To Search or to Recommend: Predicting Open-App Motivation with Neural Hawkes Process

Zhongxiang Sun
Zihua Si
Xiao Zhang
Gaoling School of Artificial Intelligence
Renmin University of China
Beijing, China
{sunzhongxiang,zihua_si,zhangx89}@ruc.edu.cn

Xiaoxue Zang
Yang Song
Kuaishou Technology Co., Ltd.
Beijing, China
{zangxiaoxue,yangsong}@kuaishou.com

Hongteng Xu
Jun Xu*
Gaoling School of Artificial Intelligence
Renmin University of China
Beijing, China
{hongtengxu,junxu}@ruc.edu.cn

ABSTRACT

Incorporating Search and Recommendation (S&R) services within a singular application is prevalent in online platforms, leading to a new task termed *open-app motivation prediction*, which aims to predict whether users initiate the application with the specific intent of information searching, or to explore recommended content for entertainment. Studies have shown that predicting users' motivation to open an app can help to improve user engagement and enhance performance in various downstream tasks. However, accurately predicting open-app motivation is not trivial, as it is influenced by user-specific factors, search queries, clicked items, as well as their temporal occurrences. Furthermore, these activities occur sequentially and exhibit intricate temporal dependencies. Inspired by the success of the Neural Hawkes Process (NHP) in modeling temporal dependencies in sequences, this paper proposes a novel neural Hawkes process model to capture the temporal dependencies between historical user browsing and querying actions. The model, referred to as **N**eural **H**awkes **P**rocess-based **O**pen-**A**pp **M**otivation prediction model (NHP-OAM), employs a hierarchical transformer and a novel intensity function to encode multiple factors, and open-app motivation prediction layer to integrate time and user-specific information for predicting users' open-app motivations. To demonstrate the superiority of our NHP-OAM model and construct a benchmark for the Open-App Motivation Prediction task, we not only extend the public S&R dataset ZhihuRec but also construct a new real-world **O**pen-**A**pp **M**otivation **D**ataset (OAMD). Experiments on these two datasets validate NHP-OAM's superiority over baseline models. Further downstream application experiments demonstrate NHP-OAM's effectiveness in predicting users' Open-App Motivation, highlighting the immense application value of NHP-OAM.

*Corresponding author. Work partially done at Engineering Research Center of Next-Generation Intelligent Search and Recommendation, Ministry of Education.
Work done when Zhongxiang Sun and Zihua Si were interns at Kuaishou.

SIGIR '24, July 14–18, 2024, Washington, DC, USA.

ACM ISBN 979-8-4007-0431-4/24/07
https://doi.org/10.1145/3626772.3657732

CCS CONCEPTS

• **Information systems** → **Users and interactive retrieval**.

KEYWORDS

Open-App Motivation, Neural Hawkes Process, Behavior Modeling

ACM Reference Format:
Zhongxiang Sun, Zihua Si, Xiao Zhang, Xiaoxue Zang, Yang Song, Hongteng Xu, and Jun Xu. 2024. To Search or to Recommend: Predicting Open-App Motivation with Neural Hawkes Process. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '24), July 14–18, 2024, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3626772.3657732

1 INTRODUCTION

To combat information overload, recommender systems and search engines are integral in applications like online video and e-commerce platforms, allowing users to access information through active and passive modalities [29–32, 35]. Consequently, this leads to the potential for users to open these applications with varying motivations. *Within these applications, two primary open-app motivations for users are to embrace suggestions from the recommendation system or employ the search engine for information seeking.* Varied open-app motivations correspond to distinct needs and behavioral patterns. As illustrated in Figure 1, a user might initiate the application driven by a specific intent, such as searching or perusing recommended videos. Notably, these motivations evolve, influenced by historical interactions as well as the user's lifestyle patterns.

An accurate prediction of the open-app motivation has significant meaning in the following aspects:

- **Enhancing user engagement:** Open-App motivation is vital for optimizing user engagement. In an experiment on a video platform with billions of users, altering the search modules from inconspicuous to a prominent position led to a significant rise in user interaction and elevated the related value of open-app to search from 0.1% to 0.8% [1]. However, this change induced competition, with 87% of the gains from the search and 13% from a decrease in returns from recommended videos. Based on the prediction of the user's open-app motivation, we can dynamically adjust the search and recommendation modules of the Apps, thereby mitigating the competition between recommendation

[1]An increase in the proportion of users open-app to search significantly improves the retention of low-activity users and can significantly increase the total usage time for users on the App.

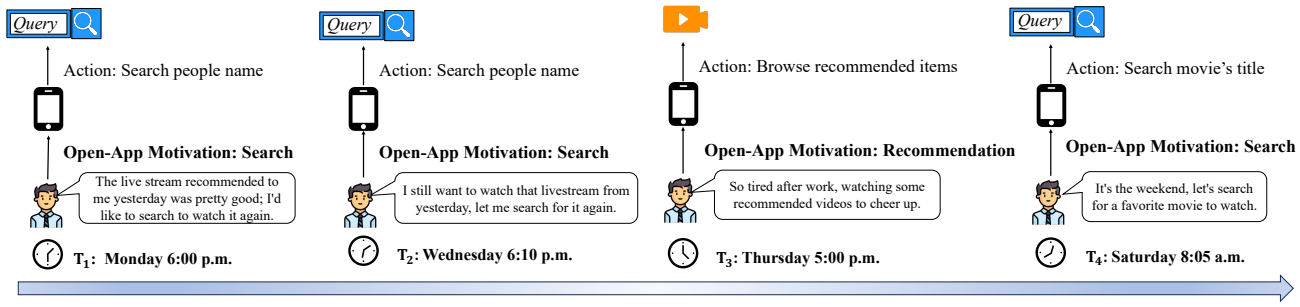


Figure 1: Examples of Open-App Motivation. Obtained through a structured questionnaire-based interview process with users.

and search to improve user engagement.

- **Application in downstream tasks:** Open-App motivation plays a critical role in various downstream applications, including (1) Based on the prediction of open-app motivation, we can obtain more precise guidance of user interest sources (search or recommendation history) in recommended tasks that need to arouse user interest in a very short time, e.g., App open advertisement and message pop-up recommendation [2] [21]. (2) Learn a better user representation, serving as a robust user characterization to enhance the recommendation system's effect.

This real-world problem relates to two research topics: User Retention Prediction [2, 7, 11, 23, 41] and User Intent Prediction [1, 17, 18, 24, 38]. User Retention Prediction forecasts if a returning user will engage with an App again. The prediction does not consider the fine-grained specifics of the open-app motivation, i.e., whether for searching or embracing recommended information. User Intent Prediction predicts what intents a user might have while utilizing an App, representing a higher-level understanding of user intents within the App's environment. Intent types are varied and hard to define explicitly. Different Apps may focus on different intents, such as purchasing in e-commerce Apps or entertainment in video Apps. We study why users open an App and acknowledge they may search or embrace recommendations for a particular intent. This leads to a new, widely applicable problem across various Apps.

Challenges in Open-App Motivation Prediction. Open-App motivation prediction faces significant challenges, which we further substantiate with behavioral patterns observed in real-world users' data (§ 3.3):

- **Time-User bias:** Open-App motivation varies based on time and individual users. Our data analysis reveals the ratio of search to recommendations exhibits an apparent **Periodicity** within each 24-hour day, as well as between workdays and rest days of the week. Combined with the case study about individual users shown in Figure 1, this validates the challenge of time-user bias.
- **Multiple factors:** The motivation is influenced by various factors, e.g., search queries and clicked recommended items in history logs. Our findings (§ 3.3) indicate two key patterns under this challenge. First, a phenomenon of **Repeat-Query** behavior suggests that users may open the App multiple times to search for the same query. Second, we observe a positive **Relevance** between the ratio of searches to clicked recommended items in a user's previous session and the motivation to search when opening the App the next time, further highlighting the multifaceted influences affecting Open-App motivation.

