International Journal of Machine Learning and Cybernetics
https://doi.org/10.1007/s13042-023-01780-2

ORIGINAL ARTICLE

Iterative convolutional enhancing self-attention Hawkes process with time relative position encoding

Wei Bian[1] · Chenlong Li[1] · Hongwei Hou[1] · Xiufang Liu[1]

Received: 3 November 2021 / Accepted: 16 January 2023

Abstract
Modeling Hawkes process using deep learning is superior to traditional statistical methods in the goodness of fit. However, methods based on RNN or self-attention are deficient in long-time dependence and recursive induction, respectively. Universal Transformer (UT) is an advanced framework to integrate these two requirements simultaneously due to its continuous transformation of self-attention in the depth of the position. In addition, migration of the UT framework involves the problem of effectively matching Hawkes process modeling. Thus, in this paper, an iterative convolutional enhancing self-attention Hawkes process with time relative position encoding (ICAHP-TR) is proposed, which is based on improved UT. First, the embedding maps from dense layers are carried out on sequences of arrival time points and markers to enrich event representation. Second, the deep network composed of UT extracts hidden historical information from event expression with the characteristics of recursion and the global receptive field. Third, two designed mechanics, including the relative positional encoding on the time step and the convolution enhancing perceptual attention are adopted to avoid losing dependencies between relative and adjacent positions in the Hawkes process. Finally, the hidden historical information is mapped by Dense layers as parameters in Hawkes process intensity function, thereby obtaining the likelihood function as the network loss. The experimental results show that the proposed methods demonstrate the effectiveness of synthetic datasets and real-world datasets from the perspective of both the goodness of fit and predictive ability compared with other baseline methods.

Keywords Hawkes process · Universal Transformer · Relative positional encoding · Convolution enhancing

1 Introduction

Human activities and natural phenomena tend to generate a large amount of event sequence data, such as user reposts on social networks, occurrences of equipment maintenance failures, admissions of hospital patients, and occurrences of natural disasters [1]. At present, with growing rapidly of the amount of event sequence data, it has become a hot and important topic to analyze event sequence data efficiently and generate business value. Specifically, one can predict and control the occurrence of future events based on the dynamic modal law of event sequence data [2, 3]. For example, indicating the direction of network forwarding behavior can control the occurrence of emergencies [4], crime analysis can be used to take corresponding measures to prevent the occurrence of crime, and accurate prediction of climate change can reduce the impact of natural disasters on human production.

Event sequence data occur at irregular time points, which is also known as asynchronous data. Exploring the potential dynamic modal law from the event sequence data is a challenging problem. Particularly, Gaussian processes [5, 6] and linear models [7, 8] used to model the continuous and discrete dynamic modal laws of time series are no longer applicable to analyze event sequence data. Regarding the event sequence data, the temporal point process is an effective mathematical tool. Different from the Gaussian process and linear models, the temporal point process treats irregular time points as random variables and models their distribution. Hawkes process is a classical temporal point process that assumes that the occurrence of an event depends on event history and uses the sequential probabilistic generative method to model marked events happening [9]. In particular, Hawkes process embodies the dependence of historical events into the mathematical

✉ Chenlong Li
lichenlong@tyut.edu.cn

[1] School of Mathematics, Taiyuan University of Technology, Taiyuan, China

Published online: 04 February 2023

formula of its conditional intensity function and then obtains the conditional probability density function of the time points. However, modeling Hawkes process needs prior knowledge to pre-select the parameterizing form of conditional intensity function, which increases the risk of incorrect model selection and limits their applications in practice.

