Multi-behavioral Sequential Prediction for Collaborative Filtering

Qiang Liu, Shu Wu, 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)
{qiang.liu, shu.wu, wangliang}@nlpr.ia.ac.cn

***Abstract*—With the rapid growth of Internet applications, sequential prediction in collaborative filtering has become an emerging and crucial task. Given the behavioral history of a specific user, predicting his or her next choice plays a key role in improving various online services. Meanwhile, there are more and more scenarios with multiple types of behaviors, while existing works mainly study sequences with a single type of behavior. As a widely used approach, Markov chain based models are based on a strong independence assumption. As two classical neural network methods for modeling sequences, recurrent neural networks can not well model short-term contexts, and the log-bilinear model is not suitable for long-term contexts. In this paper, we propose a Recurrent Log-BiLinear (RLBL) model. It can model multiple types of behaviors in historical sequences with behavior-specific transition matrices. RLBL applies a recurrent structure for modeling long-term contexts. It models several items in each hidden layer and employs position-specific transition matrices for modeling short-term contexts. Experimental results show that the proposed RLBL model yields significant improvements over the competitive compared methods on two datasets, i.e., the Movielens-1M dataset and the Tmall dataset with different numbers of behavior types.**

***Index Terms*—Collaborative filtering, sequential prediction, multi-behavior, recurrent log-bilinear.**

I. Introduction

Nowadays, Collaborative Filtering (CF) plays an important role in a large number of applications, e.g., recommender systems, information retrieval and social network analysis. Conventional CF methods focus on modeling users preference based on their historical choices of items and always ignore the sequential information. It is reasonable to assume that user preferences change with his or her behavioral sequence. Meanwhile, rather than with merely one type of behaviors, e.g., purchasing in e-commerce and clicking on websites, there are many sequential scenarios with multiple types of behaviors towards items, e.g., clicking, purchasing, adding to favorites in e-commerce and downloading, using, uninstalling in app usage. Accordingly, it is necessary to model multi-behavioral sequences and collaboratively predict what a user will prefer next under a specific behavior. For instance, multiple types of behaviors, i.e., posting, sharing and commenting, on social media has been separately modeled and studied recently, which makes great contribution to user interest detection [35]. Besides e-commerce and other Internet applications, multi-behavioral sequential prediction can be implemented for social good, such as predicting security events in a specific area [10] [30] or predicting air quality [36].

Nowadays, some efforts have been put into developing CF methods with sequential information [10] [22] [29] [34]. To the best of our knowledge, none of existing methods are designed for modeling sequences with multiple types of behaviors. And if we directly treat different behaviors towards one item as different elements in sequences, or simply ignore the differences among behaviors, conventional methods will have difficulty in revealing the correlations among behaviors and items. As shown in the example of app usage in Figure 1, different behaviors reveal users' different attitudes towards apps. Downloading and using means you may like the app, while uninstalling means you do not like the app and similar ones should not be recommended. So, it is essential to find a proper way to reveal the correlations among behaviors and items.

Moreover, existing methods still have their own limitations even for single-behavioral sequences. Markov Chain (MC) based models [32] [22] [19] have become the most popular methods for sequential prediction. MC based models aim to predict the users' next behavior based on the past behaviors. A transition matrix is estimated, which can give the probability of an action based on the previous ones. However, a major problem of MC based models is that all the components are independently combined, indicating that it makes strong independence assumption among multiple factors [29].

Recently, Recurrent Neural Networks (RNN) have been successfully employed to model temporal dependency for different applications, such as sentence modeling tasks [12] [13] [14], video modeling [5], sequential click prediction [34] and location prediction [10]. When modeling the sequential data, RNN assumes that the temporal dependency changes monotonously along with positions in a sequence. This means that, one element, e.g., a word, a frame and a product, in a sequence usually has more significant effect than the previous one for prediction. Such rules may well model words in a sentence or frames in a video, since adjacent words or frames have significant correlation. The larger the distance between two words or two frames, the smaller the correlation. However, for behavior prediction tasks, this assumption does not confirm to complex real situations, especially for the most recent

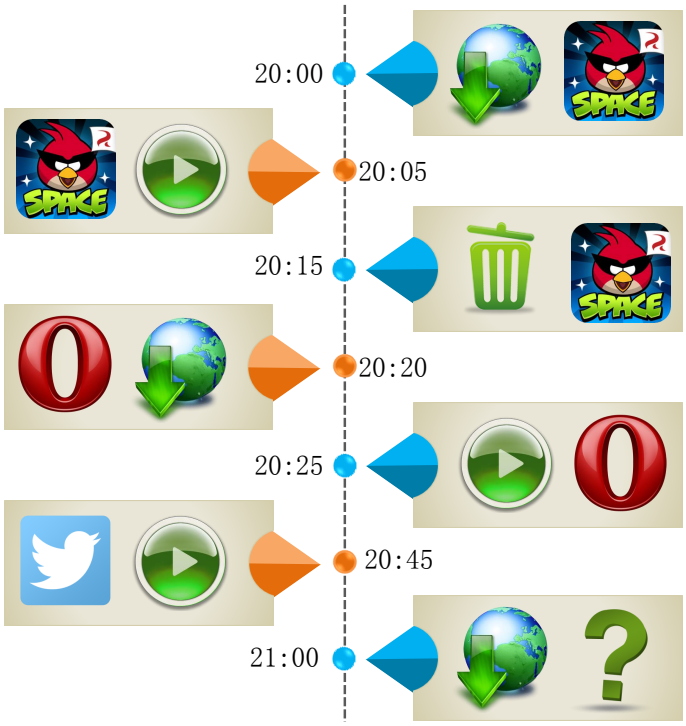

Fig. 1. Taking app usage prediction as an example of multi-behavioral sequential prediction. This example shows a user's behaviors towards apps in an hour, including downloading, using and uninstalling. We can predict what app the user is going to download or use next.