To address the challenges mentioned above, combining our analysis of users' open-app motivation's behavioral patterns, we propose a novel Neural Hawkes Process [27, 42, 47] (NHP)-based model, called NHP-OAM, to predict users' open-app motivations (§ 4). Considering that open-app motivation exhibits Periodicity and Repeat-Query features and the fact that the time intervals at which users open the App are not fixed, the neural Hawkes process can effectively capture the clear periodicity and repeated query features found in past open-app motivations without needing to assume consistent sampling time intervals like in time series modeling [25]. Unlike previous NHP models, considering that open-app motivation is influenced by multiple factors, we utilize a hierarchical transformer model as the history encoder which can not only learn the past open motivation sequence but also learn the session-level and history-level contextual representations of S&R behaviors. In the design of the intensity function for NHP-OAM, to capture the Relevance feature, we propose an intensity function that is aware of both the S&R behaviors' ratio and representations learned from the hierarchical transformer model. Finally, considering the temporal factor and users' different habits that significantly impact open-app motivation, we employ a time gate to fuse temporal information with the hierarchical user history representation and specific user embedding to predict users' open-app motivation.

The contributions of this paper are summarized as follows:

- We pioneer the study of open-app motivation prediction, which is a critical problem in real-world Apps that integrate S&R services but has not been well explored.
- We propose the NHP-OAM that effectively leverages the behavioral patterns underlying open-app motivation to address associated challenges. Utilizing the properties of NHP, we capture Periodicity and Repeat-Query features, aiding in resolving the Time-User bias challenge. A hierarchical transformer model is employed to capture the influence of Multiple factors on open-app motivation. Additionally, we introduce an intensity function aware of the Relevance features.
- We construct and open-source the first real-world **O**pen-**A**pp **M**otivation **D**ataset (OAMD), which contains users' S&R behaviors, as well as the motivation for each session when opening

[2]The introduction of App open advertisement and message pop-up recommendation: https://en.wikipedia.org/wiki/Pop-up_ad.

the App. To further validate the effectiveness of our model, we also extend the public S&R datasets-ZhihuRec [3]. Experiments on these two datasets validate NHP-OAM's superiority over baseline models. Further downstream Application experiments demonstrate NHP-OAM's effectiveness in predicting users' open-app motivation, underlining its application value.

2 RELATED WORK

User Retention Prediction. User Retention Prediction aims to forecast the likelihood of a user returning to an app for further interactions. Various techniques have been developed to tackle this issue. Xu et al. [38] utilized a hybrid approach for user retention in a short video platform. Li et al. [24] aligns with [38] in their use of ensemble methods, but specifically focused on multichannel time series for a long video Internet company. Diverging from ensemble methods, Cai et al. [1] leveraged reinforcement learning for long-term user retention in Kuaishou [1]. In contrast, Kim et al. [18] as well as Kim and Lee [17] extended the focus to offline retail, using deep survival analysis and Wi-Fi fingerprinting, respectively. In contrast to User Retention Prediction, our work on Open-App Motivation Prediction aims to identify the specific motivation why a user opens an app, adding a fine-grained study compared to existing studies.

User Intent Prediction. User Intent Prediction has been extensively studied in various domains. He et al. [11] focused on using a concept knowledge graph to characterize user intent at Alipay explicitly. In contrast, Li et al. [23] designed AutoIntent to discover implicit consumption intents from user data at Meituan. Zhang et al. [41] introduced Atten-Mixer, which leverages multi-level user intent for session-based recommendation. Chang et al. [2] proposed a probabilistic approach using variational autoencoders to infer latent user intent in sequential recommenders. Fan et al. [7] utilized a metapath-guided graph neural network to recommend user intent in mobile e-commerce platforms like Taobao. Our work shifts focus from traditional User Intent Prediction to Open-App Motivation Prediction. Instead of forecasting multiple in-app intents, we aim to pinpoint why a user initially opens an app for search or recommendation. This provides a fine-grained insight into user behavior, which is essential for integrating S&R apps.

3 EMPIRICAL STUDY

In this section, we first give the task definition of open-app motivation prediction. Then we make several observations about open-app motivation on the real-world dataset (OAMD), which serves as the foundation of our model.

3.1 Task Definition

Let $\mathcal{U}$ be a set of users and $\mathcal{I}$ be a set of items. We consider user queries by introducing an additional query set Q and word vocabulary $\mathcal{W}$, where a query $q \in Q$ consists of a list of words $\left[w_1, \ldots, w_{|q|}\right], w \in \mathcal{W}$. The interactions of user $u \in \mathcal{U}$ sorted chronologically is organized as a sequence of N sessions [4] $\mathcal{S} = \langle \{\mathcal{S}_1, m_1, t_1\}, \{\mathcal{S}_2, m_2, t_2\}, \cdots, \{\mathcal{S}_N, m_N, t_N\} \rangle$, where t_n is the time when the user u opens the App at session $\mathcal{S}_n$, and m_n is the open-app motivation for session $\mathcal{S}_n$, where

$$m_n = \begin{cases} 1, & \text{open-app motivation is search} \\ 0, & \text{open-app motivation is recommendation} \end{cases}. \quad (1)$$

Each session $\mathcal{S}_n = \left\{s_{n,1}, s_{n,2}, \cdots, s_{n,|\mathcal{S}_n|}\right\}$, where $|\mathcal{S}_n|$ is the length of this query-aware heterogeneous sequence. s_i can be an item interaction or a query action. Let δ indicates whether s_i at i-th step is a query or an item interaction:

$$s_i \in \begin{cases} \mathcal{I}, & \text{if } \delta(s_i) = 0 \\ Q, & \text{otherwise.} \end{cases} \quad (2)$$

Let $t(\mathcal{S}_n) = \left\{t_{n,1}, t_{n,2}, \cdots, t_{n,|\mathcal{S}_n|}\right\}$ represent the times of interactions or queries in session $\mathcal{S}_n$.

Goal. Given a user $u \in \mathcal{U}$ and her/his historical sessions with open-app motivations $\mathcal{S}$, we want to predict the open-app motivation m_{N+1} that u will most probably have in the next session $\mathcal{S}_{N+1}$ at time t_{N+1}.

3.2 Dataset Description

Open-App motivation prediction is a novel task, and there isn't a publicly available dataset for it. To bridge this gap, we collected search and recommendation behaviors as well as the motivation for each session when users open the App on a real video platform used by over one billion users [5], called OAMD. Search and recommendation behaviors are directly obtained from the user behavior logs. As for the open-app motivation label, we obtain it from the internal data center of the video platform. *This label is defined by whether the user actively searches within 30 seconds of opening the App. If true, the open-app motivation is "search"; otherwise, it is "recommendation".* This classification method is rooted in the platform's comprehensive data analytics and business strategy. Considering commercial confidentiality and privacy issues, we are unable to display the specific analysis data and content. To further elucidate the rationale behind this label, let's delve into two distinct perspectives: **Qualitative Perspective:** Typically, users with a deliberate intent to search will begin their search soon after App activation. Given potential latency due to factors like network or device performance, a window of 30 seconds has been determined as a reasonable threshold to gauge users' motivation. **Quantitative Analysis:** Data analysis in the dataset shows that in sessions where users actively searched within the first 30 seconds, the ratio of searched clicked items to recommended clicked items stands at *0.4082*. This is notably higher than the *0.0474* ratio in sessions without an active search in that window. This stark difference underscores the reliability of our 30-second criterion as an indicator of a user's open-app motivation.

More detailed statistics of OAMD are summarized in § 5.1.1.

