TiSSA: A Time Slice Self-Attention Approach for Modeling Sequential User Behaviors

Chenyi Lei
Alibaba Group
Hangzhou, China
chenyi.lcy@alibaba-inc.com

Shouling Ji
Zhejiang University
Hangzhou, China
sji@zju.edu.cn

Zhao Li
Alibaba Group
Hangzhou, China
lizhao.lz@alibaba-inc.com

ABSTRACT

Modeling user behaviors as sequences provides critical advantages in predicting future user actions, such as predicting the next product to purchase or the next song to listen to, for personalized search and recommendation. Recently, recurrent neural networks (RNNs) have been adopted to leverage their power in modeling sequences. However, most of the previous RNN-based work suffers from the complex dependency problem, which may lose the integrity of highly correlated behaviors and may introduce noises derived from unrelated behaviors. In this paper, we propose to integrate a novel **Ti**me **S**lice **S**elf-**A**ttention (TiSSA) mechanism into RNNs for better modeling sequential user behaviors, which utilizes the time-interval-based gated recurrent units to exploit the temporal dimension when encoding user actions, and has a specially designed time slice hierarchical self-attention function to characterize both local and global dependency of user actions, while the final context-aware user representations can be used for downstream applications. We have performed experiments on a huge dataset collected from one of the largest e-commerce platforms in the world. Experimental results show that the proposed TiSSA achieves significant improvement over the state-of-the-art. TiSSA is also adopted in this large e-commerce platform, and the results of online A/B test further indicate its practical value.

CCS CONCEPTS

• Information systems → Recommender systems; **• Computing methodologies → Neural networks**; *Ranking*.

KEYWORDS

Attention Mechanism; Recurrent Neural Networks; User Modeling

ACM Reference Format:
Chenyi Lei, Shouling Ji, and Zhao Li. 2019. TiSSA: A Time Slice Self-Attention Approach for Modeling Sequential User Behaviors. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313495

1 INTRODUCTION

Due to the increasing abundance of information on the Web, helping users filter information according to their current intents/preferences is more and more required, which emphasizes the importance of personalized search and recommendation. Analogous to the assumption in natural language processing (NLP) that the topic of an article can be represented by the sequential words or sentences of the article [22], user's current intents/preferences can be captured by his/her sequential behaviors over the timeline. To model sequential user behaviors for downstream applications, recurrent neural network (RNN) has been considered [14, 26, 29], due to its remarkable performance on many sequential learning problems. For example, [26] endowed both users and movies with a long- and short-term memory (LSTM) to predict user's future behavioral trajectories. Despite a certain success of above RNN-based methods, modeling sequential user behaviors remains a challenging problem.

One issue is that long-term dependencies are still very hard to preserve while maintaining middle- and short-term user behaviors. Recent work has indicated that attention mechanism can help to achieve state-of-the-art performance on a large number of NLP and computer vision (CV) tasks, such as neural machine translation [1] and image captioning [13]. Attention mechanism introduced into deep networks provides the functionality to focus on portions of input data or features to fulfill the given task, which might bring benefits for the problem of complex dependencies within user historical behaviors.

However, previous work along this line for modeling sequential user behaviors has some limitations [16, 19, 28, 31]. Both of [16, 19] only considered the primary attention mechanisms, which need to be better designed. Furthermore, they employ the bi-directional RNN structure, which is quite complicated and makes it challenging to be parallelized in the real world. [28, 31] abandoned RNN structures and proposed the self-attention architecture instead for different type sequential user behaviors. Nevertheless, the sequence structure information is incomplete or even lost in the RNN-free framework, whereas RNN's autoregressive modeling can, in theory, capture potentially infinitely long-term sequence with a small number of parameters [3].

Another critical issue is that most of the previous work projects each user action into an individual embedding vector. However, user sequential behaviors are usually not consistent with action time and intents, which is hard to be distinguished and encoded [32]. Consequently, the single user action embedding is not always robust. In recent work, [20] demonstrated that taking session-level representations could have very different expressive power. Nevertheless, it is not still reasonable to divide sequential user behaviors according to fix-length time window or online session of users. Furthermore, the relationships among sessions are not considered in previous session-based related work [8, 9].

These two issues need to be optimized in advanced approaches and systems. In this paper, we propose a time slice self-attention approach and corresponding ranking system, called TiSSA, for modeling sequential user behaviors, which we currently use for the personalized recommendation. We first project user sequential behaviors into session-level representations by our proposed time-interval-based gated recurrent unit (GRU). Then we split these session-level representations into slices according to the multi-scale time window. The intra-slice self-attention is performed on each slice independently, to capture the local dependency within each slice. We further perform inter-slice self-attention for outputs of each slice to exploit the global dependency. Finally, a feature fusion gate combines the outputs of intra- and inter-slice self-attention to generate context-aware representations for the user. We perform vanilla attention between these context-aware representations and the ranking item vectors, whose outputs are fed into a ranking neural network for making recommendations. The effectiveness of our adapted approach and ranking system is evaluated in our offline and online experiments on Tmall[1].

