Neural Networks 159 (2023) 84–96

Contents lists available at ScienceDirect

Neural Networks

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

Factorizing time-heterogeneous Markov transition for temporal recommendation ☆

Wen Wen [a,*], Wencui Wang [a], Zhifeng Hao [a,b], Ruichu Cai [a,c]

[a] Department of Computer Science, Guangdong University of Technology, Guangzhou, China
[b] College of Science, Shantou University, Shantou, China
[c] Guangdong Provincial Key Laboratory of Public Finance and Taxation with Big Data Application, Guangzhou, China

ARTICLE INFO

Article history:
Received 23 July 2022
Received in revised form 5 November 2022
Accepted 27 November 2022
Available online 9 December 2022

Keywords:
Temporal recommendation
Tensor factorization
Time-heterogeneous
Neural network

ABSTRACT

Temporal recommendation which recommends items to users with consideration of time information has been of wide interest in recent years. But huge event space, highly sparse user activities and time-heterogeneous dependency of temporal behaviors make it really challenging to learn the temporal patterns for high-quality recommendation. In this paper, aiming to handle these challenges, especially the time-heterogeneous characteristic of user's temporal behaviors, we proposed the Neural-based Time-heterogenous Markov Transition (NeuralTMT) model. Firstly, users' temporal behaviors are mathematically simplified as the third-order Markov transition tensors. And then a linear co-factorization model which learns the time-evolving user/item factors from these tensors is proposed. Furthermore, the model is extended to the neural-based learning framework (NeuralTMT), which is more flexible and able to capture time-heterogeneous temporal patterns via nonlinear neural network mappings and attention techniques. Extensive experiments on four datasets demonstrate that NeuralTMT performs significantly better than the state-of-the-art baselines. And the proposed method is fundamentally inspired by factorization techniques, which may also provide some interesting ideas on the connection of tensor factorization and neural-based sequential recommendation methods.

1. Introduction

Mining temporal patterns from users' daily activities has been of wide interest in the field of recommendation. Most of the existing works on this topic either only concern the sequential dependency of user's temporal behaviors (Donkers, Loepp, & Ziegler, 2017; Hidasi, Karatzoglou, Baltrunas, & Tikk, 2016; Xu et al., 2019), or consider temporal information as a special type of attribute (Wu, Shi, Dong, Huang and Chawla, 2019; Yuan, Cong, & Sun, 2014; Zhang, Cao, Shi, & Niu, 2021). However, simply reducing the temporal patterns to sequential dependency tends to neglect the time signals, while considering temporal information as a kind of attribute mixes it with other attributes (such as tags, contents et al.) and does not well reflect the unique feature of temporal information. Instead of a common type of attribute, time is actually a critical variable that inherently determines the user's temporal activities and the underlying transition patterns.

A toy example is given in Fig. 1, which demonstrates the impact of time. As shown in this figure, the item set selected by a user is highly related with time, e.g. in the morning, the user often checks in locations related with body exercise and coffee shop; while in the afternoon, he frequently checks in locations related with the workplace. Besides, the transition pattern from one temporal set to another is *time-heterogenous*, that is, how the activities in a given time segment relate with the historical periods varies with time. As shown in Fig. 1, the transition pattern of a given user's activities may differ from one time segment to another. In the morning, the user tends to check in locations highly related with those chosen in the previous morning, and weakly related with items selected in the previous afternoon and evening (i.e. illustrated as the "morning pattern"); while in the afternoon, the user tends to choose items that are highly related with those in the previous evening and weakly related with items in the previous morning and afternoon (i.e. illustrated as the "afternoon pattern").

Capturing the above-mentioned patterns is beneficial to the temporal recommendation. However, it is challenging. Firstly, the event space is usually huge, which makes it almost impossible to learn the temporal transition patterns in the original event space. Besides, different distributions of items in different time segments cause troubles in learning the discrepancy

☆ This work is financially supported by Natural Science Foundation of Guangdong Province (2021A1515011965), National Natural Science Foundation of China (61876043, 61976052), National Science Fund for Excellent Young Scholars (62122022), National Key R&D Program of China (2021ZD0111501).

* Corresponding author.
E-mail address: wwen@gdut.edu.cn (W. Wen).

https://doi.org/10.1016/j.neunet.2022.11.032
0893-6080/

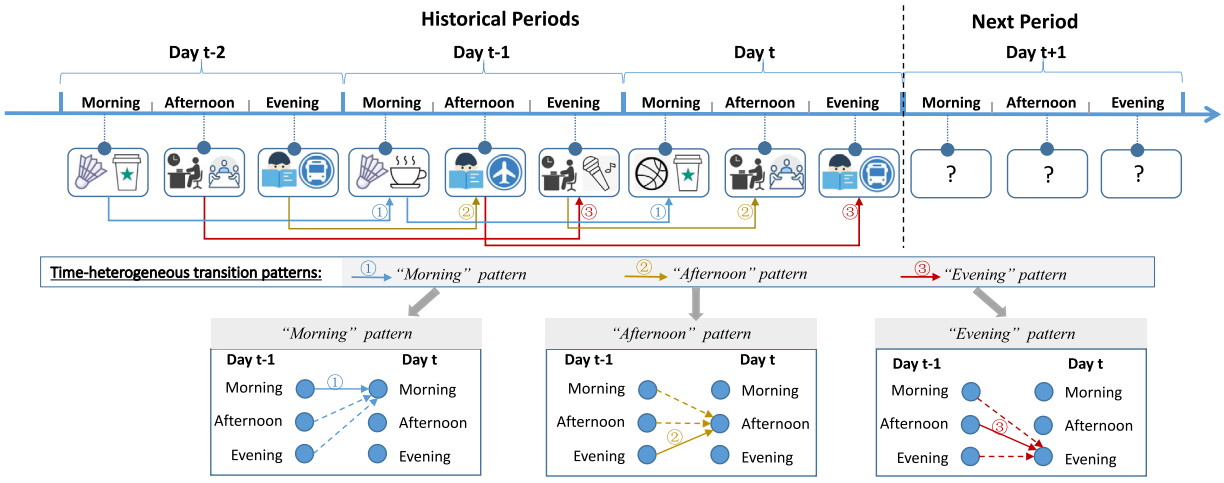

Fig. 1. An example which illustrates the time-heterogeneous transition patterns of a given user's activities in different time segments. There are three types of transition patterns in this example: "Morning", "Afternoon" and "Evening" transition patterns. The solid arrows denote strong dependencies, while the dashed arrows denote weak dependencies.

and meanwhile capturing the common patterns. Additionally, the *time-heterogeneous* and *personalized* characteristics of temporal dependency make it difficult to leverage the learning of transition patterns and collaborative filtering signals. There have been some works endeavoring to handle a part of these challenges (Donkers et al., 2017; Rendle, Freudenthaler, & Schmidt-Thieme, 2010a; Xu et al., 2019). However, existing works are largely based on the inductive bias that the dependency function of user's temporal activities does not change with time (Kang & McAuley, 2018; Rendle, Freudenthaler, & Schmidt-Thieme, 2010b; Wang et al., 2015), and the transition patterns are viewed as time-homogeneous, which is too simple to accurately describe the real temporal characteristics of users' behaviors.

In this paper, to deal with the above mentioned challenges, a tensor co-factorization method, which aims to handle the time-heterogeneous transition patterns of user's temporal activities, is proposed for the temporal recommendation. Firstly, Markov transition tensors are introduced to represent users' temporal behaviors, which consist of individualized transition information between different time segments of consecutive periods. Then to deal with the problem caused by high-dimensional event space, a tensor co-factorization model with the bayesian personalized ranking function as the objective, is designed to capture the time-heterogeneous transitions as well as the associations of temporal dependency in different time segments. Furthermore, an extension based on nonlinear neural network mappings and attention modules, is proposed to learn the temporal state dependency in a kind of more flexible and effective way.

Our contributions are as follows: (1) We focus on the time-heterogeneous transition patterns of user behaviors, which are prevalent in users' daily activities but have not been well investigated in existing works. (2) Different from existing works, we extend the assumption of Markov transition to a constraint time-heterogeneous setting, and propose a tensor factorization model to learn the time-evolving dependency function. (3) Different from the classical factorization based recommendation method, an extended neural-based framework is proposed, which is flexible to implement and provides a new perspective of the factorization-based model. (4) Extensive experiments on four real-world datasets verify our assumption and demonstrate that the proposed model outperforms the baselines.

The rest of this paper is organized as follows. Section 2 reviews the related works. And the problem is formalized in Section 3. Then the proposed model and neural-based learning framework is given in Section 4. Furthermore, disscussions on the connections of our model and existing models are presented in Section 5. Experiments and results are given in Section 6. Finally, Section 7 concludes this work.

2. Related work

In this section, we briefly review the literatures that are highly related with the topic of this paper, which includes sequential and time-sensitive models in recommendation, and factorization techniques in recommender systems,.

2.1. Sequential models in recommendation

Temporal dynamics of user behaviors have been studied from different perspectives. A popular approach is to reduce the temporal information of behaviors to behavior sequences for subsequent modeling. And the related works can be roughly divided into two streams. The first line focus on the point-wise item dependencies, in which the user–item interactions are assumed to have strict sequential order. By taking into account the sequential characteristic of historical records, neural models, such as recurrent neural network (RNN) (Donkers et al., 2017; Xu et al., 2019), convolutional networks (CNN) (Tang & Wang, 2018; Tanjim, Ayyubi, & Cottrell, 2020), graph neural networks (GNN) (Wang, Ding, Hong, Liu, & Caverlee, 2020; Wu, Tang et al., 2019), and hierarchical gating network (HGN) (Ma, Kang, & Liu, 2019), have been developed to model and capture both the long-term and short-term point-wise patterns. And most recently, attention and self-attention mechanisms have been applied to capture the sequential patterns. For example, Wang et al. (2018) proposed an Attention-based Transaction EMbedding (ATEM) model to weight each observed item in a transaction for next-item recommendation. Besides, the transformer and its extensions (Kang & McAuley, 2018; Wu, Li, Hsieh, & Sharpnack, 2020) are introduced to model the entire user sequence and adaptively consider consumed items for next item prediction. A bidirectional self-attention network inspired by BERT (Sun et al., 2019) is also introduced to model users' sequential behaviors.

The other line aims to capture the collective-item dependencies, in which the user–item interactions are assumed to have relatively flexible order with the concept of "basket" or "session".

