Neurocomputing 358 (2019) 141–149

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A hierarchical contextual attention-based network for sequential recommendation

Qiang Cui, Shu Wu*, Yan Huang, Liang Wang

Center for Research on Intelligent Perception and Computing (CRIPAC), National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (CASIA) and University of Chinese Academy of Sciences (UCAS), Beijing 100190, China

ARTICLE INFO

Article history:
Received 2 June 2018
Revised 28 March 2019
Accepted 25 April 2019
Available online 10 May 2019

Communicated by Dr. B. Hammer

Keywords:
Sequential recommendation
Recurrent neural network
Short-term interest
Context
Attention mechanism

ABSTRACT

The sequential recommendation is one of the most fundamental tasks for Web applications. Recently, recurrent neural network (RNN) based methods become popular and show effectiveness in many sequential recommendation tasks, such as next-basket recommendation and location prediction. The last hidden state of RNN is usually applied as the sequence's representation to make recommendations. RNN can capture the long-term interest with the help of gated activations or regularizers but has difficulty in acquiring the short-term interest due to the ordered modeling. In this work, we aim to strengthen the short-term interest, because it is beneficial to generate responsive recommendation according to recent behaviors. Accordingly, we propose a Hierarchical Contextual Attention-based (HCA) network. First, RNN is extended to model several adjacent factors at each time step. Such kind of multiple factors can be considered as a context where the short-term interest comes from. Then, within the context, the attention mechanism is used to find the important items that contribute to the short-term interest. This contextual attention-based technique is conducted on the input and hidden state of RNN respectively. In this way, we can relieve the limitation of ordered modeling of RNN, model the complicated correlations among recent factors, and strengthen the short-term interest. Experiments on two real-world datasets show that HCA can effectively generate the personalized ranking list and achieve considerable improvements.

1. Introduction

Recently, with the development of the Internet, some sequential applications have become numerous and multilateral, such as ad click prediction, purchase recommendation, and web page recommendation. A user's behaviors in such applications form a sequence in chronological order, and following behaviors of this user can be predicted by sequential recommendation methods. Taking online shopping for instance, after a user buys an item, it is expected to recommend a list of items that the user might purchase in the near future.

Traditional sequential methods usually employ Markov chains (MC), while RNN based methods become dominant recently. We can develop high/variable-order Markov models or combine MC with other methods like matrix factorization to improve the performance [1–3]. However, the Markov assumption makes it difficult to construct a more effective relationship among multiple factors. Lately, with the great success of deep learning, RNN based methods have achieved better performances than MC on different tasks, e.g., next basket recommendation [4], location prediction [5], product recommendations [6], and multi-behavioral prediction [7]. These RNN based methods mostly apply a common strategy: using the last hidden state of RNN as the user's final representation and then making recommendations. However, we conjecture that this strategy is not adequate to represent users and few people pay attention to this problem in the field of recommender systems.

In other fields like machine learning, we can enhance RNN by gated activations, regularizers, latent variables, and so on. The long short-term memory (LSTM) [8] and gated recurrent unit (GRU) [9] are two successful gated activations. They have been widely used in lots of fields and tasks. Many regularizers are proposed based on the Bernoulli stochastic process, which has a wide range of applications, for example, in information science [10,11]. Recently, dropout is successful for RNN and applies the null operator to force some units to zero. There are many variations of dropout to keep memory in RNN, like feed-forward dropout [12], rnnDrop [13], recurrent dropout [14], and so on. Compared with dropout, the newly developed zoneout randomly forces some units to keep their preceding values and becomes state-of-the-art [15]. Although these regularizers can resist the vanishing gradient to better

* Corresponding author.
E-mail addresses: cuiqiang2013@ia.ac.cn (Q. Cui), shu.wu@nlpr.ia.ac.cn (S. Wu), yhuang@nlpr.ia.ac.cn (Y. Huang), wangliang@nlpr.ia.ac.cn (L. Wang).

https://doi.org/10.1016/j.neucom.2019.04.073
0925-2312/

capture the long-term dependency, they do not help to enhance the user's short-term interest. On the other hand, latent variables are added to RNNs to improve performance. Usually, people would combine latent variables with hidden states in RNN to encode observed variability within the data, inform recurrent dynamics for future states, and so on, like VRNN [16], SRNN [17], Z-Forcing [18]. Most works incorporating latent variables would apply the future information when processing the current hidden state at t-th step. This is suitable for tasks like speech recognition. However, we can not do this in sequential recommendation because we do not know any future context after t-th step.

RNN cannot well model the short-term interest for users, and it may be not very appropriate to focus on better capturing the long-term dependency in the sequential recommendation. RNN has one item and one hidden state at each time step. Due to the recurrent structure and fixed transition matrices, RNN is well known for the exponential increasing or decreasing and the current item is usually more significant than the previous one. Therefore, because of this ordered modeling, RNN holds an assumption that temporal dependency has a monotonic change with the input time steps [7]. But the correlation among user behaviors is very complicated. We cannot guarantee that one item is more significant than the previous one. For example, we can consider the regular three meals a day as a sequence. When we predict the breakfast the next day by using RNN, the nearest dinner on this day is most significant, followed by lunch. Intuitively, what is most relevant to the next breakfast is previous breakfasts, instead of lunch or dinner. Small weights should be provided for such less relevant behaviors. On the other hand, people in machine learning would like to enhance RNN by using gated activation functions or regularizers to better capture the long-term dependency. This advantage allows the hidden state to be able to connect previous information for a long time. However, user interest would change over time. People would forget what they have purchased or clicked after a period of time. Therefore, there would be no need to force RNN to remember the advanced inputs, and the successful methods to enhance RNN in other fields may not be suitable in the recommender system. Consequently, the short-term interest in RNN needs to be carefully examined for the modeling of user behaviors.

To strengthen the short-term interest, the key is to acquire the complicated correlations among multiple factors. Therefore, rather than processing the factor (item or hidden state) one by one, multiple factors are considered at each time step. Such several adjacent factors form a context in our work. This contextual modeling helps RNN to relieve the limitation of ordered modeling. In some works, this contextual modeling is called union-level item relevance model [19,20], which can capture the collective influence among multiple items. Furthermore, we propose to apply the attention mechanism to acquire the complicated correlations. The attention can effectively assign an appropriate weight for each factor to tell its importance [21]. Then we apply this contextual attention-based technique on the item to explore the correlations among multiple items. Moreover, this technique is also employed on the hidden state, and we acquire a hierarchical structure.

In this paper, we propose a Hierarchical Contextual Attention-based (HCA) network. It can considerably strengthen the short-term interest. We employ the contextual attention-based technique on the input and hidden state respectively. (1) The first level is conducted on the input. We build a contextual input. A context of several recent inputs is collected and each input is assigned a weight by the attention mechanism. The contextual input is a weighted sum of this context. We deliver both the contextual input and the current input to the model. (2) The second level is executed on the hidden state. We construct a contextual hidden state based on a context of adjacent hidden states and attention. We employ both the contextual hidden state and the current hidden state to establish the user's overall interest. Finally, parameters of HCA are learned by the Bayesian Personalized Ranking (BPR) framework [22] and the Back Propagation Through Time (BPTT) algorithm [23]. In summary, the main contributions are listed as follows:

- We propose to model several factors at each time for RNN. The short-term interest is mainly established by recent factors and this contextual modeling can relieve the limitation of ordered modeling of RNN.
- We propose to apply the attention mechanism to summarize the context. The attention can distinguish the importance of each item in a context and find out which items are important for the short-term interest.
- Experiments on two large real-world datasets reveal that the HCA network is very effective and outperforms the state-of-the-art methods. Our model can automatically focus on critical information.