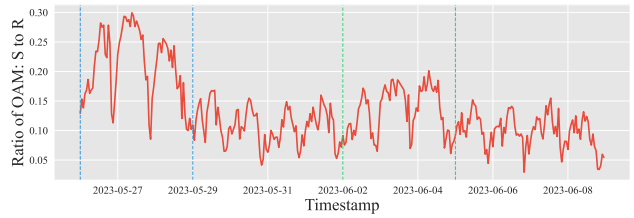
3.3 Behavioral Patterns Behind Users' Open-App Motivation

In this section, we study the behavioral patterns behind users' open-app motivation in the real-world dataset OAMD. We aim to gain insights that show what a model should focus on when predicting

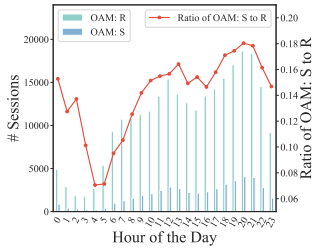
[3] We will release these datasets after acceptance.

[4] A session is defined as all the user's interactions from when they open the App to when they close it.

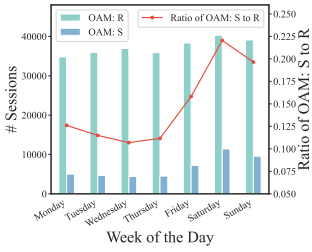
[5] We select users from who searched in the month prior to our data collection.



(a) Overall Trend: areas between the two dashed lines are weekends.



(b) Hourly Trend.



(c) Weekly Trend.

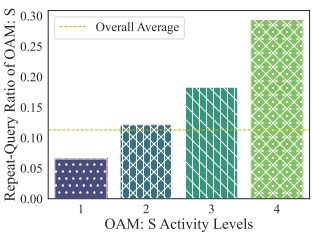
Figure 2: Periodicity statistics in Open-App Motivation. OAM: S denotes the Open-App Motivation: Search. OAM: R denotes the Open-App Motivation: Recommendation.

users' next open-app motivation.

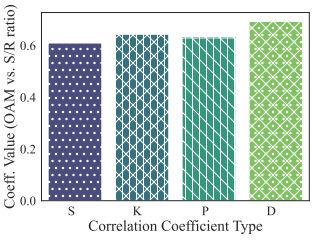
3.3.1 Periodicity in Open-App Motivation. We study how the ratio of search to recommendation in open-app motivations varies over time. Specifically, we analyze the hourly and daily trends in the ratio of searches to recommendations. Figure 2a shows that users' open-app motivations have daily and weekly patterns in the search-to-recommendation ratio. Figure 2c reveals higher open-app to search on weekends (Friday nights, Saturdays, and Sundays) than on weekdays. Figure 2b shows users prefer to open apps for searches during midday and evenings. The trends for recommendations in users' open-app motivations are the exact opposite of these patterns. This demonstrates the importance of considering the temporal dependencies in open-app motivation, which needs to be emphasized in our model predicting users' open-app motivation.

3.3.2 Repeat-Query in Open-App Motivation. The Repeat-Query in the open-app motivation task is defined as the phenomenon where users open the app due to the same search query. We first categorize users into four bins of equal quantity based on their "open-app to search" activity level. Then, within different user bins, we calculated the average proportion for each user of all their "open-app motivations to search" due to searching the same query. As shown in Figure 3a, the higher the activity level, the higher the repeat-query ratio, and the overall average repeat-query ratio per user is also a comparatively high proportion. This finding highlights the need to use models that can understand a user's past actions to better predict their open-app motivations.

3.3.3 Relevance in Open-App Motivation. Considering the potential relevance between the user's behavior within a session and the motivation to open the app, we investigate how the ratio of searches to clicks on recommended items influences a user's motivation



(a) Repeat-Query



(b) Relevance

Figure 3: Open-App Motivation Features. (a) Repeat-query statistics in open-app motivation to search. Users are divided into four activity groups by Open-App Motivate: Search (OAM: S) activity levels (high values refer to high activity levels). The overall average denotes the repeat-query ratio in all the users. (b) Relevance statistics between the ratio of clicked search list to clicked recommended list in users' past sessions and their motivation to search when they open the APP next time.

to next open the app. We employ four correlation coefficients—Spearman (S), Kendall (K), Pearson (P), and Distance (D)—for a comprehensive analysis. Distance Correlation ranges within $[0, 1]$, while the others are in $[-1, 1]$. A higher value signifies a stronger correlation for all metrics. Figure 3b reveals a positive correlation across all metrics between past search-to-click ratios and future open-app motivation. Thus, the relevance feature can serve as a direct feature for determining a user's next open-app motivation.

4 OUR APPROACH: NHP-OAM

Based on the empirical observations above, we propose a Deep Hawkes Process-based Open-App Motivation (NHP-OAM) prediction model. Some preliminaries about the Neural Hawkes Process are introduced first. Then we detailedly describe model definition and parameter learning.

4.1 Preliminaries about Neural Hawkes Process

Formally, a *Hawkes Process* [10] is a temporal point process where one event can boost the likelihood of future events. Extending this, the *Neural Hawkes Process* incorporates neural networks, offering a more flexible way to model temporal dependencies of events, thus capturing more complex patterns in data. The realization of a neural Hawkes process consists of a list of discrete events localized in time, which is denoted as $\mathcal{S}$ in the context of open-app motivation.

Given the history time of past events $\mathcal{S}$, the neural Hawkes process begins by employing a history encoder—such as a Transformer [42, 47] or LSTM [27]—to encode these past events into a history representation. This representation is then used in a conditional intensity function (CIF) $\lambda\left(t \mid \mathcal{T}_t\right)$ that provides a stochastic model for the time of the next event considering all times of previous events, where

$$\mathcal{T}_t = \langle \{\mathcal{S}_n, m_n, t_n\} : t_n < t \rangle \tag{3}$$

is the history up to time t. The CIF captures the impact of previous events and allows for intricate modeling of dependencies through

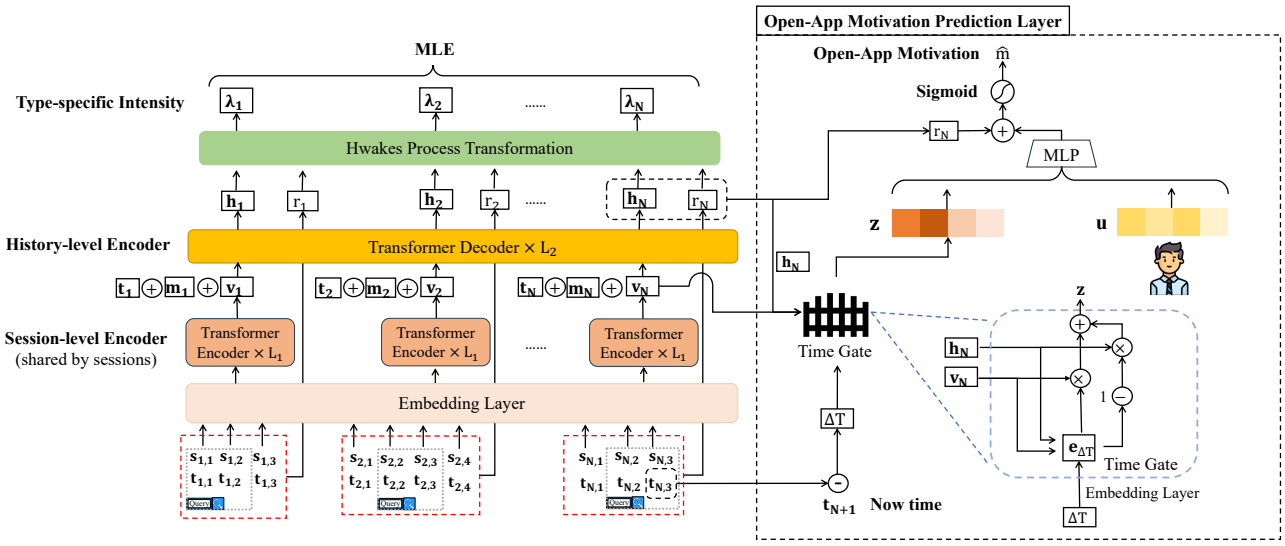


Figure 4: The overall architecture of the proposed model NHP-OAM, featuring Session-level Encoder and History-level Encoder in § 4.2, Type-specific Intensity Function in § 4.3 and Open-App Motivation Prediction Layer in § 4.4.

