ABSTRACT

The massively available data about user engagement with online information service systems provides a gold mine about users’ latent intents. It calls for quantitative user behavior modeling. In this paper, we study the problem by looking into users’ sequential interactive behaviors. Inspired by the concepts of episodic memory and semantic memory in cognitive psychology, which describe how users’ behaviors are differently influenced by past experience, we propose a Long- and Short-term Hawkes Process model. It models the short-term dependency between users’ actions within a period of time via a multi-dimensional Hawkes process and the long-term dependency between actions across different periods of time via a one dimensional Hawkes process. Experiments on two real-world user activity log datasets (one from an e-commerce website and one from a MOOC website) demonstrate the effectiveness of our model in capturing the temporal dependency between actions in a sequence of user behaviors. It directly leads to improved accuracy in predicting the type and the time of the next action. Interestingly, the inferred dependency between actions in a sequence sheds light on the underlying user intent behind direct observations and provides insights for downstream applications.

CCS CONCEPTS
- Mathematics of computing → Time series analysis; Stochastic processes; - Information systems → Task models;

KEYWORDS
Sequential data, interactive behaviors, Hawkes process

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM Reference Format:

1 INTRODUCTION

User behavior modeling is essential for understanding users’ diverse preferences and intents, which in turn provide valuable insights for online service systems to adaptively maximize their service utility in a per-user basis. A rich body of research has been developed on this topic [1, 3, 16, 25, 35]. For example, Anderson et al. [3] studied students’ learning activity patterns recorded in a Massive Online Open Courses (MOOCs) platform and developed a badge-based incentive system to improve student engagements in MOOCs. Yu et al. [35] investigated user content access patterns in a large online video-on-demand system, which provide insights on better resource allocation and performance optimization in such systems.

Most existing efforts focus on measurement studies of online user interactive behaviors, such as extracting features or implicit feedback from user activity logs for supervised model training; however, modeling the underlying dynamics that govern the generation of observed user behaviors is lacking. At a micro level, prior studies show that time intervals between users’ sequential actions carry a great deal of information about their underlying intents [7, 17, 30]. At a macro level, it has been independently observed in several different application scenarios that a series of user actions burst in a short period, referred as sessions [27] or tasks [13, 19, 31], and a sequence of a user’s interactive behaviors is usually carried out over several such periods. More importantly, quantitative analysis suggests correlation of user behaviors both within and across those short periods [12], and such correlation enables prediction of users’ future behaviors [2]. This clearly suggests that a user’s sequential interactive behaviors are not a set of independent actions, but there are internal dependency and structure that reflect and characterize his/her underlying preference and intent.

Mainstream approaches for modeling sequential user behaviors focus on fixed- or varying-order Markov models [5] or hidden Markov models [26], which capture transitional patterns between consecutive user actions in unit time steps. Semi-Markov models are used to model continuous time-intervals between actions [11]. However, it is very expensive to use Markov models to capture long-term dependency between actions, since the overall state-space
leads to significantly improved accuracy in predicting a user’s future action. A by-product of LSHP is the ability in “explaining” an observed user behavior sequence: it decomposes the generation of a current action as a combination of mutual-influence and self-influence from past actions and a spontaneous impulse caused by this action’s marginal popularity. This helps us better understand users’ underlying intent behind the observed behavior sequence and provides useful input for downstream applications, such as item recommendation and online advertising.

2 RELATED WORK

Users’ interactive behavior recorded in online information service systems is a gold mine to understand users’ underlying intents and preferences. Considerable amount of effort has been made on this direction [16, 29, 34], while most focuses on extracting task-specific features to improve a particular application. For example, Lo et al. [16] extracted a set of behavior features based on users’ interactive behaviors in Pinterest, such as search, click and bookmark a page, to classify time-variant user purchasing intent (e.g., when will a user make the purchase). Such feature engineering effort helps specific end tasks, but it can hardly unveil the underlying dynamics of observed user behaviors. It is thus of limited generality.