For example, Next-basket Recommendation (NBR) (Rendle et al., 2010b; Wang et al., 2015; Yu, Liu, Wu, Wang, & Tan, 2016) and Next-Session Recommendation (NSR) (Hidasi & Karatzoglou, 2018; Hidasi et al., 2016; Xia et al., 2021) respectively take each basket or session as the basic input unit, and perform sequential recommendation based on a sequence of baskets or sessions. Some earlier works assume that items within a basket do not have strict order, and hence the core idea is to learn and capture the long-term and short-term patterns of consecutive baskets/sessions (Hidasi et al., 2016; Yu et al., 2016). More recently, the correlation between items within a basket/session has attracted increasing attention. Le, Lauw, and Fang (2019) postulate that a basket contains coalitions of related items, rather than independent items, and incorporate information on item correlations for the next-basket recommendation. Qin, Wang, and Li (2021) focus on denoising the baskets and extracting credibly relevant items to enhance recommendation performance. Some other works (Huang et al., 2021; Wu, Tang et al., 2019; Zhang et al., 2020) seek to capture the complex intra-session and inter-session item transitional patterns by utilizing GNN.

2.2. Time-sensitive models in recommendation

Although sequential models as reviewed in the above section are popular in capturing the temporal dynamics of user behaviors, they suffer a common shortcoming: time-signals are not well considered and integrated. As found in Cho, Hyun, Kang, and Yu (2021), Ye et al. (2020) and Zhao, Zhao, Yang, Lyu, and King (2016), user behaviors are time-subtle, people may frequently interact with specific products at a certain time point. And in this paper, we focus on incorporating such kind of temporal information for recommendation, which is highly related to the topic of Time-Aware Recommendation (TAR) (Campos, Díez, & Cantador, 2014; Wang et al., 2021).

Some works on TAR consider time as one kind of contextual attribute (Campos et al., 2014; Zhang et al., 2021), and hence incorporating time information to recommendation is similar to the way of integrating other contexts (i.e. user tag, item tag et al.). However, rather than a simple property, time is usually an implicit factor that affects users' preferences and transition patterns. Therefore, other techniques are employed to model the temporal transition patterns of user preference. For example, tensor factorization based ranking methodology (TA-FPMC) is introduced to capture users' time-varying behavioral trends as well as their long-term and short-term preference (Li, Jiang, Hong, & Liao, 2017). Self-exciting point processes and low-rank models are connected to learn the recurrent temporal patterns in a large collection of user–item consumption pairs (Du, Wang, He, Sun, & Song, 2015). Attention techniques are implemented in TimelyRec (Cho et al., 2021) to jointly learn the periodical feature and time-evolving influence of recent events. And various graph-based methods are designed to unify sequential patterns and dynamic collaborative signals for temporal recommendation (Fan et al., 2021; Zhang, Wu, Yu, Liu, & Wang, 2022)

Besides, in recent years, "temporal sets", which refers to the time-stamped sequential sets, has arouse increasing attention. For example, Yu et al. (2016) proposed a dynamic recurrent model to learn the temporal dependence of the sequential sets; Sun et al. proposed a Dual Sequential Network for Temporal Sets Prediction (DSNTSP) (Sun et al., 2020), which learns both item-level and set-level dependencies via Transformer framework.

2.3. Factorization techniques in recommendation

Matrix Factorization (MF) based models in recommender system are firstly proposed by Koren, Bell, and Volinsky (2009), which capture the collaborative filtering signals by factorizing the user–item interaction matrix into low-dimensional representations of users and items. And MF based models have inspired a large number of extensions. On the one hand, to deal with the sparsity of user–item interactions in real scenarios, some works enrich the latent factors by adding auxiliary user–item connections from other perspectives. For example, CoFactor (Liang, Altosaar, Charlin, & Blei, 2016) jointly decomposes the user–item interaction and item–item co-occurrence matrix to factors with shared latent space, which is able to capture higher-order item–item connections. On the other hand, to capture the temporal dynamics of user preference, timeSVD++ (Koren, 2009), dynamic bayesian probabilistic matrix factorization (Chatzis, 2014) learn the temporal latent factors by incorporating the time-evolving changes. Besides, another important way of modeling time-evolving dynamics relies on factorizing the sequential transition matrix. For example, Factorizing Personalized Markov Chains (FPMC) (Rendle et al., 2010a) models the behavior sequences as one-order markov chains, and then factorizes the transition matrix to capture the temporal patterns.

And in recent years, with the development of deep learning, some efforts have also been devoted to extending the factorization techniques to the neural network based framework. Wang et al. proposed a Hierarchical Representation Model (HRM) for next-basket recommendation (Wang et al., 2015), which has been demonstrated to be a neural-based extension of FPMC. He et al. proposed a generalized collaborative filtering framework, named neural-network based collaborative filtering (NCF), and pointed out that matrix factorization based collaborative filtering (Koren et al., 2009) is a special case of NCF. Besides, Kim, Park, Oh, Lee, and Yu (2016) proposed a method (ConvMF) integrating convolutional neural network (CNN) into probabilistic matrix factorization (PMF), and Hong et al. (Xue, Dai, Zhang, Huang, & Chen, 2017) propose a deep matrix factorization model (DMF) with a neural network that maps the users and items into a common low-dimensional space with non-linear projections.

3. Problem formulation

Given the user's timeline which can be equally divided into regular intervals, we suppose each interval relates with a time "segment" (e.g. an hour) and M time segments form a "period" (e.g. 24 h form a day). And in the following, we use the subscript $\langle t, m\rangle$ to denote the mth time segment in the tth period. $\mathcal{S}_{t,m} \subset I$ is the set of items chosen by the user in the mth time segment of tth period. As illustrated in Fig. 2, the user's activities in the tth period are represented by the temporal set $\mathbf{S}_t = \left\{\mathcal{S}_{t,m}\right\}_{m=1}^{M}$ and his historical activities before the tth period are denoted by $\mathbb{H}_t = \{\mathbf{S}_{t-1}, \mathbf{S}_{t-2}, \ldots, \mathbf{S}_{t-W}\}$ (W is the contextual window size). Let $\mathbb{D} = \{\langle \mathbb{H}_t, \mathbf{S}_t\rangle | t = 1, 2, \ldots, T\}$ denote the training dataset. For better presentation, we formalize the problems and terminologies as follows.

Definition 1 (*Period-based Temporal Recommendation*)**.** Given the training dataset consists of different users and their historical activities $\mathbb{D} = \left\{\langle \mathbb{H}_t^u, \mathbf{S}_t^u\rangle | t = 1, 2, \ldots, T; u \in U\right\}$, the task of Period-based Temporal Recommendation (PTR) is when given $\mathbb{H}_{t^*}^u$, $t^* > T$ output $\mathbf{S}_{t^*+1}^u$, i.e. items that u will be interested in each time segment of $(t^* + 1)$th period.

Table 1
List of notations.

Notation	Description
U, I	User and item set
$\mathbf{S}_t$	The collection of temporal item sets in the tth period
$\mathcal{S}_{t,m}$	The item set in the mth time segment of the tth period
M	Number of time segments in one period
$T^{n,m}$	The third-order tensor including users' transition information from the nth time segment of $(t-1)$th period to the mth time segment of tth period for different users
$\alpha_{n,m}$	Attention weights of the nth time segment item set of previous period for the mth time segment item set of current period
$\tilde{V}_U^m$	Feature embedding matrix of users at mth time segment
$\tilde{V}_I$	Item feature embedding matrix
V^n	Transition embedding matrix of items at the nth time segment of previous period
$\tilde{V}^m$	Transition embedding matrix of items at the mth time segment of current period
d	Dimension of the embeddings
$\Phi_I^m(\cdot)$	Mapping function for transition embeddings of items at the mth time segment of previous period
$\tilde{\Phi}_I^m(\cdot)$	Mapping function for transition embeddings of items at the mth time segment of current period
$\tilde{\Phi}_U^m(\cdot)$	Mapping function for feature embeddings of users at the mth time segment of current period
$\tilde{\Phi}_I(\cdot)$	Mapping function for feature embeddings of items

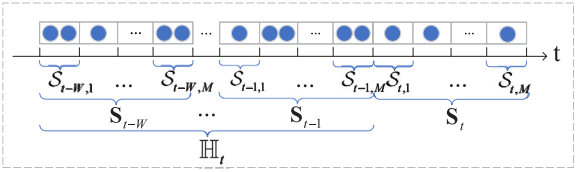

Fig. 2. Illustration of the timeline and the notations $\mathcal{S}_t, \mathbf{S}_t$ and $\mathbb{H}_t$.

In Definition 1, we simply add the superscript u to denote a specific user, i.e. $\mathbf{S}_t^u$ represent user u's activities in the tth period, and the other symbols is the same as mentioned above. Specifically, when the number of time segments in a period is set to be one (i.e. $M = 1$) and a period is taken as a basket, Period-based Temporal Recommendation (PTR) can be reduced to next-basket recommendation.

Definition 2 (*Period-based Transition Function*)**.** Noting that there are totally M temporal subsets in $\mathbf{S}_t$, the period-based transition function can be decomposed to M components, which can be written as follows.

$$\mathbf{f} = \langle f^1, f^2, \ldots, f^M \rangle$$

in which,

$$f^m(\langle \mathbb{H}_t, \mathbf{S}_t \rangle) = P(S_{t,m}|\mathbb{H}_t), m = 1, 2, \ldots, M$$

Fig. 3 gives the illustration of the period-based transition function. As mentioned in Sections 1 and 2, existing works mainly concern the case where sequential dependency of user behaviors is time-*homogeneous* (the contrary of time-heterogeneous), i.e. the transition function $\mathbf{f}$ as well as its components, does not change with time, which tends to be inconsistent with the real scenario. In this paper, we aim to extend this simple setting to the constraint time-heterogeneous case, which is featured by the following defined period-based time-heterogeneous Markov chain.

Definition 3 (*Period-based Time-heterogeneous Transition*)**.** Given the dataset $\mathbb{D} = \{\langle \mathbb{H}_t, \mathbf{S}_t \rangle | t = 1, 2, \ldots, T\}$, the Time-heterogeneous Transition refers to the case that the component of transition function varies with the index of time segment. That is, as for $\mathbf{f} = \langle f^1, f^2, \ldots, f^M \rangle$, $f^m(\langle \mathbb{H}_t, \mathbf{S}_t \rangle) = P(S_{t,m}|\mathbb{H}_t)$ has the property: when $m \neq n$, $f^m \neq f^n$.

The period-based time-heterogeneous transition as Definition 3 is the abstract description of nonstationary case illustrated in Fig. 1. We aims to handle such kind of time-heterogeneous transition for the period-based temporal recommendation, which is defined as Definition 1. Besides, we follow the rules of notations in mathematics through this paper: upper case bold letters denote matrices and lower case bold letters denote column vectors; without any specification, non-bold letters represent scalars. And some of the major symbols are listed in Table 1.

