8% (Recall@10) and 18.3% (MRR@10), respectively. Next, we will analyze the performance of each model in detail.

- **POP, UBCF, BPR and FPMC:** The POP method only recommends items with high popularity, which results in low coverage. UBCF and BPR ignore the

dependence of items in sequences and cannot model users' short-term preferences. However, in sequential recommendation, recent items usually play a large part in decision-making. The FPMC method achieves better performance than the POP and UBCF because it combines matrix factorization and Markov chains to model users' behavior sequences. However, FPMC has some limitations when it comes to retaining information in the sequence for a long time, and it does not fit well with long sequences.

- **Session-RNN:** Session-RNN mainly captures users' short-term interests but does not consider their long-term interests, which limits its performance. However, in sequential recommendation and users' long- and short-term preferences both play an important role 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 and achieves slightly better performance in most cases. However, adding time

Table 5 Recall and MRR of the proposed methods and baselines (best results are highlighted in bold)

	Recall@1	Recall@5	Recall@10	MRR@1	MRR@5	MRR@10
Game dataset						
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						
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

information directly to the input is not entirely effective. In addition, these two approaches cannot capture or preserve users' long-term preferences accurately.

- **Time-LSTM:** Time-LSTM incorporates time information into the sequence modeling process in a more effective way, so it achieves better performance than Peephole-LSTM and Peephole-LSTM with time. In particular, the lack of duration information in Time-LSTM decreases its ability to fully utilize time information or capture users' preferences accurately, which limits its performance.
- **IDLSTM and IDLSTM-E:** The 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 the IDLSTM-E is much better than that of the IDLSTM. The reason for this is that the proposed methods can utilize time interval and duration information with gate mechanisms effectively in order to perform better recommendation. Furthermore, the results also show that time interval and duration data are both important in sequence modeling, as well as for capturing users' long- and short-term preferences. The performance of the IDLSTM-E is better than that of the IDLSTM. The reason for this is that the embedding layer based on the GloVe method can effectively capture 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 the IDLSTM-C does not significantly improve upon the IDLSTM in terms of accuracy evaluation. Efficiency comparisons of each model's ability to run an epoch are listed in Table 6, which shows that the IDLSTM-C and IDLSTM-EC are approximately 6% faster than the IDLSTM and IDLSTM-E. Therefore, the efficiency of the proposed model is improved via the coupled input and forget gate, reducing the parameters that need to be trained. Traditional methods cannot achieve accurate results, although they require much less time due to their concise structure. In addition, in actual applications, recommendation models are generally pre-trained offline, so the comparison test time is more meaningful. Specifically, all recommendation methods can perform recommendation during the test phase within close and reasonable time.

In conclusion, traditional recommendation methods (such as UBCF and BPR) do not consider dynamic changes in user interests, which leads to poor results. Sequence-based methods (such as Session-RNN, Peephole-LSTM,

Table 6 Time taken for each model to run an epoch

	Training time (s/epoch)	Testing time (s)
Game dataset		
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		
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

Peephole-LSTM with time, Time-LSTM and IDLSTM(-EC)) achieve better performance than traditional recommendation methods due to the effectiveness of RNN when it comes to modeling users' behavior sequences. In particular, the proposed model can make better use of time interval and duration information, which is very important for sequence modeling and sequential prediction/recommendation. In addition, improvement of the IDLSTM-EC over the IDLSTM shows that the global information (related to users' long-term preferences) captured by IDLSTM-EC is quite important in sequential recommendation.

5.5 Effect of the number of units

In this subsection, we evaluate the influence of the number of cell units and number of embedding layer units using results from two experiments. In the first experiment, the effect of the number of cells was evaluated. The best number of units in the first experiment was then used in the second experiment to evaluate the effect of the number of embedding layer units.