2. Related work

We briefly review related works including sequential recommendation, capturing the long-term dependency, contextual modeling, and attention mechanism.

Recently, the recurrent neural network (RNN) based methods have become more powerful among sequential methods. Pooling-based representations of baskets are fed into RNN to make next basket recommendation [4]. Combined with multiple external information, RNN can predict a user's next behavior more accurately [24]. Incorporated with geographical distance and time interval, RNN gains the state-of-the-art performance for location prediction [5]. The RLBL model combines RNN and Log-BiLinear to make multi-behavioral prediction [7]. However, modeling user sequences directly using RNN is not adequate to fully acquire user expressions.

Many tasks prefer longer-term dependency, like speech recognition, language modeling, and so on. Considering that RNN can capture long-term dependency, in theory, gated activations like LSTM and GRU are devised and become popular. Recently, people design regularizers like dropout and zoneout which can help to keep more memory in RNN and further improve performance. In the beginning, people only employ dropout to the feed-forward connection in RNN [12]. Later, people take other ways. Moon et al. [13] propose a new dropout called rnnDrop for RNN. They apply dropout to recurrent connections and mainly focus on memory cells. Semeniuta et al. [14] propose recurrent dropout and apply it to the input and update gates in LSTM/GRU. Most recently, instead of forcing some units to zero in the dropout, Krueger et al. [15] design a novel regularizer called zoneout. By exactly keeping previous values, zoneout is more appropriate to preserve information than dropout for RNN and becomes state-of-the-art. However, both gated activations and regularizers focus on better capturing the long-term dependency, short-term dependency of RNN has not been fully studied yet.

Contextual modeling has been proven to be very important in different fields and tasks. When learning word vectors in word2vec, the prediction of a word is based on a context of recent words appearing before or after [25,26]. On scene labeling of computer vision, the patch is widely used. Episodic CAMN model makes contextualized representation for every referenced patch by using surrounding patches [27]. As the winner of the ImageNet object detection challenge of 2016, GBD-Net employs different resolutions and contextual regions to help identify objects [28]. Our contextual modeling mainly focuses on adjacent factors (e.g., recent inputs or hidden states) without target sequences, and do not use

any external information (e.g., time, location) which is also called contextual information [24,29–31].

Massive works benefit from the attention mechanism in recent years. Inspired by human perception, the attention mechanism enables models to concentrate on critical parts and construct a weighted sum. The mean-pooling and max-pooling, on the other hand, are unable to allocate appropriate credit assignment for individuals [21]. This mechanism is first introduced in image classification to directly explore the space around the handwritten digits by using visual attention [32]. Besides using contextual modeling, Episodic CAMN applies the attention mechanism to adaptively choose important patches [27]. In NLP related tasks, the attention mechanism is first introduced to Neural Machine Translation (NMT) [33]. This work jointly learns to align and translate words and conjectures the fixed-length vector (e.g., the last hidden state of the source language) is a bottleneck for the basic encoder-decoder structure [9].

Recently, there are some studies using both the context and the attention in the recommender system, where the context refers to the recent factor but not the external information. Some works are similar to ours, but in reality, they are considerably different and do not apply to our task. The study [34] is a regression model to predict target values. Its attention on the input is also to combine multiple elements but they are at the same time. However, modeling the short-term interest needs recent items that belong to different times. The attention on the hidden state is to build the relations between a target value and several hidden states. Unfortunately, the sequential recommendation in our work only has one input sequence for each user without the target sequence. At the same time, there are some studies that are indeed related to our network and MARank model [20] is the most related one. MARank captures long-term preference based on BPR framework and devises a multi-order attentive ranking model to represent short-term preference. In detail, MARank embeds a set of items each time and maps items' features by the residual network. By using the attention mechanism, MARank constructs the individual-level interactions. Furthermore, MARank applies another residual network to obtain union-level interaction. This work becomes the state-of-the-art for the sequential recommendation.

3. Proposed HCA network

In this section, we propose a novel network called Hierarchical Contextual Attention-based (HCA). We first formulate the problem and introduce the basic model of GRU. Then we present the contextual attention-based technique on the input and the hidden state respectively. Finally, we train the network with the BPR framework and the BPTT algorithm.

3.1. Problem formulation

In order to simplify the problem formulation of sequential recommendation, we take buying histories of online shopping as an example. Use $\mathcal{U} = \{u_1, \ldots, u_{|\mathcal{U}|}\}$ and $\mathcal{I} = \{i_1, \ldots, i_{|\mathcal{I}|}\}$ to represent the sets of users and items respectively. Let $\mathcal{I}^u = (i^u_1, \ldots, i^u_{|\mathcal{I}^u|})$ denote the items that the user u has purchased in the time order. Given each user's history $\mathcal{I}^u$, our goal is to recommend a list of items that a user may buy in the near future. The notation is listed in Table 1 for clarity.

3.2. Gated recurrent unit

GRU is very effective to deal with the gradient vanishing and exploding problem. It has an *update* gate $\boldsymbol{z}^t$ and a *reset* gate $\boldsymbol{r}^t$ to control the flow of information. The formulas are

$$\begin{aligned}
\boldsymbol{z}^t &= \sigma\left(\boldsymbol{U}_1\boldsymbol{x}_p^t + \boldsymbol{W}_1\boldsymbol{h}^{t-1} + \boldsymbol{b}_1\right)\\
\boldsymbol{r}^t &= \sigma\left(\boldsymbol{U}_2\boldsymbol{x}_p^t + \boldsymbol{W}_2\boldsymbol{h}^{t-1} + \boldsymbol{b}_2\right)\\
\tilde{\boldsymbol{h}}^t &= \tanh\left(\boldsymbol{U}_3\boldsymbol{x}_p^t + \boldsymbol{W}_3\left(\boldsymbol{r}^t \odot \boldsymbol{h}^{t-1}\right) + \boldsymbol{b}_3\right)\\
\boldsymbol{h}^t &= \left(1 - \boldsymbol{z}^t\right) \odot \boldsymbol{h}^{t-1} + \boldsymbol{z}^t \odot \tilde{\boldsymbol{h}}^t
\end{aligned} \tag{1}$$

where $\boldsymbol{x}_p^t \in \mathbb{R}^d$ is the input vector and t is the time step, d is the dimension, and $\odot$ is the element-wise product. The subscript p indicates an item is from the user sequence. $\tilde{\boldsymbol{h}}^t$ is the candidate state activated by $\tanh(x)$. The output $\boldsymbol{h}^t$ is the current hidden state. The last hidden state $\boldsymbol{h}^n$ is usually regarded as the sequence's representation, where n is the length of the sequence.

Table 1
Notation.

