Hindawi
Complexity
Volume 2020, Article ID 6136095, 13 pages
https://doi.org/10.1155/2020/6136095

Research Article

Attention with Long-Term Interval-Based Deep Sequential Learning for Recommendation

Zhao Li[1], **Long Zhang**[1], **Chenyi Lei**[1], **Xia Chen**[1], **Jianliang Gao**[2] and **Jun Gao**[3]

[1]*Alibaba Group, Hangzhou, China*
[2]*Central South University, Changsha, China*
[3]*School of EECS, Peking University, Beijing, China*

Correspondence should be addressed to Zhao Li; lizhao.lz@alibaba-inc.com

Received 19 May 2020; Revised 15 June 2020; Accepted 19 June 2020; Published 13 July 2020

Guest Editor: Shirui Pan

Modeling user behaviors as sequential learning provides key advantages in predicting future user actions, such as predicting the next product to purchase or the next song to listen to, for the purpose of personalized search and recommendation. Traditional methods for modeling sequential user behaviors usually depend on the premise of Markov processes, while recently recurrent neural networks (RNNs) have been adopted to leverage their power in modeling sequences. In this paper, we propose integrating attention mechanism into RNNs for better modeling sequential user behaviors. Specifically, we design a network featuring **A**ttention with **L**ong-term **I**nterval-based **G**ated **R**ecurrent **U**nits (ALI-GRU) to model temporal sequences of user actions. Compared to previous works, our network can exploit the information of temporal dimension extracted by time interval-based GRU in addition to normal GRU to encoding user actions and has a specially designed matrix-form attention function to characterize both long-term preferences and short-term intents of users, while the attention-weighted features are finally decoded to predict the next user action. We have performed experiments on two well-known public datasets as well as a huge dataset built from real-world data of one of the largest online shopping websites. Experimental results show that the proposed ALI-GRU achieves significant improvement compared to state-of-the-art RNN-based methods. ALI-GRU is also adopted in a real-world application and results of the online A/B test further demonstrate its practical value.

1. Introduction

Due to the increasing abundance of information on the Web, helping users filter information according to their preferences is more and more required, which emphasizes the importance of personalized search and recommendation [42–45]. Traditional methods for providing personalized content, such as item-item collaborative filtering [33], did not take into account the dynamics of user behaviors, which are recently recognized as important factors. For example, to predict the user's next action such as the next product to purchase, the profiling of both long-term preferences and short-term intents of user is required, where modeling the user's behaviors as sequences provides key advantages. Nonetheless, modeling sequential user behaviors with temporal dimension raises even more challenges than modeling them without the temporal dimension. How to identify the correlation and dependence among actions is one of the difficult issues. This problem has been studied extensively, and many methods based on the Markov process assumption, such as Factorizing Personalized Markov Chain [32] and Hierarchical Representation Model [41], have been designed and adopted in different tasks [11, 15]. These methods usually focus on factor models, which decompose the sparse user-item interaction matrix into low-dimensional matrices with latent factors. However, for modeling sequential information, it is often not clear how to integrate the dynamics of user intents into the framework of factor models.

Recently, neural-network-based algorithms have received much attention from researchers [6, 9, 18, 30, 48]. For instance, many different kinds of graph neural networks

(GNNs), instead of matrix factorization (MF) based algorithms [36], have been proposed to learn graph embedding due to their ability of learning from non-Euclidean space [30]. Many different kinds of recurrent neural networks (RNNs) have been proposed to model user behaviors due to their powerful descriptive ability for sequential data [14, 29, 39, 47, 52]. For example, Hidasi et al. [14] propose an approach based on a number of parallel RNNs with rich features to model sequential user behaviors. Wu et al. [47] endow both users and movies with a long short-term memory (LSTM) autoregressive model to predict user future behaviors. Furthermore, to utilize the temporal information better, Zhu et al. [52], Neil et al. [29], and Vassøy et al. [39] introduce time intervals between sequential actions into RNN cells to update and forget information rather than only considering the order of actions.

Despite the success of abovementioned RNN-based methods, there are several limitations that make it difficult to apply these methods into the wide variety of applications in the real world. One inherent assumption of these methods is that the importance of historical behaviors decreases over time (e.g., equation (15) in [52]), which is also the intrinsic property of RNN cells such as gated recurrent units (GRU) and long- and short-term memory (LSTM). However, this assumption does not always apply in practice, where the sequences may have complex cross-dependence [46]. For example, user's online actions are not straightforward but contain much noise and randomness. See Figure 1 for illustration, which shows a real sequence of clicked items of a user in one of the largest online shopping websites. We can conjecture the user intended to buy a T-shirt in honor of Lebron James and he/she finally bought $Item_6$. But the user also viewed items of different kind (shoes as $Item_3$ and $Item_4$). Clearly, $Item_1$ and $Item_2$ are more important than $Item_3$ and $Item_4$ to predict the final deal, although the former two items are earlier than the latter two items in the temporal dimension. This example displays the difficulty in analyzing sequential user behaviors, where the simple assumption of time interval-based correlation between actions is not enough to cope with.

In this paper, we are inspired by the attention mechanism proposed for natural language processing [1, 49] which achieves remarkable progress in the past few years. Attention mechanism introduced into deep networks provides the functionality to focus on portions of input data or features to fulfill the given task. Similarly, we expect that a trained attention mechanism helps to identify the important correlated actions from sequential user behaviors to make prediction. However, the existing attention mechanism is inefficient in modeling sequential user behaviors. Hence, we design a new attention mechanism specifically for our purpose.

Specifically, we propose a network featuring **A**ttention with **L**ong-term **I**nterval-based **G**ated **R**ecurrent **U**nits (ALI-GRU) for modeling sequential user behaviors to predict user's next action. The network is depicted in Figure 2. We adopt a series of bidirectional GRU to process the sequence of items that user had accessed. The GRU cells in our network consist of not only normal GRU but also time interval-based GRU, where the latter reflects the short-term information of time intervals. In addition, the features extracted by bidirectional GRU are used as the input of attention model, and the attention distribution is calculated at each timestamp rather than as single vector as in Seq2Seq model [1, 46]. Therefore, this attention mechanism is able to consider the long-term correlation along with short-term intervals. Our designed attention mechanism is detailed in Section 4.

We have performed a series of experiments using both well-known public datasets (LastFM and CiteULike [52]) and the dataset collected by ourselves, built from real-world data. Extensive results show that our proposed ALI-GRU outperforms the state-of-the-art methods by a significant margin on these datasets. Moreover, ALI-GRU is adopted online and we have performed online A/B test; test results further demonstrate the practical value of ALI-GRU in comparison with the well-optimized baseline in a real-world e-commerce search engine.

This paper makes the following contributions:

(i) First, we propose a bidirectional time interval-based GRU to model the long- and short-term information of user actions for better capturing temporal dynamics between actions. Time interval-based GRU is able to effectively extract short-term dynamics of user intents as driven signals of attention function and refine the long-term memory by contextual information.

(ii) Second, we design a new attention mechanism to encode long- and short-term information and identify complex correlation between actions, which attends to the driven signals at each time step along with the embedding of contextual information. This mechanism is less affected by the noise in the historical actions and is robust to extract the important correlated information between sequential user behaviors to make a better prediction.