2 BACKGROUND AND RELATED WORK

2.1 Gated Recurrent Units

A gated recurrent unit (GRU) was proposed by [5] to overcome drawbacks of normal RNNs, i.e., vanishing and exploding gradients. Specifically, the activation h_t of the GRU at time t is a linear interpolation between the previous activation h_{t-1} and the candidate activation $\tilde{h}_t$:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{1}$$

where $\odot$ is an element-wise multiplication, and an update gate z_t decides how much the unit updates its activation. The update gate is computed by

$$z_t = \sigma(W_z^{(1)} x_t + W_z^{(2)} h_{t-1}) \tag{2}$$

where x_t is input at time t, and σ is sigmoid function. The candidate activation $\tilde{h}_t$ is computed by

$$\tilde{h}_t = tanh(W_h^{(1)} x_t + W_h^{(2)} (r_t \odot h_{t-1})) \tag{3}$$

where r_t is a reset gate and computed by

$$r_t = \sigma(W_r^{(1)} x_t + W_r^{(2)} h_{t-1}) \tag{4}$$

In Eq. 2 - Eq. 4, weight parameters W_z^*, W_h^* and W_r^* connect different inputs and gates.

2.2 Attention Mechanism

Vanilla attention is introduced by [1] firstly in the encoder-decoder framework, to provide more accurate alignment for each position in the machine translation task. In particular, given token embeddings of a source sequence $\mathbf{x} = \{x_1, x_2, ..., x_n\} \in \mathcal{R}^{d \times n}$ and the vector of a query $q \in \mathcal{R}^d$, where d is the feature size of x_i and q, attention computes the alignment score between x_i and q by a function $f(x_i, q) \in \mathcal{R}$. A softmax function then transforms the scores $\{f(x_i, q)\}$ to a probability distribution $p(a|\mathbf{x}, q)$ by normalizing over all the n tokens of $\mathbf{x}$. Here a is an indicator of which token in $\mathbf{x}$ is important to q on specific task. Then the attention-based pooling can be formularized as $C = \sum_{i=1}^{n} p(a = i|\mathbf{x}, q) x_i \in \mathcal{R}^d$. In reality, additive attention is usually used, whose $f(x_i, q)$ is formulated as

$$f(x_i, q) = w^T \sigma(W^{(1)} x_i + W^{(2)} q) \tag{5}$$

where w is a weight vector to project additive function $\sigma(W^{(1)} x_i + W^{(2)} q)$ to a scalar.

Self-attentions are also studied in different mechanisms [25], which are very expressive and flexible for capturing inner-relations of the data at the encoder side. "Token2entirety" self-attention [12, 15] is one of widely used self-attention mechanism, which explores the importance between token x_i and the entire sequence $\mathbf{x}$, and compresses the sequence $\mathbf{x}$ into a vector. It removes q from Eq. 5 and adds bias term, i.e.,

$$f(x_i) = w^T \sigma(W^{(1)} x_i + b^{(1)}) + b^{(2)} \tag{6}$$

Recently, [23] further demonstrated the advantage of mask-style "token2token" self-attention mechanism, which explores the dependency between token x_i and token x_j from the same sequence $\mathbf{x}$, and generates context-aware coding for each element. At the same time, lots of work have shown that linear projections over multi-subspaces could improve the performance in various tasks [25, 31].

Another work related to our paper is hierarchical attention mechanism, which is also demonstrated as a practical design in CV/NLP tasks [17, 24, 27]. It is also a valid design for our problem, as it addresses the local and global dependency issue, which profoundly influences the accuracy of user intention summarization.

3 TWO PROPOSED IMPROVEMENTS

We propose two significant improvements to GRU structure and basic attention mechanism in this paper.

3.1 Time-GRU

Since GRU is originally designed for NLP tasks, there is no consideration of time intervals within inputs, which are very important for modeling sequential user behaviors. Furthermore, as discussed in Section 1, we argue that taking session-level inputs could have very different expressive power comparing to inputs of individual user behaviors (c.f. Section 5.4 for an in-depth analysis). Because that user behavior sessions are changing along time with the property of overlapping [7], the natural idea is to divide and embed user behaviors into sessions over behavior time smoothly. Inspired by some previous work with multi-timescale designs [18], we propose a time-interval-based GRU, called Time-GRU, to model user session-level representations. Specifically, we rewrite reset gate as a time-dependent format:

$$r_t = \sigma(W_r^{(1)} x_t + W_r^{(2)} h_{t-1} + W_r^{(3)} \triangle t_t + b_r) \tag{7}$$

Similarly, we have time-dependent update gate:

$$z_t = \sigma(W_z^{(1)} x_t + W_z^{(2)} h_{t-1} + W_z^{(3)} \triangle t_t + b_z) \tag{8}$$

In Eq. 7 and Eq. 8, $\triangle t_t \in \mathcal{R}$ is the time interval between adjacent actions, and b_* are bias terms. Our motivation is to utilize time interval information as a filter. In this way, the final activation h_t

[1]www.tmall.com, one of the largest e-commerce platforms in the world.