the history representation. Previous works have modeled the CIF using an exp-decay function with softplus [27, 42] or a linear function with softplus [47]. In the context of open-app motivation, the probability for the occurrence of a new open-app motivation given the history time $\mathcal{T}_t$ within a small time window $[t, t+dt)$ is given by the conditional intensity function:

$$\lambda_m(t \mid \mathcal{T}_t)\, dt = \mathbb{P}\{\text{event } m \text{ in } [t, t+dt) \mid \mathcal{T}_t\}.$$

The conditional intensity function for the entire open-app motivation sequence is defined by:

$$\lambda(t \mid \mathcal{T}_t) = \sum_{m=0}^{1} \lambda_m(t \mid \mathcal{T}_t).$$

Given the CIF, the probability density function (PDF) of the type-m open-app motivation is:

$$p_m(t \mid \mathcal{T}_t) = \lambda_m(t \mid \mathcal{T}_t) \exp\left(-\int_{t_j}^{t} \lambda_m(\tau \mid \mathcal{T}_\tau)\, d\tau\right). \tag{4}$$

4.2 NHP-OAM: History Encoder

Previous neural Hawkes models used historical events and time data to predict future events [27, 42, 47]. However, in our task, the motivation to open an App is influenced by both the queries searched and the items clicked on in the historical logs. To capture the rich contextual information within a user's session, as shown in Figure 4, we designed a hierarchical transformer model as NHP-OAM's history encoder. First, using the transformer encoder module, the user's query and click interaction behaviors within the session are encoded into a session-level representation. Subsequently, the past session's session-level information is combined with the motivation to open the App and the time information. This is fed into the transformer decoder module of the hierarchical transformer model, ensuring that future data is not leaked, resulting in history-level representations that can be used to compute the intensity function. Next, we'll detail the hierarchical transformer model.

4.2.1 Embedding Layer. We use embeddings $\mathrm{M}_U \in \mathbb{R}^{|\mathcal{U}|\times d}, \mathrm{M}_I \in \mathbb{R}^{|\mathcal{I}|\times d}, \mathrm{M}_W \in \mathbb{R}^{|W|\times d}, \mathrm{M}_M \in \mathbb{R}^{2\times d}, \mathrm{B} \in \mathbb{R}^{2\times d}$ to represent d-sized users, items, query words, open-app motivation type, interaction type, respectively. We represent the inputs: (1) User: Given user u, we look up corresponding embeddings M_U. (2) Item: Given item i, we look up corresponding embeddings M_I. (3) Open-App motivation: Given item m, we look up corresponding embeddings M_M. (4) Query: For query $q = [w_1, \ldots, w_{|q|}]$, we retrieve corresponding word embeddings and adopt an average pooling operation to get our query representation. (5) Interaction type: We look up $B \in \mathbb{R}^{2\times d}$ to get embeddings for different interaction types (i.e., item or query). (6) Time: For the timestamp t, we apply a positional encoding method following [47]:

$$[e_t]_i = \begin{cases} \cos\left(t/10000^{\frac{i-1}{d}}\right) & \text{if } i \text{ is odd} \\ \sin\left(t/10000^{\frac{i-1}{d}}\right) & \text{if } i \text{ is even} \end{cases}, \tag{5}$$

where e_t denotes the temporal embedding and i is the step of the interaction corresponding to t.

4.2.2 Session-level Encoder. Given a user $u \in \mathcal{U}$ and one of her/his heterogeneous historical session $\mathcal{S}_n$, the task of the session-level encoder is to generate the session embeddings $\mathbf{v}_n$ representing u's intention in the current session, using transformer encoder layers. Next, we will delve into the computation of $\mathbf{v}_n$.

For the items or queries $\{s_{n,1}, s_{n,2}, \cdots, s_{n,|\mathcal{S}_n|}\}$ of session $\mathcal{S}_n$, we assemble their embeddings as follows: if $\delta(s_{n,i})$ equals to 0, we use *Embedding Layer (2)*; otherwise, we use *Embedding Layer (4)*. The assembled embeddings form an embedding matrix $\mathbf{E}_n^s = [\mathbf{e}_{n,1}, \mathbf{e}_{n,2}, \cdots, \mathbf{e}_{n,|\mathcal{S}_n|}]^\top \in \mathbb{R}^{|\mathcal{S}_n|\times d}$. Similarly, we get the time embeddings $\mathbf{E}_n^t$ according to *Embedding Layer (6)* and interaction type embeddings $\mathbf{E}_n^b$ according to *Embedding Layer (5)*. Combining these

three embeddings, we get the session embedding matrix:

$$\mathbf{E}_n = \mathbf{E}_n^s + \mathbf{E}_n^t + \mathbf{E}_n^b,$$

where $\mathbf{E}_n \in \mathbb{R}^{|\mathcal{S}_n| \times d}$ and + denotes element-wise addition. Then, We build L_1 Transformer Encoder [36] blocks as the Session-level Encoder to learn the session-level representation $\mathbf{v}_n$:

$$\mathbf{v}_n = \text{MEAN}\left(\hat{\mathbf{E}}_n\right); \hat{\mathbf{E}}_n = \text{Encoder}(\mathbf{E}_n), \tag{6}$$

where $\mathbf{v}_n \in \mathbb{R}^d$, $\hat{\mathbf{E}}_n \in \mathbb{R}^{|\mathcal{S}_n| \times d}$ and MEAN() denotes mean pooling. The last session embedding, $\mathbf{v}_N$, is utilized as the present short-term intent embedding, as it reflects a user's immediate preference.

4.2.3 History-level Encoder. Given a user $u \in \mathcal{U}$ and all her/his heterogeneous historical sessions $\mathcal{S}$, the task of the history-level encoder is to generate the history representation $\mathbf{H}$ for the intensity function and long-term embedding for the open-app motivation prediction by using transformer decoder layers. Using the *Session-level Encoder*, *Embedding Layer (3)* and *Embedding Layer (6)* to encode $\mathcal{S}_n$, m_n, t_n, respectively, we can get the embedding of $\mathcal{S}$:

$$\mathbf{E}_{\mathcal{S}} = \left[\mathbf{v}_1 + \mathbf{e}_{m,1} + \mathbf{e}_{t,1}, \mathbf{v}_2 + \mathbf{e}_{m,2} + \mathbf{e}_{t,2}, \ldots, \mathbf{v}_N + \mathbf{e}_{m,N} + \mathbf{e}_{t,N}\right]^\top,$$

where $\mathbf{E}_{\mathcal{S}} \in \mathbb{R}^{N \times d}$ is the embedding matrix of all the user's past open-app motivation sequence. The embedding matrix $\mathbf{E}_{\mathcal{S}}$ is then fed through L_2 Transformer Decoder [36] blocks as the History-level Encoder, generating time-aware hidden representations $\mathbf{H}$ of the input open-app motivation sequence:

$$\mathbf{H} = [\mathbf{h}(t_1), \mathbf{h}(t_2), \cdots, \mathbf{h}(t_N)]^\top = \text{Decoder}(\mathbf{E}_{\mathcal{S}}), \tag{7}$$

where $\mathbf{H} \in \mathbb{R}^{N \times d}$ is composed of the hidden representations of the input sequence and d is the dimension of each representation. Each $\mathbf{h}(t_n)$ corresponds to the hidden representation of the n-th input at time t_n in the open-app motivation sequence for user u, which contains all historical information from the beginning to the n-th input. The matrix $\mathbf{H}$ is utilized to compute the intensity function. Its last position, $\mathbf{h}(t_N)$, includes all historical actions of the current user and can be used to represent the long-term user intent.