5.5.1 Effect of the number of cell units

We firstly set the number of cell units to (16, 32, 64, 128, 256 and 1024), and then evaluated the impact of the number of cell units of the IDLSTM and IDLSTM-C in terms of Recall and MRR. As shown in Fig. 5, two models had similar performance in Recall@10 and MRR@10. In addition, we found that the IDLSTM-C takes less time than the IDLSTM as the number of cell units increases. Meanwhile, the promotion of Recall@10 and MRR@10 gradually stabilizes. In particular, once the number of cell units exceeds 128, the performance of Recall@10 and MRR@10 is not much improved. Thus, the optimum number of cell units is 128, and this enables the proposed model to capture most important information.

5.5.2 Effect of the number of embedding units

The effect of different numbers of embedding units was further investigated with the number of cell units set to 128, and the results are shown in Fig. 6. In particular, the results without an embedding layer are also added at 0-abscissa for comparison. Our results indicate that when the number of cells in the embedding layer is less than 32, the performance of the proposed model is lower than that without the embedding layer model. Therefore, it is necessary to have enough units in the embedding layer to ensure that sequence information is well preserved in the recommendation model.

As shown in Fig. 6c and f, although the time varies, the overall fluctuation is not large because the number of embedding layer unit 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.

Furthermore, as shown in Fig. 6a, b, d and e, 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 and can even result in a downward trend. The reason for this is that an excessive number of units may cause overfitting. Therefore, the number of embedding layer units was set as 128.

5.6 Impact of data sparsity

We also evaluated the proposed methods against baselines for datasets with different sparsity to verify their ability to deal with sparse data. Specifically, items with a frequency of less than d were removed from the dataset, where d was set to (0, 5, 10, 15, 20, 30, 40 and 50), respectively, and the sparsity of corresponding datasets was (97.04%, 94.50%, 93.38%, 92.65%, 91.87%, 90.73%, 90.15%, 89.61%) for the game dataset and (99.72%, 98.68%, 97.87%, 96.15%, 94.74%, 88.89%, 86.23%, 85.71%) for LastFM. As shown in Fig. 7, the performance of the various methods did not significantly decrease as the sparsity increased because our task was to recommend next new items that users might have been interested in but had not yet interacted with. In particular, some items with a low frequency were excluded to change the sparsity of the dataset, which may also have removed some key items or correlations. For example, if “C” was removed from the sequence, the next-new-item recommender system (our work) performed one prediction, $\{A, A\} \rightarrow “B”$. The performance decreased if “C” was

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

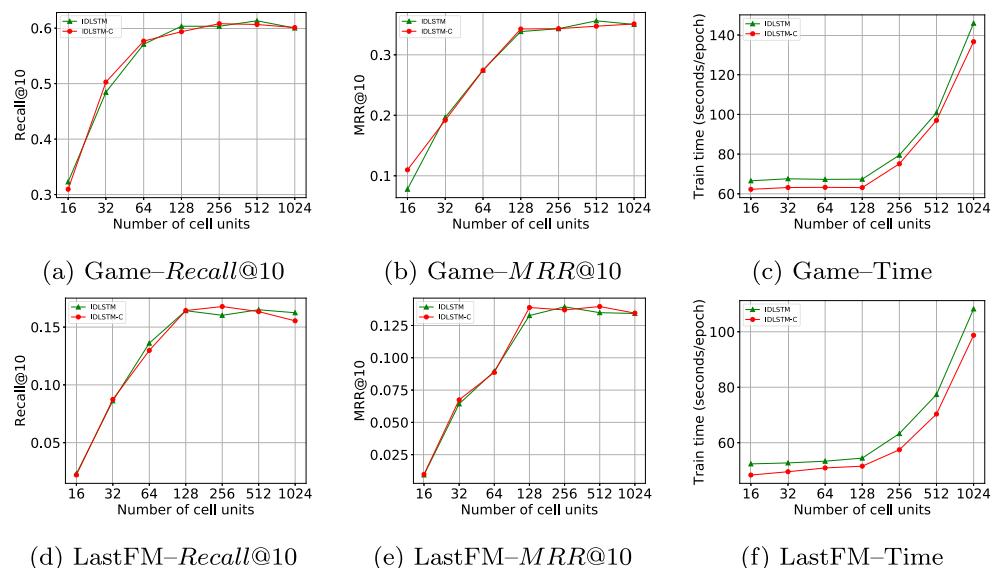


Fig. 6 The effect of different numbers of embedding units on *Recall@10*, *MRR@10* and time of an epoch