To relax the limitation of prior knowledge in order to provide a more flexible and effective form of conditional intensity function or density function, researchers have introduced the neural network-based Hawkes process model [10–13]. Neural network models have the advantage of nonlinear fitting. Related theories and applications have been widely developed and applied, such as healthy diet recognition [14], semantic segmentation [15], the Internet of Things [2, 4], and system control [16]. In general, the neural network-based Hawkes process model treats the conditional intensity function as a nonlinear function of the historical event data, which is usually parameterized using a recurrent neural network (RNN). For instance, Du et al. first proposed an RNN-based network model for modeling the Hawkes process by embedding historical information into the hidden state of RNN [1]. After that, many variant network-based point process models based on RNN emerged [10, 11]. These models either ingeniously design the form of the intensity function [1], directly output the integral of the intensity function and then derive it [10], or learn the conditional distribution [11]. These works use RNN to add a full connection layer to process the dependency information of point process sequence data. The difference is that the loss function of the output layer has its advantages. Recently, self-attention [17] has received successive attention in numerous applications, and there is evidence that self-attention is more capable than RNN. Many works explore the use of self-attention instead of RNN as a framework for point process data processing. Zhang et al. were the first to study the effectiveness of self-attention in the Hawkes process [12]. Subsequently, Zuo et al. explored the use of Transformer in point-process modeling, which incorporates structural information into the loss function with a creative method [13].

Overall, these studies focused on the challenge of obtaining a good intensity function or density function, and RNN or self-attention is nothing but the framework for processing point process data information. However, simple RNN or self-attention cannot effectively learn the potential historical expression of information in stream data. On the one hand, the conditional intensity function based on RNN is formulated as a nonlinear function of the arrival times of events, while the RNN has the gradient dispersion problem and thus suffers from the inability to learn the long-term dependency of the arrival times of events [18]. Conversely, self-attention is the weighted average of the states at the whole moment and therefore can learn the long-term dependency [13], but it does not accord with the characteristic of point process recursion.

This paper aims to solve the above two problems in the existing neural network-based Hawkes process models. For this purpose, we design an ICAHP-TR using the UT framework [19], which exerts induction on the depth of the position-wise of self-attention to combine the recursive induction of RNN with the global receptive field of self-attention. UT continuously corrects vector representations of each position by applying a transition function. This establishes the sequential nonlinear connection between the weight of self-attention and the latent history. In contrast to the previous Hawkes process modeling based on RNN or self-attention mechanism, not only does UT feature the influence of longer-interval events on subsequent events but also leverages the essential properties of the recursive inductive bias of the point process intensity function. Given the advantages of the appeal, we use UT as the main structure of the proposed model, which is also the most prominent advantage that distinguishes ICAHP-TR from the previous related work. In order to make the characteristics of the Hawkes process at relative moments and adjacent events clearer in ICAHP-TR, we propose two ingenious designs containing Relative Positional Encoding [20] on the Time Step (RPT) and Convolution Enhancing Perceptual Attention (ConvEnc) for calculating attention score weight in self-attention. Collaborative use can improve the model capacity and the model interpretability. To summarize, the main contributions of our paper are as follows:

- Combine the advantages of RNN and self-attention while eliminating defects using the Universal Transformer to learn the potential expression of historical information.
- Design the RPT to transmit information signal that contains the positional relationships of events in a sequence of point processes over time.
- Use ConvEnc as the secondary distribution of attention, which is combined with RPT to better understand the interaction of the event points at different time points.

Through the above three aspects, the proposed neural network model is more adapted to the learning of the Hawkes process. Experiments on synthetic and real-world datasets show that ICAHP-TR is superior from the perspective of both the goodness of fit and the predictive ability to recent approaches built upon RNN or self-attention.

2 Related work

This section reviews related works while focusing on the state-of-the-art models of the Hawkes process or point process based on neural network and their pros and cons.

Although various models have different loss functions designed at the output end, they can be divided into two types based on RNN and self-attention according to the information processing layer of the network.