Statistical models with Markov assumptions have been proposed for sequential user behavior modeling [26, 32]. In a MOOC environment, Shi et al. [26] combined non-parametric Bayesian with hidden Markov models to cluster students through modeling their sequential learning activities. But due to the exponential growth of Markov state space with respect to the order of modeled dependence, these Markov models can hardly capture any long-term dependence between actions in a sequence. Another type of method is based on recurrent neural networks (RNN). The success of RNN in sequential data modeling inspired researchers to apply it to sequential behavior modeling [10, 15, 36]. Hidasi et al. [16] incorporated rank loss functions into RNN for session-based recommendations in user click sequence. Li et al. [15] applied attention based RNN to session-based online shopping recommendations, where a user’s shopping intent in a session is emphasized when predicting the next action. However, such solutions do not differentiate the temporal dependency between actions, e.g., actions of the same type versus those of different types; and since they only focus on in-session short-term dependencies, it is non-trivial to extend them to full sequences for long-term dependencies modeling.

Hawkes process based models have been developed to model long-term dependencies in sequential user behaviors [8, 22, 33, 37]. However, in a standard Hawkes process, temporal dependency between actions are generally modeled as additive time decay from previous actions to current ones, which cannot distinguish influence from previous actions happening together in a short period of time versus those taking place in a long period of time. Motivated by the concepts of episodic memory and semantic memory in cognitive psychology, we separate these two types of temporal dependencies into two Hawkes processes: one focuses on mutual influence across different types of actions in a close temporal proximity, and the other focuses on self-influence within the same type of actions in a longer period of time. This provides the model with both flexibility and constraint in modeling users’ sequential interactive behaviors.
3 METHODOLOGY

In this section, we first introduce the notations and problem setup studied in this paper. Then we discuss some basics of Hawkes process. Based on these, we describe in detail our solution, Long- and Short-term Hawkes Process (LSHP), which differentiates short-term and long-term influence among sequential user actions by separating mutual-influence between actions of different types in a close temporal proximity from self-influence between actions of the same type in a longer temporal distance.

3.1 Notations and Problem Setup

To separate the long-term influence from short-term influence, we appeal to the notion of session [4] to segment an input action sequence. Our proposed solution can be easily adopted to different definitions of session, e.g., time-based sessions [17]. Formally, we denote an action as a tuple \( e_t = (v_i, t_i) \), where \( i \) is the index of this action in a sequence, \( v_i \) is its type and \( t_i \) is the timestamp of this action. Different actions may belong to the same type. We define a session \( S_k \) as a series of \( M_k \) actions observed in a chronological order from a particular user, \( S_k = \{ e_1^k, \ldots, e_{M_k}^k \} \), where \( k \) is the index of this session in the user’s behavior sequence. A sequence composed of \( K \) sessions and \( N \) actions can be represented as \( Q = \{ S_1, \ldots, S_K \} = \{ e_1^1, \ldots, e_{N}^1, \ldots, e_1^K, \ldots, e_{N}^K \} \). We assume there are in total \( V \) distinct types of actions in a corpus of \( C \) sequences.

Based on these notations, we formally define the problem of modeling users’ sequential interactive behaviors as learning a mapping from an unknown probability space to a measurable space of actions (e.g., counting based), such that the observed behavior sequence reaches the highest likelihood in this measurable space. The key in learning this probabilistic mapping is to properly specify the possible dependencies among the actions in the input sequences such that the structure of ground-truth mapping is reflected.

3.2 Hawkes Process

Before discussing details of our proposed model, we briefly introduce Hawkes process [9]. Hawkes process is a type of temporal point process, modeling sequences of timestamped actions, which assumes historical actions would influence generating intensity of future actions over a period of time [14]. It has been widely applied to modeling sequences of events over time, such as earthquake aftershocks [20]. In a Hawkes process, the conditional intensity function is used to depict generating rate of current action given historical actions and to capture the influence of historical actions on current one. For example, in one dimensional Hawkes process that models the generation of sequences with a single type of actions (e.g., type \( v \)), the conditional intensity function at time \( t \) is defined as,

\[
\lambda_v^*(t) = \mu_v + \int_0^t \alpha_v \kappa(t-s) dN_v(s),
\]

where \( \mu_v \) is the base intensity, representing the instantaneous generation rate of the action type \( v \). The kernel function \( \alpha_v \kappa(t-s) \) describes the influence of past actions \( N_v(s) \) of type \( v \) on the current action at time \( t \) in this sequence. This reflects the "self-exciting" property of Hawkes process. The parameter \( \alpha \) represents the strength of self-excitation, and \( \kappa(t-s) \) characterizes the time decay effect. Exponential kernel or power-law kernel is typically chosen to describe this time decay effect. And because of time decay, actions occurring temporally closer have stronger influence on each other than those temporally further away.