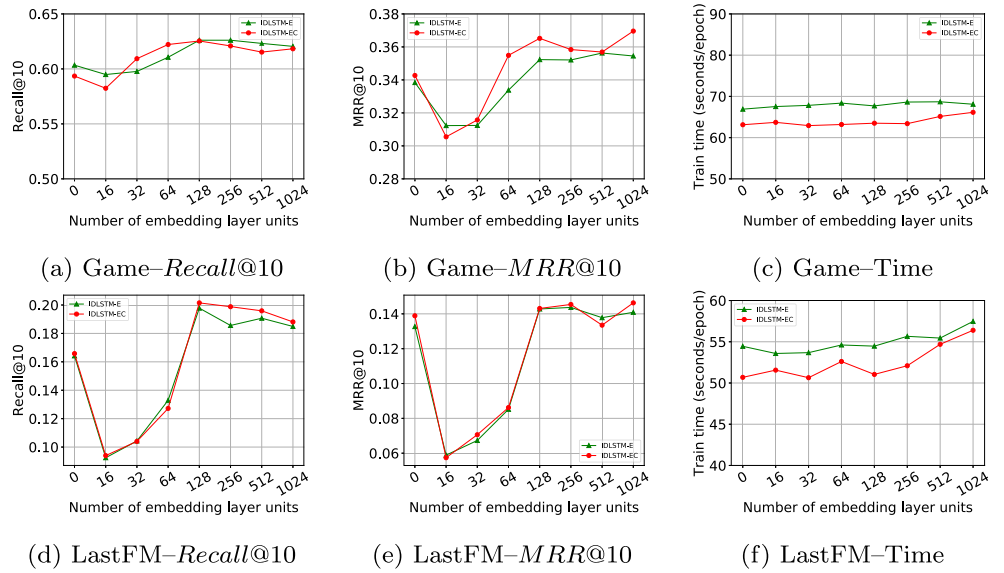