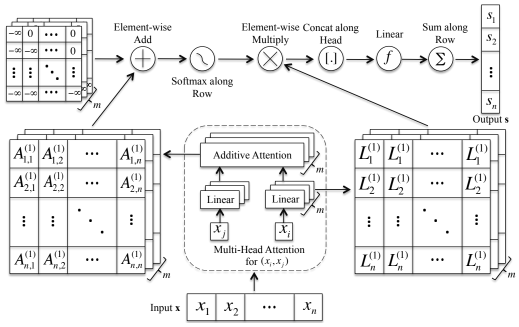

Figure 1: The proposed directional multi-head attention mechanism. Here, we use $A_{i,j}^{(k)}$ and $L_i^{(k)}$ to denote $head_{i,j}^{(k)} \in \mathcal{R}$ and $(x_i W_{ck}) \in \mathcal{R}^{d_c}$ in Eq. (9), respectively, where $k \in m$.

captures the session-level information by time t over changing time smoothly.

3.2 Directional Multi-Head Attention

Inspired by previous work [23, 31], we propose a new token2token self-attention mechanism to introduce the advantage of linear projections over multi-subspaces, which relates elements at directional positions from a single sequence by computing the multi-head attention [25] between each pair of tokens and generates context-aware coding for each $x_i \in \mathbf{x}$. In particular, as illustrated in Fig. 1, multi-head attention between each $x_i \in \mathbf{x}$ and $x_j \in \mathbf{x}$ is firstly computed. We linearly project x_i and x_j, which are both d-dimensional vectors, m times with different, learned linear projections to d_c and d_q dimensions, respectively. Then attention heads are computed. Specifically, for the k-th head of x_i and x_j, we have

$$head_{i,j}^{k} = w^T \sigma(W^{(1)}(x_i W_{ck}) + W^{(2)}(x_j W_{qk})) \tag{9}$$

where the projections are parameter matrices $W_{ck} \in \mathcal{R}^{d \times d_c}$, $W_{qk} \in \mathcal{R}^{d \times d_q}$, and $head_{i,j}^{k} \in \mathcal{R}, k \in m$. In practice, we set $d_c = d_q = d/m$. All attention heads are organized as a matrix $Rel \in \mathcal{R}^{n \times n \times m}$, whose each element is an attention scalar. Linear projections of x_i are organized as another matrix $Pro \in \mathcal{R}^{n \times n \times m}$, whose each row has the same element $(x_i W_{ck}) \in \mathcal{R}^{d_c}$.

In order to preserve temporality information in sequence, a directional mask matrix M with the same dimension with Rel is proposed,

$$M_{ij}^{k} = \begin{cases} 0, & i < j \\ -\infty, & otherwise \end{cases} \tag{10}$$

Given M and Rel, we perform an element-wise add operation, and follow a softmax normalization, each element of whose results is the attention weight for corresponding element in Pro. After weighting Pro with element-wise multiplication, matrix Pro is concatenated along attention heads as a new matrix with $n \times n$ dimensions, whose each element is a $(m \times d_c)$-dimentional vector and then projected to d-dimentional vector with a sharing learnable linear function. Finally, the projected matrix is summed along matrix row to generate token2token self-attention for all elements from $\mathbf{x}$, which is $\mathbf{s} = \{s_1, s_2, ...s_n\} \in \mathcal{R}^{d \times n}$ with the same dimensions with $\mathbf{x}$.

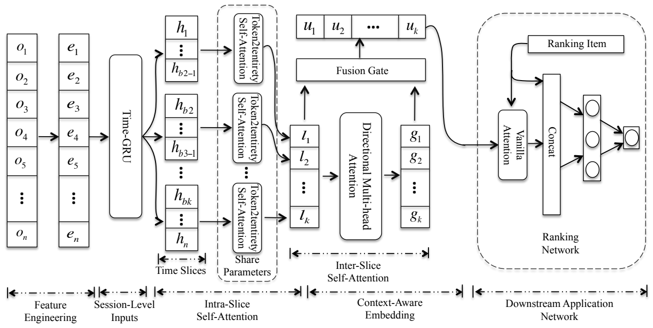

Figure 2: The proposed Time Slice Self-Attention approach and ranking system.