While one dimensional Hawkes process only considers the influence from previous actions of the same type, multi-dimensional Hawkes process can capture the dependence of different action types, where the conditional intensity function of type \( v \) at time \( t \) is,

\[
\lambda_v^*(t) = \mu_v + \sum_{v' = 1}^V \int_0^t A_{v'v} \kappa(t-s) dN_{v'}(s),
\]

where the parameter \( A_{v'v} \) represents the influence of type \( v' \) on type \( v \), and \( v' \) denotes the type associating with previous action occurring at time \( s \).

3.3 Long- and Short-Term Hawkes Process

Motivated by the cognitive psychology concepts of semantic memory and episodic memory, which describe how the past experience or knowledge influences people’s present or future behaviors, we model users’ sequential interactive behaviors as a mixture of two different stochastic processes. Specifically, we consider the actions happening in the same session of the current action as its context. As studies in cognitive psychology suggest that episodic memory would gradually lose its sensitivity in context over time, we assume the context actions only generate their influence within the same session. Hence, we adopt a multi-dimension Hawkes process to realize the concept of episodic memory as the mutual influence of past actions to this current action in this session. This forms the first stochastic process in LSHP. On the other hand, Ryan et al. [23] suggested that the semantic memory is memory recall, independent of context. We regard the action repetition of the same type in a sequence as a result of semantic memory. We employ a one dimensional Hawkes process to realize the semantic memory as self-influence driven action generation. Because semantic memory is derived from accumulated episodic memory, we assume this one dimensional Hawkes process is only influenced by actions of the same type from preceding sessions to the current action. This forms the second stochastic process in LSHP.

Using the language of Hawkes process, we incorporate these two stochastic processes into one process: for the \( i \)-th action of type \( v_i \) appearing at time \( t_i \) in an action sequence \( Q \), the conditional intensity specified by LSHP is as Eq. (1) shows, where \( S_k \) represents the session containing action \( e_k \) and \( \delta(\cdot) \) is an indicator function,

\[
\lambda_{v_i}^*(t_i) = \mu_{v_i} + \sum_{t_i < t_j} A_{v_i v_j} \kappa_{M}(t_i - t_j) \delta(S_j = S_i) \tag{1}
\]

\[
+ \sum_{t_j < t_i} B_{v_j} \kappa_{S}(t_i - t_j) \delta(S_j \neq S_i) \delta(v_j = v_i)
\]

There are three key components in LSHP. First, we introduce the base intensity \( \mu_{v_i} \) for each action type \( v_i \) with \( \mu_{v_i} > 0 \), to capture the instantaneous generation of different action types. For example, if a particular type of action occurs at the beginning of a new session and it is its first appearance in this sequence, we will accredit this occurrence to its base intensity, since the user’s episodic memory has not formed (as it is the first action in this session) and his/her
We treat time decay coefficients \( A \) to apply LSHP, we need to estimate its model parameters, i.e.,

These two types of dependency are integrated into one stochastic process to account for the heterogeneity of users’ sequential interactive behaviors, without increasing the model complexity. To solve the optimization problem defined in Eq. (4) by introducing auxiliary variable \( Z \) and dual variable \( U \), where \( \rho > 0 \) is a hyper-parameter. We solve the problem by iteratively updating \( \mu, A, B, Z, U \) with respect to the steps described below.

\[
F(\mu, A, B, Z, U) = \min_{\mu \geq 0, A \geq 0, B \geq 0, Z, U} -L(\mu, A, B) + \eta_A |\eta_A| -1
\]

\[
+ \rho Tr(U^T(A - Z)) + \frac{\rho}{2} |A - Z|^2
\]