4. Methodology

To address the problems mentioned above, we firstly introduce transition tensor to capture the time-sensitive transition patterns of different users. And then we propose a generalized tensor factorization model to capture the complex temporal dependencies, which relies on weightedly co-decompose multiple tensors with shared factors. Finally, a multi-task Bayesian optimization approach is applied to learn the factorization parameters, and all operations are implemented by an equivalent neural network framework. For simplicity, the following details are described only for a single user, and these formulas can generalize to all users.

4.1. Transition tensor

Given $\mathbb{H}_t = \{\mathbf{S}_{t-1}, \mathbf{S}_{t-2}, \ldots, \mathbf{S}_{t-W}\}$, the combinations of variables in the transition function $f^m(\langle \mathbb{H}_t, \mathbf{S}_t \rangle) = P(S_{t,m}|\mathbb{H}_t)$ increases rapidly with the contextual window size W. To make it computable, we assume that activity in current period only depends on the previous period, that is, $P(S_{t,m}|\mathbb{H}_t) = P(S_{t,m}|\mathbf{S}_{t-1})$. Hence, for a given user, we construct the following transition tensor, which is a block tensor.

$$\mathbb{T} = (T^{n,m}), 1 \leq n, m \leq M; \tag{1}$$

where $T^{n,m}$ is the subtensor and $T^{n,m} \in \mathbb{R}^{|U|*|I|*|I|}$ represents the item–item transition probability from the nth time segment of $(t-1)$th period to the mth time segment of tth period for different users. That is, the element in $T^{n,m}$ is defined as Eq. (2).

$$T^{n,m}(u, i, j) = P\left(i \in \mathcal{S}_{t,m}^u \mid j \in \mathcal{S}_{t-1,n}^u\right) \tag{2}$$

And Fig. 3 illustrates the transition tensor related with the toy example in Fig. 1.

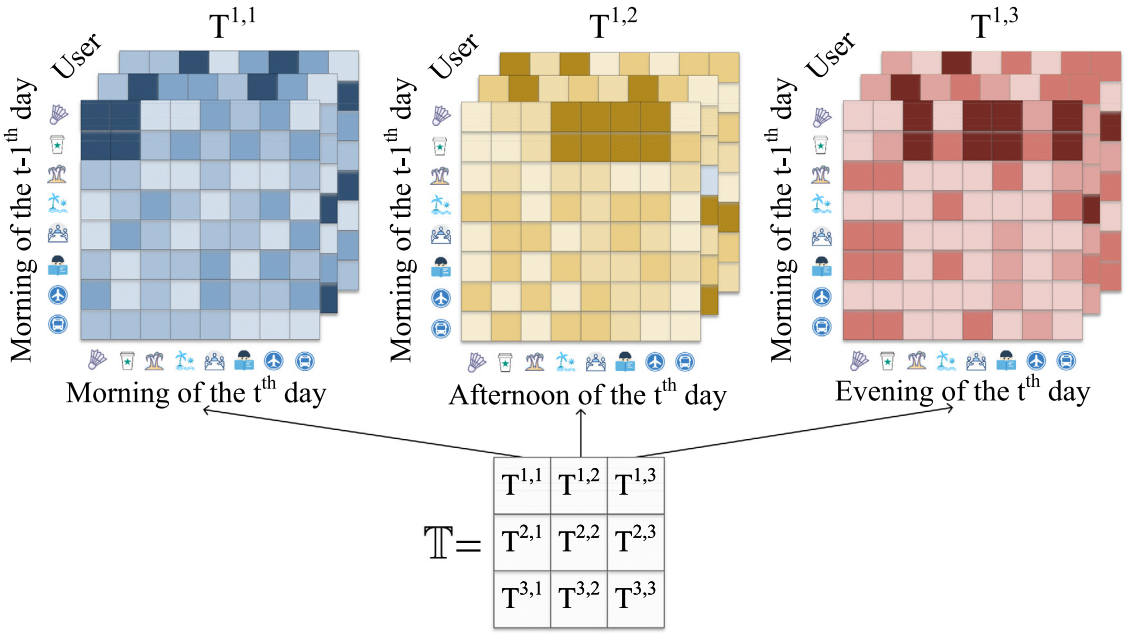

Fig. 3. Third-Order Personalized Tensors modeling the transition from the morning of $(t-1)$th period to the morning, afternoon and evening of tth period respectively.

4.2. Factorizing the time-heterogeneous transition tensor

As mentioned in Section 1, user's temporal behaviors in current period have different correlations with the time segments in the previous period. Hence, we assume that the time-heterogeneous transition function is equal to the weighted sum of the transition probabilities from one segment in the previous period to the segment in current period, which is formalized as Eq. (3). Given $\mathbf{S}_{t-1} = \{\mathcal{S}_{t-1,n}\}_{n=1}^{M}$,

$$P\left(i \in \mathcal{S}_{t,m} \mid \mathbf{S}_{t-1}\right) = \sum_{n=1}^{M} \alpha_{n,m} * P\left(i \in \mathcal{S}_{t,m} \mid \mathcal{S}_{t-1,n}\right) \tag{3}$$

where $\alpha_{n,m}$ is a parameter denoting the correlation strength between the activities in the nth time segment of the previous period and the mth time segment of current period. And $\sum_{n=1}^{M} \alpha_{n,m} = 1$. $P\left(i \in \mathcal{S}_{t,m} \mid \mathcal{S}_{t-1,n}\right)$ represents the probability of user u transferring from the item set at the nth time segment of previous period to item i at mth time segment of current period.

Similar to the literature (Rendle et al., 2010a), we assume that all items at a certain time segment of previous period have the same influence on items in current period. Then, the transition probability between item set at the nth time segment of $(t-1)$th period and item i at the mth time segment of tth period can be formulated as Eq. (4).

$$P\left(i \in \mathcal{S}_{t,m} \mid \mathcal{S}_{t-1,n}\right) = \frac{1}{\left|\mathcal{S}_{t-1,n}\right|} \sum_{j \in \mathcal{S}_{t-1,n}} T^{n,m}(u,i,j) \tag{4}$$

where $T^{n,m}(u,i,j)$ is defined as Eq. (2), which denotes the probability that user u transfers from item j at the nth time segment of $(t-1)$th period to item i at mth time segment of tth period.

However, in real scenario, huge item set, as well as high sparsity of user activities, makes it almost impossible to estimate the exact value of each element in the tensor block $T^{n,m}$. Therefore, the CANDECOMP-PARAFAC (CP) decomposition (Carroll & Chang, 1970; Harshman, 1970), is used to approximate the transition probability, which allows propagating information among similar users, items and transition pairs to alleviate the problem caused by huge item set and high data sparsity. That is,

$$T^{n,m}(u,i,j) \approx \left\langle \tilde{\boldsymbol{v}}_u^m, \tilde{\boldsymbol{v}}_i \right\rangle + \left\langle \tilde{\boldsymbol{v}}_i^m, \boldsymbol{v}_j^n \right\rangle + \left\langle \boldsymbol{v}_u^n, \boldsymbol{v}_j \right\rangle \tag{5}$$

Here, $\langle \cdot, \cdot \rangle$ denotes the inner product of two vectors; $\tilde{\boldsymbol{v}}_u^m$, $\tilde{\boldsymbol{v}}_i^m$ and $\tilde{\boldsymbol{v}}_i$ are user/item embeddings in the *current* period. The former two vectors $\tilde{\boldsymbol{v}}_u^m$, $\tilde{\boldsymbol{v}}_i^m$ are time-evolving, which respectively denote user u's and item i's embedding at the mth time segment, while the third vector $\tilde{\boldsymbol{v}}_i$ does not change with time, which can be viewed as the long-term representation of item i. Similarly, $\boldsymbol{v}_u^n$ and $\boldsymbol{v}_j^n$ are respectively user u's and item j's embedding at the nth time segment of *previous* period, and $\boldsymbol{v}_j$ is the long-term representation of item j at the *previous* period. Besides, all the factors in Eq. (5) has the dimension d.

Substituting Eqs. (5), (4) to Eq. (3), we have the following co-factorization model:

$$\begin{aligned}
&P\left(i \in \mathcal{S}_{t,m} \mid \mathcal{S}_{t-1,n}, n = 1, \ldots, M\right) \\
&= \sum_{n=1}^{M} \alpha_{n,m} * P\left(i \in \mathcal{S}_{t,m} \mid \mathcal{S}_{t-1,n}\right) \\
&= \sum_{n=1}^{M} \Bigg(\alpha_{n,m} * \Bigg(\left\langle \tilde{\boldsymbol{v}}_i, \tilde{\boldsymbol{v}}_u^m, \right\rangle \\
&\quad + \frac{1}{\left|\mathcal{S}_{t-1,n}\right|} \sum_{j \in \mathcal{S}_{t-1,n}} \left(\left\langle \tilde{\boldsymbol{v}}_i^m, \boldsymbol{v}_j^n \right\rangle + \left\langle \boldsymbol{v}_u^n, \boldsymbol{v}_j \right\rangle \right) \Bigg) \Bigg) \\
&= \sum_{n=1}^{M} \left(\alpha_{n,m} * \frac{1}{\left|\mathcal{S}_{t-1,n}\right|} \sum_{j \in \mathcal{S}_{t-1,n}} \left\langle \tilde{\boldsymbol{v}}_i^m, \boldsymbol{v}_j^n \right\rangle \right) \\
&\quad + \left\langle \tilde{\boldsymbol{v}}_u^m, \tilde{\boldsymbol{v}}_i \right\rangle + \sum_{n=1}^{M} \left(\alpha_{n,m} * \frac{1}{\left|\mathcal{S}_{t-1,n}\right|} \sum_{j \in \mathcal{S}_{t-1,n}} \left\langle \boldsymbol{v}_u^n, \boldsymbol{v}_j \right\rangle \right)
\end{aligned} \tag{6}$$

The last term in Eq. (6) does not influence the ranking of a given item i, because it only depends on the historical records

(i.e. $\mathcal{S}_{t-1,n}, n = 1, \ldots, M$), which is constant for all candidate items. Hence we have the following approximation of the probability transition.

$$P\left(i \in \mathcal{S}_{t,m} \mid \mathcal{S}_{t-1,n}, n = 1, \ldots, M\right) \propto \underbrace{\left(\sum_{n=1}^{M}\left(\alpha_{n,m} * \frac{1}{\left|\mathcal{S}_{t-1,n}\right|} \sum_{j \in \mathcal{S}_{t-1,n}} \left\langle \tilde{\boldsymbol{v}}_i^m, \boldsymbol{v}_j^n \right\rangle\right)\right)}_{\text{Short-term}} + \underbrace{\left\langle \tilde{\boldsymbol{v}}_i, \tilde{\boldsymbol{v}}_u^m \right\rangle}_{\text{Long-term}} \tag{7}$$

Some interesting facts are revealed by Eq. (7), given user behaviors in the previous period, the probability of user–item interactions in current period can be approximated by a combination of two components: one is the "short-term" component, i.e. how the user's time-evolving patterns of the previous period encoded in v_j^n influences his preference in current period; and the other is the "long-term" component, i.e. how the long-term pattern encoded in the user's personalized embedding $\tilde{v}_u^m$ influences his preference in current period. Furthermore, taking into account of the different impact of short-term and long-term components, we generalize Eqs. (7) to (8) via adding a learnable fusion weight z_m, which is formulated as follows.

$$\begin{aligned} &P\left(i \in \mathcal{S}_{t,m} \mid \mathcal{S}_{t-1,n}, n = 1, \ldots, M\right) \\ &= z_m * \left(\sum_{n=1}^{M}\left(\alpha_{n,m} * \frac{1}{\left|\mathcal{S}_{t-1,n}\right|} \sum_{j \in \mathcal{S}_{t-1,n}} \left\langle \tilde{\boldsymbol{v}}_i^m, \boldsymbol{v}_j^n \right\rangle\right)\right) \\ &\quad + (1 - z_m) * \left\langle \tilde{\boldsymbol{v}}_i, \tilde{\boldsymbol{v}}_u^m \right\rangle \end{aligned} \tag{8}$$

where $z_m \in [0, 1]$ leverages a user's short-term and long-term preference at the mth time segment for predicting his next-period activity.