4.3 NHP-OAM: Intensity Function

Since the intensity function of Hawkes processes is history-dependent, we use $\mathbf{H}$ learned by the History Encoder which contains rich historical behavior information to compute the type-specific intensity function. Building upon previous explorations of the intensity function [26, 47], and taking into account time efficiency as well as the complexity of user behaviors, we adopted a linear interpolation time module to ensure the continuous of the conditional intensity function. Additionally, we utilized a type-specific nonlinear transformation that transforms the hidden states $\mathbf{h}(t_n)$ into a scalar, enhancing the model's capability to capture intricate behavior patterns. Furthermore, considering the relevance feature of the open-app motivation in § 3.3, we add the query (recommendation) ratio aware score $r_{t,m}$. This score captures the proportion of user search or interaction with the recommendation at the current moment t in the current session. Given that the search ratio is typically low and might introduce noise, we performed batch normalization on $r_{t,m}$. This normalization step ensures a consistent scale for the ratio, reducing the influence of outliers and allowing for more stable learning. Finally, we express the intensity function as follows:

$$\lambda_m\left(t \mid \mathcal{T}_t\right) = f\left(\alpha_m \frac{t - t_n}{t_n} + \phi\left(\mathbf{w}^\top \mathbf{h}\left(t_n\right)\right) + r_{t,m}\right), \tag{8}$$

where f is the softplus function to constrain the intensity function to be positive, α_m is the hyperparameter which modulates the importance of the interpolation, ϕ is the activation function which can be GELU [13] or Tanh [16], and $\mathbf{w} \in \mathbb{R}^{d \times 1}$ are learnable parameters. For brevity, we omitted batch normalization in Equation 8.

4.4 NHP-OAM: Prediction Layer

In this section, we introduce the open-app motivation prediction layer to predict the user's next open-app motivation. Drawing inspiration from [47], we found that adding an additional prediction layer on top of the neural Hawkes processes model yields better performance. The prediction layer can integrate user historical information, which contains the short-term user intent $\mathbf{v}_N$ and long-term user intent $\mathbf{h}(t_N)$, the time information t_{N+1} as well as user-specific information u.

Specifically, we use a time-gate approach to integrate $\mathbf{v}_N$ and $\mathbf{h}(t_N)$. Inspired by the relative position encoding methods [3, 5, 33], we use the difference between the current time t_{N+1} and the user's last interaction time $t_{N,|\mathcal{S}_N|}$ to obtain the relative time Δt:

$$\Delta t = t_{N+1} - t_{N,|\mathcal{S}_N|}.$$

Then, we use the *Embedding Layer (6)* to get the time encoding $\mathbf{e}_{\Delta t} \in \mathbb{R}^d$. The gating vector can be computed by:

$$\mathbf{g} = \text{Sigmoid}\left(\mathbf{W}_l \mathbf{h}(t_N) + \mathbf{W}_s \mathbf{v}_N + \mathbf{e}_{\Delta t}\right),$$

where $\mathbf{W}_l$ and $\mathbf{W}_s \in \mathbb{R}^{d \times d}$ are the learnable weight matrices. The integrated history embedding $\mathbf{z}$ is calculated by:

$$\mathbf{z} = \mathbf{g} \otimes \mathbf{v}_N + (1 - \mathbf{g}) \otimes \mathbf{h}(t_N), \tag{9}$$

where $\otimes$ represents element-wise product.

To introduce user-specific information, we use *Embedding Layer (1)* to encode the user ID u, obtaining $\mathbf{u}$, which along with $\mathbf{z}$ is fed into the widely adopted two-layer MLP [45, 46] to model feature interaction. Considering that the sigmoid function is monotonically increasing, we directly add r_N to make use of the relevance feature in open-app motivation:

$$\hat{m}_{N+1} = \text{Sigmoid}\left(\text{MLP}\left(\mathbf{z} \| \mathbf{u}\right) + r_N\right), \tag{10}$$

where $\|$ denotes the concatenation operation and $\hat{m}_{N+1}$ denotes the prediction score of u's next open-app motivation.

4.5 Training

In this section, we describe how our model is trained. For a given user u, in accordance with the existing methods for neural Hawkes process training [26, 27, 47], we maximize the log-likelihood across all the user's history of open-app motivations:

$$\mathcal{L}_\lambda^u = \frac{1}{N} \sum_{n=1}^{N} \log \lambda_{m_n}\left(t_n \mid \mathcal{T}_{t_n}\right) - \int_{t_1}^{t_N} \lambda\left(t \mid \mathcal{T}_t\right) dt \tag{11}$$

where the first term of equation (11) represents the open-app motivation that happened at the times they happened and the second term is an integral of the total intensities over the observation interval $[t_1, t_N]$, which can be approximated using Monte Carlo

Table 1: Statistics of OAMD and ZhihuRec. # denotes the number of per user; R-C denotes recommended items that were clicked; Q denotes query actions; Q-C denotes clicked items from the search results (ZhihuRec misses this feature); OAM denotes Open-App Motivation.

Information	OAMD	ZhihuRec
Users	25,355	798, 086
# R-C, Q, Q-C	285.73, 17.51, 22.64	33.81, 4.89, —
# Sessions	20.31	5.27
Time span (days)	16	10
OAM: search	66,546	715,792
OAM: recommendation	340,842	3,314,396

integration [28].

For the training of the open-app motivation prediction Layer, we utilize the Binary Cross Entropy loss for optimization:

$$\mathcal{L}_m^u = -\frac{1}{N}\sum_{n=1}^{N}(m_n \log(\hat{m}_n) + (1-m_n)\log(1-\hat{m}_n)). \tag{12}$$

Finally, for all users in $\mathcal{U}$, the loss $\mathcal{L}$ is computed as:

$$\mathcal{L} = \sum_{u\in\mathcal{U}}(-\alpha\mathcal{L}_\lambda^u + \mathcal{L}_m^u), \tag{13}$$

where α are hyperparameters that control the importance of the two parts of loss.

5 EXPERIMENTS

In this section, we empirically verify the efficiency of NHP-OAM. The source code, OAMD and extended ZhihuRec datasets have been shared at https://github.com/Jeryi-Sun/NHP_OAM.

5.1 Experimental Settings

5.1.1 Datasets. NHP-OAM requires both user session-level S&R behavior logs and open-app motivations simultaneously. In the following experiments, we evaluated the models on the new real-world Open-App Motivation dataset (OAMD in § 3.2) and the extended public S&R dataset–ZhihuRec [9]. As OAMD has already been detailed in Section 3.2, this section mainly focuses on how to extend the public S&R datasets–ZhihuRec. Table 1 reports the statistics of both datasets.

To the best of our knowledge, there doesn't exist a public dataset that contains both explicitly session-level S&R behaviors and open-App motivation. We extend a public S&R dataset, the ZhihuRec dataset [9], by generating session-level information and open-app motivations. Specifically, consider that ZhihuRec was collected from a fixed day and includes a fixed number of user interactions, while our task requires a continuous history of user interactions. Therefore, we first preprocess ZhihuRec to make it suitable for our task requirements [6]. Subsequently, we separate a user's entire behavior sequence into sessions, using 30 minutes of inactivity as the interval [8, 39]. Finally, we get users' open-app motivations based on whether the user actively searches within 30 seconds of opening the App. If true, the open-app motivation is "search"; otherwise, it is "recommendation".

To more closely approximate real-world application scenarios, we adopt the temporal ordering time-ratio-based splitting strategy [44] to split both datasets. Specifically, for both datasets, to ensure that each user has sufficient history for building a user profile, we treat the log data from the first three days as the historical set. The last day serves as the test set, the second-to-last day as the validation set, and the remaining data as the training set.