**Step 1: Update \( \mu, A, B \).** The terms in Eq. (4) that are relevant to the update of \( \mu, A, B \), include,

\[
F(\mu, A, B) = \min_{\mu \geq 0, A \geq 0, B \geq 0} -L(\mu, A, B) + \rho Tr(U^T(A - Z)) + \frac{\rho}{2} |A - Z|^2
\]

To solve the optimization problem defined in \( F(\mu, A, B) \) by introducing a set of branching parameters \( p_{ii}, p_{il} \) and \( p_{li} \). One advantage of using majorization-minimization to minimize the upper bound of this objective function is that we can obtain closed form solutions for \( \mu, A, B \) independently; and in the meanwhile, the non-negativity constraints are satisfied automatically. Replacing Eq. (2) and Eq. (1) into \( F(\mu, A, B) \), we obtain,

\[
F(\mu, A, B) =
\]

\[
\min_{\mu \geq 0, A \geq 0, B \geq 0} - \sum_{c=1}^{C} \sum_{i=1}^{N} \log \left( \mu_{ci} + \sum_{j \neq i} A_{vij} \kappa_m(t_i - t_j) \delta(S_i = S_j) \right) + \sum_{v \neq i} B_{vii} \kappa_s(t_i - t_j) \delta(S_i \neq S_j) \delta(v = v_j)
\]

\[
\times C \int_0^T \lambda^2_v(t) dt + \rho Tr(U^T(A - Z)) + \frac{\rho}{2} |A - Z|^2
\]

\[
\leq - \sum_{c=1}^{C} \sum_{i=1}^{N} \left( p_{ii} \log \frac{\mu_{ci}}{p_{ii}} + \sum_{j \neq i} p_{ii} \delta(S_i = S_j) \log \frac{A_{vij} \kappa_m(t_i - t_j)}{p_{ij}} \right) + \sum_{v \neq i} p_{ii} \delta(S_i \neq S_j) \delta(v = v_i) \log \frac{B_{vii} \kappa_s(t_i - t_j)}{p_{ij}}
\]

\[
+ C \int_0^T \lambda^2_v(t) dt + \frac{\rho}{2} |A - Z + U|^2
\]

The branching parameter \( p_{ii} = \frac{\mu_{ii}}{\lambda^2_v(t_i)} \) can be considered as the probability that the \( i \)-th action is generated from the base intensity. And the branching parameter \( p_{ji} = \frac{A_{vij} \kappa_m(t_i - t_j) \delta(S_i = S_j) \delta(v = v_j) \lambda^2_v(t_j)}{\lambda^2_v(t_i)} \) indicates the probability that the \( j \)-th action within this session leads to the \( i \)-th action. Likewise, the branching parameter \( p_{ij} = \frac{B_{vii} \kappa_s(t_i - t_j) \delta(S_i \neq S_j) \delta(v_j = v_i) \lambda^2_v(t_i)}{\lambda^2_v(t_j)} \) represents the probability that the \( j \)-th action in previous sessions leads to the \( i \)-th action in current session.
Setting the gradients of these parameters to zero, we obtain the updating rule of $\mu, A, B$ as follows:

$$\mu_v = \frac{\sum_{c=1}^{C} \sum_{i=1}^{N} \rho_{i} \delta(v_i = v)}{\sum_{c=1}^{C} M_{k}}$$  \hspace{0.5cm} (5)$$

$$A_{uv'} = \frac{1}{2\rho} \left( X + \sqrt{X^2 - 4\rho Y} \right)$$  \hspace{0.5cm} (6)$$

$$X = \rho(U_{uv'} - Z_{uv'}) + \sum_{c=1}^{C} \sum_{k=1}^{M_{k}} \int_{t_i}^{t_{M_{k}}} \kappa_{m}(t - t_i) dt$$

The updating rules of $A$ suggest that the value $A_{uv'}$, corresponding to the mutual influence between action type $v$ and $v'$, correlates with both the frequency of action type $v$ and $v'$ co-occurring in the same session, and the time interval between actions of these two types. The shorter the time they are to each other, the stronger mutual influence they would have on each other. Besides, the updating rules for $\mu$ and $B$ suggest that not only the frequency of an action type in a sequence but also the action’s relative temporal duration in a sequence affect these intensities.