RNN-based models Theoretically, suppose a sequence of strictly increasing arrival times $(t_1, \ldots, t_N)$. The intensity function based on RNN can be formulated as $h_n = f\left(Uh_{n-1} + Wt_n + b\right) \iff h_n = g(t_n|h_{n-1})$, in which U, W, b are parameters. It is obvious that if h_{n-1} and h_n are regarded as two adjacent historical conditions of arrival times, and RNN can represent the recursive relationship between them. As RNN conforms to the recursive characteristics of the Hawkes process and is proposed earlier, the preliminary research work mainly focuses on RNN. Du et al. (2016) suggested RMTPP using RNN to encode event history as the vector h_i, and then using h_i to define the conditional intensity function, such as the *time t_i-constant intensity* model $\lambda^*(t_i) = \exp\left(v^T h_i + b\right)$ [18, 19] or a more flexible *exponential intensity* model $\lambda^*(t_i) = \exp(w(t_i - t_{i-1}) + v^T h_i + b)$ [1]. This exponential intensity corresponds to a Gompertz distribution [21]. The authors first modeled the point process with RNN. However, the unimodal distribution does not match the flexibility of the model [22]. Then, Mei and Eisner proposed a novel RNN architecture that could model complex intensity functions [23] with the explanatory power. While the cost of this flexibility is the inability to assess the possibility of closed-form and thus, Monte Carlo integration was required. Therefore, Omi et al. introduced a flexible full neural network (FullyNN) intensity model, in which a neural network was used to compute the cumulative intensity function, and the conditional intensity was obtained by differentiating it [10]. This method could not only obtain a flexible model of the intensity function but also accurately calculate the logarithmic likelihood function, including the integration of the intensity function, without any numerical approximation. However, this model also exists some disadvantages: (1) not defining a valid probability density function; (2) requiring enormous computation power when sampling that requires iterative root-finding; (3) not calculating the expectation in a closed-form; (4) not normalizing the probability density function of FullyNN model to one since the suboptimal selection of network architecture; (5) giving non-zero probabilities of negative event time intervals [11]. Shchur et al. proposed parameterizing the conditional density function using a mixed log-normal distribution to address shortcomings such as insufficient flexibility, lack of closed-form likelihoods, and inability to generate samples analytically [11]. The mixture model matches the flexibility of the flow-based model, and the sampling and calculating arrival times are computed in closed form. However, its shortcoming lies in the clipping of the variance range is an uncontrollable factor in the time prediction when the model is programmed. More importantly, not only do the above models have their deficiencies, but they also have difficulties learning the long-term dependence relationship caused by the gradient diffusion phenomenon in the training RNN.

Self-attention-based models Unlike the RNN-based models, self-attention is the weighted average of the states at the moment because $h_n = \sum_{i=1}^{n} w_i^n(t_1, \ldots, t_n)v(t_i), h_{n-1} = \sum_{i=1}^{n-1} w_i^{n-1}(t_1, \ldots, t_n)v(t_i)$ and $w_{:}^{n}$ is independent of $w_{:}^{n-1}$. Therefore, self-attention-based models can learn the influence of events of a larger span. Researchers set out to explore whether self-attention could be a more effective substitute for RNN [24]. Zhang et al. firstly filled the gap in point-process modeling of self-attention [14, 25]. They employed self-attention to capture the impact of historical events on subsequent events to predict when the next event is most likely to occur and introduced a more reasonable embedding of position and content-encoding [12]. This model fails on long-term horizon predictions. Zuo et al. did similar work simultaneously [13]. They leveraged the self-attention mechanism for model training so that the model can easily learn the long dependence due to the global receptive field and meanwhile enjoyed computational efficiency. Moreover, they designed a causal structural loss function which can incorporate additional structural knowledge. However, since the mean square error of the arrival time is also used as a loss, the training tends to fall into a suboptimal solution. The above two works fill the gap of self-attention as a sequence processing tool. Compared with RNN, self-attention does not have the problem of gradient disappearance, and it is easier to learn the relationship between long-range events. But this will ignore the interdependent relationship between events, which does not conform to the recursive feature of the point process.

After reviewing related works, we discover that it is challenging to choose a deep learning network for the Hawkes process due to problems of RNN or self-attention in training or theory. Generally, a good network should avoid long-term dependencies as much as possible while capturing the dependencies of previous and previous events. The main framework of the proposed work, UT, is such a network. UT continuously corrects vector representations of each position by applying a transition function. This establishes the sequential connection between h_{n-1} and h_n, i.e., $w_i^n\left(t_1, \ldots, t_n\right) = f^{'}(t_n|h_{n-1})$. Then $h_n = \sum_{i=1}^{n} f_i^{'}(t_n|h_{n-1})v(t_i) \iff h_n = g^{'}(t_n|h_{n-1})$. Therefore, according to the above reasoning analysis, we believe that UT can consider both recursive induction and global receptive field, which Hawkes process modeling requires. It is worth noting that UTHP [26] recently used the original UT framework to build a Hawkes process model. Still, UT is designed for the natural language, resulting in the limited effect of the UTHP for the Hawkes process. In other words, the UT framework cannot simply be ported

to Hawkes modeling, but rather needs to be fixed. In addition to deep learning approaches, other works have considered alternative training methods the largest possible target for the process of marked time series points [11]. Examples include noise contrast estimation [27], Wasserstein distance [28] and reinforcement learning [29, 30].