5.1.2 Evaluation Metrics. In our specific use case, within Apps such as video platforms that incorporate both Search and Recommendation (S&R) services, the emphasis is often placed more heavily on the recommendation module. Failing to accurately predict a user's primary motivation for opening the app as *search* can detrimentally impact the user experience. As such, achieving high *precision* in prediction becomes paramount. Consequently, the chosen evaluation metrics include **Accuracy**, **Precision**, **F1-Score**, **$F_{0.5}$-Score**, and **AUC** [7].

5.1.3 Baseline Models. Considering the absence of pre-existing models tailored for open-app motivation prediction, we adapt five categories of baseline models to this task: Sequential Recommendation Models, Joint Search-Recommendation Sequential Models, Time-Aware Sequential Recommendation Models, Time-Series Models, and Neural Hawkes Process Models. Each category is modified to predict open motivations, mapping outputs to a [0, 1] interval:

Sequential Recommendation Models (SR): **GRU4Rec** [14] employs Gated Recurrent Units to encode users' historical sequences. In our task, we use the same architecture but modify the output layer to a linear layer mapping to [0, 1], with the training objective being open motivation prediction. **SASRec** [15] employs transformer decoders, adapted similarly. **BERT4Rec** [34] utilizes transformer encoders, adapted with mask token embedding.

Joint Search-Recommendation Sequential Models (SRS): **NRHUB** [37] combines search and recommendation histories with self-attention, adapted with MLP output. **Query-SeqRec** [12] merges histories using a transformer encoder. **USER** [39] operates at the session level, combining search and recommendation history into a single sequence and using a transformer encoder for encoding.

Time-Aware Sequential Recommendation Models (TSR): **TiSASRec** [22] extends SASRec with interaction timestamps. **RESETBERT4Rec** [43] enhances BERT4Rec with time information. **TLSRec** [3] is session-based, uses time lag gate, adapted with MLP.

Time-Series Models (TS): **LSTNet** [20] captures dependencies in time-series data with CNN and RNN layers. **Autoformer** [4] Uses series decomposition to obtain trend and seasonal information, relying on FFT and IFFT-based Auto-correlation. **DLinear** [40] decomposes time series, predicts with FFN.

Neural Hawkes Process Models (NHP): **CTNHP** [27] models events in continuous time with LSTM. **SAHP** [42] applies self-attention for event prediction. **THP** [47] employs self-attention in point-process-based RNN models.

[6] We remove the last day to ensure that users' data distribution is not concentrated on the final day. At the same time, if we don't remove the last day, a large number of users' behaviors would be concentrated in the test set according to our time-based validation and testing method, leading to severe distribution drift.

[7] Specifically, $F_{0.5}$-score is a variant of $F1$-score that assigns greater weight to *precision* over *recall* [6].

Table 2: Performance comparisons between NHP-OAM and the baselines. The boldface represents the best performance. '†' indicates that the improvements over all of the baselines are statistically significant (t-tests, p-value < 0.05).

Types	Models	OAMD					ZhihuRec				
		Accuracy	Precision	F1-Score	$F_{0.5}$-Score	AUC-Score	Accuracy	Precision	F1-Score	$F_{0.5}$-Score	AUC-Score
SR	SASRec	0.9193	0.8156	0.4941	0.6472	0.7162	0.7247	0.5431	0.5352	0.5399	0.6611
	BERT4Rec	0.9170	0.8310	0.4602	0.6285	0.7269	0.7551	0.6145	0.5491	0.5865	0.6826
	GRU4Rec	0.9228	0.8789	0.5046	0.6778	0.7181	0.7521	0.6061	0.5479	0.5814	0.6827
SRS	NRHUB	0.9055	0.7289	0.3384	0.4987	0.7336	0.7451	0.6023	0.5294	0.5709	0.6853
	Query-SeqRec	0.9166	0.7626	0.4783	0.6161	0.8012	0.7532	0.6120	0.5429	0.5823	0.7237
	USER	0.9103	0.7304	0.4136	0.5591	0.7986	0.7533	0.6132	0.5410	0.5821	0.7254
TSR	TiSASRec	0.9164	0.7387	0.4908	0.6145	0.8100	0.7501	0.6007	0.5468	0.5779	0.6919
	RESETBERT4Rec	0.9159	0.7429	0.4824	0.6109	0.8117	0.7552	0.6146	0.5491	0.5866	0.6937
	TLSRec	0.9107	0.7119	0.4334	0.5664	0.7982	0.7547	0.6131	**0.5494**	0.5859	0.6997
TS	Dlinear	0.9221	0.7743	0.5467	0.6638	0.8316	0.6182	0.3940	0.4419	0.4118	0.5871
	Autoformer	0.9120	0.6502	0.5322	0.5972	0.8171	0.6518	0.4439	0.5201	0.4715	0.6757
	LSTNet	0.9226	0.7973	0.5385	0.6688	0.8352	0.7116	0.5190	0.5338	0.5248	0.7003
NHP	THP	0.9393	0.9019	0.6669	0.8050	0.9063	0.7163	0.5273	0.5321	0.5292	0.7064
	CTNHP	0.9409	0.8883	0.6912	0.7974	0.9017	0.7161	0.5268	0.5336	0.5295	0.7085
	SAHP	0.9409	0.9014	0.6638	0.7885	0.9076	0.7153	0.5253	0.5343	0.5289	0.7064
Ours	NHP-OAM	**0.9463**†	**0.9320**†	**0.7100**†	**0.8284**†	**0.9326**†	**0.7601**†	**0.6326**†	0.5462	**0.5950**†	**0.7541**†

5.1.4 Implementation details. NHP-OAM's hyperparameters are tuned using grid search on the validation set with Adam [19]. The batch size is tuned among $\{64, 128, 256\}$. The learning rate η is tuned among $\{1e-5, 1e-4, 1e-3\}$. The embedding dimension d is set to 32. The number of history encoder layers L_1 and L_2 are both set to 2. The α is tuned among $\{0.1, 0.001, 0.0001, 0.00001, 0.000001\}$. For all models, the maximum session number is set to 10 on the OAMD dataset and 5 on the ZhihuRec dataset. The maximum sequence length of behaviours in a session is set to 20 on the OAMD dataset and 5 on the ZhihuRec dataset. Since the output of NHP-OAM falls within the range $(0, 1)$, we need to establish a threshold within $(0, 1)$ for classification purposes. In our open-app motivation prediction task, we opt for the $F_{0.5}$-score as our evaluation metric because we place a greater emphasis on precision while also considering recall. The threshold is determined based on achieving the optimal $F_{0.5}$-score on the validation set. The search range for the threshold starts from 0 and increases by increments of 0.01, up to and including 1. The optimal threshold is then selected from these candidates, which achieves the highest $F_{0.5}$-score on the validation set, to be applied to the test set for final classification results. For baselines, we tuned the parameters using the grid search around the optimal values in the original paper.

5.2 Overall Performance

Table 2 shows prediction performances of NHP-OAM versus baselines on OAMD and ZhihuRec. Based on the results presented in Table 2, we found: (1) NHP-OAM almost outperformed the baseline models in terms of five evaluation metrics on OAMD and ZhihuRec, with statistical significance (t-tests, p-value < 0.05). These results verified the efficiency of NHP-OAM in predicting users' open-app motivations. (2) NHP-OAM outperformed Time-Aware Sequence Recommendation Models and Sequence Recommendation Models, demonstrating that relying solely on history-level information was insufficient. It was also essential to model and fully utilize the behavioral patterns behind users' open-app motivations. At the same time, Time-Aware Sequence Recommendation Models and Sequence Recommendation Models outperformed Sequence Recommendation Models, indicating the importance of incorporating time information and unified modeling for search and recommendation in predicting open-app motivations. (3) NHP-OAM and Neural Hawkes Process Models outperformed Time-Series Models, proving that the neural Hawkes process effectively captured the features found in past open-app motivations without needing to assume consistent sampling time intervals like time-series models. (4) NHP-OAM performed better than the Neural Hawkes Process baselines, showing the effectiveness of our improvements in applying the Hawkes Process to open-app motivation prediction. (5) The results on ZhihuRec are generally lower than those on OAMD, mainly because ZhihuRec is adapted to perform this task.