**Step 2: Update $Z$.** With the updated parameters $\mu, A, B$, we update $Z$ through solving the following optimization problem,

$$Z = \arg \min_{Z} \eta_{A}||Z||_{1} + \rho T r(U^{T}(A - Z)) + \frac{2}{2}(||A - Z||^{2})$$

The updating rule depending on the magnitude of $A + U$ is

$$Z_{uv'} = \begin{cases} (A_{uv'} + U_{uv'}) - \frac{\eta_{A}}{\rho}, & A_{uv'} + U_{uv'} \geq \frac{\eta_{A}}{\rho} \\ (A_{uv'} + U_{uv'}) + \frac{\eta_{A}}{\rho}, & A_{uv'} + U_{uv'} \leq \frac{\eta_{A}}{\rho} \\ 0, & |A_{uv'} + U_{uv'}| < \frac{\eta_{A}}{\rho} \end{cases}$$

As the equation suggests, the auxiliary variable $Z$ is introduced to handle the L1 regularizer on the mutual influence matrix $A$.

**Step 3: Update $U$.** Given the updated parameters $\mu, A, B$ and auxiliary variable $Z$, we update the dual variable

$$U_{new} = U_{old} + (A_{new} - Z_{new})$$

where $A_{new}, Z_{new}$ represent updated mutual-influence matrix and auxiliary variable respectively.

### 4 EXPERIMENTS

In this section, we evaluate the proposed model for sequential user behavior modeling. First, we describe the evaluation datasets and preprocessing steps. Then we provide qualitative analysis of our proposed model LSHP in identifying the underlying dynamics of user behaviors. Lastly, we compare LSHP with other baselines in the tasks of predicting the type and time of users’ future actions.

We collected two evaluation datasets, one contains users’ online browsing activities in a major e-commerce website in the U.S. and another contains students’ video watching behaviors in a MOOC course. In the e-commerce dataset, we collected five months user browsing logs under the cellphone category. We randomly selected a subset of users who have made at least one purchase in this five-month period. Each user is associated with a sequence of product page browsing activities. And each action consists of the browsed product ID and click timestamp. We filtered out sequences with fewer than 5 actions and the products which appear fewer than 5 times in this collection. In the MOOC dataset, we collected students’ video watching activities from an edX course, i.e., "Statistical Learning" Winter 2015. Each action contains the name of the video and watching timestamp. We filtered out sequences which have fewer than 5 actions, and removed videos which appeared fewer than 5 times in total in this dataset. As sessions are typically defined by a 30-minute threshold of inactivity [4, 12], we adopt this strategy to segment sequences into sessions. Based on the segmented sessions, we linearly scaled the recorded timestamps by 30 minutes for numerical purposes. For both datasets, we also filter out sequences which have fewer than 2 sessions. The statistics of the processed datasets1 are shown in Table 1.

#### 4.1 Qualitative Analysis

We first perform several qualitative evaluations to study the dependency structure identified by LSHP.

4.1.1 Dependence among different types of actions. To illustrate the dependence among action types captured by LSHP, we visualize its learned mutual influence matrix $A$. The mutual influence among seven selected products from the e-commerce dataset is shown in Figure 1, and it depicts the influence from products listed on the vertical axis to those on the horizontal axis. From the figure, we have the following observations: (1) The mutual influence matrix is asymmetric. We could see that product 4320 has strong influence on product 992, but the opposite direction does not exist. (2) The mutual influence matrix is sparse. We could observe that some products have no influence on others: for example, product 992 has no influence on product 751. (3) Strong influence exists between similar products, e.g., products only differ in color or with similar

---

1Due to business concerns, several fields of e-commerce dataset are filled in N/A.