4.3. A generalized neural-based factorization framework

Although the factors in Eq. (8) might be obtained by matrix optimization method, such as the alternating direction method of multipliers (ADMM) (Boyd, Parikh, Chu, Peleato, Eckstein, et al., 2011), it is inflexible for capturing more complicated and nonlinear behavioral dependency between consecutive periods. Therefore, based on the factorization model in the above subsection, a generalized neural-based factorization framework, which is named by the Neural-based Time-heterogeneous Markov Transition (NeuralTMT) model, is proposed to learn the user/ item's representation for period-based temporal recommendation. The whole architecture is illustrated as Fig. 4. Firstly, an Embedding Layer is designed to obtain the low-rank user/item representations relating with the tensor factorization Eq. (5), where items and users (one-hot vectors) are projected to different feature spaces via the mapping functions. Then based on Eq. (4), an aggregation layer is devised to generate the latent factors of different time segments, which can be used to obtain the collective effect of all items at the previous time segment on the current time segment. After that, according to Eq. (3), an attention layer is designed to learn the time-evolving state transition between the consecutive periods. Finally, on the basis of Eq. (8), a fusion layer is utilized to fuse the long-term and short-term components.

4.3.1. Embedding layer

As given in Eq. (8), for each item, there are three types of factors: $\tilde{\boldsymbol{v}}_i^m$ is the factor related to the mth time segment in *current* period; $\boldsymbol{v}_i^m$ relates to the mth time segment in *previous* period; and $\boldsymbol{v}_i$ is the factor that does not change with time. Besides, for a given user, we only consider one type of factor, which varies with the index of time segment: $\tilde{\boldsymbol{v}}_u^m$. These factors can be respectively obtained by introducing the following embedding functions:

$$\begin{aligned} &\tilde{\Phi}_I^m : I \to \mathbb{R}^d; \quad m = 1, \ldots, M \\ &\Phi_I^m : I \to \mathbb{R}^d; \quad m = 1, \ldots, M \\ &\Phi_I : I \to \mathbb{R}^d; \\ &\tilde{\Phi}_U^m : U \to \mathbb{R}^d; \quad m = 1, \ldots, M \end{aligned} \tag{9}$$

Hence, Eq. (8) can be generalized to the following form in the neural-based learning framework.

$$\begin{aligned} &P\left(i \in \mathcal{S}_{t,m} \mid \mathcal{S}_{t-1,n}, n = 1, \ldots, M\right) \\ &= z_m * \left(\sum_{n=1}^{M}\left(\alpha_{n,m} * \frac{1}{\left|\mathcal{S}_{t-1,n}\right|} \sum_{j \in \mathcal{S}_{t-1,n}} \left\langle \tilde{\Phi}_I^m(i), \Phi_I^n(j) \right\rangle\right)\right) + (1 - z_m) * \left\langle \tilde{\Phi}_I(i), \tilde{\Phi}_U^m(u) \right\rangle \end{aligned} \tag{10}$$

4.3.2. Aggregation layer

The embedding of each item in the input subsequence $\mathcal{S}_{t-1,n}$ are then aggregated to obtain the latent representation of $\mathcal{S}_{t-1,n}$, which is denoted by $\boldsymbol{x}_n \in \mathbb{R}^d$ in Fig. 4. And meanwhile, we use the function $\boldsymbol{f}_{agg}(\cdot)$ to denote this operation. That is,

$$\boldsymbol{x}_n = \boldsymbol{f}_{agg}\left(\Phi_I^n(j); j \in \mathcal{S}_{t-1,n}\right) \tag{11}$$

Therefore Eq. (10) can be furthermore generalized to Eq. (12).

$$\begin{aligned} &P\left(i \in \mathcal{S}_{t,m} \mid \mathcal{S}_{t-1,n}, n = 1, \ldots, M\right) \\ &= z_m * \left(\sum_{n=1}^{M} \alpha_{n,m} * \left\langle \tilde{\Phi}_I^m(i), \boldsymbol{x}_n \right\rangle\right) \\ &\quad + (1 - z_m) * \left\langle \tilde{\Phi}_I(i), \tilde{\Phi}_U^m(u) \right\rangle \end{aligned} \tag{12}$$

In this paper, mean pooling is used as the aggregation function,i.e.

$$\boldsymbol{f}_{agg}\left(\Phi_I^n(j); j \in \mathcal{S}_{t-1,n}\right) = \frac{1}{\left|\mathcal{S}_{t-1,n}\right|} \sum_{i \in \mathcal{S}_{t-1,n}} \Phi_I^n(j) \tag{13}$$

where all the items contribute equally to the next-item transition.

4.3.3. Attention layer

An attention layer is designed to learn the time-heterogenous transition weight $\alpha_{n,m}$. Let $\boldsymbol{X} = [\boldsymbol{x}_1; \boldsymbol{x}_2; \cdots; \boldsymbol{x}_M] \in \mathbb{R}^{d \times M}$ denote the output of the aggregation layer. The attention layer is designed as follows.

$$\boldsymbol{h}_m^i = \sum_{n=1}^{M} \alpha_{n,m}^i \boldsymbol{x}_i \quad \sum_{n=1}^{M} \alpha_{n,m}^i = 1 \tag{14}$$

$$\alpha_{n,m}^i = \frac{\exp\left(e\left(\boldsymbol{x}_n, \tilde{\Phi}_I^m(i)\right)\right)}{\sum_{k=1}^{M} \exp\left(e\left(\boldsymbol{x}_k, \tilde{\Phi}_I^m(i)\right)\right)} \tag{15}$$

$$e\left(\boldsymbol{x}_n, \tilde{\Phi}_I^m(i)\right) = \frac{\boldsymbol{x}_n \boldsymbol{W}^m \tilde{\Phi}_I^m(i)}{\sqrt{d}} \tag{16}$$

where $\boldsymbol{W}^m \in \mathbb{R}^{d \times d}$ is a learnable weight matrix, $\tilde{\Phi}_I^m(i) \in \mathbb{R}^d$ is the embedding of candidate item i at mth time segment of current period. Noting that $\alpha_{n,m}^i$ is an extension of $\alpha_{n,m}$ in Eq. (8), the $\alpha_{n,m}^i$ does not only relate with the index of time segment n, m, but also the item i. That is, $\alpha_{n,m}^i$ can be viewed as an normalized weight based on the function $e(\cdot) : \mathbb{R}^d \to \mathbb{R}$ and the softmax function, which is not only dependent on the latent state representation embeddings of nth time segment of $(t-1)$th period, but also the embedding of the candidate item i. The scale factor $\sqrt{d}$ in Eq. (16)

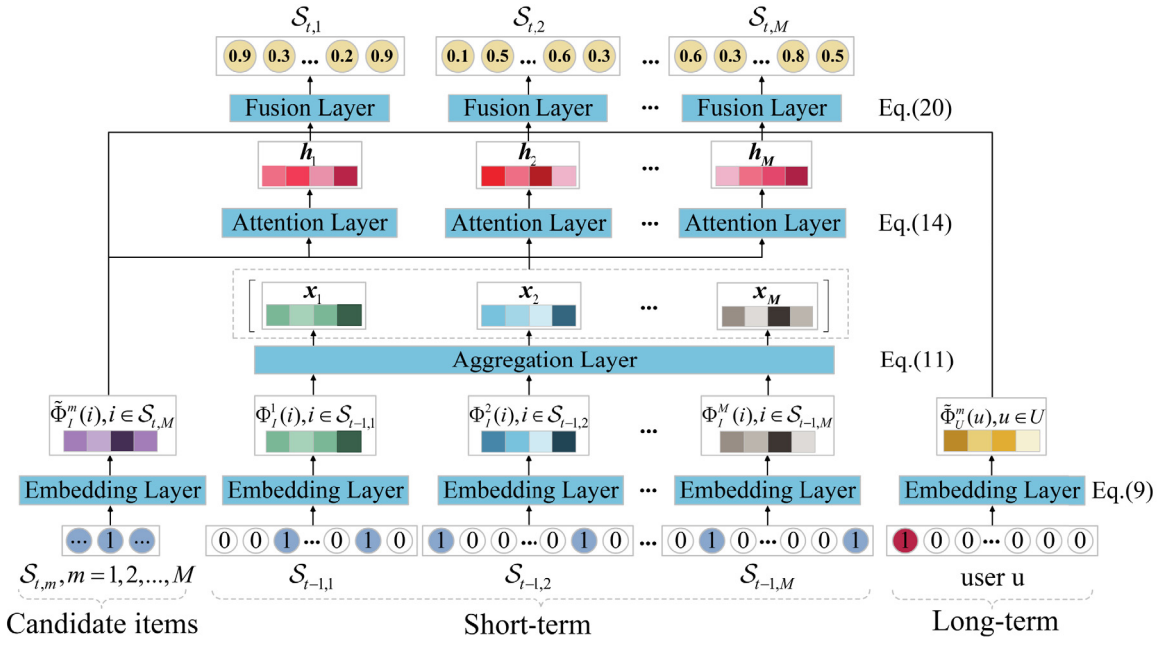

Fig. 4. The architecture of the generalized neural-based factorization framework, where one color indicates unique vector space. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

is to avoid overly large values of the inner product, especially when the dimensionality is high. Finally, the item-level hidden vector (current time state latent factor) $\boldsymbol{h}_m^i$ for the candidate item $i \in S_{t,m}$ is built as a weighted sum of $\boldsymbol{x}_n$. We use the function $\boldsymbol{f}_{att}(\cdot)$ to denote the attention operation. Then we have the following equation.

$$P\left(i \in \mathcal{S}_{t,m} \mid \mathcal{S}_{t-1,n}, n = 1, \ldots, M\right) = z_m * \boldsymbol{f}_{att}(\tilde{\Phi}_I^m(i), \boldsymbol{X}) + (1 - z_m) * \left\langle \tilde{\Phi}_I(i), \tilde{\Phi}_U^m(u) \right\rangle \tag{17}$$

4.3.4. Fusion layer

Finally, a fusion layer is implemented to incorporate the short-term and long-term components of user behaviors.