Notation	Explanation
$\mathcal{U}$, $\mathcal{I}$, $\mathcal{I}^u$	Set of users, set of items, sequence of items of user u
$\boldsymbol{x}_p$, $\boldsymbol{x}_q$	Positive item, negative item
$\hat{x}^t_{upq}$	Difference of preference of user u towards $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$ at the tth time
d	Dimension of the input item
$\boldsymbol{x}_p$, $\boldsymbol{h}$	Input item and hidden state of GRU
$\boldsymbol{x}_c$, $\boldsymbol{h}_c$	Contextual input, contextual hidden state
w_{x}, w_{h}	Window widths
$\boldsymbol{U}$, $\boldsymbol{W}$, $\boldsymbol{b}$	Transition matrices and bias of GRU
$\boldsymbol{V}$	Transition matrix for $\boldsymbol{x}_c$
$\boldsymbol{E}$, $\boldsymbol{F}$	Embedding matrices for $\boldsymbol{h}$ and $\boldsymbol{h}_c$
$\boldsymbol{h}_o$	Overall interest of $\boldsymbol{h}$ and $\boldsymbol{h}_c$
$\boldsymbol{r}_{\mathrm{x}}/\boldsymbol{Q}_{\mathrm{x}}$, $\boldsymbol{a}_{\mathrm{x}}$	Weight vector/matrix and attention weight for input item
$\boldsymbol{r}_{\mathrm{h}}/\boldsymbol{Q}_{\mathrm{h}}$, $\boldsymbol{a}_{\mathrm{h}}$	Weight vector/matrix and attention weight for hidden state

3.3. HCA

3.3.1. Contextual attention-based modeling on the input

The first level of HCA is on the input and the modeling process is represented in the bottom part of Fig. 1. At each time step, we collect a context of several recent items and build a contextual input $\boldsymbol{x}_c$ to explore the complicated correlations. Going along with the current original input $\boldsymbol{x}_p$, $\boldsymbol{x}_c$ is also delivered to the GRU unit to enhance the representation ability of the hidden state.

Take the current tth time step for instance. Let $\boldsymbol{C}^t_{\mathrm{x}}$ be a context matrix consisting of recent w_{x} inputs, where w_{x} is the window width of the context. Then the following attention mechanism will generate a vector $\boldsymbol{a}_{\mathrm{x}}$ consisting of w_{x} weights and form a weighted sum $\boldsymbol{x}^t_c$. The important information in $\boldsymbol{C}^t_{\mathrm{x}}$ will be automatically reserved.

$$\boldsymbol{C}^t_{\mathrm{x}} = \left[\boldsymbol{x}_p^{t-w_{\mathrm{x}}+1}; \cdots; \boldsymbol{x}_p^t\right] \qquad \boldsymbol{C}^t_{\mathrm{x}} \in \mathbb{R}^{w_{\mathrm{x}} \times d} \tag{2}$$

$$\boldsymbol{e}_{\mathrm{x}} = \boldsymbol{r}^{\mathrm{T}}_{\mathrm{x}} \tanh\left(\boldsymbol{Q}_{\mathrm{x}}(\boldsymbol{C}^t_{\mathrm{x}})^{\mathrm{T}}\right) \qquad \boldsymbol{e}_{\mathrm{x}} \in \mathbb{R}^{w_{\mathrm{x}}} \tag{3}$$

$$\boldsymbol{a}_{\mathrm{x}} = \mathrm{softmax}(\boldsymbol{e}_{\mathrm{x}}) \qquad \boldsymbol{a}_{\mathrm{x}} \in \mathbb{R}^{w_{\mathrm{x}}} \tag{4}$$

$$\boldsymbol{x}^t_c = \left(\boldsymbol{a}_{\mathrm{x}}\boldsymbol{C}^t_{\mathrm{x}}\right)^{\mathrm{T}} \qquad \boldsymbol{x}^t_c \in \mathbb{R}^d \tag{5}$$

where $\boldsymbol{r}_{\mathrm{x}} \in \mathbb{R}^d$ and $\boldsymbol{Q}_{\mathrm{x}} \in \mathbb{R}^{d \times d}$ are the weight vector and matrix for attention. $\boldsymbol{C}^t_{\mathrm{x}}$ is updated iteratively through the recurrent structure of GRU. The beginning of a sequence does not contain enough inputs and thus we apply zero vectors to make up this context matrix. The subscript x represents variables which are associated with the input. $\boldsymbol{x}^t_c$ is the contextual attention-based input.

Next, we add $\boldsymbol{x}^t_c$ to the basic GRU in Eq. (1) to rewrite the formula as

$$\begin{aligned}
\boldsymbol{z}^t &= \sigma\left(\boldsymbol{U}_1\boldsymbol{x}_p^t + \mathbf{V}_1\boldsymbol{x}_c^t + \boldsymbol{W}_1\boldsymbol{h}^{t-1} + \boldsymbol{b}_1\right)\\
\boldsymbol{r}^t &= \sigma\left(\boldsymbol{U}_2\boldsymbol{x}_p^t + \mathbf{V}_2\boldsymbol{x}_c^t + \boldsymbol{W}_2\boldsymbol{h}^{t-1} + \boldsymbol{b}_2\right)\\
\tilde{\boldsymbol{h}}^t &= \tanh\left(\boldsymbol{U}_3\boldsymbol{x}_p^t + \mathbf{V}_3\boldsymbol{x}_c^t + \boldsymbol{W}_3\left(\boldsymbol{r}^t \odot \boldsymbol{h}^{t-1}\right) + \boldsymbol{b}_3\right)\\
\boldsymbol{h}^t &= \left(1 - \boldsymbol{z}^t\right) \odot \boldsymbol{h}^{t-1} + \boldsymbol{z}^t \odot \tilde{\boldsymbol{h}}^t
\end{aligned} \tag{6}$$

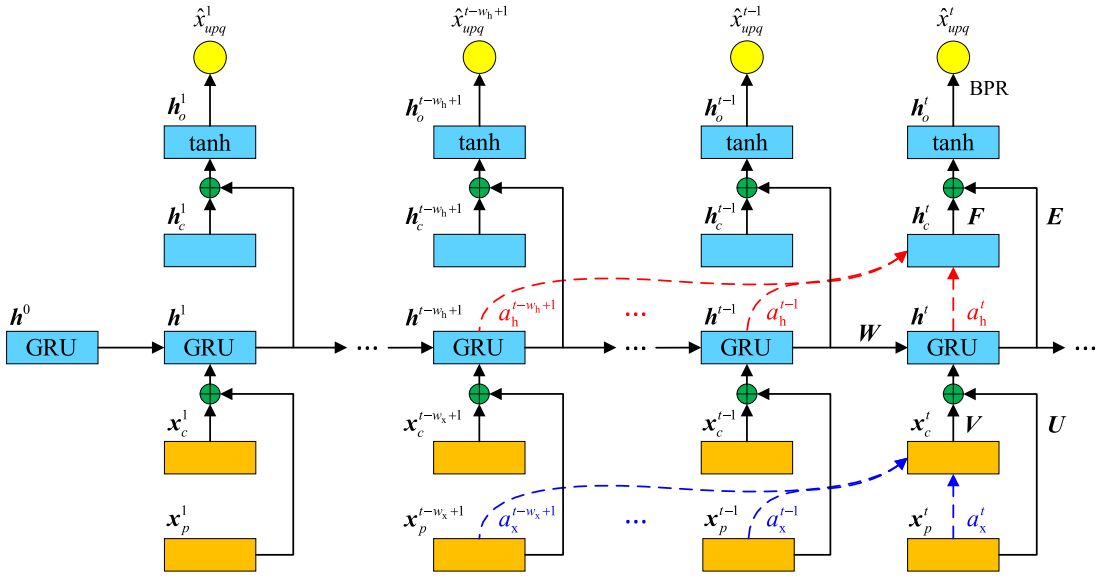


Fig. 1. Diagram of the HCA network. The contextual attention-based modeling is conducted on the input and hidden state respectively. Context of several recent inputs $(\boldsymbol{x}_p^{t-w_x+1}, \ldots, \boldsymbol{x}_p^{t-1}, \boldsymbol{x}_p^t)$ is used to explore the correlation among items. Context of several recent hidden states $(\boldsymbol{h}^{t-w_h+1}, \ldots, \boldsymbol{h}^{t-1}, \boldsymbol{h}^t)$ is used to strengthen the short-term interest and construct the user's overall interest. We only illustrate the whole modeling process at the tth time step in detail and select the same window widths.

where $\boldsymbol{V} \in \mathbb{R}^{3\times d\times d}$ formed by $\boldsymbol{V}_1$, $\boldsymbol{V}_2$, $\boldsymbol{V}_3$ is the transition matrix for $\boldsymbol{x}_c^t$. In this way, $\boldsymbol{h}^t$ contains not only information of original input $\boldsymbol{x}^t$ but also critical information of several recent inputs represented by $\boldsymbol{x}_c^t$. The representation ability of each hidden state is greatly enhanced.