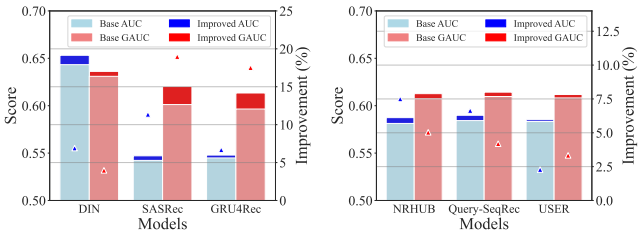
Table 3: Ablation Study of NHP-OAM on OAMD.

Models	Accuracy	Precision	F1-Score	$F_{0.5}$-Score	AUC-Score
w/o SLE	0.9436	0.9003	0.6858	0.8002	0.9144
w/o UE	0.9460	0.9123	0.7077	0.8177	0.9323
w/o TG	0.9448	0.8965	0.6960	0.8039	0.9307
w/o PL	0.9433	0.9056	0.6817	0.8005	0.9289
w/o QR	0.9457	0.9153	0.6975	0.8137	0.9302
w/o HP	0.9372	0.8672	0.6452	0.7623	0.9212
NHP-OAM	**0.9463**	**0.9320**	**0.7100**	**0.8284**	**0.9326**

5.3 Ablation Study

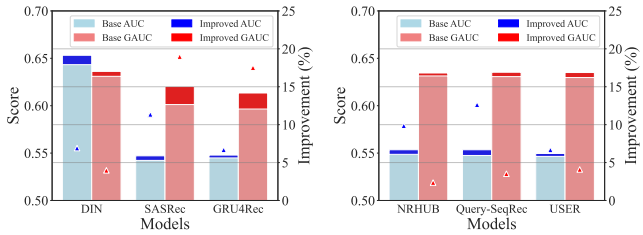
NHP-OAM consists of several key components, and to understand the effects of each component, we conducted several ablation experiments for NHP-OAM. The studies involved removing the Session-level Encoder (**w/o SLE**), excluding the query (recommendation) ratio aware score (**w/o QR**), using LSTM in place of the Hawkes process (**w/o HP**), removing user-specific information (**w/o UE**), eliminating the time-gate (**w/o TG**), and removing the Prediction Layer (**w/o PL**).

Table 3 presents the performance of various model variants on



(a) SR Models (b) SRS Models

Figure 5: Application on Item Recommendation.



(a) SR Models (b) SRS Models

Figure 6: Application on Open-App Item Recommendation.

the OAMD [8]. The results reveal several insights: First, the performance of the model without the Session-level Encoder (w/o SLE) falls short of that of NHP-OAM, emphasizing the importance of capturing rich contextual information within a user's session for improved model effectiveness. Second, the absence of the query (recommendation) ratio aware score (w/o QR) also leads to inferior performance, which underscores the significance of modeling relevance features. Third, excluding the Hawkes process (w/o HP) results in a significant drop in performance, highlighting the advantages of using this process to model open-app motivation. Lastly, we observe a decline in performance when User-specific information (w/o UE), Time-gate (w/o TG), and Prediction Layer (w/o PL) are removed, demonstrating the efficacy of using the Prediction Layer to model both time information and user-specific information.

5.4 Application Experiments

In this section, we evaluate the performance of two downstream recommendation tasks when deploying NHP-OAM with them. Specifically, the two applications are Item Recommendation and Open-App Item Recommendation. The performance of NHP-OAM on these applications reflects its effectiveness in learning better user representations, especially for tasks that require arousing user interest in a very short time, highlighting the immense application value of NHP-OAM.

5.4.1 Item Recommendation. Item Recommendation aims to generate a list of recommendations for a user u. We utilized the embedding $\mathbf{z}$ in Equation 9 trained by NHP-OAM to improve the item recommendation models [14, 15, 46], which can be formulation as:

$$p(i|u) = \sigma(\boldsymbol{i}^\top \cdot \boldsymbol{u}),$$

where $\boldsymbol{u} = \text{FFN}([\mathbf{u}_{reco}; \mathbf{z}])$, $[\cdot;\cdot]$ concatenates vectors and σ is the Sigmoid function. The $\mathbf{u}_{reco}$ vector is sourced from the original Item Recommendation models. For Joint Search-Recommendation Sequential Models [12, 37, 39], we can also utilize the predicted open-app motivation score to control the weight of Search items and Recommendation items within the current session history. For the dataset, we use items clicked by users in OAMD as positive items and unclicked items as negative items, adopting the same splitting method as in § 5.1.1, which aligns well with practical applications. Evaluation metrics include AUC, GAUC, and we separately calculate the relative improvement after applying NHP-OAM followed [46]. The item recommendation model and NHP-OAM are trained simultaneously. Evidenced by Figure 5, there is a noticeable improvement, serving as an indicator of our model's effectiveness in learning better user representations.

5.4.2 Open-App Item Recommendation. Open-App Item Recommendation refers to a type of Item Recommendation that has only a single opportunity to recommend, such as App Open Advertisement Recommendation and Message Pop-up Recommendation [21]. We employed the same model architecture, evaluation metrics, and training set as used in the above Item Recommendation. For the test set, we selected items from the user's first interaction, which includes both searched and recommended items. An interaction is considered a positive item if it is clicked, and a negative item otherwise. As evident from Figure 6, there is a significant improvement, which serves as evidence for NHP-OAM's effectiveness in capturing users' immediate interests upon app opening.

6 CONCLUSIONS

In this work, we introduce a novel problem, *open-app motivation prediction*, which is pivotal for platforms providing both search and recommendation services. We propose the NHP-OAM model, effectively tackling associated challenges. Leveraging the Neural Hawkes Process (NHP), our model captures temporal patterns and employs a hierarchical Transformer architecture, along with an intensity function aware of the relevance feature. We also construct a new real-world dataset, called OAMD, to advance the study of open-app motivation prediction. Comprehensive experiments demonstrate significant improvement over baseline models, underscoring the considerable application value of NHP-OAM.

ACKNOWLEDGMENTS

This work was funded by the National Key R&D Program of China (2023YFA1008704), the National Natural Science Foundation of China (No. 62376275, 62377044), Beijing Key Laboratory of Big Data Management and Analysis Methods, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, PCC@RUC, funds for building world-class universities (disciplines) of Renmin University of China. Supported by the Outstanding Innovative Talents Cultivation Funded Programs 2024 of Renmin University of China, Kuaishou Technology.

[8] To ensure the accuracy of the experiment, we conducted a series of analytical experiments on the real-world dataset OAMD.

REFERENCES

[1] Qingpeng Cai, Shuchang Liu, Xueliang Wang, Tianyou Zuo, Wentao Xie, Bin Yang, Dong Zheng, Peng Jiang, and Kun Gai. 2023. Reinforcing User Retention in a Billion Scale Short Video Recommender System. In *Companion Proceedings of the ACM Web Conference 2023*. 421–426.

[2] Bo Chang, Alexandros Karatzoglou, Yuyan Wang, Can Xu, Ed H Chi, and Minmin Chen. 2023. Latent User Intent Modeling for Sequential Recommenders. In *Companion Proceedings of the ACM Web Conference 2023*. 427–431.

[3] Lihua Chen, Ning Yang, and Philip S Yu. 2022. Time lag aware sequential recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 212–221.

[4] Minghao Chen, Houwen Peng, Jianlong Fu, and Haibin Ling. 2021. Autoformer: Searching transformers for visual recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*. 12270–12280.

[5] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2978–2988.