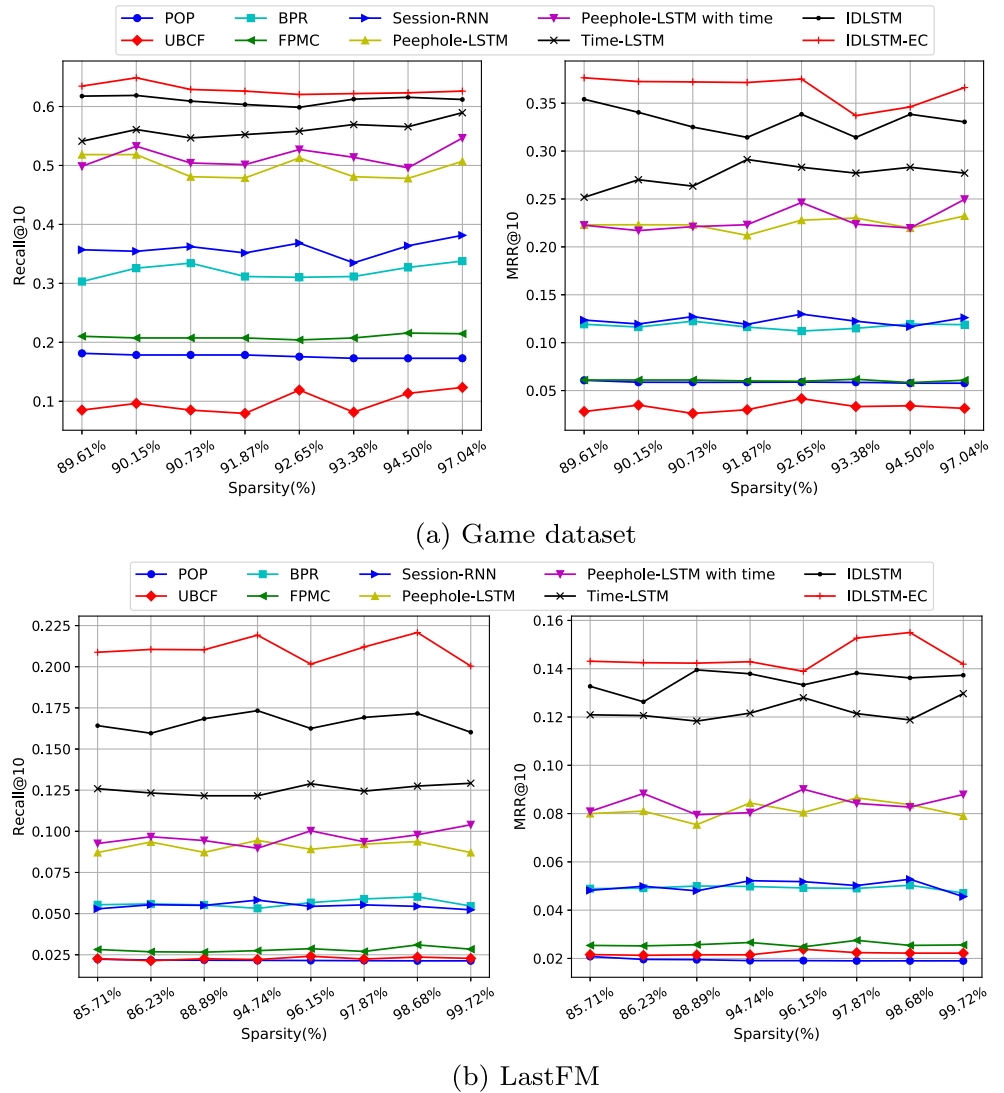