elements in historical sequences. Sometimes, several most recent elements have similar effects on users' next behavior. For instance, if you went to the gym, the restaurant and the shopping market yesterday morning, afternoon and evening respectively, these three behaviors may have similar effects on your behaviors today. Sometimes, most recent elements have more complex effects on the future. For instance, going to the gym yesterday has dominant effects on how you exercise today, and what you ate at the restaurant yesterday or what you bought at the shopping market yesterday can affect what you want to eat today a lot. There is no guarantee that one element has more or less significant effect than the previous one. The effects of most recent elements in modeling human behaviors are much more complicated than that in modeling sentences or videos. But RNN can only tell us that behaviors in yesterday morning have more significant effects than behaviors in yesterday afternoon, and behaviors in yesterday afternoon have more significant effects than behaviors in yesterday evening. Accordingly, we can say that, RNN can not well model short-term contexts in a sequence.

Different from the recurrent architecture in RNN based language models [12] [13] [14], the Log-BiLinear (LBL) model [16] represents each word in a sentence, i.e., each position in a sequence, with a specific matrix. It can better model the complex situations of local contexts in sequences. But when the sequence is too long, a maximal length is usually set. And in real behavior prediction scenarios, length of behavioral sequences is usually not fixed. So, LBL can not well model long-term contexts in a sequence.

In this paper, to overcome above shortcomings of conventional methods and model multi-behavioral sequences, we propose a novel sequential prediction methods, i.e., **Recurrent Log-BiLinear (RLBL)** model. *First*, to capture the properties of different types of behaviors in historical sequences, we employ behavior-specific transition matrices in our model. To the best of our knowledge, this is the first work which is designed for predicting multi-behavioral sequences *Second*, we design RLBL model as a recurrent architecture to capture long-term contexts in sequences. It models several elements in each hidden layer and uses position-specific transition matrices to capture short-term contexts of the historical sequence. Our RLBL not only can model the subtle characteristics of the most recent items in a sequence, but also can deal with long-term contexts with a recurrent structure.

The main contributions of this work are listed as follows:

- We firstly address the problem of multi-behavioral sequential prediction, which is a significant problem in sequential prediction. And we use behavior-specific matrices to represent the effects of different types of behaviors.
- The RLBL model incorporates position-specific matrices and the recurrent structure, which can well model both short- and long-term contexts in historical sequences.
- Experiments conducted on two real-world datasets show that RLBL is effective and clearly outperforms the state-of-the-art methods.

II. Related Works

In this section, we review several types of methods for sequential prediction and time-aware prediction, i.e., time-aware factorization methods, markov chain based methods and neural network based methods.

A. Time-aware Factorization Methods

Matrix factorization (MF) based methods [18] [9] have become the state-of-the-art approach to collaborative filtering. The basic objective of MF is to factorize a user-item rating matrix into two low rank matrices, each of which represents the latent factors of users or items. The original rating matrix can be approximated via the multiplying calculation. Nowadays, MF based methods have been extended for more general and complex situations [20] [23]. Among them, time-aware factorization based models have been extensively studied. Tensor Factorization (TF) [1] [31] treats time slices as another dimension and generates latent vectors of time slices via factorization to capture the underlying properties in the behavioral history. TimeSVD++ [8] learns time-aware representations for users and items in different time slices. However, factorization based models have difficulties in generating latent representations for time slices which has never or seldom appeared in the training data. Thus, factorization based models are not able to accurately predict item in the future time slices.

B. Markov Chain Based Methods

The MC based methods are widely used models for sequential prediction tasks [32]. MC based models predict users' next behaviors via estimating a transition matrix, which gives the probability of an action based on the previous ones. Via factorization of the personalized probability transition matrices of users, Factorizing Personalized Markov Chain (FPMC) [22] can provide more accurate prediction for each sequence. FPMC is also extended by using the user group [19] or incorporating the location constraint [3]. However, the main drawback of MC based models is the independent combination of the past components, which lies in a strong independence assumption and confines the prediction accuracy. Then MC based methods are extended by using representation learning. Hierarchical Representation Model (HRM) [29] learns the hierarchical representation of behaviors in the last transaction and in the past history of a user to predict behaviors in the next transaction. And Personalized Ranking Metric Embedding (PRME) [6] learns embeddings of users according to distances between locations. These methods still face a problem that they only model items in the most recent history and previous items can only be modeled by constant user latent vectors. Thus, except items in the most recent history, other items after model training will be ignored. User representations can not change dynamically along with behavioral sequences.