On the one hand, with the item-level hidden state $\boldsymbol{h}_m^i$ (the output of the attention layer) and the embedding $\tilde{\Phi}_I^m(i)$, we can calculate the probability that user u will interact with candidate item i at the mth time segment of next period, which is determined by the user's short-term preference.

$$o_{short}^{m,i} = \boldsymbol{h}_m^i \tilde{\Phi}_I^m(i) \tag{18}$$

On the other hand, with the time-evolving feature embedding of user $\tilde{\Phi}_U^m(u)$ and the embedding of candidate item $\tilde{\Phi}_I(i)$, the long-term component is as follows.

$$o_{long}^{m,i} = \tilde{\Phi}_U^m(u) \tilde{\Phi}_I(i) \tag{19}$$

Then the output layer of the network is finally calculated as follows, which is the estimation of transition probability:

$$P\left(i \in \mathcal{S}_{t,m} \mid \mathcal{S}_{t-1,n}, n = 1, \ldots, M\right) = z_m * o_{short}^{m,i} + (1 - z_m) * o_{long}^{m,i} \tag{20}$$

Here, we use the function $\hat{f}^m(\cdot)$ to denote the estimation of the transition probability. That is

$$\hat{f}^m(\mathbf{S}_{t-1}, i) = P\left(i \in \mathcal{S}_{t,m} \mid \mathcal{S}_{t-1,n}, n = 1, \ldots, M\right) \tag{21}$$

4.4. Loss function

Bayesian Personalized Ranking (BPR) loss with consideration of the index of time segment is used as the objective function, whose basic assumption is that rather than items unselected in the past, users prefer items that have been selected. And recommendation in different time segments can be viewed as multiple recommendation tasks with underlying association. That is, the following loss function Eq. (22) is used to train the network.

$$\ell = \sum_{u \in U} \sum_{t=1}^{T} \sum_{m=1}^{M} \sum_{i \in \mathcal{S}_{t,m}^u} \sum_{s \notin \mathcal{S}_{t,m}^u} -\ln \sigma(\hat{f}^m(\mathbf{S}_{t-1}, i) - \hat{f}^m(\mathbf{S}_{t-1}, s)) + \frac{\lambda}{2} \|\Theta\|^2 \tag{22}$$

where item s is negative samples for candidate item i in $\mathcal{S}_{t,m}^u$, λ is a constant to control the power of regularization. σ is a non-linear function denoting logistic function $\sigma(x) = \frac{1}{1+e^{-x}}$. Symbol $\Theta = \{\Theta_{\tilde{\Phi}_U^m}, \Theta_{\tilde{\Phi}_I^m}, \Theta_{\Phi_I^m}, \Theta_{\tilde{\Phi}_I} \mid n, m = 1, \ldots, M\}$ denotes all the parameters in the Embedding Layer that need to learn, in which $\Theta_{\tilde{\Phi}_U^m}, \Theta_{\tilde{\Phi}_I^m}, \Theta_{\Phi_I^m}, \Theta_{\tilde{\Phi}_I}$ denote the parameters for the embedding functions. Besides, the parameters in Attention Layer and Fusion Layer $W^m, z_m, m = 1, \ldots, M$ are also required for optimization. We assume that the model parameters are drawn from a Gaussian distribution $\Theta \sim N(0, \sigma_\Theta I)$ and Adam optimizer is used to optimize Eq. (22). In the prediction stage, the probability of all items in each time segment of next period can be calculated via the network, and then items with Top-N probability will be added to the recommendation list.

4.5. Complexity analysis

In the training stage, the computational time is mainly cost in sampling and calculating the transition probability. (1) **Sampling**: Given that there are averagely l_1 positive samples in one time segment, and l_2 negative samples are produced for each positive sample, totally $l_1 l_2$ samples are generated for the subsequence in one time segment. Therefore, it takes $O(Ml_1l_2)$ for sampling

procedure. (2) **Calculating the probability of one item**: the computational complexity of this step consists of two parts:attention layer with $O(Md)$, and fusion layer with $O(2d)$. Therefore the overall time cost for one user in a sequential time segment approximates to $O\left(M^2 l_1 l_2 d + 2Ml_1 l_2 d\right)$, which mainly depends on the number of negative samples l_2 for one positive sample and the dimension of embedding d.

In the prediction stage, the computational complexity of temporal recommendation is composed of two parts: the cost of calculating the probabilities of items in M time segments, which is $O(M|I|d + 2|I|d)$; and the cost of finding items with Top-N probability ($N \ll |I|$), which is approximately $O(|I|)$. Since $|I| \gg M$ and $|I| \gg d$, the prediction complexity is approximately $O(|I|)$.

5. Model discussions

5.1. Connection to time-aware models

As one of the time-aware recommendation models, TimelyRec (Cho et al., 2021) is highly-related to our model (NeuralTMT), because both of them focus on the heterogeneous and hierarchically temporal patterns of user behaviors. However, TimelyRec is technically different from NeuralTMT. That is, TimelyRec relies on a positional encoder for integrating the hierarchically temporal information to the attention module and is more suitable to capture the periodically repeat pattern; while NeuralTMT is generalized from tensor cofactorization, which is better for learning correlational (but not necessarily repeat) patterns in the latent factor space.

Besides, there are also hierarchically temporal models in the field of real-valued time series (Athanasopoulos, Hyndman, Kourentzes, & Petropoulos, 2017), where the temporal hierarchy is constructed by means of non-overlapping temporal aggregation. And predictions constructed at all aggregation levels are combined to result in temporally reconciled and robust forecasts. But since users' behavioral sequences consist of high-dimensional events (i.e. user–item interactions), which are essentially discrete values, the generalized-least-square regression framework in Athanasopoulos et al. (2017) is not directly applicable for the temporal recommendation.

5.2. Connection to sequential models

NeuralTMT can be viewed as a special case of sequential models. However, different from the classical item-based and basket (or session)-based sequential models, the proposed NeuralTMT seeks to handle the period-based behavior sequences so as to answer the question "what item will the user prefer during different time segments (morning/afternoon/evening) of next period (day)". Therefore, as illustrated by Fig. 1, the input data in NeuralTMT are period-based sequences, which involves temporal sets of items. And the time-heterogeneous transitional patterns are captured by Neural-TMT.

As it should be, NeuralTMT can be adapted to learn patterns from item-based and basket-based sequences. More specifically, by ignoring the temporal hierarchy (i.e. "period" $\rightarrow$ "time segment") and confining the input unit to consist of only one item in each time segment $\mathcal{S}_{t,m}$, NeuralTMT is equivalent to capture the point-wise item dependency in behavior sequences, so it can be used for item-based sequential recommendation. Besides, by ignoring the concept of time segment, and taking each period as a basket (i.e. setting $M = 1$), the proposed NeuralTMT can also be reduced to capture the basket-based sequential dependency of user behaviors, so it is able to make basket-based sequential recommendation.

5.3. Connection to Markov-based models

Existing Markov-based models, including FPMC (Rendle et al., 2010b) and HRM (Wang et al., 2015), mainly concern the transition patterns of consecutive baskets. And the underlying assumption of these models is the *homogeneousity* of basket-based transition. That is, the transition function does not vary with time, which is equivalent to restricting "$f^m = f^n, (m \neq n)$" in Definition 3. Therefore, in FPMC and HRM, basket-based recommendations are transformed to learn low-dimensional factors from the transition matrix of consecutive baskets.

On the contrary, NeuralTMT focuses on the *heterogeneousity* of transition function (as Definition 3). Temporal hierarchy is involved and the transition function is assumed to be varied with the index of time segments, which make NeuralTMT suited for more complicated cases and no longer strictly require the one-order Markov assumption on the data. In essence, via a neural-based extension of the transition-tensor factorization technique, NeuralTMT does not only capture the Markov dependency, but also a kind of skip-chain dependency of behaviors in consecutive periods.

6. Experiments

In this section, we will answer the following questions through conducting extensive experiments on four real-world datasets.

RQ1: Does the proposed method outperform the state-of-the-art methods?

RQ2: How about the performance of different variants of our method ?

RQ3: Is the proposed method able to learn complex temporal dependencies and meaningful patterns related with time?

RQ4: How about the computational efficiency of NeuralTMT in practical implementation?

6.1. Experiment setup

6.1.1. Datasets

We evaluate our method on four public datasets: NYC, TKY, Gowalla and Weeplace.

- **NYC:** A widely-used dataset collected from Foursquare, which contains check-ins of 1083 users in New York City, ranging from Apr. 3, 2012 to Feb. 15, 2013.
- **TKY:** This dataset is also collected from Foursquare, which consists of check-in records of 2293 users in TKY, ranging from Apr. 4, 2012 to Feb. 16, 2013.
- **Gowalla:** The check-in data collected from the social network Gowalla, which contains behavioral sequences of 1186 users, ranging from Feb. 2009 to Oct. 2010.
- **Weeplace:** This dataset is collected from Weeplace, a website that aims to visualize users' check-in activities in LSBN, which contains behavior records of 13,819 users for the whole year of 2010.

The statistics of each dataset are summarized in Table 2. For all of these datasets, a *period* is a day, which contains four time *segments*: dawn (00:00–06:59), morning (07:00–11:59), afternoon (12:00–18:59) and evening (19:00–23:59). For each dataset, user activities are sequentially divided into two subsequences which respectively contain 80% and 20% samples. And the former (80% samples) are used as the training set and the rest as testing set.

Table 2
Training time per epoch comparison on all datasets.

Dataset	#(users)	#(items)	#(periods)	Sparsity
New York	1,083	9,989	85,185	99.51%
Tokyo	2,293	15,177	182,281	99.58%
Gowalla	1,186	41,275	185,630	99.61%
Weeplace	13,819	37,471	1,429,620	99.86%

6.1.2. Evaluation metrics

In the testing stage, the probabilities of all items are calculated and top K items are generated for a given user at a specific time segment. The average of two standard evaluation metrics: Hit Ratio(HR@K) and Normalized Discounted Cumulative Gain ($NDCG@K$) are used to evaluate all of the methods. $HR@K$ represents the percentage of correct items in the top K recommendations, $NDCG@K$ is applied to measure the ranking quality of the recommendation list, which assigns larger weights to higher positions. The value for HR@K and NDCG@K are calculated as follows:

$$HR@K = \frac{|P_u \cap T_u|}{min\,(k, |T_u|)} \tag{23}$$

where P_u is the top K predictions for user u, T_u represents the ground truth of the user u.

$$NDCG@K = \frac{\sum_{k=1}^{K} p_k / \log_2(k+1)}{\sum_{k=1}^{min(K,|T_u|)} 1/(\log_2(k+1))} \tag{24}$$

where p_k indicates the correlation of recommendation results at position k of the top K recommendation list.