(iii) Third, we conduct a series of experiments on two well-known public datasets and a large-scale dataset constructed from a real-world e-commerce platform. Extensive experimental results show that our proposed ALI-GRU obtains significant improvement compared to state-of-the-art RNN methods. In addition, ALI-GRU is adopted and we conducted online A/B test, and the results further demonstrate the practical value in a real-world e-commerce search engine.

The remainder of this paper is organized as follows. Section 2 discusses related works. The problem of modeling sequential user behaviors is formulated in Section 3, followed by detailed description of our proposed ALI-GRU in Section 4. Experimental results are reported in Section 5 and concluding remarks in Section 6.

2. Related Work

We give brief overview of related work at two aspects, modeling of sequential user behaviors and attention mechanism.

FIGURE 1: An example of sequential actions of a user in one of the largest online e-commerce platforms. The user clicked several items sequentially and finally purchased $Item_6$. The red frame around each item indicates the corresponding (estimated) contribution of that item to the final deal; darker color stands for more contribution.

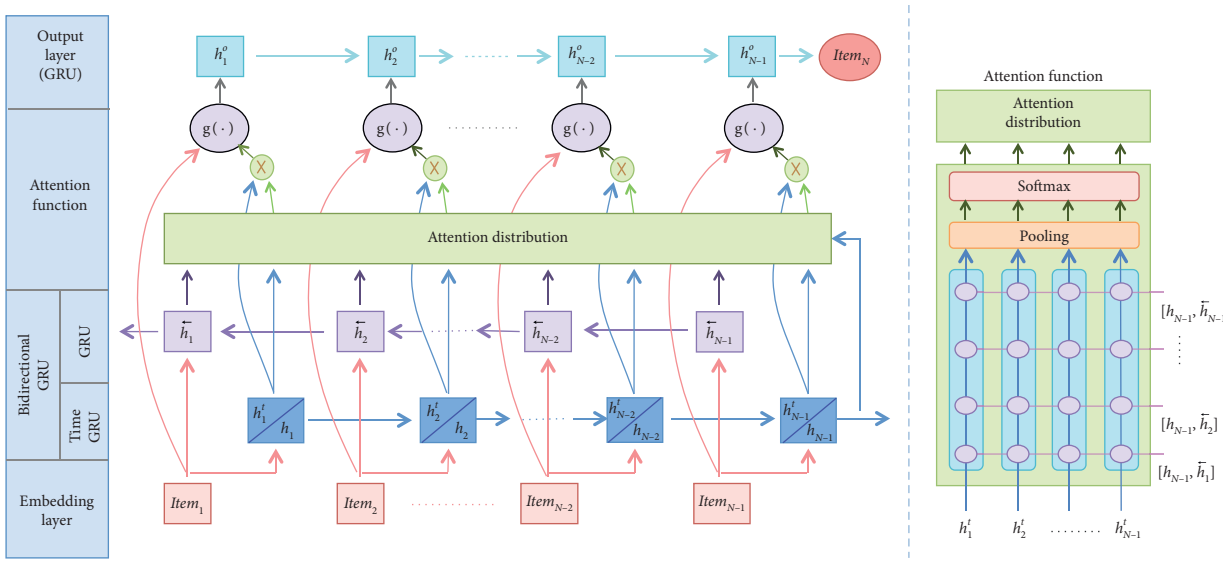

FIGURE 2: The proposed framework for modeling sequential user behaviors (left) and the designed attention mechanism (right).

2.1. Modeling Sequential User Behaviors. Due to the significance of user-centric tasks such as personalized search and recommendation, modeling sequential user behaviors has attracted great attention in both industry and academia. Most of the pioneering work relies on model-based Collaborative Filtering (CF) to analyze user-item interaction matrix. There have been a variety of such algorithms including Bayesian methods [28] and matrix factorization (MF) methods [31, 50]. Due to the characteristics of sequential information, several CF works take the temporal dynamics into account, often based on the assumption of Markov processes [11, 21, 32]. For the task of sequential recommendation, Rendle et al. [32] propose Factorizing Personalized Markov Chain (FPMC) to combine matrix factorization of user-item matrix with Markov chains. He and McAuley [11] further integrate similarity-based methods [20] into FPMC to tackle the problem of sequential dynamics.

The major problems of the abovementioned work are that these methods independently combine several components, rely on low-level hand-crafted features of user or item, and have difficulty to handle long-term behaviors. On the contrary, along with the development of deep neural networks, Lei et al. [22] and Zheng et al. [51] employ deep learning to learn effective representations of user/item automatically. Furthermore, with the success of recurrent neural networks (RNNs) in the past few years, a paucity of work has made attempts to utilize RNNs [14, 25, 47]. For example, Liu et al. [25] consider jointly the contextual information such as weather into the RNN architecture to improve modeling performance. The insight that RNN-based solutions achieve success in modeling sequential user

behaviors is that RNN has well-demonstrated ability of capturing patterns in the sequential data. Recent studies [10, 29, 39, 52] also indicate that time intervals within sequential signal are a very important clue to update and forget information in RNN architecture. Zhu et al. [52] design several time gates in LSTM units to improve modeling performance. He et al. [10] embed items into a "transition space," where users are modeled as translation vectors operating on item sequences. Liu et al. [26] employ adaptive context-specific input matrices and adaptive context-specific transition matrices to capture external situations and how lengths of time intervals between adjacent behaviors in historical sequences affect the transition of global sequential features, respectively. But, in practice, there is complex dependence and correlation between sequential user behaviors, which requires deeper analysis of relation among behaviors rather than simply modeling the presence, order, and time intervals. To summarize, how to design an effective RNN architecture to model sequential user behaviors effectively is still a challenging open problem.

2.2. Attention Mechanism. Attention mechanism is now a commonly adopted ingredient in various deep learning tasks such as machine translation [1, 27], image captioning [24], question answering [38], and speech recognition [5], which has been shown to be effective against capturing the contribution and correlation between different components in the network. The success of attention mechanism is mainly due to the reasonable assumption that human beings do not tend to process the entire signal at once; instead, they only focus on selected portions of the entire perception space when and where needed [17]. To avoid the limitation of normal networks that the entire source must be encoded to one hidden layer, the attention-based network contains a set of hidden representations that scale with the size of the source. The network learns to assign attention weights to perform a soft selection of these representations.

With the development of attention mechanism, recent researches start to leverage different attention architectures to improve performance of related tasks [1, 3, 34, 40, 46, 49]. For example, Bahdanau et al. [1] conjecture that the use of a fixed-length vector is a bottleneck in improving the performance of this basic encoder-decoder architecture; therefore, they design a model to automatically search for parts of a source sentence which are relevant to predicting a target word. Yang et al. [49] propose a hierarchical attention network at word and sentence level, respectively, to capture contributions of different parts of a document. Vaswani et al. [40] utilize multihead attention mechanism to improve performance. Wang et al. [46] propose a coverage strategy to combat the misallocation of attention caused by the memorylessness of traditional attention mechanism. Nevertheless, most of the previous work calculates attention distribution according to the interaction of every source vector with a single embedding vector of contextual or historical information (such as translated words in the sentence), which may lead to information loss caused by early summarization and noise caused by incorrect previous attention. In particular, Shen et al. [35] propose an attention-based language understanding method without any other network structure (e.g., RNN). In [35], the input sequence is processed by directional (forward and backward) self-attentions to model context dependency and produce context-aware representations for all tokens. Then, a multidimensional attention computes a vector representation of the entire sequence.