In Eq. (6), we still need to feed $\boldsymbol{x}_p^t$ into the GRU unit in order to guarantee the modeling of original long-term interest in $\boldsymbol{h}^t$. However, as $\boldsymbol{x}_p^t$ and $\boldsymbol{x}_c^t$ are both in the Eq. (6), the consequence is that $\boldsymbol{x}_p^t$ still has the highest weight among the recent input items. This is a flaw in the acquired complicated correlations among items. Hence we execute further contextual attention-based modeling on the hidden state.

3.3.2. Contextual attention-based modeling on hidden state

The second level of HCA is on the hidden state and we illustrate the modeling process in the upper part of Fig. 1. At each time step, a context of several recent hidden states is collected and a contextual hidden state $\boldsymbol{h}_c$ is constructed to capture the critical short-term interest. Then we combine the $\boldsymbol{h}_c$ and the current $\boldsymbol{h}$ to obtain the user's overall interest $\boldsymbol{h}_o$.

The construction of contextual attention-based hidden state $\boldsymbol{h}_c^t$ is similar to that of $\boldsymbol{x}_c^t$:

$$\boldsymbol{C}_{\mathrm{h}}^t = \left[\boldsymbol{h}^{t-w_{\mathrm{h}}+1}; \ldots; \boldsymbol{h}^t\right] \qquad \boldsymbol{C}_{\mathrm{h}}^t \in \mathbb{R}^{w_{\mathrm{h}}\times d} \tag{7}$$

$$\boldsymbol{e}_{\mathrm{h}} = \boldsymbol{r}_{\mathrm{h}}^{\mathrm{T}} \tanh\left(\boldsymbol{Q}_{\mathrm{h}}(\boldsymbol{C}_{\mathrm{h}}^t)^{\mathrm{T}}\right) \qquad \boldsymbol{e}_{\mathrm{h}} \in \mathbb{R}^{w_{\mathrm{h}}} \tag{8}$$

$$\boldsymbol{a}_{\mathrm{h}} = \mathrm{softmax}(\boldsymbol{e}_{\mathrm{h}}) \qquad \boldsymbol{a}_{\mathrm{h}} \in \mathbb{R}^{w_{\mathrm{h}}} \tag{9}$$

$$\boldsymbol{h}_c^t = \left(\boldsymbol{a}_{\mathrm{h}}\boldsymbol{C}_{\mathrm{h}}^t\right)^{\mathrm{T}} \qquad \boldsymbol{h}_c^t \in \mathbb{R}^d \tag{10}$$

where $\boldsymbol{C}_{\mathrm{h}}^t$ is the context matrix of recent w_{h} hidden states. The operations on $\boldsymbol{C}_{\mathrm{h}}^t$ are the same with that on $\boldsymbol{C}_{\mathrm{x}}^t$. Variables associated with the hidden state have a subscript h. $\boldsymbol{h}_c^t$ holds the strengthened short-term interest in recent w_{h} time steps.

The final representation is a non-linear combination of $\boldsymbol{h}_c^t$ and $\boldsymbol{h}^t$. This is inspired by some works in NLP tasks, like recognizing textual entailment [35] and aspect-level sentiment classification [36].

$$\boldsymbol{h}_o^t = \tanh\left(\boldsymbol{E}\boldsymbol{h}^t + \boldsymbol{F}\boldsymbol{h}_c^t\right) \qquad \boldsymbol{h}_o^t \in \mathbb{R}^d \tag{11}$$

where $\boldsymbol{E} \in \mathbb{R}^{d\times d}$ and $\boldsymbol{F} \in \mathbb{R}^{d\times d}$ are embedding matrices. $\boldsymbol{h}_o^t$ is the user's current overall interest.

3.3.3. Network learning

The proposed network can be trained under the BPR framework and by using the classical BPTT algorithm. BPR is a powerful pairwise method for implicit feedback. Many RNN based methods have successfully applied BPR to train their models [4,5,7].

The training set $\mathcal{S}$ is formed by (u, p, q) triples:

$$\mathcal{S} = \{(u, p, q) | u \in \mathcal{U} \wedge \boldsymbol{x}_p \in \mathcal{I}^u \wedge \boldsymbol{x}_q \in \mathcal{I} \setminus \mathcal{I}^u\} \tag{12}$$

where u denotes the user. Used as subscripts, p or q indicates the item $\boldsymbol{x}$ is positive or negative. Item $\boldsymbol{x}_p$ is from the user's history $\mathcal{I}^u$, while item $\boldsymbol{x}_q$ is randomly chosen from the rest items. Then we calculate the user's difference of preferences for positive and negative items at each time step.k

$$\hat{x}_{upq}^t = \left(\boldsymbol{h}_o^t\right)^{\mathrm{T}}\left(\boldsymbol{x}_p^{t+1} - \boldsymbol{x}_q^{t+1}\right) \tag{13}$$

where $\boldsymbol{x}_p^{t+1}$ and $\boldsymbol{x}_q^{t+1}$ represent next positive and negative items respectively.

The objective function minimizes the following formula:

$$\Theta^* = \underset{\Theta}{\mathrm{argmin}} \sum_{(u,p,q)\in\mathcal{S}} -\ln\sigma\left(\hat{x}_{upq}\right) + \frac{\lambda_\Theta}{2}\|\Theta\|^2 \tag{14}$$

where $\Theta = \{\boldsymbol{X}, \boldsymbol{U}, \boldsymbol{W}, \boldsymbol{V}, \boldsymbol{b}, \boldsymbol{r}_{\mathrm{x}}, \boldsymbol{Q}_{\mathrm{x}}, \boldsymbol{r}_{\mathrm{h}}, \boldsymbol{Q}_{\mathrm{h}}, \boldsymbol{E}, \boldsymbol{F}\}$ is the set of parameters, and $\sigma(x) = 1/(1+e^{-x})$ is the *logistic* function. $\boldsymbol{X}$ is latent features of all items. $\boldsymbol{U}$ is a matrix formed by $\boldsymbol{U}_1$, $\boldsymbol{U}_2$, $\boldsymbol{U}_3$ used in Eqs. (1) and (6), which is similar to $\boldsymbol{W}$, $\boldsymbol{V}$, $\boldsymbol{b}$. $\lambda_\Theta \geqslant 0$ is the regularization parameter. Then, HCA-GRU is learned by stochastic gradient descent and BPTT. During the test process, we need to recalculate each user's final overall interest $\boldsymbol{h}_o^n$ by using the fixed parameter Θ.