3 Model

This section will explain the design of our UT-based deep learning network for Hawkes process modeling. In the structure of this section, we first formulate the global framework of ICAHP-TR's network level and operational details in Sect. 3.1 and then introduce RPT and ConvEnc as supplements to the network details in Sects. 3.2 and 3.3.

3.1 Universal Transformer Frame

From the perspective of the global framework of ICAHP-TR, after the sequential event data pass through the ***input layer***, the hidden historical state is encoded by the ***Universal Transformer layer*** and then fed into the ***output layer***, yielding the conditional intensity function and categorical probabilities. The architecture of the ICAHP-TR network is visible in Fig. 1. We now describe how to adapt the Universal Transformer mechanism to the Hawkes process.

Given a sequence of events $e = \{(t_i, m_i)\}_{i=1}^{L}$ with the length L, where each arrival time t_i corresponds to a category $m_i \in \{1, 2, \ldots, M\}$ with a total number of M categories, the task of Hawkes Process modeling is to estimate the conditional intensity function $\lambda^*(t)$.

Input Layer Event sequences need to be embedded as dense vectors to be used as signal inputs. Event embedding is used to implement this process by exerting a linear embedding layer in category one-hot vectors and then concatenating it with time interval vectors:

$$C_{em} = cW^c,$$
$$E_{em} = \text{Concat}\left(C_{em}, \Delta t\right),$$

where c is the one-hot vector of categories, W^c is the embedding matrix, Δt is the time interval vector, C_{em} is the categorical embedding vector, Concat$(\cdot)$ is the concatenate operation, and E_{em} is the event embedding vectors.

Remark: We adhere to the standard trainable event embedding that differs from the event embedding incorporated in sinusoidal position embedding used in SAHP and THP. There are two purposes for adopting this method: (1) to avoid the embedding encoding range from being forced to be controlled between -1 and 1 to provide a comprehensive feature space; (2) we use relative positional encoding, which