C. Neural Network Based Methods

Recently, some prediction models, especially language models [15], are proposed based on neural networks. The most classical neural language model is proposed via a single layer neural network [2]. Among variety language models, RNN has been the most successful one in modeling sentences [12] [13] [14]. It has successfully applied in variety natural language processing tasks, such as machine translation [4] [27], conversation machine [25] [26] and image caption [11] [28]. Recently, RNN based models also achieve successive results in other areas. For video analyzing, RNN brings satisfying results for action recognition [5]. Incorporating users' each clicking as an input element of each layer, RNN has greatly improved the performance of sequential click prediction [34]. Spatial-Temporal Recurrent Neural Netwrks (ST-RNN) [10] learns geographical distance-specific transition matrices in RNN framework for location prediction. And Dynamic REcurrent bAsket Model (DREAM) [33] uses pooling methods in each layer of RNN for aggregating items in one transaction and achieves state-of-the-art performance in next basket recommendation [33]. However, when modeling sequential data, RNN assumes that temporal dependency changes monotonously along with the positions in a sequence, which means one element in a sequence usually has more significant effect than the previous one for prediction. This is usually suitable for words in sentences or frames in videos. But it does not confirm to practical situations for predicting behaviors, especially for the most recent elements of a historical sequence. Several most recent elements may usually have similar or even more complex effects on a user's next choice. But RNN can only tell us that the most recent item has more significant effect than the previous items. So, we can say that RNN can not well model short-term contexts in behavior modeling. LBL [16] is another widely-used language model, which represents elements at each position in a sequence with specific matrices. And a hierarchical softmax [17] is utilized to accelerate LBL model. However, when sequences are too long, a maximal length is usually set and long-term contexts are discarded. So, LBL can not well model long-term contexts in sequences, which often exist in real behavior prediction situations.

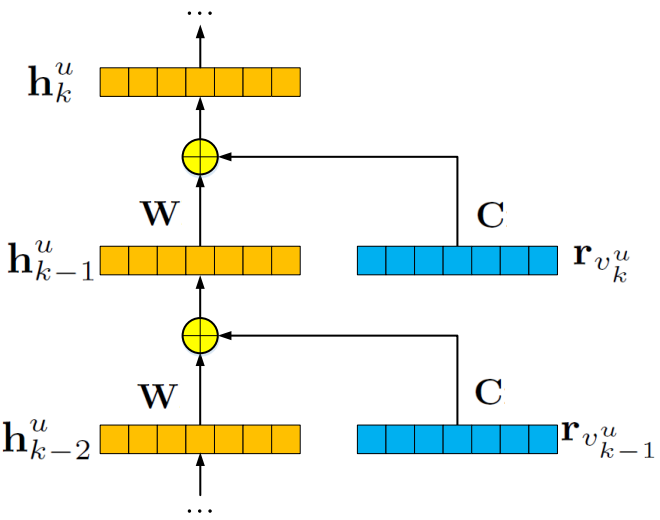

Fig. 2. Illustration of the Recurrent Neural Networks (RNN) model. RNN is a recurrent architecture with multiple hidden layers. The hidden status of RNN changes dynamically along with sequences, where the trend is monotonous. RNN has difficulty in modeling short-term contexts in behavioral sequences.

III. Problem Definition

The multi-behavioral sequential prediction problem we study in this work can be formulated as follows. We have a set of users and a set of items denoted as $U = \{u_1, u_2, ...\}$ and $V = \{v_1, v_2, ...\}$ respectively. Multiple types of behaviors are denoted as $B = \{b_1, b_2, ...\}$. Each behavior of user u is associated with a behavioral type and a timestamp. Then the sequential behavioral history of user u consists of items $V^u = \{v_1^u, v_2^u, ...\}$ and corresponding behavioral types $B^u = \{b_1^u, b_2^u, ...\}$. Given behavioral history of users towards items, the task is to predict what a specific user will choose next under a specific behavior.

Here, taking the application in e-commerce as an example, there will be four types of behaviors (i.e., clicking, purchasing, adding to favorites and adding to shopping chart) denoted as $\{b_1, b_2, b_3, b_4\}$. The task is to predict which item a user would like to click, purchase, add to favorites or add to shopping chart next. Similarly, in the app usage, there will be three types of behaviors (i.e., downloading, using and uninstalling) denoted as $\{b_1, b_2, b_3\}$. Then the task becomes predicting which app a user would like to download, use or uninstall next.

IV. Recurrent Log-bilinear Model (RLBL)

In this section, we present the recurrent log-bilinear model. We first introduce the RNN model and LBL model, then detail the architecture of RLBL with a single type of behaviors and introduce how RLBL can be employed to model multiple types of behaviors.

A. Recurrent Neural Networks

The architecture of RNN is shown in Figure 2. It consists of an input layer, an output unit, multiple hidden layers, as well as inner weight matrices [34]. The activation values of the hidden layers are computed as:

$$\mathbf{h}_k^u = f\left(\mathbf{W}\mathbf{h}_{k-1}^u + \mathbf{C}\mathbf{r}_{v_k^u}\right) \ , \tag{1}$$

where $\mathbf{h}_k^u \in \mathbb{R}^d$ denotes the hidden representation of user u at position k in a sequence, $\mathbf{r}_{v_k^u} \in \mathbb{R}^d$ denotes the representation of the kth input item of user u. $f(x)$ is the activation function. $\mathbf{C} \in \mathbb{R}^{d\times d}$ and $\mathbf{W} \in \mathbb{R}^{d\times d}$ mean the transition matrix for the current items and the previous status respectively. $\mathbf{W}$ can propagate sequential signals, and $\mathbf{C}$ can capture users' current behavior. This activation process can be repeated iteratively and then the status at each position in a sequence can be calculated.