[6] Benjamin H Ellis, Michael Deceglie, and Anubhav Jain. 2019. Automatic detection of clear-sky periods from irradiance data. *IEEE Journal of Photovoltaics* 9, 4 (2019), 998–1005.

[7] Shaohua Fan, Junxiong Zhu, Xiaotian Han, Chuan Shi, Linmei Hu, Biyu Ma, and Yongliang Li. 2019. Metapath-guided heterogeneous graph neural network for intent recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 2478–2486.

[8] Songwei Ge, Zhicheng Dou, Zhengbao Jiang, Jian-Yun Nie, and Ji-Rong Wen. 2018. Personalizing search results using hierarchical RNN with query-aware attention. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 347–356.

[9] Bin Hao, Min Zhang, Weizhi Ma, Shaoyun Shi, Xinxing Yu, Houzhi Shan, Yiqun Liu, and Shaoping Ma. 2021. A Large-Scale Rich Context Query and Recommendation Dataset in Online Knowledge-Sharing. *arXiv preprint arXiv:2106.06467* (2021).

[10] Alan G Hawkes. 1971. Spectra of some self-exciting and mutually exciting point processes. *Biometrika* 58, 1 (1971), 83–90.

[11] Yacheng He, Qianghuai Jia, Lin Yuan, Ruopeng Li, Yixin Ou, and Ningyu Zhang. 2023. A Concept Knowledge Graph for User Next Intent Prediction at Alipay. In *Companion Proceedings of the ACM Web Conference 2023*. 45–48.

[12] Zhankui He, Handong Zhao, Zhaowen Wang, Zhe Lin, Ajinkya Kale, and Julian Mcauley. 2022. Query-Aware Sequential Recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 4019–4023.

[13] Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415* (2016).

[14] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *Proc. of ICLR*, Yoshua Bengio and Yann LeCun (Eds.).

[15] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*. IEEE, 197–206.

[16] Bekir Karlik and A Vehbi Olgac. 2011. Performance analysis of various activation functions in generalized MLP architectures of neural networks. *International Journal of Artificial Intelligence and Expert Systems* 1, 4 (2011), 111–122.

[17] Sundong Kim and Jae-Gil Lee. 2018. Utilizing in-store sensors for revisit prediction. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 217–226.

[18] Sundong Kim, Hwanjun Song, Sejin Kim, Beomyoung Kim, and Jae-Gil Lee. 2020. Revisit Prediction by Deep Survival Analysis. In *Advances in Knowledge Discovery and Data Mining: 24th Pacific-Asia Conference, PAKDD 2020, Singapore, May 11–14, 2020, Proceedings, Part II 24*. Springer, 514–526.

[19] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).

[20] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*. 95–104.

[21] Xiaochong Lan, Chen Gao, Shiqi Wen, Xiuqi Chen, Yingge Che, Han Zhang, Huazhou Wei, Hengliang Luo, and Yong Li. 2023. NEON: Living Needs Prediction System in Meituan. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 5292–5302.

[22] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time interval aware self-attention for sequential recommendation. In *Proceedings of the 13th international conference on web search and data mining*. 322–330.

[23] Yinfeng Li, Chen Gao, Xiaoyi Du, Huazhou Wei, Hengliang Luo, Depeng Jin, and Yong Li. 2022. Automatically Discovering User Consumption Intents in Meituan. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3259–3269.

[24] Zhiruo Li, Zhihui Cui, and Shunyao Wu. [n. d.]. An Effective Ensemble Framework with Multichannel Time Series for User Retention Prediction. ([n. d.]).

[25] Bryan Lim and Stefan Zohren. 2021. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A* 379, 2194 (2021), 20200209.

[26] Haitao Lin, Cheng Tan, Lirong Wu, Zhangyang Gao, Stan Li, et al. 2021. An Empirical Study: Extensive Deep Temporal Point Process. *arXiv preprint arXiv:2110.09823* (2021).

[27] Hongyuan Mei and Jason M Eisner. 2017. The neural hawkes process: A neurally self-modulating multivariate point process. *Advances in neural information processing systems* 30 (2017).

[28] Christian P Robert, George Casella, and George Casella. 1999. *Monte Carlo statistical methods*. Vol. 2. Springer.

[29] Teng Shi, Zihua Si, Jun Xu, Xiao Zhang, Xiaoxue Zang, Kai Zheng, Dewei Leng, Yanan Niu, and Yang Song. 2024. UniSAR: Modeling User Transition Behaviors between Search and Recommendation. arXiv:2404.09520

[30] Zihua Si, Xueran Han, Xiao Zhang, Jun Xu, Yue Yin, Yang Song, and Ji-Rong Wen. 2022. A Model-Agnostic Causal Learning Framework for Recommendation using Search Data. In *Proceedings of the ACM Web Conference 2022 (WWW '22)*. Association for Computing Machinery, New York, NY, USA, 224–233. https://doi.org/10.1145/3485447.3511951

[31] Zihua Si, Zhongxiang Sun, Xiao Zhang, Jun Xu, Yang Song, Xiaoxue Zang, and Ji-Rong Wen. 2023. Enhancing Recommendation with Search Data in a Causal Learning Manner. *ACM Trans. Inf. Syst.* 41, 4, Article 111 (apr 2023), 31 pages. https://doi.org/10.1145/3582425

[32] Zihua Si, Zhongxiang Sun, Xiao Zhang, Jun Xu, Xiaoxue Zang, Yang Song, Kun Gai, and Ji-Rong Wen. 2023. When Search Meets Recommendation: Learning Disentangled Search Representation for Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23)*. Association for Computing Machinery, 1313–1323. https://doi.org/10.1145/3539618.3591786

[33] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864* (2021).

[34] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1441–1450.

[35] Zhongxiang Sun, Zihua Si, Xiaoxue Zang, Dewei Leng, Yanan Niu, Yang Song, Xiao Zhang, and Jun Xu. 2023. KuaiSAR: A Unified Search And Recommendation Dataset *(CIKM '23)*. Association for Computing Machinery, New York, NY, USA, 5407–5411. https://doi.org/10.1145/3583780.3615123

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[37] Chuhan Wu, Fangzhao Wu, Mingxiao An, Tao Qi, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Neural news recommendation with heterogeneous user behavior. In *EMNLP-IJCNLP*. 4874–4883.

[38] Haocheng Xu, Siyi Liu, and Jiaxin Wu. [n. d.]. A Hybrid approach for Users Retention Rate Prediction. ([n. d.]).

[39] Jing Yao, Zhicheng Dou, Ruobing Xie, Yanxiong Lu, Zhiping Wang, and Ji-Rong Wen. 2021. USER: A unified information search and recommendation model based on integrated behavior sequence. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2373–2382.

[40] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are transformers effective for time series forecasting?. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 37. 11121–11128.

[41] Peiyan Zhang, Jiayan Guo, Chaozhuo Li, Yueqi Xie, Jae Boum Kim, Yan Zhang, Xing Xie, Haohan Wang, and Sunghun Kim. 2023. Efficiently leveraging multi-level user intent for session-based recommendation via atten-mixer network. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 168–176.

[42] Qiang Zhang, Aldo Lipani, Omer Kirnap, and Emine Yilmaz. 2020. Self-attentive Hawkes process. In *International conference on machine learning*. PMLR, 11183–11193.

[43] Qihang Zhao. 2022. RESETBERT4Rec: A pre-training model integrating time and user historical behavior for sequential recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*. 1812–1816.

[44] Wayne Xin Zhao, Zihan Lin, Zhichao Feng, Pengfei Wang, and Ji-Rong Wen. 2022. A revisiting study of appropriate offline evaluation for top-N recommendation algorithms. *ACM Transactions on Information Systems* 41, 2 (2022), 1–41.

[45] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5941–5948.

[46] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui

Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1059–1068.

[47] Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. 2020. Transformer hawkes process. In *International conference on machine learning*. PMLR, 11692–11702.