4. Experimental results and analysis

In this section, we conduct experiments on two real-world datasets. First, we introduce the used datasets, evaluation metrics, and several baselines. Then we make a comparison between HCA and baselines. Finally, we present the window width selection and attention visualization.

Table 2
Datasets. We list the numbers of users, items, feedbacks, average sequence length, average number of an item and sparsity of both datasets.

Dataset	#users	#items	#feedbacks	Avg. seq. len.	Avg. num. of an item	Sparsity
Taobao	36,986	267,948	1,640,433	44.35	6.12	99.9834%
Outbrain	65,573	69,210	833,646	12.71	12.05	99.9816%

4.1. Experimental settings

4.1.1. Datasets

Experiments are carried out on two datasets collected from *Taobao*[1] and *Outbrain*[2]. We choose users by the length and do not delete items that appear less frequently.

- *Taobao* is a dataset for clothing matching competition on the *TianChi*[3] platform. User historical behavior data is applied to make the sequential recommendation. We hold users who purchase at least 30 times ($|\mathcal{I}^u| \geqslant 30$).
- *Outbrain* is a dataset for click prediction on Kaggle. We apply the page views log here. As the log is too tremendous and over 2 billion rows, we choose a sample version and convert it to the user sequence. Finally, we keep users who have no less than 10 views ($|\mathcal{I}^u| \geqslant 10$).

The basic statistics of two datasets are listed in Table 2. Both datasets have massive sequential implicit feedbacks. Taobao has far more items than Outbrain, which naturally creates a huge search space and may degrade performance.

4.1.2. Evaluation metrics

Performance is evaluated on the test set under metrics consisting of Recall and Normalized Discounted Cumulative Gain (NDCG) [37]. The former one is an evaluation of unranked retrieval sets while the latter one is an evaluation of ranked retrieval results. Here we consider Top-k (e.g., $k = 5$, 10 and 15) recommendation. The top-bias property of NDCG is considerable for recommendation [38]. We select the first 80% of each user sequence as the training set and the rest 20% as the test set.

4.1.3. Comparison

We compare HCA with several comparative methods:

- *Random*: Items are randomly recommended for all users.
- *POP*: This method recommends the most popular items in the training set to users.
- *BPR*: This method refers to the BPR-MF for implicit feedback [22]. This BPR framework is state-of-the-art among pairwise methods.
- *GRU*: RNN is the state-of-the-art sequential baseline for recommendation [4]. We apply an extension of RNN called GRU as it can capture the long-term interest. Compared with our proposed method, GRU models sequences without any contextual attention-based modeling.
- *Zoneout*: Zoneout is state-of-the-art for regularizing RNN by stochastically preserving previous hidden activations [15]. We implement this method based on GRU and execute zoneout on the hidden states.
- *MARank*: MARank is the state-of-the-art for sequential recommendation [20]. It balances the long-term and short-term preferences by unifying individual-level and union-level item interactions.

Our HCA has 3 networks which are abbreviated as HCA(xw_x, hw_h), HCA(xw_x) and HCA(hw_h), where w_x and w_h are values of window widths. The first one is the complete network. The rest two are subnetworks where we only conduct contextual attention-based modeling on the input or hidden state. Besides, the parameter Θ is initialized by a uniform distribution $[-0.5, 0.5]$. The dimension is set as $d = 20$. The learning rate is set as $\alpha = 0.01$ and the epochs are 100. Regularization parameter is set as $\lambda_\Theta = 0.001$. As for zoneout, it applies random identity-mask to keep previous hidden states. The corresponding best zoneout probabilities are set as $z_h = (0.5, 0.1)$ on two datasets respectively. The code is written by using Theano and is available on GitHub[4].

4.2. Analysis of experimental results

Table 3 illustrates the performances of all compared methods on two datasets. We list the performance of the HCA network under best window widths which will be discussed in the next section. Please note that all values in Tables 3, 5 and Fig. 2 are represented in percentage, and the '%' symbol is omitted in these tables and figures. All methods have run four times and we use average values as the results.

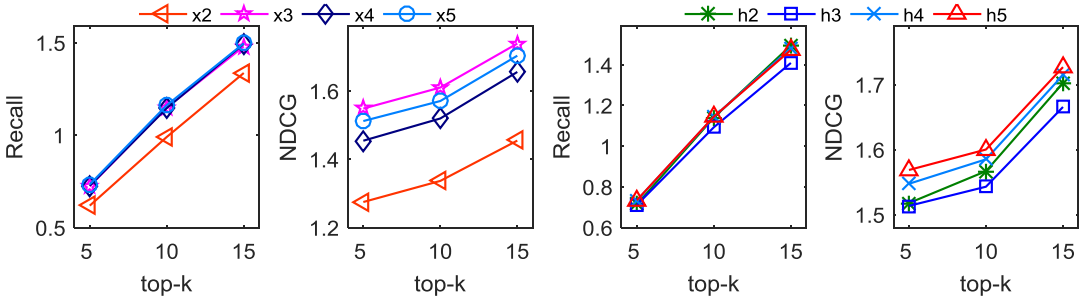
Performance of Our HCA. From a global perspective, our HCA is very effective and outperforms the baselines. Within HCA, the subnetwork performances of HCA(xw_x) and HCA(hw_h) are very close, and the latter is slightly better than the former. The complete HCA(xw_x, hw_h) obtains the best performance. In terms of the 2 metrics, HCA performs differently. It has a great improvement on Recall@5, but the improvement visibly gets smaller on Recall@10. In contrast, the improvement on NDCG is very considerable and quite stable. The complete HCA is about 50% more efficient than the GRU on NDCG. HCA has a very prominent performance in sorting. This is because HCA can effectively capture the complicated correlations among recent items and distinguish which items the user are more interested in. When we compare the performance of the two datasets, we find HCA dose very well. Although the absolute values of two datasets vary greatly, HCA obtains similar relative improvements.

Comparison with different methods. In this part, we compare the differences between these methods. BPR is a widely used static method. As it can not model the dynamics in user sequences (e.g., changes of user interest), it is obviously less effective than GRU. As for zoneout, it is more effective than GRU on Taobao, while it is competitive with GRU or is worse than that on Outbrain. The reason is that blindly enhancing the long-term dependency is not a perfect fit for the modeling of user sequences, because user interest would change over time. On Taobao, people purchase commodities and user interest usually changes slowly. In such a situation, zoneout can improve performance by capturing longer-term dependency than GRU. However, Outbrain is collected from users' page views log, and the average sequence length of Outbrain is much shorter than that of Taobao. We can think that GRU is enough to capture long-term dependency. Besides, user interest can change sharply when clicking web pages. Zoneout's

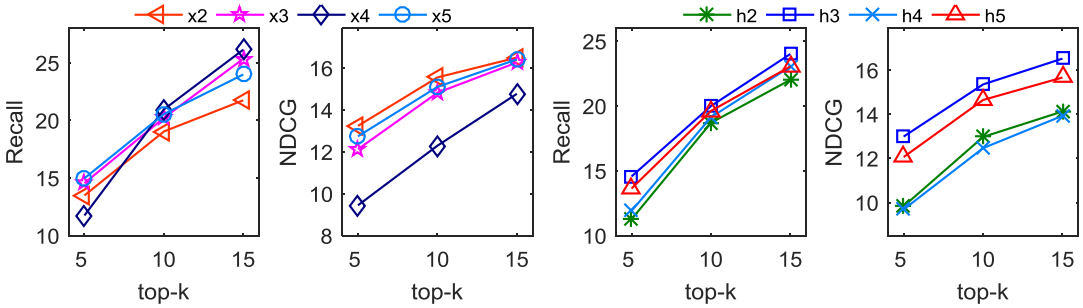
[1] https://tianchi.shuju.aliyun.com/datalab/dataSet.htm?id=13.
[2] https://www.kaggle.com/c/outbrain-click-prediction.
[3] https://tianchi.aliyun.com/.
[4] https://github.com/cuiqiang1990/HCA.