B. Log-bilinear Model

The Log-BiLinear (LBL) model [16] is a deterministic model that may be viewed as a feedforward neural network with a single linear hidden layer [7]. Using LBL for the sequential prediction problem, the final predicted representation of a sequence is generated based on the input items and the transition matrices at each position. As shown in Figure 3, in the LBL model, the representation at next position is a linear prediction:

$$\mathbf{h}_k^u = \sum_{i=0}^{n-1} \mathbf{C}_i \mathbf{r}_{v_{k-i}^u} \ , \tag{2}$$

where $\mathbf{C}_i \in \mathbb{R}^{d\times d}$ denotes the transition matrix for the corresponding position in a sequence, and n is the number of elements modeled in a sequence.

C. Modeling Single Type of Behaviors

As discussed in the previous sections, though both RNN and LBL have achieved satisfying results, they still have their own drawbacks. RNN can not well handle short-term contexts in a sequence, while LBL can not well model long-term contexts.

To capture short-term and long-term contexts in historical sequences simultaneously, instead of modeling only one element in each hidden layer in RNN, we model several elements in each hidden layer and incorporate position-specific matrices into the recurrent architecture. As illustrated in Figure 4, given a user u, the hidden representation of the user at the position k in a sequence can be computed as:

$$\mathbf{h}_k^u = \mathbf{W}\mathbf{h}_{k-n}^u + \sum_{i=0}^{n-1} \mathbf{C}_i \mathbf{r}_{v_{k-i}^u} \ , \tag{3}$$

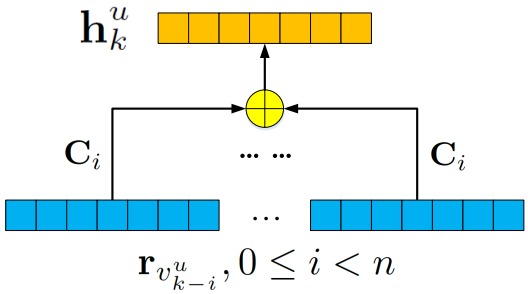

Fig. 3. Illustration of the Log-BiLinear (LBL) model. LBL is a feedforward neural network with a single linear hidden layer. In LBL, each position in sequences is modeled with a specific transition matrix. And a maximal number of modeled elements is usually set. LBL has difficulty in modeling long-term contexts in behavioral sequences.

where n is the number of input items modeled in one layer of RLBL, which is called the window width in this paper. The position-specific transition matrices $\mathbf{C}_i \in \mathbb{R}^{d\times d}$ captures the impact of short-term contexts, i.e., the ith item in one layer of RLBL, on user behaviors. And the characteristics of users' long-term history are modeled via the recurrent framework. Moreover, when we only consider one input item in each layer and set the window width $n = 1$, the formulation of RLBL will be as the same as that of RNN ignoring the nonlinear activation function.

Notice that, when the sequence is shorter than the window width or the predicted position is at the very first part of a sequence, i.e., $k < n$. Equation 3 should be rewritten as:

$$\mathbf{h}_k^u = \mathbf{W}\mathbf{h}_0^u + \sum_{i=0}^{k-1} \mathbf{C}_i \mathbf{r}_{v_{k-i}^u} \ , \tag{4}$$

where $\mathbf{h}_0^u = \mathbf{u}_0$, denoting the initial status of users. The initial status of all users should be the same because personal information does not exist when a user has not selected an item. This representation $\mathbf{u}_0$ can be used to model cold start users. The equation in this special situation can be viewed as the same as that of a regular LBL model.

D. Modeling Multiple Types of Behaviors

Although there exist some scenarios with one type of behavior, e.g., purchasing in e-commerce and clicking on websites, there are much more applications with multiple types of behaviors towards items. For instance, users will click items, purchase items and add items to favorites in e-commerce. And users may download apps, use apps and uninstall apps. Thus, it is necessary to model multi-behavioral sequences and collaboratively predict what a user will choose next under a specific behavior.

We can simply ignore different types of behaviors, or treat different behaviors towards one item as different elements in conventional models. However, it is hard to model the correlation among different behaviors towards one item. Here, we incorporate behavior-specific matrices to capture properties