4 TIME SLICE SELF-ATTENTION APPROACH AND RANKING SYSTEM

A user can be represented as all his/her sequential behaviors $u = \{(o_i, t_i) | i = 1, 2, 3, ..., n\}$, where (o_i, t_i) indicates the interaction between user u and object o_i at timestamp t_i. For recommendation task, the objective of modeling sequential user behaviors is to predict the conditional probability of user's next action $p(o|u)$ for a certain given user u.

In the following of this section, we introduce our proposed time slice self-attention approach (TiSSA) and ranking system in detail, and the overall system is illustrated in Fig. 2. We divide the framework into several parts for a better description.

Feature Engineering. In our work, we take the user behavior object with deep and wide features as inputs, which is illustrated in Fig. 3. Each object o is represented by a multi-hot vector $o = \{v_1, v_2, ..., v_F\}$, where v_i indicates i-th one-hot feature such as ID-features and discretized statistical features. Following multi-hot vector inputs, behavior embedding layer transforms the multi-hot vector of o_i into a low-dimensional dense vector $e_i \in \mathcal{R}^{d_e}$ by linear mapping each feature of o_i, e.g., item id, to a dense vector and concatenating them. Note that, we enable embedding sharing when objects have features in common, e.g, item id and shop id, referring to the same entities. Then we have the behavior embedding vector for acted object o_i,

$$e_i = [W_{emb}^{(1)} v_1, W_{emb}^{(2)} v_2, ..., W_{emb}^{(F)} v_F] \tag{11}$$

where [.] denotes concatenating operation and $W_{emb}^{(i)}$ is learnable embedding function parameters for i-th feature.

Session-Level Inputs. As discussed in Section 1 and 2, we prefer to employ session-level representations as inputs for better capacity. In particular, we put behavior embedding vectors $e \in \mathcal{R}^{d_e \times n}$ into our proposed Time-GRU sequentially, whose hidden states $\mathbf{h} \in \mathcal{R}^{d_h \times n}$ represent the session-level inputs. In Time-GRU, sessions of user behaviors are distinguished by time intervals smoothly.

Hierarchical Self-Attention. Since much previous work indicates that time information has substantial power in modeling user intents, we further divide session-level inputs $\mathbf{h}$ into k multi-scale time slices, which is proved as a simple yet efficient method (c.f. Section 5.4). Different user behavior platforms have different granularities of time slicing, which are hyper-parameters. After that,

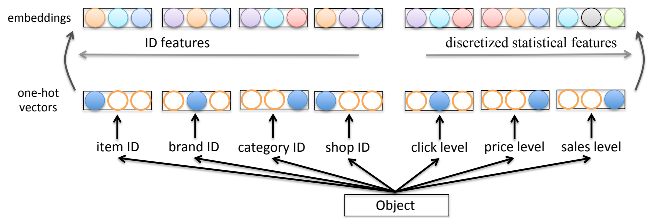

Figure 3: Feature engineering, which integrates both ID-features and statistical features.

we perform intra-slice and inter-slice self-attention hierarchically for modeling sequence dependencies at different levels.

Intra-Slice Self-Attention. We have two considerations for this level of self-attention. One is to project variable-length behaviors in different time slices into a fix-length encoding vector (zero-vector for empty time slice). The other is to capture the local dependency within a time slice. In particular, we employ token2entirety self-attention (c.f. Section 2.2) for this purpose, where shares all parameters in Eq. 6. Finally, local-context representations $\mathbf{l} = \{l_1, l_2, ..., l_k\} \in \mathcal{R}^{d_l \times k}$ at intra-slice level are generated by intra-slice self-attention, where k is the amount of time slices.

Inter-Slice Self-Attention. Another important challenging for modeling sequential user behaviors is to exploit the long-range global dependency among time slices. We perform our proposed directional multi-head self-attention (c.f. Section 3.2) after the intra-slice self-attention mechanism, which can not only capture dependencies but also preserve temporal orders. The outputs of inter-slice self-attention operation have the same dimension with inputs $\mathbf{l}$, which are denoted as $\mathbf{g} = \{g_1, g_2, ..., g_k\} \in \mathcal{R}^{d_g \times k}$.

Feature Fusion. The final context-aware representations $\mathbf{u} \in \mathcal{R}^{d_u \times k}$, where $d_u = d_g = d_l$, for each time slice is obtained by combining local and global context representations with a fusion gate. Motivated by the update processing of GRU, we design feature fusion gate as follows,

$$F = \sigma(W_F^{(1)}\mathbf{g} + W_F^{(2)}\mathbf{l} + b_F) \tag{12}$$

$$\mathbf{u} = F \odot \mathbf{l} + (1 - F) \odot \mathbf{g} \tag{13}$$

where W_F^* and b_F are learnable parameters.