Fig. 7 Comparison of different models using data with different sparsity



the key item for prediction of “*B*”. But even so, we found that the proposed method performs better than baselines in terms of Recall@10 and MRR@10. Therefore, we conclude that our methods can deal with data with different sparsity effectively.

6 Conclusions

In this paper, we have proposed a novel time-aware sequence modeling method and have applied it to next-new-item recommendation. Specifically, the proposed method introduces two gates, i.e., a duration gate for modeling user preferences and an interval gate for modeling the impact of the current item on the next item in sequences. In addition, we adopted the GloVe method to take advantage of global context information and further improve its efficiency with a coupled input and forgot gate. Experiments on real-world datasets show that the proposed model outperforms state-of-the-art baselines, including LSTM and its variants. Furthermore, the experimental results also demonstrate the effectiveness of the proposed methods when handling sparse data.

In the future, we will try utilizing an attention mechanism to extract key features and their relevance from sequences. It is generally agreed that users’ personalized interests play an important role in recommendations. Therefore, we will try enhancing the model’s ability to adapt 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

- Dakhel AM, Malazi HT, Mahdavi M (2018) A social recommender system using item asymmetric correlation. *Appl Intell* 48(3):527–540
- Duchi J, Hazan E, Singer Y (2011) Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research* 12:2121–2159
- Eppler MJ, Mengis J (2004) 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
- Fu C (2020) User correlation model for question recommendation in community question answering. *Appl Intell* 50(2):634–645
- Garcin FF, Dimitrakakis C, Faltings B (2013) Personalized news recommendation with context trees. In: 7Th ACM recommender systems conference (recsys 2013), CONF
- Greff K, Srivastava RK, Koutník J, Steunebrink BR, Schmidhuber J (2016) Lstm.: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems* 28(10):2222–2232
- Guan Y, Wei Q, Chen G (2019) Deep learning based personalized recommendation with multi-view information integration. *Decis Support Syst* 118:58–69
- Gupta S, Varma V (2017) Scientific article recommendation by using distributed representations of text and graph. In: Proceedings of the 26th international conference on world wide Web companion, International World Wide Web Conferences Steering Committee, pp 1267–1268
- Hidasi B, Karatzoglou A, Baltrunas L, Tikk D (2016) Session-based recommendations with recurrent neural networks. In: 4Th international conference on learning representations, ICLR 2016
- Hidasi B, Tikk D (2016) General factorization framework for context-aware recommendations. *Data Min Knowl Disc* 30(2):342–371
- Hu L, Chen Q, Zhao H, Jian S, Cao L, Cao J (2018) Neural cross-session filtering: Next-item prediction under intra-and inter-session context. *IEEE Intell Syst* 33(6):57–67
- Linden G, Smith B, York J (2003) Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing* 7(1):76–80
- Ma H, Zhou D, Liu C, Lyu MR, King I (2011) Recommender systems with social regularization. In: Proceedings of the fourth ACM international conference on Web search and data mining, ACM, pp 287–296
- Nisha C, Mohan A (2019) A social recommender system using deep architecture and network embedding. *Appl Intell* 49(5):1937–1953
- Pan W, Ming Z (2017) Collaborative recommendation with multiclass preference context. *IEEE Intell Syst* 32(2):45–51
- Pazzani MJ, Billsus D (2007) Content-based recommendation systems. In: The adaptive web: methods and strategies of Web personalization. Springer, Berlin, pp 325–341
- Pennington J, Socher R, Manning C (2014) Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543
- Rendle S, Freudenthaler C, Gantner Z, Schmidt-Thieme L (2009) Bpr: Bayesian personalized ranking from implicit feedback. In: Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence, AUAI Press, pp 452–461
- Rendle S, Freudenthaler C, Schmidt-Thieme L (2010) Factorizing personalized markov chains for next-basket recommendation. In: Proceedings of the 19th international conference on World wide Web, ACM, pp 811–820
- Shi H, Chen L, Xu Z, Lyu D (2019) Personalized location recommendation using mobile phone usage information. *Appl Intell* 49(10):3694–3707
- Twardowski B (2016) Modelling contextual information in session-aware recommender systems with neural networks. In: Proceedings of the 10th ACM conference on recommender systems, ACM, pp 273–276
- Wang D, Deng S, Xu G (2018) Sequence-based context-aware music recommendation. *Information Retrieval Journal* 21(2-3):230–252
- Wang D, Deng S, Zhang X, Xu G (2018) Learning to embed music and metadata for context-aware music recommendation. *World Wide Web* 21(5):1399–1423
- Wang D, Zhang X, Yu D, Xu G, Deng S (2020) Came: Content- and context-aware music embedding for recommendation. *IEEE Transactions on Neural Networks and Learning Systems*
- Wang P, Guo J, Lan Y, Xu J, Wan S, Cheng X (2015) Learning hierarchical representation model for nextbasket recommendation. In: Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval, ACM, pp 403–412
- Wang S, Hu L, Cao L, Huang X, Lian D, Liu W (2018) Attention-based transactional context embedding for next-item