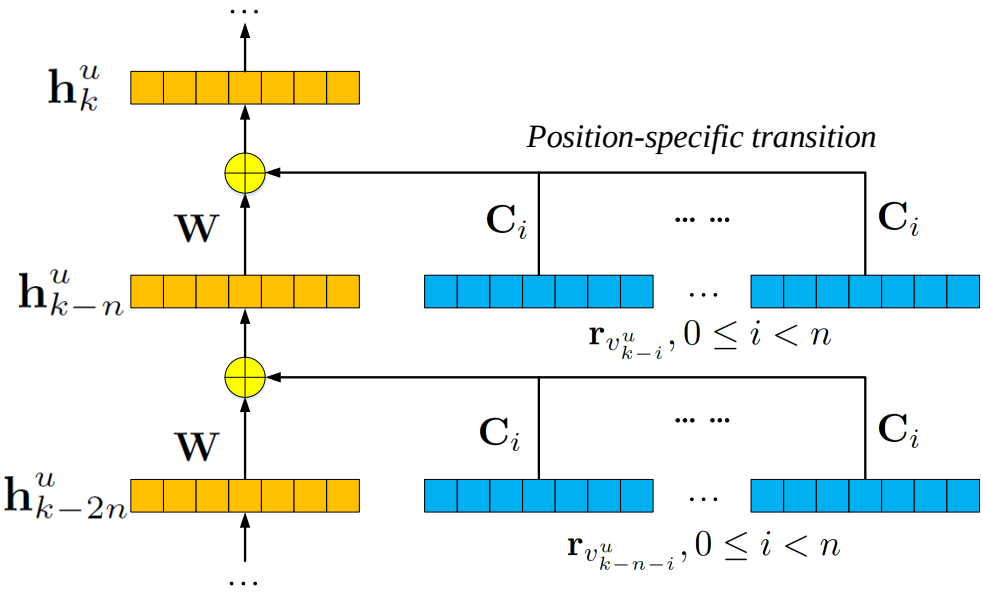

Fig. 4. Illustration of the Recurrent Log-BiLinear (RLBL) model. RLBL employs a recurrent architecture to capture long-term contexts. It models several elements in each hidden layer and incorporates position-specific transition matrices to capture short-term contexts in a historical sequence. Behavior-specific matrices can be incorporated in RLBL to capture multiple types of behaviors in sequences.

of multiple types of behaviors. Then, the representation of user u at position k can be calculated as:

$$\mathbf{h}_k^u = \mathbf{W}\mathbf{h}_{k-n}^u + \sum_{i=0}^{n-1} \mathbf{C}_i \mathbf{M}_{b_{k-i}^u} \mathbf{r}_{v_{k-i}^u} \ , \tag{5}$$

where $\mathbf{M}_{b_i^u} \in \mathbb{R}^{d \times d}$ denotes a behavior-specific transition matrix modeling the corresponding behavior on the ith item of user u. Note that, behavior-specific matrices can be omitted if there is only one type of behavior. Incorporating behavior-specific matrices, RLBL is the first approach which can be used to model the underlying properties of different types of behaviors in historical sequences.

Now, via calculating inner product, the prediction of whether user u would conduct behavior b on item v at the sequential position $k+1$ can be made as:

$$y_{u,k+1,b,v} = (\mathbf{s}_k^u)^T \mathbf{M}_b \mathbf{r}_v = (\mathbf{h}_k^u + \mathbf{u}_u)^T \mathbf{M}_b \mathbf{r}_v \ , \tag{6}$$

where $\mathbf{s}_k^u$ denotes the representation for the status of user u at the sequential position k, containing dynamic representation $\mathbf{h}_k^u$ and static latent representation $\mathbf{u}_u \in \mathbb{R}^d$.

E. Parameter Learning

In this subsection, we introduce the learning process of RLBL with Bayesian Personalized Ranking (BPR) [21] and Back Propagation Through Time (BPTT) [24].

BPR [21] is a state-of-the-art pairwise ranking framework for the implicit feedback data. BPR has been used as objective function for learning of RNN based models in behavioral prediction tasks [10] [33]. The basic assumption of BPR is that a user prefers a selected element than a negative one. Formally, we need to maximize the following probability:

$$p(u, k+1, b, v \succ v') = g(y_{u,k+1,b,v} - y_{u,k+1,b,v'}) \ , \tag{7}$$

where v' denotes a negative sample, and $g(x)$ is a nonlinear function which is selected as:

$$g(x) = \frac{1}{1+e^{-x}} \ . \tag{8}$$

Incorporating the negative log likelihood, we can minimize the following objective function equivalently:

$$J = \sum \ln(1 + e^{-(y_{u,k+1,b,v} - y_{u,k+1,b,v'})}) + \frac{\lambda}{2} \|\mathbf{\Theta}\|^2 \ , \tag{9}$$

where $\mathbf{\Theta} = \{\mathbf{U}, \mathbf{R}, \mathbf{W}, \mathbf{C}, \mathbf{M}\}$ denotes all the parameters to be estimated, λ is a parameter to control the power of regularization. And the derivations of J with respect to the parameters can be calculated as:

$$\frac{\partial J}{\partial \mathbf{u}_u} = \sum \frac{\mathbf{M}_b(\mathbf{r}_{v'} - \mathbf{r}_v) l(u, k+1, b, v \succ v')}{1 + l(u, k+1, b, v \succ v')} + \lambda \mathbf{u}_u \ ,$$

$$\frac{\partial J}{\partial \mathbf{r}_v} = -\sum \frac{(\mathbf{M}_b)^T (\mathbf{h}_k^u + \mathbf{u}_u) l(u, k+1, b, v \succ v')}{1 + l(u, k+1, b, v \succ v')} + \lambda \mathbf{r}_v \ ,$$

$$\frac{\partial J}{\partial \mathbf{r}_{v'}} = \sum \frac{(\mathbf{M}_b)^T (\mathbf{h}_k^u + \mathbf{u}_u) l(u, k+1, b, v \succ v')}{1 + l(u, k+1, b, v \succ v')} + \lambda \mathbf{r}_{v'} \ ,$$

$$\frac{\partial J}{\partial \mathbf{M}_b} = \sum \frac{(\mathbf{h}_k^u + \mathbf{u}_u)(\mathbf{r}_{v'} - \mathbf{r}_v)^T l(u, k+1, b, v \succ v')}{1 + l(u, k+1, b, v \succ v')} + \lambda \mathbf{M}_b \ ,$$

$$\frac{\partial J}{\partial \mathbf{h}_k^u} = -\sum \frac{\mathbf{M}_b(\mathbf{r}_{v'} - \mathbf{r}_v) l(u, k+1, b, v \succ v')}{1 + l(u, k+1, b, v \succ v')} \ ,$$