Indeed, the attention mechanism is very important to the task of modeling sequential user behaviors. However, to the best of our knowledge, there are a few works concentrating on this paradigm. Chen et al. [2] consider the attention mechanism into a multimedia recommendation task with multilayer perceptron. Song et al. [37] propose a recommender system for online communities based on a dynamic-graph-attention neural network. They model dynamic user behaviors with a recurrent neural network and context-dependent social influence [23] with a graph-attention neural network, which dynamically infers the influencers based on users' current interests. In this paper, an effective solution with attention mechanism for better modeling sequential user behaviors is to be investigated.

3. Problem Formulation

We start our discussion with the definitions of some notations. Let $\mathcal{U}$ be a set of users and let $\mathcal{I}$ be a set of items in a specific service such as products in online shopping websites. For each user $u \in \mathcal{U}$, his/her historical behaviors are given by $\mathcal{H}^u = \{(i_k^u, t_k^u) \mid i_k^u \in \mathcal{I}, t_k^u \in \mathcal{R}^+, k = 1, 2, \ldots, N_u\}$, where k denotes the user's action and (i_k^u, t_k^u) denotes the interaction between user u and item i_k^u at time t_k^u; interaction has different forms in different services, such as clicking, browsing, and adding to favorites. The objective of modeling sequential user behaviors is to predict the conditional probability of the user's next item $p(i_{N_u+1}^u \mid \mathcal{H}^u, t_{N_u+1}^u)$ for a certain given user u.

We take RNN as the basic model, which generates the conditional probability in multiple steps sequentially. At step k, the k-th item i_k^u is vectorized into x_k and then fed into RNN units by nonlinear transformation, e.g., multilayer perceptron. Then, it updates the hidden state of RNN units, i.e., $h_k = \mathrm{RNN}(x_k, h_{k-1})$, as well as the output of RNN units. The representations of hidden state and output are trained to predict the next item vectorized as x_{k+1} given h_k. To train the RNN, we aim to maximize the likelihood of historical behaviors of a set of users $\mathcal{U}$:

$$p(\mathcal{U}) = \prod_u \prod_{k=1}^{N_u-1} p\left(i_{k+1}^u \mid \mathcal{H}_k^u, t_{k+1}^u\right) = \prod_u \prod_{k=1}^{N_u-1} p\left(i_{k+1}^u \mid h_k^u, t_{k+1}^u\right), \tag{1}$$

where i_{k+1}^u is the target item for the given user u. In other words, we aim to minimize the negative logarithmic likelihood, that is, the objective function:

$$\min_\theta \mathcal{L}(\mathcal{U}) = -\sum_u \sum_{k=1}^{N_u-1} \log p\left(i_{k+1}^u \mid h_k^u, t_{k+1}^u\right), \tag{2}$$

where θ is the set of parameters in the RNN model.

To fulfill this learning, it requires us to design an effective RNN architecture including the inner functions of RNN cells and the overall network structure, which together approximate a highly nonlinear function for obtaining the probability distribution of next item. In this process, RNN usually suffers from the complex dependency problem, especially when we deal with user actions that have much noise and randomness. Attention mechanism is a possible solution, which constructs a pooling layer on top of the RNN cells at each step to characterize the dependence between the current intent and all of the historical actions. We will describe our designed network architecture with attention mechanism in the next section.

4. ALI-GRU

As illustrated in the left part of Figure 2, our designed network features an attention mechanism with long-term interval-based gated recurrent units for modeling sequential user behaviors. This network architecture takes the sequence of items as raw signal. There are four stages in our network. The embedding layer maps items to a vector space to extract their basic features. The bidirectional GRU layer is designed to capture the information of both long-term preferences and short-term intents of user; it consists of normal GRUs and time interval-based GRUs (see Figure 3). The attention function layer reflects our carefully designed attention mechanism, which is illustrated in the right part of Figure 2. Finally, there is an output layer to integrate the attention distribution and the extracted sequential features and utilize normal GRUs to predict the conditional probability of next item.

4.1. Embedding Layer. The purpose of the embedding layer is to map the raw data of items into a rectified vector space, where the vectorized representations of items still keep the semantics of the items; e.g., semantically relevant items have small distance in the vector space. Usually items can be first represented as one-hot vectors and then processed by several fully connected layers [52]. If however the number of items is too large, pretrained encoding network is useful to process the items, which encodes not only the basic properties such as category of items but also the crowdsourcing properties such as sales of items [4]. In this paper, we adopt these two strategies for different datasets, respectively.

4.2. Bidirectional GRU Layer with Time-GRU. This layer is designed to extract driven signals from input sequence and to refine the long-term memory by contextual information. We now detail our method for these two targets.

In the previous work for natural language processing tasks, the attention function is driven by a single vector of input [1, 27, 40]. That model works well because of the relatively stable syntax and semantics of input words. However, sequential user behaviors contain much noise and randomness that make the simple model problematic. We propose a new network structure with time-GRU to extract short-term dynamics of user intents as driven signal of the attention function.

The structure of time-GRU in comparison with normal GRU is shown in Figure 3, where the black lines denote the network links of normal GRU and the red lines denote new links of time-GRU. The normal GRU are as follows:

$$z_N = \sigma\left(W_z I_N + U_z h_{N-1} + b_z\right), \tag{3}$$

$$r_N = \sigma\left(W_r I_N + U_r h_{N-1} + b_r\right), \tag{4}$$

$$\widetilde{h}_N = \tanh\left(W_h I_N + r_N \odot \left(U_h h_{N-1}\right) + b_h\right), \tag{5}$$

$$h_N = \left(1 - z_N\right) \odot h_{N-1} + z_N \odot \widetilde{h}_N, \tag{6}$$

where I_N denotes the N-th sequence item vector. h_{N-1} denotes the $(N-1)$-th hidden state vector. $\widetilde{h}_N$ is the candidate activation. z_N represents the update gate, which decides how much the unit updates its activation. r_N is the reset gate to control how much the last state contributes to the current activation. σ represents the sigmoidal nonlinearities function and tanh represents the tanh nonlinearities function, and $\odot$ is an element-wise multiplication. Weight parameters W_z, W_h, W_r and U_z, U_h, U_r connect different inputs and gates; parameters b_z, b_h, b_r are biases.

The above equations imply that normal GRU is good at capturing the general sequential information. Since GRU is originally designed for NLP tasks, there is no consideration of time intervals within inputs, which are very important for modeling sequential user behaviors. To include the short-term information, we augment the normal GRU with a time gate T_N:

$$\begin{aligned} T_N &= \sigma\left(W_t \triangle t_N + U_t I_N + b_t\right) \\ &\text{s.t. } W_t < 0, \end{aligned} \tag{7}$$

where $\triangle t_N$ is the time interval between adjacent actions. The constraint $W_t < 0$ is to utilize the simple assumption that smaller time interval indicates larger correlation. Moreover, we generate a time-dependent hidden state h_N^t in addition to the normal hidden state h_N; that is,

$$h_N^t = \left(1 - z_N \odot T_N\right) \odot h_{N-1}^t + z_N \odot T_N \odot \widetilde{h}_N^t, \tag{8}$$

where we utilize the time gate as a filter to modify the update gate z_N so as to capture short-term information more effectively.