Downstream Application Network. With generated context-aware representations, we can ensemble various kinds of neural networks according to downstream task requirement. In this paper, we focus on evaluating predictions of interactions within the user and items, and we set the downstream application network to be a point-wise ranking network. For each candidate item, its embedding vector q is calculated by the same embedding layer in our framework, and vanilla attention between $\mathbf{u}$ and q is performed to produce the attention pooling vector for the given user. After a concatenating operation and two fully connected layers, we have final sigmoid cross entropy loss function:

$$-\sum_{u,q} y_q \, log\, \sigma(f(u,q)) + (1 - y_q) \, log\,(1 - \sigma(f(u,q))) \tag{14}$$

where $y_q \in \{0, 1\}$ is the ground truth that indicates the user interacts the candidate item or not, and f is the downstream application network.

It is notable that feature engineering strategy can apply to many machine learning models where the designed features and ID-features exist. Time slice self-attention approach can be generalized to handle tasks involving sequential data with time intervals. The final context-aware representations can be further utilized by various downstream applications such as the clustering task.

5 EXPERIMENT

5.1 Offline Dataset

This huge dataset is real data and collected from Tmall, which describe the entire page viewed items and user feedbacks. Each instance contains user historical sequential clicked items with timestamps and the observed items currently. All items contain item id, category id, shop id, brand id and pre-designed discretized statistical features, including price level, sales level and click level. In our task, the target is to predict whether the user would have click/purchase behavior for each item in currently viewed items. 15 days user click-through behaviors are sampled for training, and we evaluate the data of the next day. Related statistics are shown in Table. 1.

5.2 Compared Approaches

We compared our proposed TiSSA approach and ranking system against several baselines from different aspects.

libFM. The task in this paper can also employ traditional recommendation methods that capture the content and collaborative information. In our work, we use the popular libFM model [21] as a representative of these methods.

Wide and Deep network (WD) [4]. This method only employs the wide and deep features of current items to construct deep neural network without consideration of sequential information.

Time-GRU. The method in [30] is employed, whose LSTM structure are replaced as our proposed Time-GRU in this paper because of the adoption of Time-GRU in our framework.

Time-GRU + Attention. A vanilla attention is added on top of Time-GRU method.

EDRec [16]. EDRec employs a bi-directional RNNs as an encoder for modeling user sequential behaviors and a traditional vanilla-attention-based RNN as a decoder to predict the user's next action.

ATRank[2] [31] and **SHAN** [28]. Two very recent state-of-the-art work on modeling user behaviors, which only utilize fully connected layer with specially designed attention mechanism. Both of them employ the same downstream application network and loss function with TiSSA.

Variant TiSSA. To justify each component in our proposed TiSSA approach, we separate out each component in TiSSA systematically, including TiSSA without RNNs (TiSSA w/o RNNs), TiSSA without Time Slices (TiSSA w/o TiS), TiSSA without Inter-Slice Self-Attention (TiSSA w/o Inter-SSA), TiSSA without multi-head attention (TiSSA w/o Multi-head) and TiSSA with Fix-Scale Time Slices (TiSSA w. Fix-TiS).

5.3 Implementations and Metrics

Our method and all compared approaches use common raw features as inputs, whose hyper-parameters are tuned on the validation

[2]https://github.com/jinze1994/ATRank

Table 1: Statistics of offline dataset. PV denotes page view. Avg. Len. and Med. Len. indicate average and median length of user historical sequential behaviors, respectively.

Data	Users	Items	Categories	Shops	Brand	PV	Avg. Len.	Med. Len.
Amount	62.6 M	27.5 M	10.6 K	0.18 M	0.16 M	1.2 B	85.8	46

set using random search [2]. All approaches are implemented on the TensorFlow machine learning system, utilizing 100 parameter servers and 2000 workers, each of which runs with 15 CPU cores.

For libFM, we extract tuples $\langle user\_id, item\_id, label\rangle$, where the label is 1 if the user has an action on the item and 0 otherwise. Then the multi-hot vectors of ID-features and discretized statistical features are concatenated to be $\mathbf{x}$ and the label is $\mathbf{y}$ in libFM. For other methods based on neural network, we have following settings.

Feature Engineering. All methods based on neural network employ the same feature embedding network (c.f. Section 4). All embeddings are randomly initialized with mean value 0.0 and standard deviation 1.0. We set the embedded dimension size of item IDs and category IDs, shop IDs, brand IDs as 64, 32, 32, 32, respectively. All discretized statistical features are embedded into 16 dimensions, which include click level, price level and sales level. All of these embedding dimensions result from tuning on the validation set.