where

$$l(u, k+1, b, v \succ v') = e^{-(y_{u,k+1,b,v} - y_{u,k+1,b,v'})} \ .$$

The derivations of the output layer have been calculated. Under each layer of the recurrent structure, similar to the conventional RNN model, RLBL can be trained by using the Back Propagation Through Time (BPTT) algorithm [24], which has been used in practical sequential prediction models [10] [34]. For user u, given the derivation $\frac{\partial J}{\partial \mathbf{h}_k^u}$ of the representation $\mathbf{h}_k^u$ at sequential position k, the corresponding gradient of parameters at the hidden layer can be calculated as:

$$\frac{\partial J}{\partial \mathbf{h}_{k-n}^u} = \mathbf{W}^T \frac{\partial J}{\partial \mathbf{h}_k^u} \ ,$$

$$\frac{\partial J}{\partial \mathbf{W}^T} = \frac{\partial J}{\partial \mathbf{h}_k^u} (\mathbf{h}_{k-n}^u)^T \ ,$$

$$\frac{\partial J}{\partial \mathbf{r}_{v_{k-i}^u}} = (\mathbf{M}_{b_{k-i}^u})^T (\mathbf{C}_i)^T \frac{\partial J}{\partial \mathbf{h}_k^u} \ ,$$

$$\frac{\partial J}{\partial \mathbf{C}_i} = \frac{\partial J}{\partial \mathbf{h}_k^u} (\mathbf{r}_{v_{k-i}^u})^T (\mathbf{M}_{b_{k-i}^u})^T \ ,$$

$$\frac{\partial J}{\partial \mathbf{M}_{b_{k-i}^u}} = (\mathbf{C}_i)^T \frac{\partial J}{\partial \mathbf{h}_k^u} (\mathbf{r}_{v_{k-i}^u})^T \ .$$

This process can be repeated iteratively, and the gradients of all the parameters are obtained. Then, the model can be learned via Stochastic Gradient Descent (SGD) until converge.

V. Experiments

In this section, we empirically investigate the performance of RLBL. We conduct our experiments on two scenarios with different numbers of behavioral types. We first introduce our experimental settings. Then we conduct experiments to compare RLBL with different window width. We also conduct experiments to compare performances of single behavior and multiple behaviors, as well as other methods with single behavior. Finally, we study the performance of models under different length of behavioral history.

A. Experimental Settings

Our experiments are conducted on two real datasets with different numbers of behavioral types:

- **Movielens-1M**[1] is a widely used dataset, associated with timestamps, for the rating prediction in recommender systems. The ratings are divided into five levels, indicating users' different levels of preference, which can be viewed as five different types of behaviors. With this dataset, we plan to predict which movie a user will rate 5 or 4 stars next.
- **Tmall**[2] is a dataset collected from Tmall[3], one of the biggest online shopping websites in China. The temporal information of the dataset is extracted based on the day level. It contains four different types of behaviors: clicking, purchasing, adding to favorites and adding to shopping cart. It suits for the task of collaborative prediction on multi-behavioral sequences. On this dataset, we plan to predict what users will purchase next.

For each behavioral sequence of these three datasets, we use first 70% of the items in the sequence for training, following 10% data as the validation set for tuning parameters, e.g., the dimensionality of latent representations, and remaining 20% for testing. The regularization parameter is set as $\lambda = 0.01$. And we use line search to select learning rates in each iteration.

We compare RLBL and TA-RLBL with some state-of-the-art sequential methods.

- **FPMC** [22] extends conventional MC methods and factorizes personalized probability transition matrices of users. It is a widely-used method for sequential prediction and next basket recommendation.
- **HRM** [29] learns the representation of behaviors in the previous transaction and predicts next behaviors. It has become a state-of-the-art method for next basket recommendation.
- **RNN** [34] is a state-of-the-art method for the sequential prediction. It has been successfully applied in some applications, such as sentence modeling, click prediction, location prediction and next basket recommendation.

Considering FPMC and HRM predict future behaviors according to behaviors in the last transaction, we need to split transactions in different datasets according to corresponding application scenarios. So, we set the length of transaction in the Movielens dataset, the Global Terrorism Database and the Tmall dataset as one week, one month and one day respectively.

As above methods can not model multi-behavioral sequences, when conducting compared methods on multi-behavioral datasets, we ignore different types of behaviors in behavioral histories. This means we treat different behaviors towards one item as the same.

Moreover, to investigate the performance of our proposed methods and compared methods, we select several widely-used evaluation metrics for our experiments.

- **Recall@k** and **F1-score@k** are two important metrics for ranking tasks. The evaluation score for our experiments is computed according to where the next selected item appears in the predicted list. We report Recall@k and F1-score@k with $k = 1$, 2, 5 and 10 in our experiments. The larger the value, the better the performance.
- **Mean Average Precision (MAP)** is another widely used global evaluation in ranking tasks, which measure the quality of the whole ranking list. Top-bias property of MAP is particularly significant in evaluating ranking tasks such as top-n recommendation. The larger the value, the better the performance.

[1] http://grouplens.org/datasets/movielens/

[2] https://102.alibaba.com/competition/addDiscovery/index.htm

[3] https://www.tmall.com/

TABLE I
COMPARISON OF RLBL WITH VARYING WINDOW WIDTH n AND DIMENSIONALITY $d = 8$.