In addition, we want to utilize contextual information to extract long-term information with as little information loss as possible. Recent methods usually construct a bidirectional RNN and add or concatenate two output vectors (forward and backward) of bidirectional RNN. Bidirectional RNN outperforms unidirectional one but still suffers from the embedding loss, since the temporal dynamics are not considered enough. On the contrary, we propose to combine the output of forward normal GRU (h_N in equation (6)) with all the outputs of backward GRU at different steps (the output of backward GRU at step k is denoted by $\overleftarrow{h}_k$ in Figure 2). Specifically, we produce concatenated vectors $[h_{N-1}, \overleftarrow{h}_{N-1}]$, $[h_{N-1}, \overleftarrow{h}_{N-2}]$, ..., $[h_{N-1}, \overleftarrow{h}_1]$, as shown in the

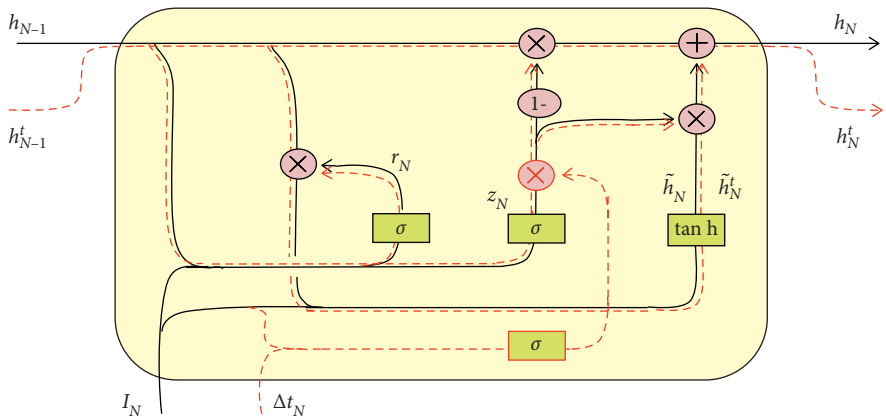

Figure 3: The structure of our proposed time-GRU. In addition to the network links of normal GRU (shown in black lines), we design network links and a new gate to capture time interval information (shown in red-dotted lines).

right part of Figure 2, where [,] stands for concatenation of vectors. This design effectively captures the contextual information as much as possible.

4.3. Attention Function Layer. Attention function layer is responsible for linking and analyzing the dependence and contribution over driven signals and contextual long-term information provided by the previous layers. Unlike previous attention mechanisms, we do not simply summarize the contextual long-term information into individual feature vectors, e.g., using $p(i_1, h_{N-1})$ for calculating the attention weight of item i_1 to the hidden state at the $(N-1)$-th step [46]. Instead, we design to attend to the driven signals at each time step along with the embedding of contextual information.

Specifically, as shown in the right part of Figure 2 and as already discussed in the last subsection, we use $\mathbf{H}_k = [h_{N-1}, \overleftarrow{h}_k] \in \mathscr{R}^{2d}, k = 1, 2, \ldots, N-1$, where d is the dimension of GRU states, to represent the contextual long-term information. $h^t_k \in \mathscr{R}^d$ denotes the short-term intent reflected by item i_k. We then construct an attention matrix $A \in \mathscr{R}^{(N-1)*(N-1)}$, whose elements are calculated by

$$A_{ij} = \alpha\left(\mathbf{H}_i, h^t_j\right) \in \mathscr{R}, \tag{9}$$

where the attention weight,

$$\alpha\left(\mathbf{H}_i, h^t_j\right) = v^T \tanh\left(W_a \mathbf{H}_i + U_a h^t_j\right), \tag{10}$$

is adopted to encode the two input vectors. v^T are the weight parameters. There is a pooling layer, for example, average or max pooling, along the direction of long-term information, and then there is a Softmax layer to normalize the attention weights of each driven signal. Let a_k be the normalized weight on h^t_k; then the attended short-term intent vector is $\hat{h}^t_k = a_k h^t_k \in \mathscr{R}^d$. At last, we use $g(i_k, \hat{h}^t_k) = [i_k, \hat{h}^t_k, |i_k - \hat{h}^t_k|, i_k \odot \hat{h}^t_k] \in \mathscr{R}^{4d}$ as the output to the next layer, where i_k is the embedded vector of the item at the k-th step.

We want to emphasize the insight of our carefully designed attention mechanism described above, which is different from the existing methods, to reduce the loss of contextual information caused by early summarization. Furthermore, since driven signals are attended to the long-term information at different steps, the attentions can obtain the trending change of user's preferences, being more robust and less affected by the noise in the historical actions.

4.4. Output Layer. Given $g(i_k, \hat{h}^t_k)$ produced by the attention function layer, we use a layer of normal GRUs to produce the embedding vector (h^o_k in Figure 2), which is expected to contain the contextual long-term information about all of the user's historical actions with respect to the single item and short-term intents. The embedding vector is then decoded to produce the final result. For example, we use a Softmax function after a fully connected layer to obtain the probability distribution of different items in the next action:

$$p(i_N) = \text{softmax}\left(w^T_p h^o_{N-1}\right). \tag{11}$$

If the number of candidate items is too big, we shall use a slightly different decoding function, which will be detailed in Section 5.3.

5. Experiments

In this section, we first describe the used datasets and several state-of-the-art approaches that were compared as baselines in this paper. Then, we report and discuss experimental results on different datasets.

5.1. Datasets. To verify our proposed ALI-GRU, we conduct a series of experiments on two well-known public datasets (LastFM (http://www.dtic.upf.edu/~ocelma/MusicRecommendationDataset/lastfm-1K.html) and CiteULike (http://www.citeulike.org/faq/data.adp)). Additionally, we also perform offline and online experiments on the real data from one of the largest online shopping websites. Table 1 shows the statistics of LastFM and CiteULike:

(i) LastFM contains <*user_id*, *timestamp*, *artist_id*, *song_id*> tuples collected from Last.fm API (https://www.last.fm/api/). It represents the whole listening

Table 1: Statistics of LastFM and CiteULike.

	LastFM	CiteULike
Number of users	987	1625
Number of items	5,000	5,000
Number of behaviors	818,767	35,834

habits (till 5 May 2009) for 1000 users. We extract tuples <*user_id*, *song_id*, *timestamp*> from the original dataset to conduct experiments, where each *song_id* represents an item and each tuple represents the action or behavior that the user *user_id* listens to the song *song_id* at time *timestamp*.

(ii) CiteULike consists of the tuples <*user_id*, *paper_id*, timestamp, *tag*>, where each tuple represents that the user *user_id* annotates the paper *paper_id* with *tag* at time *timestamp*. One user annotating one research paper (i.e., item) at a certain time may have several records, in order to distinguish different tags. We merge them as one record and extract tuples <*user_id*, *paper_id*, *timestamp*> to construct dataset like in [52].