6.1.3. Baselines

To evaluate the effectiveness of our method, three groups of the state-of-the-art recommendation methods: Markov-based, sequential and time-aware models, are selected as the baselines.

Markov-based Model

- FPMC (Rendle et al., 2010b): A hybrid model combines Markov transition and matrix factorization for next basket recommendation, which takes both collective-item dependencies and user's personal preferences into account.
- HRM (Wang et al., 2015): A hierarchical representation model for next basket recommendation, where non-linear combination of both the collective-item dependencies and users' preferences is considered.

Sequential Model

- SASRec(2018) (Kang & McAuley, 2018): A sequential recommendation model which captures point-wise dependencies of users' behaviors via the state-of-art transformer framework.
- DynamicRec (Tanjim et al., 2020): A sequential model that relies on dynamic convolutions to model the sequential behaviors in the session-based setting.
- CLEA(2021) (Qin et al., 2021): A basket-based recommendation method which denoises the history baskets and keeps exact items relevant to the target item via contrastive learning.

Time-aware Model

- DSNTSP (Sun et al., 2020): A temporal recommendation model which captures multi-level correlations of items and sets, as well as the time signals via a co-transformer module.
- TimelyRec (Cho et al., 2021): A time-aware recommendation method which jointly considers the periodical feature of user preference and the time-evolving influence of recent events.
- DGSR (Zhang et al., 2022): A GNN-based recommendation model that connects different user sequences through a dynamic graph structure, with consideration of time and order information.

Specifically, the baselines have the following features. (1) Both FPMC and HRM focus on the Markov transition of user behaviors. The former (FPMC) resorts to matrix factorization, and the latter (HRM) relies on neural-based technique to represent and capture the temporal patterns in the low-dimensional representation space. Both of them are highly related with our method. (2) SASRec, DynamicRec and CLEA are the state-of-the-art sequential recommendation methods, which assume the temporal dependency of items (or item sets) can be reduced to sequential dependency. And these methods focus on capturing the sequential correlations via neural-based techniques. (3) DSNTSP, TimelyRec and DGSR concern the time-evolving item–item transitions, which consider that time is an important variable that influences the transition patterns of user behaviors. And they incorporate time signals in various ways. Comparison against these baselines provides an investigation on the validity of our method.

6.1.4. Implementation details

For our method, the regularization parameter λ in Eq. (22) is gridly searched from the set $\{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$. The learning rate and batchsize are respectively set as 10^{-5} and 256. The dimension d of the latent factor is set to be 500, and 30 negative samples are generated for each positive sample. Our source code will be shared on github.[1]

For the baselines, the dimension d is set to be consistent with our method, other implementation parameters are tuned by grid search according to the relevant literatures. Specifically, for the Markov-based and sequential baselines, user activities at the same time segment of consecutive periods are treated as the item or basket sequence so as to make the period-based recommendation. For the time-aware methods, i.e. DSNTSP, TimelyRec and DGSR, timestamps are incorporated according to their literatures. All the experiments are conducted on the GeForce RTX 2080 Ti GPUs with fixed random seeds.

6.2. Overall performance (RQ1)

The overall performance of the proposed NeuralTMT and the baselines are given in Table 3. As shown in this table, NeuralTMT significantly outperforms the baselines by a large margin. Especially on Gowalla, the relative improvement is 16.17% and 22.81% for HR@5 and NDCG@5 respectively. The results demonstrate that our method is effective to learn the time-heterogeneous transition and enhance the performance of temporal recommendation.

We also have some observations on the baselines. Firstly, Markov-based models FPMC and HRM are comparable with the other baselines on four datasets, which indicates that short-term transition patterns play an important role in temporal recommendation. However, since they are based on the assumption that the Markov transition is time-homogeneous, therefore they cannot capture the time-evolving temporal dependency that is more complicated. Besides, as for the sequential models SASRec, CLEA and DynamicRec, since they simply model the sequential dependencies of users' behaviors and meanwhile ignore the personalized temporal patterns, they are demonstrated to be disadvantageous on the temporal recommendation task. Furthermore, as for the first two Time-aware models, DNSTSP and TimelyRec, they perform worse than the proposed method. This is probably because both of them only consider temporal information as a

[1] https://github.com/wangwencui896/NeuralTMT.

Table 3
Experiment results of different methods. The best results among all baselines are denoted in underline, and our model's results are in bold.

Dataset	Metrics	FPMC	HRM	SASRec	DynamicRec	CLEA	DSNTSP	TimelyRec	DGSR	NeuralTMT	Imp.%
New York	HR@5	0.4575	0.4680	0.4273	0.4422	0.4260	0.4217	0.4014	0.4431	**0.5156**	10.17%
	HR@10	0.5751	0.5818	0.5245	0.5342	0.5219	0.5435	0.5295	0.5475	**0.5981**	2.80%
	NDCG@5	0.3698	0.3873	0.3525	0.3718	0.3381	0.3357	0.3193	0.3582	**0.4477**	1 5.59%
	NDCG@10	0.4102	0.4267	0.3870	0.4042	0.3683	0.3768	0.3619	0.3919	**0.4758**	1 1.51%
Tokyo	HR@5	0.4383	0.4148	0.3726	0.3953	0.4005	0.4305	0.3507	0.4212	**0.4558**	3.99%
	HR@10	0.5369	0.5146	0.4576	0.4680	0.4871	0.5225	0.4518	0.5059	**0.5393**	0.45%
	NDCG@5	0.3858	0.3572	0.3368	0.3593	0.3394	0.3654	0.2848	0.3643	**0.4180**	8.35%
	NDCG@10	0.4249	0.3972	0.3725	0.3887	0.3720	0.4007	0.3295	0.3963	**0.4494**	5.77%
Gowalla	HR@5	0.3728	0.3518	0.3331	0.3614	0.3188	0.3409	0.3068	0.3696	**0.4331**	16.17%
	HR@10	0.4498	0.4317	0.3974	0.4316	0.3917	0.4348	0.3919	0.4494	**0.5037**	11.98%
	NDCG@5	0.3310	0.3144	0.3069	0.3314	0.2795	0.2962	0.2578	0.3177	**0.4103**	23.81%
	NDCG@10	0.3682	0.3530	0.3411	0.3688	0.3135	0.3371	0.2960	0.3560	**0.4460**	20.93%
Weeplace	HR@5	0.4556	0.5287	0.4701	0.5330	0.5178	0.5210	0.4237	0.5401	**0.5826**	7.87%
	HR@10	0.5680	0.6334	0.5715	0.6003	0.6001	0.6283	0.5335	0.6133	**0.6617**	3.74%
	NDCG@5	0.3734	0.4457	0.4179	0.4845	0.4404	0.4366	0.3457	0.4531	**0.5270**	8.77%
	NDCG@10	0.4123	0.4808	0.4494	0.5126	0.4677	0.4725	0.3865	0.4854	**0.5514**	7.57%

Table 4
Performance comparison of some different variants.

Method	New York		Tokyo		Gowalla		Weeplace	
	HR@5	NDCG@5	HR@5	NDCG@5	HR@5	NDCG@5	HR@5	NDCG@5
NeuralTMT	**0.5156**	**0.4477**	**0.4558**	**0.4180**	**0.4331**	**0.4103**	**0.5827**	**0.5271**
Variant 1: Time-homogeneous	0.4858	0.4067	0.4420	0.3936	0.3999	0.3623	0.5641	0.4862
Variant 2: Remove fusion weight	0.5052	0.4387	0.4378	0.4005	0.4230	0.4003	0.5653	0.5157
Variant 3: Remove personalized factor	0.3005	0.2481	0.2477	0.2137	0.3012	0.2852	0.3912	0.3521

kind of attribute and mix it with other attributes, which hence does not reflect the unique feature of time. Another Time-aware model DGSR also produces inferior results than our model, probably because it mainly focuses on the dynamic collaborative filtering signals but overlooks the personalized patterns in constructing the time-stamped user–item interaction graph.

6.3. Ablation study (RQ2)

To verify the validity of assumptions and strategies in the proposed method, we conduct an ablation study by removing some modules or changing some strategies. Table 4 shows the overall performance of the variants, and discussions are provided for each variant in the following subsections.

6.3.1. Effect of time-heterogenous factors

In order to investigate the importance of considering the time-heterogeneous features of transition patterns, we conduct an ablation study by making the embeddings: $\tilde{v}_i^m$, v_i^m and $\tilde{v}_u^m$ shared across different time-segments, which is denoted as "variant 1: Time-homogeneous". And the results are given in the Table 4. It is demonstrated that the time-homogeneous variant obviously performs worse than the proposed NeuralTMT on all of the datasets, which verifies that transition patterns of user behaviors are different from one time segment to another, and hence it is necessary to take into account the time-evolving features by adding time signals in the representation module.

6.3.2. Effect of fusing the long-term and short-term components

To investigate the importance of fusing the long-term and short-term components, we conduct an ablation study by removing the learnable fusion weight, that is, the long-term and short-term components are set to equally contribute to the loss function. This variant is denoted as "Variant 2: Remove fusion weight". As shown in Table 4, on all the datasets, NeuralTMT outperforms variant 2, which indicates long-term and short-term components have different effects on temporal recommendation, and taking into account the different influences is conducive to improving the recommendation performance. Specifically, we will visualize the fusion weights in Section 6.4.2 to investigate the different roles of long-term and short-term components on different datasets.

6.3.3. Effect of personalized factor

To investigate the effect of considering the personalized characteristic of temporal transition, we remove the user's feature embedding $\tilde{v}_u^m$ from NeuralTMT and then the proposed model is reduced to a variant that co-factorizes multiple un-personalized transition matrices, which is denoted as "Variant 3". As demonstrated in Table 4, removing personalized factors dramatically reduces the performance, which is consistent with the observation that users' temporal behaviors are greatly influenced by their own interests. Besides, the embedding $\tilde{v}_u^m$ also facilitates integrating user's long-term preference, and hence it is crucial for the temporal recommendation.