Table 3
Evaluation of different methods on two datasets. We generate Top-5/10/15 items for each user. Values are represented in percentage, and so do the other tables and figures.

Dataset	Method	@5		@10		@15	
		Recall	NDCG	Recall	NDCG	Recall	NDCG
Taobao	Random	0.002	0.003	0.003	0.003	0.004	0.003
	POP	0.025	0.046	0.100	0.085	0.138	0.104
	BPR	0.373	0.860	0.560	0.872	0.723	0.925
	GRU	0.478	1.088	0.810	1.099	1.123	1.178
	Zoneout (z_h=0.5)	0.596	1.222	0.946	1.281	1.244	1.403
	MARank	**0.814**	**1.810**	1.101	**1.754**	1.305	**1.840**
	HCA(x3)	0.724	1.548	1.149	1.610	1.480	1.739
	HCA(h5)	0.735	1.569	1.145	1.600	1.472	1.727
	HCA(x5,h5)	0.765	1.618	**1.191**	1.659	**1.520**	1.787
Outbrain	Random	0.007	0.006	0.015	0.009	0.022	0.012
	POP	0.026	0.019	1.188	0.535	1.191	0.536
	BPR	4.092	4.050	8.523	5.992	23.750	10.935
	GRU	11.145	9.451	17.560	11.987	21.413	12.969
	Zoneout (z_h=0.1)	11.052	8.192	18.384	11.351	20.900	12.050
	MARank	10.571	7.807	16.428	9.445	24.918	12.351
	HCA(x2)	13.485	13.237	19.033	15.551	21.744	16.498
	HCA(h3)	14.562	12.988	19.993	15.353	24.010	16.522
	HCA(x2,h3)	**16.396**	**14.225**	**23.328**	**17.097**	**26.836**	**18.280**



(a) Taobao



(b) Outbrain

Fig. 2. The election of different window widths for two subnetworks on two datasets. We generate Top-5,10 and 15 items. Subnetworks are HCA(x2 ~ 5) and HCA(h2 ~ 5), and we omit the 'HCA()' for conciseness in the figures.

better modeling of long-term dependency is harmful to the modeling of user's current interest in such case. Different from other tasks like speech recognition, under the condition of ensuring long-term dependency by using GRU/LSTM, it is more advantageous to enhance RNN to do sequential recommendation by strengthening the modeling of user's short-term interest.

Comparison with MARank. Obviously, MARank is a strong baseline. On Taobao, MARank performs best on Top-5 and is comparable with our HCA on Top-10/15. While on Outbrain, although MARank considerably outperforms the BPR, it is evidently not as good as our model. Briefly, MARank and our HCA are based on BPR and GRU, respectively. Both models can greatly strengthen short-term interest. On the one hand, MARank captures multi-order item interactions by using the residual network and the attention, which helps to relieve the data sparsity. The average numbers of an item are 6.12 and 12.05 on Taobao and Outbrain, respectively. MARank tends to have better performance on the sparser Taobao dataset. On the other hand, as mentioned in the previous paragraph, user interest changes slowly on Taobao while it would have a sharp change on Outbrain. The sequential effect is less significant on Taobao. Therefore, BPR and GRU perform close to each other on Taobao while GRU is obviously better than BPR on Outbrain. However, MARank is based on BPR to capture long-term interest, which leads to defects in capturing sequential patterns. The two reasons cause a difference in the performance of MARank on two datasets. Accordingly, our HCA has the advantage to better capture the sequential pattern than MARank.

Table 4
Training time of each model in each iteration on two datasets. Statistics are expressed in seconds and the batch size is one.

Dataset	(a) BPR	(b) GRU	(c) Zoneout	(d) MARank	(e) HCA	d vs. e
Taobao	90	188	192	3682	362	1017%
Outbrain	45	128	132	967	194	498%

In this paragraph, we illustrate another advantage of the speed of our HCA by analyzing the computational complexity between HCA and MARank. We obtain the training time of each iteration on two datasets and the statistics are in Table 4. The training time of MARank is ten times and five times as long as our HCA on Taobao and Outbrain, respectively. Actually, MARank has much more operations than our HCA each time when modeling a user sequence. MARank has an individual-level interaction with two k-layer residual networks for the current user vector and several items' vectors. This process has $2k$ recurrent activations and k+2 attentions. Furthermore, the union-level interaction in MARank also has a k-layer residual network, which results in k activations. In total, MARank has $3k$ activations and k+2 attentions. In our HCA, it has 1 attention on items, 1 attention on hidden states, 3 activations in a GRU unit (Eq. (6)) and 1 activation in computing the final user vector (Eq. (11)). Our HCA has a total of 4 activations and 2 attentions. Therefore, compared with MARank, although our HCA is not always best, it is evidently faster and more efficient than MARank.

In conclusion, our HCA performs best for the whole, enhances RNN significantly, has a fast speed, and can better model user behaviors in different scenarios.

4.3. Analysis of window width

In this subsection, we explore how many factors modeled each time are most conducive to the modeling of short-term interest because the short-term interest mainly comes from recent behaviors. In another word, we investigate the best window width for HCA. Both window widths w_x and w_h range in [2, 3, 4, 5]. The basic GRU can be considered as a special case of our network when both w_x and w_h are 1.

4.3.1. Subnetwork: HCA(xw_x), HCA(hw_h)

First we select the best window widths for two subnetworks on two datasets. Fig. 2 illustrates the performance of 2 metrics. For two subnetworks with different window widths, the difference is slight on Recall, while there is an obvious difference on NDCG. Consequently, the best window widths for two subnetworks are chosen as $w_x = 3,\ w_h = 5$ on Taobao and $w_x = 2,\ w_h = 3$ on Outbrain, respectively, according to NDCG. Both w_x and w_h on Taobao are larger than those on Outbrain. This is caused by differences in user behaviors. Taobao consists of purchase history while Outbrain contains page views log. If a user buys clothes online, his short-term interest changes slowly. However, user's short-term interest often has a quick or even sharp change when he views web pages. Therefore, items with complicated correlations tend to appear in small windows in Outbrain.

4.3.2. Complete network: HCA(xw_x, hw_h)

Based on the best window widths of the two subnetworks, we pick proper w_x, w_h for the complete HCA network. Grid search over combinations of $w_x \in [2, 3, 4, 5]$ and $w_h \in [2, 3, 4, 5]$ is very time-consuming. Instead, we fix one with its best value and adjust the other. Results are shown in Table 5. Combinations $w_x = 5,\ w_h = 5$ and $w_x = 2,\ w_h = 3$ are best for Taobao and Outbrain respectively. The combination of the best window widths of the subnetworks can generate good results but it does not guarantee the best. We can also see that the hierarchical structure can leverage the advantages of the two subnetworks and further achieve better performance.