- recommendation. In: Thirty-second AAAI conference on artificial intelligence
27. Wang S, Hu L, Wang Y, Sheng QZ, Orgun M, Cao L (2019) Modeling multi-purpose sessions for nextitem recommendations via mixture-channel purpose routing networks. In: Proceedings of the 28th international joint conference on artificial intelligence, AAAI Press, pp 1–7
 28. Wu Y, Li K, Zhao G, Qian X (2019) Long-and short-term preference learning for next poi recommendation. In: Proceedings of the 28th ACM international conference on information and knowledge management, pp 2301–2304
 29. Xiang L, Yuan Q, Zhao S, Chen L, Zhang X, Yang Q, Sun J (2010) 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
 30. Xiao T, Shen H (2019) Neural variational matrix factorization for collaborative filtering in recommendation systems. *Appl Intell* 49(10):3558–3569
 31. Xing S, Wang Q, Zhao X, Li T et al (2019) Content-aware point-of-interest recommendation based on convolutional neural network. *Appl Intell* 49(3):858–871
 32. Yap GE, Li XL, Philip SY (2012) Effective next-items recommendation via personalized sequential pattern mining. In: International conference on database systems for advanced applications, Springer, pp 48–64
 33. Yu D, Xu K, Wang D, Yu T, Li W (2019) Point-of-interest recommendation based on user contextual behavior semantics. *Int J Softw Eng Knowl Eng* 29(11n12):1781–1799
 34. Yu Z, Lian J, Mahmood A, Liu G, Xie X (2019) Adaptive user modeling with long and short-term preferences for personalized recommendation. In: Proceedings of the 28th international joint conference on artificial intelligence, AAAI Press, pp 4213–4219
 35. Zhang Y, Dai H, Xu C, Feng J, Wang T, Bian J, Wang B, Liu TY (2014) Sequential click prediction for sponsored search with recurrent neural networks. In: Proceedings of the Twenty-Eighth AAAI conference on artificial intelligence, AAAI'14, AAAI Press, pp 1369–1375
 36. Zhao G, Liu Z, Chao Y, Qian X (2020) Caper: Context-aware personalized emoji recommendation. *IEEE Transactions on Knowledge and Data Engineering*
 37. Zhu Y, Li H, Liao Y, Wang B, Guan Z, Liu H, Cai D (2017) What to do next: modeling user behaviors by time-lstm. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence, AAAI Press, pp 3602–3608

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Dongjing Wang received the B.S. and Ph.D. degrees in computer science from Zhejiang University, Hangzhou, China, in 2012 and 2018, respectively. He is currently a Lecturer with Hangzhou Dianzi University, China. His current research interests include recommender systems, machine learning, and business process management.



Dengwei Xu received the M.S. degree in computer science from Hangzhou Dianzi University, Hangzhou, China, in 2020. His research interests include machine learning and information retrieval.



Dongjin Yu is currently a Professor with Hangzhou Dianzi University, China, where he is also the Director of the Institute of Big Data and the Institute of Computer Software. His research efforts include big data, business process management, and software engineering. He is a member of IEEE, a member of ACM, and a Senior Member of China Computer Federation (CCF). He is also a member of the Technical Committee of Software Engineering, CCF, and the Technical Committee of Service Computing, CCF.



Guandong Xu is a Full Professor in Data Science at School of Computer Science and Advanced Analytics Institute, University of Technology Sydney with PhD degree in Computer Science. His research interests cover Data Science, Data Analytics, Recommender Systems, Web Mining, User Modelling, NLP, Social Network Analysis, and Social Media Mining. He has published three monographs in Springer and CRC press, and 190+ journal and conference papers including ACM Transactions on Information Systems, ACM Transaction on Intelligent Systems and Technology, IEEE Transactions on Neural Networks and Learning Systems, IEEE Transactions on Services Computing, IEEE Transactions on Information Forensics and Security, IEEE Intelligent Systems, Information Sciences, IJCAI, AAAI, WWW, ICDM, ICDE, CVPR, and CIKM conferences. He is the assistant Editor-in-Chief of World Wide Web Journal and has been serving in editorial board or as guest editors for several international journals, such as Social Network Analysis and Mining, the Computer Journal, Journal of Systems and Software, World Wide Web Journal, Multimedia Tools and Applications, and Online Information Review. He has received a number of Industry Awards from Australian industry community, such as 2018 Top-10 Australian Analytics Leader Award.