### Table 1: Statistics of two evaluation datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Sequences per sequence</th>
<th>#Clicks per sequence</th>
<th>#Sessions per sequence</th>
<th>#Items</th>
<th>#Clicks per session</th>
<th>#Duration per sequence (30 minutes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>e-commerce</td>
<td>7540</td>
<td>N/A</td>
<td>6.58+/-6.72</td>
<td>4602</td>
<td>N/A</td>
<td>1992.69+/-1115.78</td>
</tr>
<tr>
<td>MOOC</td>
<td>4382</td>
<td>52.97+/-54.93</td>
<td>14.69+/-12.98</td>
<td>71</td>
<td>3.60+/-3.42</td>
<td></td>
</tr>
</tbody>
</table>

---

N/A: Not Available.
specifications under the same brand. We should note that such features are not provided to LSHP for mutual influence learning, but the model identifies the pattern from the logged users’ interactive behaviors. This indirectly supports LSHP’s ability in recognizing the temporal dependency among user actions, and also suggests such domain specific features can be introduced to further improve LSHP’s modeling quality.

4.1.2 Dependence among actions in the same sequence. To illustrate the dependence of an action on previous actions in the same sequence captured by LSHP, we select a sequence of product browsing actions from the e-commerce dataset as an example. This sequence contains 11 actions over three sessions. For each action, we compute its conditional intensity according to Eq. (1) and use a vertical bar to visualize the ratio of base intensity and influence from previous actions normalized by the conditional intensity of the action. In Figure 2, the height of each segment in a bar denotes the ratio. The observations are: (1) Actions at the beginning of a session are usually introduced by either base intensity or self-influence from actions of the same type in previous sessions. This reflects the people’s semantic memory over time. (2) At the later stage of a session, most actions are generated by the mutual influence between actions in the same session, like the last few actions in session 1 and 2 in Figure 2. This reflects people’s episodic memory. More importantly, such a decomposition of temporal influence can serve as a form of explanation of users’ underlying intent of their sequential behaviors. For example, for the last action in session 2, we can understand its appearance is caused by all previous actions from actions of the same type in previous sessions. This reflects people’s semantic memory over time. (2) At the later stage of a session, most actions are generated by the mutual influence between actions in the same session, like the last few actions in session 1 and 2 in Figure 2. This reflects people’s episodic memory. More importantly, such a decomposition of temporal influence can serve as a form of explanation of users’ underlying intent of their sequential behaviors. For example, for the last action in session 2, we can understand its appearance is caused by all previous actions from actions of the same type in previous sessions. This reflects people’s semantic memory over time. (2) At the later stage of a session, most actions are generated by the mutual influence between actions in the same session, like the last few actions in session 1 and 2 in Figure 2. This reflects people’s episodic memory. More importantly, such a decomposition of temporal influence can serve as a form of explanation of users’ underlying intent of their sequential behaviors. For example, for the last action in session 2, we can understand its appearance is caused by all previous actions from actions of the same type in previous sessions. This reflects people’s semantic memory over time.
our model with a set of baselines in predicting the type of future actions. The task is defined as follows: given the first \( n \) actions in a sequence \( Q = \{ e_1, \ldots, e_n \} \), we are interested in predicting what type of action the user will take next. After making a prediction of action \( e_{n+1} \), we include the ground-truth action \( e_{n+1} \) into the sequence and move onto the prediction of next action, until the end of this input sequence. On the MOOC dataset, we compare models in predicting which video the student would watch next. On the e-commerce dataset, we compare models in predicting which cell-phone the user would browse next. Presumably, a model that better identifies the dependence of next action on previous actions can provide more accurate prediction of the user’s next action.

4.2.1 Hyper-parameter tuning. Before comparing LSHP with any baselines, we first investigate how LSHP’s performance is affected by the time decay hyper-parameters \( \beta_m \) and \( \beta_s \), which are manually specified in LSHP. We calculate the conditional intensity of all possible action types according to Eq. (1) and rank them in a descending order. We measure MAP and P@1 of LSHP in predicting the type of next action with different settings of these two hyper-parameters. For both e-commerce dataset and MOOC dataset, we use 80% of sequences for training, and the rest for testing. When evaluating LSHP on testing sequences, we assume each sequence in testing dataset has the first \( y \) portion of actions given to initiate the prediction, and we evaluate MAP and P@1 on the rest of actions.