5.2. Compared Approaches. We compare ALI-GRU with the following state-of-the-art approaches for performance evaluation:

(i) *Factorized Sequential Prediction with Item Similarity Models (Fossil)* [11]. This is a state-of-the-art factorized sequential prediction method based on Markov processes. Fossil also considers the similarity of explored items to those already consumed/liked by user, which achieves a certain success to handle the long-tail problem. We have used the implementation provided by the authors (https://drive.google.com/file/d/0B9Ck8jw-TZUEeEhSWXU2WWloc0k/view).

(ii) *Basic GRU/Basic LSTM* [7]. This method directly uses normal GRU/LSTM as the primary network. For fair comparison, we set the network to use the same embedding layer and the same decoding function as our method.

(iii) *Session RNN* [13]. Hidasi et al. propose an RNN-based method to capture the contextual information according to sessions of user behaviors. In our experiments, we use a commonly adopted method described in [16] to identify sessions so as to adopt this baseline.

(iv) *Time-LSTM* [52]. This method utilizes LSTM to model the pattern of sequential user behaviors. Compared to normal LSTM units, Time-LSTM considers the time intervals within sequential signal and designs several time gates in LSTM units similarly as our time-GRU.

(v) *Simplified Version 1 (SV1).* This approach is designed to verify the effectiveness of our designed attention mechanism. The SV1 approach is identical to ALI-GRU, with the only difference being that SV1 uses an attention mechanism provided in [1, 46] which simply summarizes the contextual long-term information into individual feature vectors. Specifically, user contextual behaviors are modeled as a vector $[h_{N-1}, \overleftarrow{h}_1]$, and all driven signals are attended to this vector.

(vi) *Simplified Version 2 (SV2).* This approach is designed to verify the effectiveness of our proposed time-GRU for generating driven signals according to short-term information. Compared to ALI-GRU, the only difference is that SV2 uses single item at each step (the embedded vector) to attend to contextual information.

All RNN-based models are implemented with the open source deep learning platform TensorFlow (https://www.tensorflow.org/). Training was done on a single GeForce Tesla P40 GPU with 8 GB graphical memory.

5.3. Experiments on LastFM and CiteULike. We first evaluate our method on two well-known public datasets for the task of sequential recommendation.

5.3.1. Datasets. In this experiment, we use the same datasets as those adopted in [52], i.e., LastFM and CiteULike. Table 1 presents the statistics of these two datasets. Both datasets can be formulated as a series of tuples <*user_id, item_id, timestamp*>. Our target is to recommend songs in LastFM and papers in CiteULike for users according to their historical behaviors.

For fair comparison, we follow the segmentation of training set and test set as described in [52]. Specifically, 80% users are randomly selected for training. The remaining users are for testing. For each test user u with k historical behaviors, there are $k-1$ test cases, where the k-th test case is to perform recommendations at time t^u_{k+1} given the user's previous k actions, and the ground-truth is i^u_{k+1}. The recommendation can also be regarded as a multiclass classification problem. For more details, please refer to [52].

5.3.2. Implementation. Following the method in [52], we use one-hot representations of items as inputs to the network and one fully connected layer with 8 nodes for embedding. The length of hidden states of GRU-related layers including both normal GRU and time-GRU is 16. A Softmax function is used to generate the probability prediction of next items. For training, we use the AdaGrad [8] optimizer, which is a variant of Stochastic Gradient Descent (SGD). Parameters for training are minibatch size of 16 and initial learning rate of 0.001 for all layers. The training process takes about 8 hours.

5.3.3. Evaluations. In the test stage, following the evaluation method in [52], we select 10 items with top probabilities as final recommendations. We use Recall@10 to measure

whether the ground-truth item is in the recommendation list. Recall@10 is defined as

$$\text{Recall@10} = \frac{n_{\text{hit}}}{n_{\text{testcase}}}, \tag{12}$$

where n_{hit} is the number of test cases where i_g is in the recommendation list and n_{testcase} is the number of all test cases. We further use MRR@10 (Mean Reciprocal Rank) to consider the rank of ground truth in the recommendation list. This is the average of reciprocal ranks of i_g in the recommendation list. The reciprocal rank is set to 0 if the rank is above 10.

5.3.4. Overall Performance. The results of sequential recommendation tasks on LastFM and CiteULike are shown in Table 2. It can be observed that our approach performs the best on both LastFM and CiteULike for all metrics, which demonstrates the effectiveness of our proposed ALI-GRU. Specifically, ALI-GRU obtains significant improvement over Time-LSTM, which is the best baseline, averagely by 4.70% and 6.55% for Recall@10 and MRR@10, respectively. It owes to the superiority of introducing attention mechanism into RNN-based methods, especially in capturing the contribution of each historical action.

5.3.5. Performance of Cold-Start. Cold-start refers to the lack of enough historical data for a specific user, which often decreases the efficiency of making recommendations. We analyze the influence of cold-start on the LastFM dataset and the results are given in Figure 4. In this figure, test cases are separately counted for different numbers of historical actions, and small number refers to cold-start. We can observe that, for cold users with only 5 actions, ALI-GRU performs slightly worse than the state-of-the-art methods. This is because ALI-GRU considers short-term information as driven signals, which averages source signal to some extent and leads to less accurate modeling for cold users. Along with the increase of historical actions, ALI-GRU achieves significantly better performance than the baselines, which indicates that bidirectional GRU and attention mechanism can better model the long-term preferences for making recommendations.

5.4. Offline Experiments. We have collected a large-scale dataset from a real-world e-commerce website for further performance evaluation. ALI-GRU is also adopted online and results of online A/B test will be reported in the next section.

5.4.1. Dataset. User behaviors of this dataset are randomly sampled from the logs of clicking and purchasing in seven days (a beginning week of July 2017) on a real-world e-commerce website. The dataset is again formulated as a series of tuples <*user_id, item_id, timestamp*>.

We focus on the task of the personalized search of e-commerce websites. So we define positive cases as those purchasing behaviors led by the e-commerce search engine mentioned above, while negative cases are those clicks without purchases (if there is no purchase around the click within 5 actions). Finally, we have 24, 282, 032 positive cases, 84, 322, 922 negative cases, 30, 602, 427 users, and 10, 808, 463 items. We randomly select 80% users for training, and the remaining users are for testing. For each positive or negative case i_k in the sequence, our target is to predict whether the user would purchase i_k according to his/her historical behaviors, which is a typical binary classification problem.

Table 2: Comparison results on LastFM and CiteULike.