4.4. Analysis of attention weights

After we acquire the best window widths, we discuss which factors in a context contribute to the short-term interest. The attention mechanism generates a vector to summarize contextual information, and the attention weights can be obtained in Eqs. (4) and (9). We take the best networks HCA(x5,h5) on Taobao and HCA(x2,h3) on Outbrain for instance. We choose one sequence from each of the two datasets. Weights of the two sequences at the last time step are listed in Table 6.

On one hand, we focus on weights of inputs. We look at each line of weights. For example, in the third line on Taobao, the weight of item x^{n-6} is 0.396 but x^{n-3} only gets 0.013. This gap is about 30 times. Obviously, attention can capture the most important item with the highest weight. We check each column. If an item is very important in a context, it would be probably also very important in the next context as it has critical information, and vice versa. For example, items x^{n-6} and x^{n-2} on Taobao have about 0.3, but the weights of x^{n-3} are all less than 0.03. On the other hand, we study weights of hidden states. The weights have little difference with each other on Taobao and hold a descending order on Outbrain respectively. This is because each hidden state has its own important information for the modeling of short-term interest. To sum up, our HCA relieves the limitation of the

Table 5
Evaluation of different window widths for the complete HCA network on two datasets.

Dataset	Method	@5		@10		@15	
		Recall	NDCG	Recall	NDCG	Recall	NDCG
Taobao	HCA(x3,h2)	0.693	1.472	1.084	1.521	1.392	1.639
	HCA(x3,h3)	0.675	1.476	1.054	1.517	1.372	1.646
	HCA(x3,h4)	0.678	1.472	1.051	1.515	1.382	1.644
	HCA(h5,x2)	0.715	1.526	1.113	1.570	1.453	1.701
	HCA(h5,x3)	0.735	1.556	1.170	1.613	**1.533**	1.753
	HCA(h5,x4)	0.735	1.567	1.141	1.604	1.482	1.740
	HCA(h5,x5)	**0.765**	**1.618**	**1.191**	**1.659**	1.520	**1.787**
Outbrain	HCA(x2,h2)	13.515	12.902	20.706	15.800	24.194	17.173
	HCA(x2,h3)	16.396	14.225	**23.328**	**17.097**	**26.836**	**18.280**
	HCA(x2,h4)	**16.698**	**14.306**	21.721	16.747	25.893	18.106
	HCA(x2,h5)	14.799	11.886	22.144	14.992	26.750	16.519
	HCA(h3,x3)	13.183	10.883	20.448	14.070	25.481	15.501
	HCA(h3,x4)	12.482	9.733	21.996	13.512	25.712	14.974
	HCA(h3,x5)	12.368	10.869	20.264	14.161	25.073	15.946

Table 6
Examples of attention weights of two users' sequences on two datasets, respectively. We apply the best HCA(x5,h5) on Taobao and HCA(x2,h3) on Outbrain. The HCA(x2,h3) is taken as an example to explain how weights are organized. Its window widths are $w_x = 2$, $w_h = 3$. The context of the last hidden state (h^n) has 3 hidden states (h^{n-2}, h^{n-1}, h^n). Every one of these three hidden state has 2 weights of the input. For example, the corresponding context of input for h^n is x^{n-1} and x^n.

(a) Taobao - HCA(x5,h5)

Hidden states	h^{n-8}	h^{n-6}	h^{n-6}	h^{n-5}	h^{n-4}	h^{n-3}	h^{n-2}	h^{n-1}	h^n
Weights					0.207	0.187	0.203	0.199	0.204
Inputs	x^{n-8}	x^{n-7}	x^{n-6}	x^{n-5}	x^{n-4}	x^{n-3}	x^{n-2}	x^{n-1}	x^n
Weights	0.170	0.271	**0.289**	0.144	0.126				
		0.323	**0.345**	0.171	0.150	0.011			
			0.396	0.196	0.172	0.013	0.223		
				0.252	0.220	0.016	**0.287**	0.225	
					0.290	0.021	**0.378**	0.296	0.015

(b) Outbrain - HCA(x2,h3)

Hidden states	h^{n-3}	h^{n-2}	h^{n-1}	h^n
Weights		**0.366**	0.327	0.307
Inputs	x^{n-3}	x^{n-2}	x^{n-1}	x^n
Weights	**0.661**	0.339		
		0.597	0.403	
			0.319	**0.681**

ordered modeling of RNN by modeling several factors (item, hidden state) each time. The attention mechanism can capture the most important information and assign time-independent non-monotonic weights.

5. Conclusion

In this work, we have proposed a novel network called Hierarchical Contextual Attention-based (HCA) network for the sequential recommendation. As GRU can capture the long-term interest, we focus on strengthening the short-term interest. We propose a contextual attention-based technique and apply it to the input and the hidden state, respectively. A contextual input is built to capture the correlation among adjacent items and enhance the representation ability of each hidden state. Then we construct a contextual hidden state to extract the short-term interest within adjacent hidden states. The contextual and current hidden states are combined together as the user's overall interest. Experiments have verified the state-of-the-art performance of our proposed network, especially in terms of NDCG. The results also demonstrate that the contextual attention-based technique can effectively relieve the limitation of ordered modeling as well as capture the critical information, and the hierarchical structure can fully leverage the advantage of different factors.

In the future, we would like to apply the proposed model to other scenarios, like next-basket recommendation, music recommendation, and so on. For the model itself, we would like to further enhance RNN by explicitly modeling user's long-term interest. At each time step, instead of using long-term interest acquired in the current hidden state, we would apply the attention mechanism to all the hidden states up to now to form the long-term interest. After that, we can achieve a better balance between short-term and long-term interest.

Acknowledgements

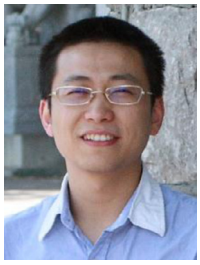
This work is jointly supported by National Natural Science Foundation of China (61772528, 61871378) and National Key Research and Development Program of China (2016YFB1001000).