6.3.4. Impact of the embedding dimension

As a factorization-based model, the dimension of latent factor d may have a vital impact on the recommendation performance. Therefore, we set d to be different values to investigate its influence on the recommendation performance. Specifically, d is set in $\{50, 200, 500, 800\}$, and HR@5 is used to evaluate the effect. The results on four datasets are plotted in Fig. 5. It is demonstrated that on all the datasets the performance of NeuralTMT gradually improves as the embedding dimension increases, and reaches a stable point when $d = 500$, which indicates that the dimension size $d = 500$ is a fairly appropriate value to capture the complex temporal transition dependencies on these datasets.

6.4. Discussions on some of the learnable weights (RQ3)

6.4.1. Visualizations of attention weights

To further investigate whether our model can effectively capture the time-heterogeneous dependencies between consecutive periods and learn meaningful transition patterns, we make statistical analysis on the attention weights $\alpha_{n,m}^i$ from the training

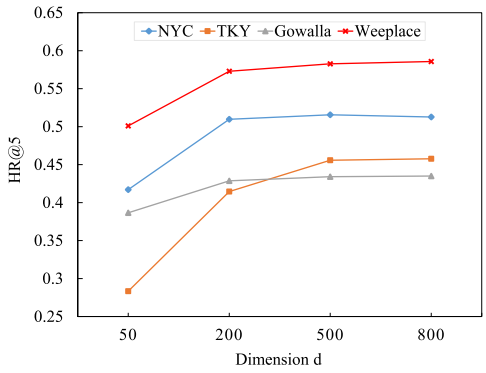

Fig. 5. Impact of the dimension d.

dataset. And the average value of attention weights over different items are plotted in Fig. 6. As shown in this figure, four heatmaps illustrate the average attention weights that respectively correspond to NYC, TKY, GOWALLA and Weeplace datasets. As is shown in subfigure (a) and (c), large attention weights are mainly distributed along the diagonal. This indicates that on NYC and Gowalla, users' behaviors at the current time segment are closely related to the same time segment of the previous period and slightly influenced by other time segments of the previous period. However, for the datasets TKY and Weeplace, as illustrated in subfigure (b) and (d), large values of attention weight are mainly distributed along the same line (i.e. morning). This indicates that for these two datasets, users' behaviors in different time segments are highly related to the morning of the previous period. A probable cause is that for these two datasets users' historical activities are fairly sparse in all of the time segments except morning, which makes it difficult to capture the temporal dependencies between the time segments with low user activities.

6.4.2. Visualizations of fusion weights

The weights of short-term and long-term components after training NeuralTMT on four datasets are plotted in Fig. 7. As shown in this figure, on two datasets (NYC, Gowalla), z_m is significantly larger than $1 - z_m$, which indicates the short-term component is weighted more important in calculating the transition probability. On the contrary, z_m is significantly smaller than $1-z_m$ on the Weeplace, while relatively smaller on the TKY, which implies short-term components are weighted significantly less important on Weeplace while slightly less important on the TKY. The variation of the short-term and long-term weights on different datasets suggests that it is necessary to make the weights learnable from the data. And this also explains why our model outperforms the baselines FPMC and HRM, as our model has the ability to well leverage the short-term and long-term components by the learnable weights, which is conducive to improving the overall recommendation performance.

6.5. Discussions on runtime (RQ4)

To further investigate the computational cost of the proposed method, the runtime of NeuralTMT and the baselines is given in Tables 5 and 6. For the training stage, as shown in Table 5, NeuralTMT costs more time per epoch than the Markov-based and sequential models, the possible reason is that our model aims to capture time-varying dependencies of user behaviors, which makes it more complex among these methods. However, the proposed NeuralTMT is almost the most efficient when compared with the time-aware baselines (i.e.DSNTSP, TimelyRec and DGSR), probably because our method is based on the tensor-factorization technique, and requires relatively shallow neural network modules to learn the personalized temporal patterns. For the testing stage, as demonstrated in Table 6, the proposed NeuralTMT is fairly efficient compared with the up-to-date sequential and time-aware baselines, which verifies its applicability in real scenario.

Table 5
Training time per epoch comparison on all datasets.

Method	NYC	TKY	Gowalla	Weeplace
FPMC	7	25	42	198
HRM	6	23	38	132
SAS	12	40	47	204
CLEA	3	5	10	43
DynamicRec	4	9	12	78
DSNTSP	77	110	290	1794
TimelyRec	29	56	101	354
DGSR	18	58	118	336
NeuralTMT	22	41	83	312

#The time unit is minute in this table.

Table 6
Execution time per user per period.

Method	NYC	TKY	Gowalla	Weeplace
FPMC	5	5	6	5
HRM	3	3	5	3
SAS	4	6	7	5
CLEA	11	13	17	12
DynamicRec	5	6	6	5
DSNTSP	976	1095	1295	1794
TimelyRec	9	10	13	9
DGSR	19	28	45	19
NeuralTMT	8	9	12	8

#The time unit is millisecond in this table.

6.6. Qualitative analysis

As demonstrated in Table 3, the advantage of NeuralTMT is significant on Gowalla, while relatively marginal on TKY dataset. In order to investigate what factors affect its advantages, we make data exploration on these two datasets. For each user, we calculate the Pearson's correlation (as Eq. (25)) of his activities in different time segments of consecutive periods.

$$\rho_{n,m} = \frac{\mathbb{C}\mathbb{o}\mathbb{v}(\mathcal{S}_{t,n}, \mathcal{S}_{t-1,m})}{\sigma_{\mathcal{S}_{t,n}}\sigma_{\mathcal{S}_{t-1,m}}}, 1 \leq n, m \leq M; \tag{25}$$

Here, $\mathbb{C}\mathbb{o}\mathbb{v}$ denotes the covariance of two vectors, σ denotes the standard deviation of a vector.

Figs. 8 and 9 illustrate the distribution of different users' Pearson's correlation on Gowalla and TKY datasets. As shown by these figures, the heterogeneity of users' temporal behaviors on Gowalla dataset are much more significant than TKY dataset: both the medium value and distribution of pearson correlation on Gowalla dataset is more diverse than TKY dataset. Besides, another factor: the proportion of long-term and short-term patterns, seems also have an impact on the advantage of neuralTMT, because as illustrated by Fig. 7, short-term weight is overwhelming on Gowalla dataset, which may indicate that NeuralTMT is more adept at capturing short-term temporal patterns.

7. Conclusion

In this paper, a tensor co-factorization based method, which aims to capture the time-heterogeneous transition patterns of user activities, is proposed for the temporal recommendation.

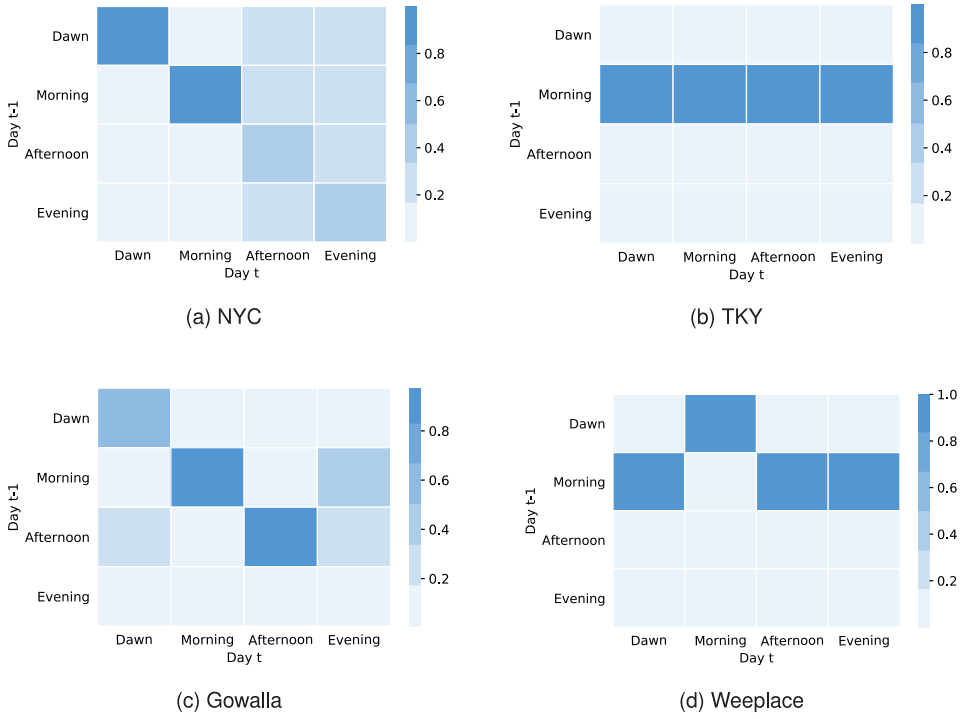

Fig. 6. Visualizations of the average value of attention weights $\alpha^{i}_{n,m}$. A darker color represents a more close link. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

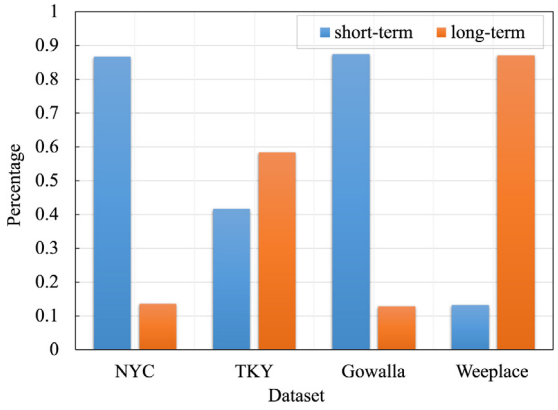

Fig. 7. Visualizations of the fusion weights on four datasets.

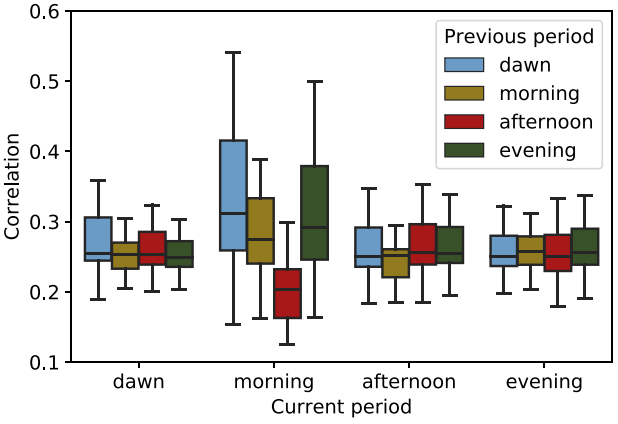

Fig. 8. Pearson correlation coefficient of different time-segments in consecutive periods (Gowalla Dataset).