	LastFM		CiteULike	
	Recall@10	MRR@10	Recall@10	MRR@10
Basic-LSTM	0.2451	0.0892	0.6824	0.2889
Session RNN	0.3405	0.1573	0.7129	0.2997
Time-LSTM	0.3990	0.2657	0.7586	0.3660
ALI-GRU	**0.4752**	**0.2697**	**0.7764**	**0.4930**

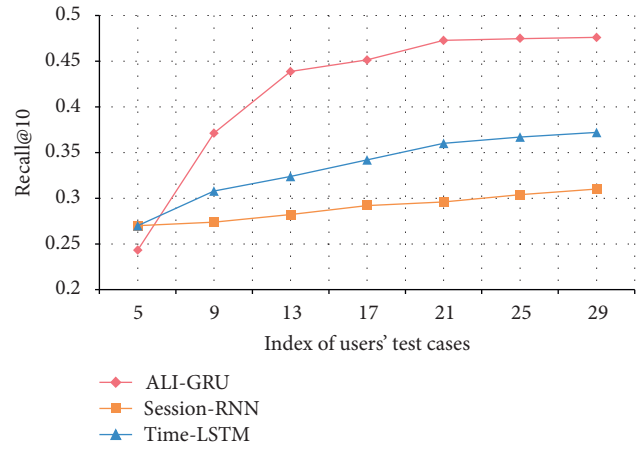

Figure 4: Recall@10 evaluated on different indexes of users' test cases in LastFM.

5.4.2. Implementation. Since the amount of items is too huge in this dataset, it is inconvenient to employ the one-hot representations as inputs to RNN-based models. Instead, we use pretrained embedding vectors of items as inputs and additionally use two fully connected layers, both with 128 nodes, to reembed the item vectors. Also, we have followed the wide & deep learning approach in [4] to convert the outputs of the final fully connected layer, whose size is 48, into representations of corresponding items. For fair comparison, all RNN-based approaches employ the pretrained item representations as inputs. Hidden state size of GRU-related layers is 128. We finally use the sigmoid function to predict whether the user would purchase i_k. For training, the loss function is cross-entropy and AdaGrad optimizer is employed with a minibatch size of 256 and an initial learning rate of 0.001 for all layers. The entire training process takes about 50 hours.

Table 3: Comparison results on offline dataset.

	Fossil	Basic-GRU	Session RNN	Time-LSTM	SV1	SV2	ALI-GRU
AUC	0.7017	0.6609	0.7543	0.7712	0.7833	0.8062	**0.8370**

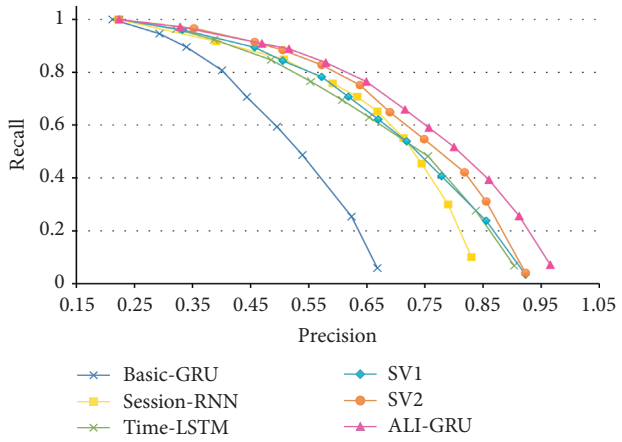

Figure 5: Precision-Recall curves of different methods on offline dataset.

5.4.3. Evaluations. In the test stage, we use Precision-Recall of positive cases to measure the performance. Moreover, AUC (Area under ROC Curve) is also adopted, which is widely used in imbalanced classification tasks [12]. The larger the value of AUC is, the better the performance is.

5.4.4. Overall Performance. Table 3 shows the AUC results for measuring the overall performance. We can observe that all RNN-based methods except Basic-GRU outperform Fossil that is based on matrix factorization with Markov processes, which indicates the advantage of RNN for modeling sequential data. Furthermore, for different views of the capabilities of different RNN-based approaches, we also report Precision-Recall curves shown in Figure 5 and make some comparisons and summarize our findings as follows.

5.4.5. Basic GRU versus Session RNN versus Time-LSTM. Session RNN and Time-LSTM achieve significant improvement compared to Basic GRU, which is consistent with the previous results on public datasets. This is due to the limitation of Basic-GRU/Basic-LSTM in modeling complex long-term sequential data. Compared to Session RNN, Time-LSTM achieves better performance for high precision range (precision larger than about 0.73), which owes to the advantage of short-term intents for predicting highly confident items. On the contrary, Session RNN outperforms Time-LSTM for low precision range (precision lower than about 0.73), since Session RNN introduces session view to better model contextual information and then benefits from recalling items based on long-term preferences of user.

5.4.6. Session RNN versus Time-LSTM versus ALI-GRU. By adopting time gate, which is strong at modeling short-term dynamics, and bidirectional RNN, which leads to advantages for modeling long-term information, ALI-GRU better analyzes the complex dependence among items and user intents, together with a novel matrix-form attention mechanism to enhance the performance. ALI-GRU outperforms Session RNN and Time-LSTM by as much as 10.96% and 8.53% for AUC (Table 3), respectively. Observing Precision-Recall curves, we found that ALI-GRU beats Session RNN and Time-LSTM over the entire range, and the improvement is more significant for the high precision range. The superior performance of ALI-GRU on various datasets and views demonstrates its efficacy to handle long-term sequential user behaviors with dynamical short-term intents.

5.4.7. SV1 and SV2 versus Others. We also show results of SV1 and SV2 for ablation analyses (Figure 5). Observing the curve of SV1, we can find that the previous attention mechanism with bidirectional GRU only achieves slight improvement compared to Time-LSTM, which indicates the limitation of the previously studied attention mechanism for capturing dynamical importance of items in sequential user behaviors. On the contrary, SV2 outperforms both Session RNN and Time-LSTM consistently, especially for low precision range. It suggests that our proposed matrix-form attention mechanism with bidirectional GRU has superior capacity in distinguishing items' importance for modeling long-term preferences of user. Nevertheless, the curve of SV2 drops a lot when the precision is larger than about 0.82, where it is comparable to SV1 and Time-LSTM. This is because user behaviors and intents are dynamic with a certain randomness, and single item is not robust enough for calculating attention distribution and capturing short-term intents. Last but not least, we can find that ALI-GRU leads to performance boost against SV1 and SV2 consistently. It demonstrates the advantages of our carefully designed matrix-form attention with long-term interval-based GRU framework for modeling sequential user behaviors.

5.4.8. Case Study and Insights. We present three cases in Figure 6 for comprehensive study to give some insights of our proposed approach. Each case consists of one user's historical items ordered by click time and shows the attention heat map for each item and the final item (click or purchase) with prediction and ground truth.

Case A. The user clicked items of several classes such as watch, handbag, and dress and finally purchased the watch that was clicked long before. We make a few observations as follows: (1) ALI-GRU gives higher weights to most watches than other items, which is consistent with the user's contextual intent (purchasing a watch). It suggests that our proposed approach has capacity to capture user's real intents from historical behaviors. (2) The 1st, 3rd, and 5th watches, which are the same as or similar to the final purchased watch, have higher weights than the other watches, especially for the

Figure 6: Case study of making prediction whether the user purchases the next item according to historical sequential behaviors. Each case consists of one user's clicked items ordered by click time and shows the attention heat map for each item and the final item (clicked or purchased) with prediction and ground truth. Darker color in the heat map indicates higher attention weight.

1st watch, though it was clicked earliest for a long time ago. More interestingly, we can observe that the 6th watch is for women, and the user is probably a woman (according to the dresses he/she clicked), but the 6th watch has the relative lowest weight in all watches. These observations indicate that ALI-GRU successfully distinguishes the user's current intent for purchasing a watch for men.