(a) Performance on the Movielens dataset.

method	n	Recall@1	Recall@2	Recall@5	Recall@10	F1-score@1	F1-score@2	F1-score@5	F1-score@10	MAP
RLBL	2	0.0067	0.0103	0.0333	0.0508	0.0067	0.0069	0.0111	0.0093	0.0377
	3	0.0070	0.0104	0.0334	0.0510	0.0070	0.0070	0.0111	0.0093	0.0381
	4	0.0070	0.0107	0.0338	0.0520	0.0070	0.0072	0.0113	0.0095	0.0385
	5	0.0070	0.0108	0.0343	0.0527	0.0070	0.0072	0.0114	0.0096	0.0386
	6	**0.0071**	**0.0112**	**0.0354**	**0.0538**	**0.0071**	**0.0074**	**0.0118**	**0.0098**	**0.0395**
	7	0.0070	0.0111	0.0354	0.0543	0.0070	0.0074	0.0118	0.0099	0.0393
	8	0.0070	0.0108	0.0351	0.0535	0.0070	0.0072	0.0117	0.0097	0.0390

(b) Performance on the Tmall dataset.

method	n	Recall@1	Recall@2	Recall@5	Recall@10	F1-score@1	F1-score@2	F1-score@5	F1-score@10	MAP
RLBL	2	0.1507	0.2170	0.3712	0.4690	0.1507	0.1447	0.1237	0.0853	0.2704
	3	0.1480	0.2515	**0.4118**	0.5176	0.1480	0.1677	**0.1373**	0.0941	0.2781
	4	0.1467	0.2311	0.3646	0.4953	0.1467	0.1541	0.1215	0.0901	0.2689
	5	**0.1600**	0.2158	0.3975	0.5519	**0.1600**	0.1439	0.1325	0.1003	**0.2836**
	6	0.1502	0.2272	0.3822	**0.5596**	0.1502	0.1515	0.1274	**0.1017**	0.2806
	7	0.1493	**0.2553**	0.4074	0.5272	0.1493	**0.1702**	0.1358	0.0959	0.2819
	8	0.1387	0.2324	0.4019	0.5395	0.1387	0.1549	0.1340	0.0981	0.2770

B. Comparison among different window width

To compare the performances of our proposed RLBL model with different window width, we conduct experiments on the two datasets with varying window size n. The results evaluated by Recall, F1-score and MAP are illustrated in Table I. These experimental results provide some hints in selecting the best window width n for RLBL in our experiments. Performances of our models on Movielens are stable and the best performances are obviously achieved at $n = 6$. On the Tmall dataset, the performances are not so stable evaluated by different metrics. We can select the best parameters according to the global metric MAP, which considers all the positions in a ranking list. Then the best window width for the Tmall dataset is $n = 5$. For the rest of our experiments, we report the performances of RLBL under the best window width. Moreover, on the Tmall dataset, for metrics Recall@p and F1-score@p, there seems existing a rough pattern. For smaller p, better recall values and F1-score values of RLBL are achieved with smaller window width n. While for larger p, better recall values and F1-score values of RLBL are achieved with larger window width n.

C. Multiple Behaviors VS. Single Behavior

We have analyzed performances of RLBL modeling multiple behaviors. To investigate the impact of multiple behaviors and single behavior on prediction effectiveness, we need to obtain performances of RLBL modeling a single behavior. So, we can ignore multiple types of behaviors in behavioral sequences when implementing RLBL to obtain their performances under a single type of behavior. We also compare our methods with some other methods with single behavior.

The performance comparison of modeling multiple behaviors and single behavior evaluated by Recall, F1-score and MAP on three datasets is shown in Table II. We can clearly observe the significant improvements brought by modeling multiple behaviors. Comparing with modeling single behavior, MAP improvements of RLBL modeling multiple behaviors are 2.92% and 6.41% on two datasets respectively. Moreover, we can also see that, even ignoring multiple types of behaviors, RLBL can still outperform conventional RNN with a relatively significant advantage, which indicates the effectiveness of position-specific transition. Meanwhile, comparing results of RLBL in Table I with results of RNN in Table II, even not with the best window width, most of results of RLBL are still better than the performance of RNN. This indicates the effectiveness and stability of RLBL with varying window width.

TABLE II
COMPARISON OF MULTIPLE BEHAVIORS AND SINGLE BEHAVIOR.

(a) Performance on the Movielens dataset with dimensionality $d = 8$ and window width $n = 6$.

method	Recall@1	Recall@1	Recall@1	MAP
FPMC	0.0054	0.0255	0.0440	0.0332
HRM	0.0060	0.0288	0.0462	0.0345
RNN	0.0063	0.0318	0.0484	0.0362
RLBL (Single)	0.0068	0.0343	0.0519	0.0384
RLBL (Multiple)	**0.0071**	**0.0354**	**0.0538**	**0.0395**

(b) Performance on the Tmall dataset with dimensionality $d = 8$ and window width $n = 5$.

method	Recall@1	Recall@1	Recall@1	MAP
FPMC	0.0866	0.2253	0.3405	0.1811
HRM	0.0956	0.2488	0.3703	0.2001
RNN	0.1283	0.3410	0.4397	0.2432
RLBL (Single)	0.1389	0.3581	0.5277	0.2666
RLBL (Multiple)	**0.1600**	**0.3822**	**0.5519**	**0.2836**

VI. CONCLUSIONS

In this paper, we have proposed two novel multi-behavioral sequential prediction methods, i.e. recurrent log-bilinear model and time-aware recurrent log-bilinear model. We build our

model under a recurrent structure. RLBL models several elements in each hidden layer and incorporate position-specific transition matrices. With such architecture, RLBL can well model both short- and long-term contexts in a historical sequence. Besides, to capture multiple types of behavior in behavioral sequences, behavior-specific matrices are designed and applied for each type of behavior. Then, to incorporate time difference information in behavioral sequences, we further extend the RLBL model and propose a time-aware recurrent log-bilinear model with time-specific transition matrices. Modeling time difference information, TA-RLBL can further improves the performance of RLBL in sequential prediction. The experimental results on three real datasets show that both RLBL and TA-RLBL outperforms state-of-the-art sequential prediction models.