Setting \( y = 80\% \), the MAP and P@1 of LSHP with different settings of \( \beta_m \) and \( \beta_s \) on the e-commerce dataset are reported in Figure 3. We have performed the same analysis on the MOOC dataset and very similar findings were obtained on it. But due to the space limit, we only report the analysis on the e-commerce dataset here. From Figure 3 (a) and (b), we observe that when \( \beta_s = 2.0 \), MAP and P@1 improve with an increasing \( \beta_m \) till \( \beta_m = 3.0 \), and they gradually become worse with a further increasing \( \beta_m \). When \( \beta_m \) is small, such as 0.1, the kernel function \( \exp(-\beta_m \Delta t) \) cannot sufficiently reduce the mutual influence among actions within the same session and thus actions in the session have similar influence on the next action regardless of their different temporal proximity to it. The resulting poor MAP and P@1 suggest that it is important to account for the temporal information when evaluating the influence of previous actions within the session towards the next action. When \( \beta_s \) is large, like 10.0, the kernel function almost turns off mutual-influence of previous actions within a session which occur further away to the next action. The poor MAP and P@1 suggest that the occurrence of the next action does depend on actions occurring long before the current one. To summarize, the settings of \( \beta_m \) suggest that the type of the next action not only depends on actions temporally close within a session, but also on previous actions temporally away within the session. What is more, the dependence should be weighted by their temporal proximity to differentiate their impact.

As Figure 3 (c) and (d) show, when \( \beta_m = 3.0 \), MAP and P@1 improve with an increasing \( \beta_s \) till \( \beta_s = 0.01 \), and gradually MAP and P@1 become worse with a further increasing \( \beta_s \). This suggests that the type of the next action depends on the historical actions of the same type across sessions, and the dependence is influenced by the temporal distance. We can also notice that because the time intervals between two actions within a session are much shorter than those between two actions across sessions, the optimal value of \( \beta_m \) is much larger than \( \beta_s \).

4.2.2 Baselines for comparison. We compare LSHP with the following baselines in predicting the type of the next action.

- **Global Popularity (globalPop).** Rank action types according to their frequency in the training dataset in a descending order.
- **Sequence Popularity (seqPop).** Rank action types according to the frequency of action types in the target sequence. The frequency of action types is updated as more observations in the sequence become available. We use global popularity to break the tie.
- **First Order Markov Model (FOM).** We estimate the first order transition probability between action types on the training dataset, and rank action types according to the transition probability with respect to the last observed action.
- **Standard Multi-dimension Hawkes Process (standardHP).** Following [18, 33], the intensity function is defined as \( \lambda_{v_i}(t_i) = \mu v_i + \sum_{t_t < t_t} A_{v_i t} \kappa(t_t - t_i) \), which assumes the next action depends on all preceding actions in a sequence.
- **Sparse Hawkes Process (sparseHP).** This is an extension of standard Multi-dimension Hawkes Process. The matrix \( A \) capturing mutual influence is constrained to be sparse by imposing a \( L1 \) regularization over \( A \).
- **Session-based Hawkes Process (sessionHP).** To verify whether considering actions across session is beneficial to predict the type of the next action, we developed this variant of LSHP as a baseline. The intensity function is defined as \( \lambda_{v_i}(t_i) = \mu v_i + \sum_{t_t < t_t} A_{v_i t} \kappa(t_t - t_i) \delta(S_t = S_t) \), where the indicator function \( \delta(S_t = S_t) \) includes only influence of actions belonging to the same session.
- **Recurrent Temporal Point Process (RTPP).** Du et al. [7] propose a recurrent marked temporal point process to model both the type and time of an action, where LSTM is used as the recurrent layer. On two datasets, the embedding dimension of action type is
Figure 4: MAP and P@1 for models with different ratios of given actions in a testing sequence on the e-commerce dataset.

Figure 5: MAP and P@1 for models with different ratios of given actions in a testing sequence on the MOOC dataset.