Case B. If items were inherently important or repeated or had low frequency, models without attention mechanism might work well since such model could automatically assign low weights to irrelevant items and vice versa. However, the importance of items and user intents is highly dependent on context and is consistent to a certain degree. In Case B, the user finally purchased a coat hanger that belongs to the class he/she never clicked. Nevertheless, ALI-GRU looks at the context of his/her recent behaviors, conjectures the intent is possible for something about laundry, and correctly figures out this is a positive case.

Case C. It is not a correct prediction case according to ground truth. Observing the historical behaviors and attention distribution of the user, we can find that ALI-GRU chooses to ignore various items before the first suit; these actions had a long time interval (about two days) from the latter actions. Furthermore, ALI-GRU conjectures that the user wants to buy something for formal wearing. Therefore, ALI-GRU predicts it is a negative case for purchasing a USB cable, which is purchased by the user finally. In such cases, there exist choppy and decisive intents of user, which is a great challenge left for future exploration.

5.5. Online Test. Online test with real-world e-commerce users is carried out to study the effectiveness of our proposed method. In particular, we integrate ALI-GRU into the e-commerce search engine mentioned above, which has billions of clicks per day. A standard A/B test is conducted online. Users of the search engine are randomly divided into multiple buckets, and we randomly select two buckets for experiments. For users in bucket A, we use the existing highly optimized ranking solution of the search engine, which performs Learning to Rank (LTR) and Reinforce Learning (RL) with several effective algorithms such as wide & deep learning and CF prediction. For users in bucket B, we further integrate the results produced by ALI-GRU. Specifically, for a given user, his/her sequential behaviors (clicked items and timestamps) are collected from the entire service, and the user's intent vector is predicted by ALI-GRU in real time. When the user provides a query, we combine the calculated user intent vector with all the retrieved items to calculate purchasing probability, which is similar to the method of offline experiments. Finally, we integrate the purchasing probability into the existing ranking strategy.

Measures for the online A/B test include Gross Merchandise Volume (GMV), user Click Through Rate (uCTR), Click Conversion Rate (CVR), Per Customer Transaction (PCT), and Unique Visitor Value (UV_Value), which are all frequently used metrics in e-commerce [19]:

$$\begin{aligned} \text{uCTR} &= \frac{\text{number of user clicking}}{\text{number of user browsed items}}, \\ \text{CVR} &= \frac{\text{number of user trading}}{\text{number of user clicking}}, \\ \text{PCT} &= \frac{\text{gross merchandise volume}}{\text{number of user trading}}, \\ \text{UV\_Value} &= \frac{\text{gross merchandise volume}}{\text{number of unique visitor}}. \end{aligned} \tag{13}$$

The test was performed within one week in July 2017. Comparative results are given in Table 4, where the absolute values are omitted for business confidentiality. The results show that ALI-GRU achieves better performance for all the metrics. As we expected, uCTR and CVR are improved, which means that users are more likely to click the reranked items, and there is higher probability to purchase these

Table 4: The improvements (%) of online test.

GMV	uCTR	CVR	PCT	UV_value
+3.08	+1.71	+1.28	+1.29	+2.57

items. More interesting is the improvement of PCT and UV_Value, which is due to the increase of number of transactions per user with purchasing actions. This result suggests that our model provides kinds of recommendation functionality into search engine, such as case B in Figure 6. In summary, our proposed ALI-GRU consistently improves the high-quality baseline of one of the largest online e-commerce platforms that has been optimized for several years. Such improvements are very important for e-commerce search engine systems and have significant business value. ALI-GRU has been adopted into the search engine before this paper is prepared.

6. Conclusions

Modeling user behaviors as sequential learning plays an important role for predicting future user actions, such as personalized search and recommendation. However, most of RNN-based methods assume that the importance of historical behaviors decreases over time and fail to consider the cross-dependence in sequences, which makes it difficult to apply to the real-world scenarios. To address these problems, we propose a novel and efficient approach called Attention with Long-term Interval-based Gated Recurrent Units (ALI-GRU) for better modeling sequential user behaviors. We first propose a bidirectional time interval-based GRU to identify complex correlation between actions and capture both long-term preferences and short-term intents of users as driven signals. Then, we design a new attention mechanism to attend to the driven signals at each time step for predicting the next user action. The empirical evaluations on two public datasets for sequential recommendation task show that ALI-GRU achieves better performance than state-of-the-art solutions. Specifically, ALI-GRU outperforms Session RNN and Time-LSTM by as much as 10.96% and 8.53% in terms of AUC. In addition, online A/B tests in a real-world e-commerce search engine further demonstrate its practical value. As GRU cannot be calculated in parallel, it takes a lot of time to train the model. In the future, we will adopt a parallel approach to solve this problem.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. 61873288), Alibaba-PKU Joint Program, and Zhejiang Lab (nos. 2019KE0AB01 and 2019KB0AB06).

References

[1] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, https://arxiv.org/abs/1409.0473.

[2] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, "Attentive collaborative filtering: multimedia recommendation with item-and component-level attention," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 335–344, Tokyo, Japan, August 2017.

[3] H. Cheng, H. Fang, X. He, J. Gao, and Li Deng, "Bi-directional attention with agreement for dependency parsing," 2016, https://arxiv.org/abs/1608.02076.

[4] H.-T. Cheng, L. Koc, J. Harmsen et al., "Wide & deep learning for recommender systems," 2016, https://arxiv.org/abs/1606.07792.

[5] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," 2015, https://arxiv.org/abs/1506.07503.

[6] C. Chu, L. Zhao, B. Xin et al., "Deep graph embedding for ranking optimization in e-commerce," in *Proceedings of the ACM International Conference on Information and Knowledge Management*, Turin, Italy, 2018.

[7] J. Chung, C. Gulcehre, K. H. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, https://arxiv.org/abs/1412.3555.

[8] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[9] L. Guo, Q. Zhang, W. Hu, Z. Sun, and Y. Qu, "Learning to complete knowledge graphs with deep sequential models," *Data Intelligence*, vol. 1, no. 3, pp. 224–2243, 2019.

[10] R. He, W.-C. Kang, and J. McAuley, "Translation-based recommendation: a scalable method for modeling sequential behavior," in *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 5264–5268, Stockholm, Sweden, 2018.

[11] R. He and J. McAuley, "Fusing similarity models with Markov chains for sparse sequential recommendation," in *Proceedings of the International Conference on Data Mining*, pp. 191–200, Barcelona, Spain, 2016.

[12] R. He and J. McAuley, "VBPR: visual bayesian personalized ranking from implicit feedback," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 144–150, Phoenix, AZ, USA, 2016.

[13] B. Hidasi, A. Karatzoglou, L. Baltrunas, and T. Domonkos, "Session-based recommendations with recurrent neural networks," 2015, https://arxiv.org/abs/1511.06939.

[14] B. Hidasi, M. Quadrana, A. Karatzoglou, and T. Domonkos, "Parallel recurrent neural network architectures for feature-rich session-based recommendations," in *Proceedings of the ACM Conference on Recommender Systems*, pp. 241–248, Boston, MA, USA, 2016.