Network Shape. All hidden states sizes in the RNNs structure are set to be 64. For variant TiSSA except for TiSSA w. Fix-TiS, we set time slices as $[0, 1), [1, 2), [2, 4), ..., [2^7, +\infty)$ (#hour) based on the prior knowledge of the property of time decay, and set the amount of attention heads of inter-slice self-attention as 8 according to [25]. Time slices of TiSSA w. Fix-TiS are $[0, 2), [2, 4), [4, 6), ..., [14, +\infty)$ (#day) for comparison. Sizes of remaining fully connected layers of all approaches are set to be 128, which also results from tuning on the validation set.

Batch Size. The batch size of all methods is set to be 512. All the batches in the data set are trained for two epochs.

Optimizer. We use AdaGrad [6] as the optimizer for all approaches. The initial learning rate is 0.01 for variant TiSSA, and 0.1 for rest approaches. The l_2-loss weight of regularization is set to 10^{-6} for RNN-based methods, and 5×10^{-5} for rest methods.

Note that, items embedding can be pre-computed and stored, and each hidden state of GRU only depends on the last one. In the inference procedure of real-world applications, TiSSA can save historical hidden states and employ parallel computing processes similar to RNN-free models, e.g., ATRank, after computing one GRU cell. Specifically, in our practical online large e-commerce platform, for one user and a batch of 3K items, TiSSA runs less than 50ms to give ranking results.

To measure the performance of each model, we follow the practice in [31]. Specifically, Area Under ROC Curve (AUC) is employed. The larger the value of AUC, the better the performance is.

5.4 Experimental Results on Offline Dataset

Firstly, we report the overall performance. Table. 2 shows the AUC results for measuring the performance of different methods. According to overall AUC results, we can observe that TiSSA achieves significant improvement than all other methods, which demonstrates the rationality of our motivations and the effectiveness of TiSSA approach on modeling sequential user behaviors, on the macro level. Next, we make some comparisons and summarize our findings as follows.

Table 2: Comparison results on offline dataset. Overall AUC denotes AUC for the entire dataset, while Long-Term AUC indicates AUC for users with more than 150 historical behaviors and 3 active days. Bold typeset indicates the best performance. The improvements from baselines to TiSSA is statistically significant according to two-tailed t-test ($p < 0.05$).

Method	Overall AUC	Long-Term AUC
libFM	0.696	0.695
WD	0.697	0.697
Time-GRU	0.702	0.698
Time-GRU + Attention	0.709	0.714
EDRec	0.709	0.711
ATRank	0.724	0.721
SHAN	0.714	0.711
TiSSA w/o RNNs	0.732	0.726
TiSSA w/o TiS	0.729	0.733
TiSSA w/o Inter-SSA	0.735	0.732
TiSSA w/o Multi-head	0.740	0.739
TiSSA w. Fix-TiS	0.737	0.738
TiSSA	**0.742**	**0.744**

Baselines libFM and WD captures content and collaborative filtering features at the same time but fails to capture the sequential information of user behaviors, while other methods capture all these information. This fact explains why libFM and WD perform worse than other models.

Due to the limitation of Time-GRU in modeling complex dependency, Time-GRU performs worse than Time-GRU + Attention and TiSSA, which reveals the benefits of attention mechanism. TiSSA w/o RNNs exhibits the improvement of overall AUC by a remarkable margin comparing to ATRank and SHAN. This result owes to the advantage that our proposed time slices hierarchical self-attention mechanism cannot only capture local dependency but the long-range global dependency of user historical behaviors efficiently. TiSSA, ATRank and SHAN outperform EDRec by a significant improvement indicate that special designed attention-mechanism brings much more benefits.

Comparing TiSSA to TiSSA w/o RNNs, TiSSA achieves improvements as much as 1.4% for AUC. As the only difference between

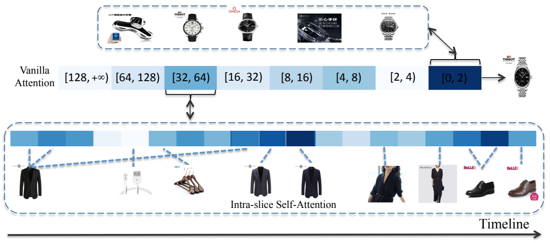

Figure 4: (Best view in color). Case study of making prediction on whether the user interacts with next item according to historical sequential behaviors. Deeper color in the heat map indicates higher attention weight.

them is whether taking session-level representations as inputs, this result validates the motivation of adoption of Time-GRU for transforming individual behaviors into session-level representations. In the comparison results of TiSSA w/o TiS, TiSSA w/o Inter-SSA and TiSSA, we demonstrate the effectiveness of the hierarchical self-attention designs. AUC declines when we remove multi-head attention in inter-slice attention mechanism. One explanation is that multi-head attention captures the relationship of inter-slice more detailed by projecting source token into multiple subspaces. Moreover, the result of TiSSA w. Fix-TiS is worse than TiSSA, which is consistent with previous studies on user behaviors on e-commerce websites [10], e.g., the property of time decay.