For all methods, we utilize 80% of sequences for model training and the rest for testing. And also a portion of actions in a testing sequence is provided to the algorithms to initiate subsequent predictions. To reduce potential bias introduced by the number of actions used to initiate the prediction, and to study how different models perform with only a few actions available for initialization, we compared these models with different portions of actions provided initially in a testing sequence. For LSHP, standardHP, sparseHP and sessionHP, we rank action types according to their estimated intensities in a descending order.

MAP and P@1 over the two datasets are reported in Figure 4 and Figure 5. Without modeling self-influence across sessions, sessionHP cannot capture the influence of previous actions of the same type outside its current session, which means the long-term dependence is missing, and consequently it performs worse than LSHP. Although seqPop makes use of dependence between the next action and previous actions of the same type across sessions, ignoring the dependence between actions of different types within a session leads to its worse performance. StandardHP and sparseHP do not differentiate temporal dependence within nor across sessions, and use a universal time decay to model the temporal influence, which leads to their less accurate modeling of dependency, and thus worse performance in predicting the type of the next action. FOT only considers the influence of the last action and ignores the influence of any other preceding actions, so that it only achieves limited prediction accuracy. As globalPop does not consider the dependence of the next action on previous actions, its MAP and P@1 are much worse than all other algorithms and we do not include them in the results. RNN-based models including RTPP and NARM do not differentiate the dependence between actions of
the same type from those of different types effectively. RTPP does not explicitly separate actions within a session from those across sessions. NARM does not model either the temporal information or actions across sessions. In conclusion, because LSHP separates the temporal influence of actions of different types in the same session from those of the same type cross sessions on the next action, it can better capture both long-term and short-term dependence among actions and thus predict the type of the next action more accurately.

Among different ratios of actions used to initiate the prediction, LSHP outperforms the baselines in all settings. In addition, we could observe that when the number of actions used to initiate the prediction in a testing sequence is limited, i.e., around 10%, LSHP could still outperform all baselines.

4.3 Time Prediction

Modeling time as a random variable enables Hawkes process based solutions to predict the time of the next action. Presumably a model which can more accurately recognize the temporal dependence among actions in a sequence can better predict the arrival time of the next action. The task of predicting the time of next action is defined as: given n actions in a sequence Q = {e1, ..., en}, we are interested in predicting time tn+1 of the next action en+1. With the intensity function λ∗n+1(t), the probability of the next action occurring at time t is defined as:

\[ f^*(t) = \lambda_{n+1}^*(t) \exp \left( -\int_{t_n}^{t} \lambda_{n+1}^*(\tau) d\tau \right) \]

We estimate the time of the (n + 1)-th action via the expectation of its predicted time, as Eq. (9) shows. Since the integral in Eq. (8) does not have an analytic solution, we employ a numerical integration method Simpson’s rule [24] to approximate the expectation.

\[ t_{n+1} = \mathbb{E}[f(t)|t] \]

We follow the same setting as that for evaluating next action type prediction in Section 4.2. For Hawkes process models, we plug their conditional intensity functions into Eq. (8) and Eq. (9) to obtain the corresponding predicted time of the next action. As sessionHP only considers actions within a session, it cannot predict timestamp of the next action at the beginning of a session. As a result, we decompose the comparison of time prediction into two parts: First, we compare models in predicting the time of the next action within a session. Second, we compare models excluding sessionHP in predicting the time of the next action at the beginning of a session.
the effectiveness of LSHP on modeling the temporal dependence among users’ sequential interactive behaviors.

Identifying the underlying dependence structure among sequential interactive behaviors is crucial in user modeling and understanding. In this work, we utilize LSHP to bridge the gap between the studies in cognitive psychology of user behaviors and the computational modeling of user behaviors. This opens several important future directions. First, external features can be incorporated into LSHP to improve its temporal dependence modeling. For example, the similarity between actions based on external taxonomy. Second, we currently assume all users share the same set of model parameters. It would be beneficial to relax this constraint and estimate individualized models for different (groups) of users. Third, we have assumed static base intensities among different action types. But in practice they might also change over time, reflecting the dynamics of their global popularity. Another layer of stochastic process can be introduced to model this level of temporal process.

6 ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful comments. This paper is based on work supported by the National Science Foundation under grant IIS-1718216 and IIS-1553568.

REFERENCES