References

[1] S. Rendle, C. Freudenthaler, L. Schmidt-Thieme, Factorizing personalized Markov chains for next-basket recommendation, in: Proceedings of the WWW, 2010, pp. 811–820.
[2] J. Chen, C. Wang, J. Wang, A personalized interest-forgetting Markov model for recommendations, in: Proceedings of the AAAI, 2015, pp. 16–22.
[3] R. He, J. McAuley, Fusing similarity models with Markov chains for sparse sequential recommendation, in: Proceedings of the ICDM, 2016, pp. 191–200.
[4] F. Yu, Q. Liu, S. Wu, L. Wang, T. Tan, A dynamic recurrent model for next basket recommendation, in: Proceedings of the SIGIR, 2016, pp. 729–732.
[5] Q. Liu, S. Wu, L. Wang, T. Tan, Predicting the next location:a recurrent model with spatial and temporal contexts., in: Proceedings of the AAAI, 2016, pp. 194–200.
[6] B. Hidasi, M. Quadrana, A. Karatzoglou, D. Tikk, Parallel recurrent neural network architectures for feature-rich session-based recommendations, in: Proceedings of the RecSys, 2016, pp. 241–248.
[7] Q. Liu, S. Wu, L. Wang, Multi-behavioral sequential prediction with recurrent log-bilinear model, TKDE 29 (6) (2017) 1254–1267.
[8] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[9] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, in: Proceedings of the EMNLP, 2014.
[10] J. Wang, K. Shi, Q. Huang, S. Zhong, D. Zhang, Stochastic switched sampled–data control for synchronization of delayed chaotic neural networks with packet dropout, Appl. Math. Comput. 335 (2018) 211–230.
[11] K. Shi, Y. Tang, S. Zhong, C. Yin, X. Huang, W. Wang, Nonfragile asynchronous control for uncertain chaotic Lurie network systems with bernoulli stochastic process, Int. J. Robust Nonlinear Control 28 (5) (2018) 1693–1714.
[12] V. Pham, T. Bluche, C. Kermorvant, J. Louradour, Dropout improves recurrent neural networks for handwriting recognition, in: Proceedings of the ICFHR, IEEE, 2014, pp. 285–290.
[13] T. Moon, H. Choi, H. Lee, I. Song, Rnndrop: a novel dropout for RNNS in ASR, in: Proceedings of the ASRU, IEEE, 2015, pp. 65–70.
[14] S. Semeniuta, A. Severyn, E. Barth, Recurrent dropout without memory loss, in: Proceedings of the COLING, 2016, pp. 1757–1766.
[15] D. Krueger, T. Maharaj, J. Kramár, M. Pezeshki, N. Ballas, N.R. Ke, A. Goyal, Y. Bengio, A. Courville, C. Pal, Zoneout: regularizing rnns by randomly preserving hidden activations, in: Proceedings of the ICLR, 2017.
[16] J. Chung, K. Kastner, L. Dinh, K. Goel, A.C. Courville, Y. Bengio, A recurrent latent variable model for sequential data, in: Proceedings of the NIPS, 2015, pp. 2980–2988.
[17] M. Fraccaro, S.K. Sønderby, U. Paquet, O. Winther, Sequential neural models with stochastic layers, in: Proceedings of the NIPS, 2016, pp. 2199–2207.
[18] A.G.A.P. Goyal, A. Sordoni, M.-A. Côté, N.R. Ke, Y. Bengio, Z-forcing: training stochastic recurrent networks, in: Proceedings of the NIPS, 2017, pp. 6713–6723.
[19] J. Tang, W. Ke, Personalized top-n sequential recommendation via convolutional sequence embedding, in: Proceedings of the WSDM, 2018, pp. 565–573.
[20] L. Yu, C. Zhang, S. Liang, X. Zhang, Multi-order attentive ranking model for sequential recommendation, in: Proceedings of the AAAI, 2019.

[21] S. Zhai, K.-h. Chang, R. Zhang, Z.M. Zhang, Deepintent: learning attentions for online advertising with recurrent neural networks, in: Proceedings of the SIGKDD, 2016, pp. 1295–1304.
[22] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, BPR: Bayesian personalized ranking from implicit feedback, in: Proceedings of the UAI, 2009, pp. 452–461.
[23] P.J. Werbos, Backpropagation through time: what it does and how to do it, Proc. IEEE 78 (10) (1990) 1550–1560.
[24] Q. Liu, S. Wu, D. Wang, Z. Li, L. Wang, Context-aware sequential recommendation, in: Proceedings of the CDM, 2016, pp. 1053–1058.
[25] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Proceedings of the NIPS, 2013, pp. 3111–3119.
[26] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: Proceedings of the ICML, 2014, pp. 1188–1196.
[27] A.H. Abdulnabi, B. Shuai, S. Winkler, G. Wang, Episodic CAMN: contextual attention-based memory networks with iterative feedback for scene labeling, in: CVPR, 2017.
[28] X. Zeng, W. Ouyang, J. Yan, H. Li, T. Xiao, K. Wang, Y. Liu, Y. Zhou, B. Yang, Z. Wang, et al., Crafting gbd-net for object detection Crafting GBD-NET for object detection, TPAMI 40 (9) (2017) 2109–2123.
[29] X. Wang, D. Rosenblum, Y. Wang, Context-aware mobile music recommendation for daily activities, in: Proceedings of the ACM MM, 2012, pp. 99–108.
[30] N. Hariri, B. Mobasher, R. Burke, Context-aware music recommendation based on latenttopic sequential patterns, in: Proceedings of the RecSys, 2012, pp. 131–138.
[31] A. Beutel, P. Covington, S. Jain, C. Xu, J. Li, V. Gatto, E.H. Chi, Latent cross: making use of context in recurrent recommender systems, in: Proceedings of the WSDM, ACM, 2018, pp. 46–54.
[32] V. Mnih, N. Heess, A. Graves, et al., Recurrent models of visual attention, in: Proceedings of the NIPS, 2014, pp. 2204–2212.
[33] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, in: Proceedings of the ICLR, 2015.
[34] Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, G. Cottrell, A dual-stage attention-based recurrent neural network for time series prediction, in: Proceedings of the IJCAI, 2017.
[35] T. Rocktäschel, E. Grefenstette, K.M. Hermann, T. Kočiskỳ, P. Blunsom, Reasoning about entailment with neural attention, in: Proceedings of the ICLR, 2016.
[36] Y. Wang, M. Huang, X. Zhu, L. Zhao, Attention-based LSTM for aspect-level sentiment classification, in: Proceedings of the EMNLP, 2016, pp. 606–615.
[37] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, X. Cheng, Learning hierarchical representation model for next basket recommendation, in: Proceedings of the SIGIR, 2015, pp. 403–412.
[38] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, A. Hanjalic, N. Oliver, TFMAP: optimizing map for top-n context-aware recommendation, in: Proceedings of the SIGIR, 2012, pp. 155–164.

Qiang Cui received his B.S. degree from Shandong University, China, in 2013. He is currently working toward the Ph.D. degree in Center for Research on Intelligent Perception and Computing (CRIPAC) at National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (CASIA), Beijing, China. His research interests include data mining, machine learning, recommender systems and information retrieval. He has published papers in the leading international journals and conferences such as TKDE and PAKDD.



Shu Wu received his B.S. degree from Hunan University, China, in 2004, M.S. degree from Xiamen University, China, in 2007, and his Ph.D. degree from University of Sherbrooke, Quebec, Canada. He is an Associate Professor in Center for Research on Intelligent Perception and Computing (CRIPAC) at National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (CASIA). His research interests include data mining, information retrieval, and recommendation systems. He has published more than 20 papers in the areas of data mining and information retrieval at international journals and conferences, such as IEEE TKDE, IEEE THMS, AAAI, ICDM, SIGIR, and CIKM.

Yan Huang received the B.S. degree from University of Electronic Science and Technology of China (UESTC) in 2012, and the Ph.D. degree from University of Chinese Academy of Sciences (UCAS) in 2017. Since July 2017, he has joined the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (CASIA) as an assistant professor. His research interests include machine leaning and pattern recognition. He has published papers in the leading international journals and conferences such as TPAMI, TIP, NIPS, CVPR, ICCV and ECCV.

Liang Wang received both the B.S. and M.S. degrees from Anhui University in 1997 and 2000, respectively, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences (CASIA) in 2004. From 2004 to 2010, he was a research assistant at Imperial College London, United Kingdom, and Monash University, Australia, a research fellow at the University of Melbourne, Australia, and a lecturer at the University of Bath, United Kingdom, respectively. Currently, he is a full professor of the Hundred Talents Program at the National Lab of Pattern Recognition, CASIA. His major research interests include machine learning, pattern recognition, and computer vision. He has widely published in highly ranked international journals such as IEEE TPAMI and IEEE TIP and leading international conferences such as CVPR, ICCV, and ICDM. He is an IEEE Fellow and an IAPR Fellow.