For the in-depth analysis of the problem of long- and short-term dependencies, we also report long-term AUC results in Table. 2, which relies heavily on the modeling of long- and short-term dependencies. The results show that TiSSA achieves the best performance against all other methods. We also observe that performances of methods without RNNs or attention mechanism drop a lot compared with overall AUC, which is reasonable for the advantage of RNNs + Attention framework at modeling long-term sequential data. Long-term AUC of TiSSA w/o Inter-SSA also drops due to the capacity of Inter-SSA for capturing long-range global dependency.

Table. 3 shows average attention scores for different time slices over all offline Dataset, which decrease with time generally except middle slices, e.g., [8, 16). It is because of the properties of user behaviors on e-commerce websites, such as time decay and cross-time dependency. Then we present a real case in Fig. 4. The middle heat map is attention scores for different time slices with target item (a mechanical watch), which is easy to understand the reason that [0, 2) slice has the highest attention score (the top line of Fig. 4). It is interesting to find that [32, 64) slice has the second high attention score. With in-depth analysis, we can find that the user has lots of interactions with formal wear at [32, 64) slice (the bottom line of Fig. 4), which is highly related to the target item and captured by our context-aware representations produced by TiSSA. Intra-slice self-attentions of [32, 64) slice are also shown in Fig. 4.

5.5 Online A/B Test on Tmall

Online test with real-world e-commerce users is carried out to study the effectiveness of our proposed method. In particular, we integrate TiSSA into the search engine of Tmall. A standard A/B test is conducted online [11]. Users of Tmall are randomly divided into multiple buckets, and we randomly select two buckets for experiments. For users in A bucket, we use the existing highly optimized ranking solution of Tmall search engine, which ensembles results from many powerful models. For users in B bucket, we further integrate the results produced by TiSSA. Specifically, for a given user, his/her context-aware representations are produced by TiSSA in real time according to collected sequential behaviors. All recall items of the search engine are preformed downstream application network in this paper and produced an interacting probability to attend existing online item ranking strategy, i.e., LTR. The test was performed within 12 days, and comparative results are given in Table. 4. It can be observed that our proposed TiSSA achieves 1.56% and 2.09% relative improvement for uCTR and uCVR than the baseline, respectively. We perform the statistical test and the resulting p-values for uCTR and uCVR are 9.1×10^{-12} and 0.0414, respectively, both less than 0.05. Thus, we think the improvement is significant. Such improvements are significant for Tmall search engine systems and have significant business value. Furthermore, the key metric to optimize in Tmall, i.e. GMV (Gross Merchandise Volume), has an improvement of 3.66%. Considering the traffic of Tmall, it would result in a significant boost in revenue. TiSSA has been adopted into the search engine of Tmall.

6 CONCLUSION

In this paper, we propose a time slice hierarchical self-attention approach to generate user context-aware representations for modeling sequential user behaviors called TiSSA, which captures both local and long-range global dependency of user actions. We also propose a Time-GRU structure to produce session-level inputs for better robustness and a directional multi-head attention. The empirical evaluations on a huge dataset collected from the real world for prediction of user interaction task show that our proposed approach can be the new state-of-the-art solution for related tasks. The ablation analyses on the separate parts of our TiSSA demonstrate the importance of each proposed component in our model. Results of online A/B test on Tmall search engine further demonstrate its practical value.

7 ACKNOWLEDGEMENT

This work was partly supported by NSFC under No. 61772466 and U1836202, the Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars under No. LR19F020003, the Provincial Key Research and Development Program of Zhejiang, China under No. 2017C01055, and the Alibaba-ZJU Joint Research Institute of Frontier Technologies.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.
[2] James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research* (2012), 281–305.
[3] Shiyu Chang, Yang Zhang, Wei Han, Mo Yu, Xiaoxiao Guo, Wei Tan, Xiaodong Cui, Michael Witbrock, Mark Hasegawa-Johnson, and Thomas S. Huang. 2017. Dilated recurrent neural networks. In *NIPS*.
[4] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide and deep Learning for recommender systems. *arXiv:1606.07792* (2016).

Table 3: Average vanilla attention scores for different time slices over the entire offline dataset.