[15] B. Hidasi and T. Domonkos, "Fast ALS-based tensor factorization for context-aware recommendation from implicit feedback," in *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 67–82, Bristol, UK, September 2012.

[16] X. Huang, F. Peng, A. An, and S. Dale, "Dynamic web log session identification with statistical language models," *Journal of the American Society for Information Science and Technology*, vol. 55, no. 14, pp. 1290–1303, 2004.

[17] R. Hübner, M. Steinhauser, and C. Lehle, "A dual-stage two-phase model of selective attention," *Psychological Review*, vol. 117, no. 3, pp. 759–784, 2010.

[18] S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip, "A survey on knowledge graphs: representation, acquisition and applications," 2020, https://arxiv.org/abs/2002.00388.

[19] P. Jiang, Y. Zhu, Yi Zhang, and Y. Quan, "Life-stage prediction for product recommendation in e-commerce," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2015.

[20] S. Kabbur, X. Ning, and K. George, "FISM: factored item similarity models for top-n recommender systems," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 659–667, Chicago, IL, USA, 2013.

[21] Y. Koren, "Collaborative filtering with temporal dynamics," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 447–456, Paris France, June 2009.

[22] C. Lei, L. Dong, W. Li, Z.-J. Zha, and H. Li, "Comparative deep learning of hybrid representations for image recommendations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2545–2553, Las Vegas, NV, USA, 2016.

[23] S. Li, T. Cai, Ke Deng, X. Wang, T. Sellis, and F. Xia, "Community-diversified influence maximization in social networks," *Information Systems, Amsterdam, The Netherlands*, vol. 92, Article ID 101522, 2020.

[24] C. Liu, J. Mao, F. Sha, and A. Yuille, "Attention correctness in neural image captioning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 4176–4182, San Francisco, CA, USA, 2017.

[25] Q. Liu, S. Wu, D. Wang, Z. Li, and L. Wang, "Context-aware sequential recommendation," in *Proceedings of the IEEE 16th International Conference on Data Mining (ICDM)*, pp. 1053–1058, Barcelona, Spain, 2016.

[26] Q. Liu, S. Wu, D. Wang, Z. Li, and L. Wang, "Context-aware sequential recommendation," *IEEE International Conference on Data Mining*, pp. 1053–1058, 2016.

[27] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015, https://arxiv.org/abs/1508.04025.

[28] K. Miyahara and M. J. Pazzani, "Collaborative filtering with the simple Bayesian classifier," in *Proceedings of the Pacific Rim International Conference on Artificial Intelligence*, pp. 679–689, Melbourne, Australia, August 2000.

[29] D. Neil, M. Pfeiffer, and S.-C. Liu, "Phased LSTM: accelerating recurrent network training for long or event-based sequences," 2016, https://arxiv.org/abs/1610.09513.

[30] S. Pan, R. Hu, S.-F. Fung, G. Long, J. Jiang, and C. Zhang, "Learning graph embedding with adversarial training methods," *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2475–2487, 2020.

[31] A. Paterek, "Improving regularized singular value decomposition for collaborative filtering," *Proceedings of KDD Cup and Workshop*, vol. 2007, pp. 5–8, 2007.

[32] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized Markov chains for next-basket recommendation," in *Proceedings of the International Conference on World Wide Web*, pp. 811–820, Raleigh, CA, USA, 2010.

[33] B. Sarwar, K. George, K. Joseph, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the tenth international conference on World Wide Web*, pp. 285–295, Hong Kong, China, 2001.

[34] M. Seo, A. Kembhavi, F. Ali, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," 2016, https://arxiv.org/abs/1611.01603.

[35] T. Shen, T. Zhou, G. Long, J. Jiang, S. Pan, and C. Zhang, "DiSAN: directional self-attention network for RNN/CNN-free language understanding," 2018, https://arxiv.org/abs/1709.04696.

[36] X. Shen, S. Pan, W. Liu, Y.-S. Ong, and Q.-S. Sun, "Discrete network embedding," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pp. 3549–3555, Stockholm, Sweden, July 2018.

[37] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, and J. Tang, "Session-based social recommendation via dynamic graph attention networks," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 555–563, Melbourne, Australia, 2019.

[38] S. Sukhbaatar, J. Weston, R. Fergus et al., "End-to-end memory networks," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2440–2448, Montreal, Canada, 2015.

[39] B. Vassøy, M. Ruocco, E. d. S. d. Silva, and E. Aune, "Time is of the essence: a joint hierarchical RNN and point process model for time and item predictions," in *Proceedings of the ACM International Conference on Web Search and Data Mining*, pp. 591–599, Los Angeles, CA, USA, 2019.

[40] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 5998–6008, Long Beach, CA, USA, 2017.

[41] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, and X. Cheng, "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, Santiago, Chile, 2015.

[42] P. Wang, L. Zhao, X. Pan, D. Ding, X. Chen, and Y. Hou, "Density matrix based preference evolution networks for e-commerce recommendation," in *Proceedings of the International Conference on Database Systems for Advanced Applications*, pp. 366–383, Chiang Mai, Thailand, 2019.

[43] P. Wang, L. Zhao, Y. Zhang, Y. Hou, and L. Ge, "QPIN: a Quantum-inspired preference interactive network for e-commerce recommendation," in *Proceedings of the ACM International Conference on Information and Knowledge Management*, pp. 2329–2332, Beijing, China, 2019.

[44] X. Wang, Y. Liu, and F. Xiong, "Improved personalized recommendation based on a similarity network," *Physica A: Statistical Mechanics and Its Applications*, vol. 456, pp. 271–280, 2016.

[45] X. Wang, Y. Liu, G. Zhang, Y. Zhang, H. Chen, and J. Lu, "Mixed similarity diffusion for recommendation on bipartite networks," *IEEE Access*, vol. 5, no. 2017, pp. 21029–21038, 2017.

[46] Y. Wang, H. Shen, S. Liu, J. Gao, and X. Cheng, "Cascade dynamics modeling with attention-based recurrent neural network," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pp. 2985–2991, Melbourne, Australia, 2017.

[47] C.-Y. Wu, A. Ahmed, A. Beutel, A. J. Smola, and H. Jing, "Recurrent recommender networks," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 495–503, Cambridge, UK, 2017.

[48] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: multivariate time series forecasting with graph neural networks," 2020, https://arxiv.org/abs/2005.11650.

[49] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1480–1489, San Diego, CA, USA, 2016.

[50] H.-F. Yu, C.-J. Hsieh, S. Si, and I. Dhillon, "Scalable coordinate descent approaches to parallel matrix factorization for recommender systems," in *Proceedings of the IEEE 12th International Conference on Data Mining*, pp. 765–774, Brussels, Belgium, 2012.

[51] L. Zheng, V. Noroozi, and S. Y. Philip, "Joint deep modeling of users and items using reviews for recommendation," in *Proceedings of the ACM International Conference on Web Search and Data Mining*, pp. 425–434, Cambridge, UK, 2017.

[52] Yu Zhu, H. Li, Y. Liao et al., "What to do next: modeling user behaviors by time-LSTM," in *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 3602–3608, Melbourne, Australia, August 2017.