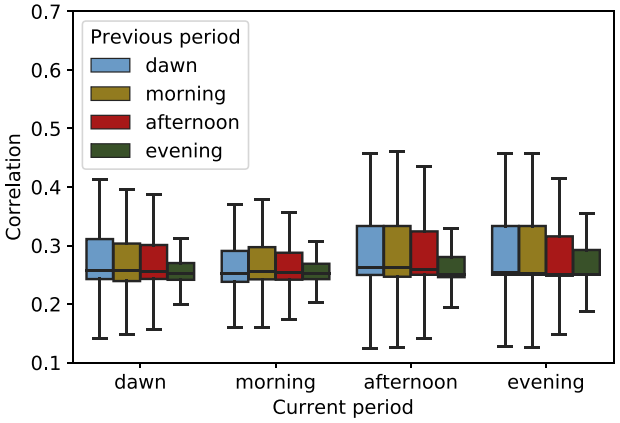

Fig. 9. Pearson correlation coefficient of different time-segments in consecutive periods (TKY Dataset).

In the proposed method, Markov transition tensors are introduced to represent user's temporal behaviors and a tensor cofactorization model with a bayesian-personalized ranking objective is designed to capture the time-heterogeneous transitions. And an extension based on nonlinear neural-network mappings and attention modules is proposed to learn the time-heterogeneous temporal dependency in a kind of flexible and effective way.

The major advantage of the proposed method is that it focuses on the time-heterogeneous property of user behaviors and effectively learns the time-evolving temporal dependencies. Besides, the proposed method is theoretically based on tensor cofactorization techniques and the two-level temporal hierarchy (period→time segment) is considered so as to make a multi-granularity temporal recommendation, where the time granularity can be further extended (e.g. period→time segment → time slice). However, the limitation of NeuralTMT is that higher-order

temporal dependency that involves more than two periods has not been taken into account. In the future, we would like to extend NeuralTMT to include higher-order temporal dependency so as to capture more complicated temporal patterns.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Athanasopoulos, G., Hyndman, R. J., Kourentzes, N., & Petropoulos, F. (2017). Forecasting with temporal hierarchies. *European Journal of Operational Research*, *262*(1), 60–74.

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, *3*(1), 1–122.

Campos, P. G., Díez, F., & Cantador, I. (2014). Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modelling User Adaptive Interactions*, *24*(1–2), 67–119.

Carroll, J. D., & Chang, J.-J. (1970). Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika*, *35*(3), 283–319.

Chatzis, S. (2014). Dynamic bayesian probabilistic matrix factorization. In *AAAI, Vol. 28* (pp. 1731–1737).

Cho, J., Hyun, D., Kang, S., & Yu, H. (2021). Learning heterogeneous temporal patterns of user preference for timely recommendation. In *WWW '21: The web conference 2021, virtual event / Ljubljana, Slovenia, April 19-23, 2021* (pp. 1274–1283).

Donkers, T., Loepp, B., & Ziegler, J. (2017). Sequential user-based recurrent neural network recommendations. In P. Cremonesi, F. Ricci, S. Berkovsky, & A. Tuzhilin (Eds.), *Proceedings of the eleventh ACM conference on recommender systems, RecSys 2017, Como, Italy, August 27-31, 2017* (pp. 152–160). ACM.

Du, N., Wang, Y., He, N., Sun, J., & Song, L. (2015). Time-sensitive recommendation from recurrent user activities. *Advances in Neural Information Processing Systems*, *28*, 3492–3500.

Fan, Z., Liu, Z., Zhang, J., Xiong, Y., Zheng, L., & Yu, P. S. (2021). Continuous-time sequential recommendation with temporal graph collaborative transformer. In *Proceedings of the 30th ACM international conference on information & knowledge management* (pp. 433–442).

Harshman, R. A. (1970). *Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis*: *UCLA working papers in phonetics 16*, (pp. 1–84). University of California at Los Angeles Los Angeles.

Hidasi, B., & Karatzoglou, A. (2018). Recurrent neural networks with top-k gains for session-based recommendations. In *CIKM* (pp. 843–852).

Hidasi, B., Karatzoglou, A., Baltrunas, L., & Tikk, D. (2016). Session-based recommendations with recurrent neural networks. In *ICLR*.

Huang, C., Chen, J., Xia, L., Xu, Y., Dai, P., Chen, Y., et al. (2021). Graph-enhanced multi-task learning of multi-level transition dynamics for session-based recommendation. In *Proceedings of the AAAI conference on artificial intelligence, Vol. 35* (pp. 4123–4130).

Kang, W.-C., & McAuley, J. (2018). Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)* (pp. 197–206). IEEE.

Kim, D., Park, C., Oh, J., Lee, S., & Yu, H. (2016). Convolutional matrix factorization for document context-aware recommendation. In *Proceedings of the 10th ACM conference on recommender systems* (pp. 233–240).

Koren, Y. (2009). Collaborative filtering with temporal dynamics. In *SIGKDD* (pp. 447–456).

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, *42*(8), 30–37.

Le, D., Lauw, H. W., & Fang, Y. (2019). Correlation-sensitive next-basket recommendation. In *IJCAI* (pp. 2808–2814).

Li, X., Jiang, M., Hong, H., & Liao, L. (2017). A time-aware personalized point-of-interest recommendation via high-order tensor factorization. *ACM Transactions on Information Systems (TOIS)*, *35*(4), 1–23.

Liang, D., Altosaar, J., Charlin, L., & Blei, D. M. (2016). Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence. In *Proceedings of the 10th ACM conference on recommender systems* (pp. 59–66).

Ma, C., Kang, P., & Liu, X. (2019). Hierarchical gating networks for sequential recommendation. In *SIGKDD* (pp. 825–833).

Qin, Y., Wang, P., & Li, C. (2021). The world is binary: Contrastive learning for denoising next basket recommendation. In *SIGIR* (pp. 859–868).

Rendle, S., Freudenthaler, C., & Schmidt-Thieme, L. (2010a). Factorizing personalized markov chains for next-basket recommendation. In *WWW* (pp. 811–820).

Rendle, S., Freudenthaler, C., & Schmidt-Thieme, L. (2010b). Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on world wide web* (pp. 811–820).

Sun, L., Bai, Y., Du, B., Liu, C., Xiong, H., & Lv, W. (2020). Dual sequential network for temporal sets prediction. In *SIGIR* (pp. 1439–1448).

Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., et al. (2019). BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management* (pp. 1441–1450).

Tang, J., & Wang, K. (2018). Personalized Top-N sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining, WSDM 2018* (pp. 565–573). Marina Del Rey, CA, USA.

Tanjim, M. M., Ayyubi, H. A., & Cottrell, G. W. (2020). DynamicRec: A dynamic convolutional network for next item recommendation.. In *CIKM* (pp. 2237–2240).

Wang, J., Ding, K., Hong, L., Liu, H., & Caverlee, J. (2020). Next-item recommendation with sequential hypergraphs. In *SIGIR* (pp. 1101–1110).

Wang, P., Guo, J., Lan, Y., Xu, J., Wan, S., & Cheng, X. (2015). Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval* (pp. 403–412).

Wang, S., Hu, L., Cao, L., Huang, X., Lian, D., & Liu, W. (2018). Attention-based transactional context embedding for next-item recommendation. In *Proceedings of the AAAI conference on artificial intelligence, Vol. 32*.

Wang, X., Liu, X., Li, L., Chen, X., Liu, J., & Wu, H. (2021). Time-aware user modeling with check-in time prediction for next POI recommendation. In C. K. Chang, E. Daminai, J. Fan, P. Ghodous, M. Maximilien, Z. Wang, R. Ward, & J. Zhang (Eds.), *2021 IEEE international conference on web services, ICWS 2021, Chicago, IL, USA, September 5-10, 2021* (pp. 125–134). IEEE.

Wu, L., Li, S., Hsieh, C., & Sharpnack, J. (2020). SSE-PT: sequential recommendation via personalized transformer. In *RecSys 2020: Fourteenth ACM conference on recommender systems, virtual event, Brazil, September 22-26, 2020* (pp. 328–337). ACM.

Wu, X., Shi, B., Dong, Y., Huang, C., & Chawla, N. V. (2019). Neural tensor factorization for temporal interaction learning. In *Proceedings of the Twelfth ACM international conference on web search and data mining* (pp. 537–545).

Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., & Tan, T. (2019). Session-based recommendation with graph neural networks. In *AAAI, Vol. 33* (pp. 346–353).

Xia, X., Yin, H., Yu, J., Wang, Q., Cui, L., & Zhang, X. (2021). Self-supervised hypergraph convolutional networks for session-based recommendation. In *Thirty-fifth AAAI conference on artificial intelligence, AAAI 2021, thirty-third conference on innovative applications of artificial intelligence, IAAI 2021, the eleventh symposium on educational advances in artificial intelligence, EAAI 2021, virtual event, February 2-9, 2021* (pp. 4503–4511). AAAI Press.

Xu, C., Zhao, P., Liu, Y., Xu, J., Sheng, V. S., Cui, Z., et al. (2019). Recurrent convolutional neural network for sequential recommendation. In *The world wide web conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019* (pp. 3398–3404). ACM.

Xue, H.-J., Dai, X., Zhang, J., Huang, S., & Chen, J. (2017). Deep matrix factorization models for recommender systems. In *IJCAI, Vol. 17* (pp. 3203–3209). Melbourne, Australia.

Ye, W., Wang, S., Chen, X., Wang, X., Qin, Z., & Yin, D. (2020). Time matters: Sequential recommendation with complex temporal information. In *SIGIR* (pp. 1459–1468).

Yu, F., Liu, Q., Wu, S., Wang, L., & Tan, T. (2016). A dynamic recurrent model for next basket recommendation. In *Proceedings of the 39th international ACM SIGIR conference on research and development in information retrieval* (pp. 729–732).

Yuan, Q., Cong, G., & Sun, A. (2014). Graph-based point-of-interest recommendation with geographical and temporal influences. In *Proceedings of the 23rd ACM international conference on conference on information and knowledge management, CIKM 2014, Shanghai, China, November 3-7, 2014* (pp. 659–668). ACM.

Zhang, Q., Cao, L., Shi, C., & Niu, Z. (2021). Neural time-aware sequential recommendation by jointly modeling preference dynamics and explicit feature couplings. *IEEE Transactions on Neural Networks and Learning Systems*.

Zhang, M., Wu, S., Gao, M., Jiang, X., Xu, K., & Wang, L. (2020). Personalized graph neural networks with attention mechanism for session-aware recommendation. *IEEE Transactions on Knowledge and Data Engineering*.

Zhang, M., Wu, S., Yu, X., Liu, Q., & Wang, L. (2022). Dynamic graph neural networks for sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering*.

Zhao, S., Zhao, T., Yang, H., Lyu, M., & King, I. (2016). STELLAR: Spatial-temporal latent ranking for successive point-of-interest recommendation. In *AAAI, Vol. 30*.