Time Slices (#Hour)	[0, 2)	[2, 4)	[4, 8)	[8, 16)	[16, 32)	[32, 64)	[64, 128)	[128, +∞)
Avg. Attention Score	0.2067	0.1242	0.1139	0.1446	0.1329	0.1182	0.0916	0.0678

Table 4: Results of online A/B test. The user Click Through Rate (uCTR), user Click Conversion Rate (uCVR) and Gross Merchandise Volume (GMV) are adopted for measuring. Improvement indicates a relative growth of Group B compared to Group A, e.g. $1.56\% \approx (44.8567 - 44.1667)/44.1667$**. The absolute values of GMV are omitted for business confidential.**

	uCTR	uCVR	GMV
Group A (baseline)	44.1667 ± 0.261%	8.8767 ± 0.22%	–
Group B (TiSSA)	44.8567 ± 0.234%	9.0622 ± 0.227%	–
Improvement	1.56%	2.09%	3.66%

[5] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *arXiv:1406.1078*.

[6] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12 (July 2011), 2121–2159.

[7] Carlos A. Gomez-Uribe and Neil Hunt. 2016. The netflix recommender system: algorithms, business value, and innovation. *ACM Transactions on Management Information Systems* 6, 13 (January 2016).

[8] BalÃązs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *arXiv:1706.03847*.

[9] BalÃązs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *RecSys*. ACM, Boston, Massachusetts, 241–248.

[10] Peng Jiang, Yadong Zhu, , Yi Zhang, and Quan Yuan. 2015. Life-stage Prediction for Product Recommendation in E-commerce. In *KDD*. ACM, Sydney, NSW, Australia, 1879–1888.

[11] Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M. Henne. 2009. Controlled experiments on the web: survey and practical guide. *Data Mining and Knowledge Discovery* 18, 1 (February 2009), 140–181.

[12] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *ICLR*.

[13] Chenxi Liu, Junhua Mao, Fei Sha, and Alan Yuille. 2017. Attention correctness in neural image captioning. In *AAAI Conference on Artificial Intelligence*. AAAI, 4176–4182.

[14] Qiang Liu, Shu Wu, and Liang Wang. 2017. Multi-behavioral sequential prediction with recurrent log-bilinear model. *IEEE Transactions on Knowledge and Data Engineering* 29, 6 (June 2017), 1254–1267.

[15] Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. 2016. Learning natural language inference using bidirectional lstm model and inner-attention. In *arXiv:1605.09090*.

[16] Pablo Loyola, Chen Liu, and Yu Hirate. 2017. Modeling user session and intent with an attention-based encoder-Decoder architecture. *RecSys* (2017), 147–151.

[17] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2017. Hierarchical question-image co-attention for visual question answering. In *NIPS*.

[18] Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. 2016. Phased LSTM: accelerating recurrent network training for long or event-based sequences. In *NIPS*.

[19] Wenjie Pei, Jie Yang, Zhu Sun, Jie Zhang, Alessandro Bozzon, and David M. J. Tax. 2017. Interacting attention-gated recurrent networks for recommendation. In *CIKM*. ACM, Singapore, Singapore, 1459–1468.

[20] Massimo Quadrana, Alexandros Karatzoglou, BalÃązs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *RecSys*. ACM, Como, Italy.

[21] Steffen Rendle. 2012. Factorization machines with libfm. *TIST* 3 (2012).

[22] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hananneh Hajishirzi. 2017. Bi-directional attention flow for machine comprehension. In *ICLR*. Toulon, France.

[23] Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. DiSAN: directional self-Attention network for RNN/CNN-free language understanding. In *AAAI Conference on Artificial Intelligence*. AAAI.

[24] Tao shen, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. 2018. Bi-directional block self-attention for fast and memory-efficient sequence modeling. *arXiv:1804.00857* (2018).

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

[26] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. 2017. Recurrent recommender networks. In *WSDM*. ACM, Cambridge, United Kingdom, 495–503.

[27] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *NAACL*.

[28] Haochao Ying, Fuzhen Zhuang, Fuzheng Zhang, Yanchi Liu, Guandong Xu, Xing Xie, Hui Xiong, and Jian Wu. 2018. Sequential recommender system based on hierarchical attention network. In *IJCAI*.

[29] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A dynamic recurrent model fornext basket recommendation. In *SIGIR*. ACM, Pisa, Italy, 729–732.

[30] Yuyu Zhang, Hanjun Dai, Chang Xu, Jun Feng, Taifeng Wang, Jiang Bian, Bin Wang, and Tie-Yan Liu. 2014. Sequential click prediction for sponsored search with recurrent neural networks. In *AAAI Conference on Artificial Intelligence*. AAAI.

[31] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2018. ATRank: an attention-based user behavior modeling framework for recommendation. In *AAAI Conference on Artificial Intelligence*. AAAI.

[32] Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2017. What to do next: modeling user behaviors by time-lSTM. In *IJCAI*. 3602–3608.