REFERENCES

[1] M. T. Bahadori, Q. R. Yu, and Y. Liu. Fast multivariate spatio-temporal analysis via low rank tensor learning. In *Annual Conference on Neural Information Processing Systems*, pages 3491–3499, 2014.

[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003.

[3] C. Cheng, H. Yang, M. R. Lyu, and I. King. Where you like to go next: Successive point-of-interest recommendation. In *International Joint Conference on Artificial Intelligence*, pages 2605–2611, 2013.

[4] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734, 2014.

[5] Y. Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1110–1118. IEEE, 2015.

[6] S. Feng, X. Li, Y. Zeng, G. Cong, Y. M. Chee, and Q. Yuan. Personalized ranking metric embedding for next new poi recommendation. In *International Joint Conference on Artificial Intelligence*, pages 2069–2075, 2015.

[7] R. Kiros, R. Zemel, and R. R. Salakhutdinov. A multiplicative model for learning distributed text-based attribute representations. In *Annual Conference on Neural Information Processing Systems*, pages 2348–2356, 2014.

[8] Y. Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[9] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.

[10] Q. Liu, S. Wu, L. Wang, and T. Tan. Predicting the next location: A recurrent model with spatial and temporal contexts. In *AAAI Conference on Artificial Intelligence*, pages 194–200, 2016.

[11] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *International Conference on Learning Representations*, 2015.

[12] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur. Recurrent neural network based language model. In *INTERSPEECH*, volume 2, page 3, 2010.

[13] T. Mikolov, S. Kombrink, L. Burget, J. H. Cernocky, and S. Khudanpur. Extensions of recurrent neural network language model. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5528–5531. IEEE, 2011.

[14] T. Mikolov, S. Kombrink, A. Deoras, L. Burget, and J. Cernocky. Rnnlm-recurrent neural network language modeling toolkit. In *Automatic Speech Recognition and Understanding Workshop*, pages 196–201. IEEE, 2011.

[15] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Annual Conference on Neural Information Processing Systems*, pages 3111–3119, 2013.

[16] A. Mnih and G. Hinton. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648. ACM, 2007.

[17] A. Mnih and G. E. Hinton. A scalable hierarchical distributed language model. In *Annual Conference on Neural Information Processing Systems*, pages 1081–1088, 2009.

[18] A. Mnih and R. Salakhutdinov. Probabilistic matrix factorization. In *Annual Conference on Neural Information Processing Systems*, pages 1257–1264, 2007.

[19] N. Natarajan, D. Shin, and I. S. Dhillon. Which app will you use next?: Collaborative filtering with interactional context. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 201–208. ACM, 2013.

[20] S. Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):57, 2012.

[21] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.

[22] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 811–820. ACM, 2010.

[23] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 635–644. ACM, 2011.

[24] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5:3, 1988.

[25] I. V. Serban, A. Sordoni, Y. Bengio, A. Courville, and J. Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 3776–3784, 2016.

[26] L. Shang, Z. Lu, and H. Li. Neural responding machine for short-text conversation. *Annual Meeting of the Association for Computational Linguistics*, pages 1577–1586, 2015.

[27] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Annual Conference on Neural Information Processing Systems*, pages 3104–3112, 2014.

[28] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164, 2015.

[29] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, and X. Cheng. Learning hierarchical representation model for next basket recommendation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 403–412. ACM, 2015.

[30] S. Wu, Q. Liu, P. Bai, L. Wang, and T. Tan. Sape: A system for situation-aware public security evaluation. In *AAAI Conference on Artificial Intelligence*, pages 4401–4402, 2016.

[31] L. Xiong, X. Chen, T.-K. Huang, J. G. Schneider, and J. G. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *Proceedings of the SIAM conference on Data Mining*, pages 211–222. SIAM, 2010.

[32] Q. Yang, J. Fan, J. Wang, and L. Zhou. Personalizing web page recommendation via collaborative filtering and topic-aware markov model. In *Proceedings of the IEEE conference on Data Mining*, pages 1145–1150. IEEE, 2010.

[33] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan. A dynamic recurrent basket recommendation model. In *Proceedings of the 39nd international ACM SIGIR conference on Research and development in information retrieval*, pages 729–732. ACM, 2016.

[34] Y. Zhang, H. Dai, C. Xu, J. Feng, T. Wang, J. Bian, B. Wang, and T.-Y. Liu. Sequential click prediction for sponsored search with recurrent neural networks. In *AAAI Conference on Artificial Intelligence*, pages 1369–1376, 2014.

[35] Z. Zhao, Z. Cheng, L. Hong, and E. H. Chi. Improving user topic interest profiles by behavior factorization. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1406–1416. ACM, 2015.

[36] Y. Zheng, X. Yi, M. Li, R. Li, Z. Shan, E. Chang, and T. Li. Forecasting fine-grained air quality based on big data. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2267–2276. ACM, 2015.