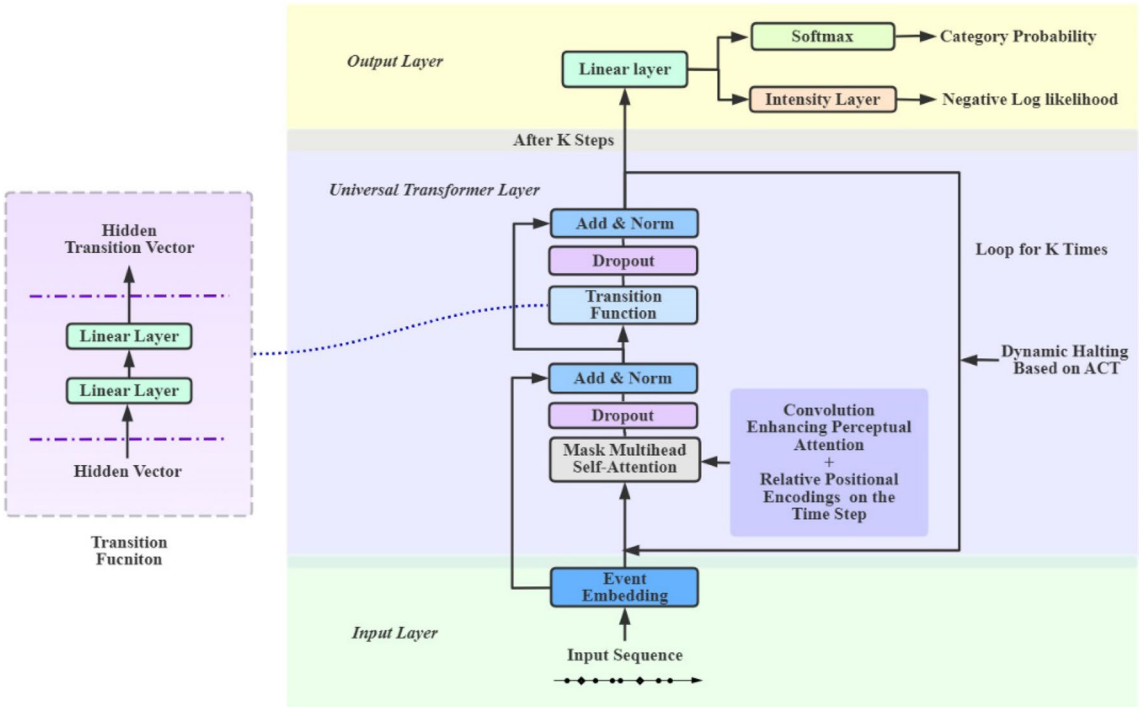

Fig. 1 The Architecture of ICAHP-TR

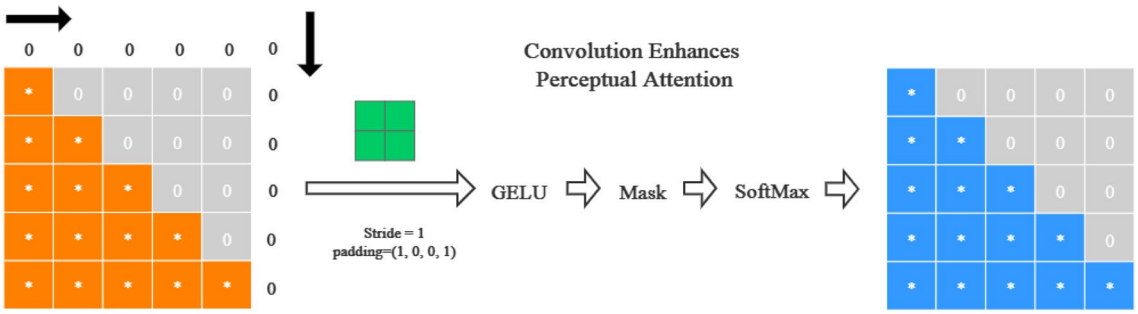

Fig. 2 Flow chart of convolution enhancement operation

is reflected in the attention score weight rather than in the input. Stem from the reason of reducing the redundancy of input information and reducing the parameters, our event embedding encoding does not include positional encoding.

Universal Transformer Layer Given the event embedding, we need to compute the hidden historical state to consider previous events' influence. We use the UT framework to do it. Specifically, UT starts with the event embedding as an initialized embedding matrix $H^0 \in R^{L\times d}$. Then, UT iteratively calculates the representation H_t of all T timestamps by applying the masked multi-head dot-product self-attention in the t-th step, followed by the transition function layer applying dropout [31] and layer normalization [32] to each block. This process is illustrated in ***Universal Transformer Layer*** of Fig. 1. In detail, we use the multi-head version with k heads:

$$\text{MultiHead Self Attention}(H_t) = \text{Concat}(\text{head}_1, \ldots, \text{head}_k)W^O,$$
$$\text{head}_i = \text{Self Attention}\left(H^t W_i^Q, H^t W_i^K, H^t W_i^V\right),$$

where $W_i^Q \in R^{d\times d/k}$, $W_i^K \in R^{d\times d/k}$, $W_i^V \in R^{d\times d/k}$, $W^O \in R^{d\times d}$ and d is the hidden state size.

At step t, UT iteratively computes revised hidden state $H^t \in R^{L\times d}$ for all L events as follows:

$$H^t = \text{LayerNorm}\left(A^t + \text{Transtion}\left(A^t\right)\right),$$
$$A^t = \text{LayerNorm}\left(H^{t-1} + \text{MultiHead Self Attention}\left(H^{t-1}\right)\right),$$

where *Transition* function is two fully-connected neural networks with residual connection shown on the left of Fig. 2, note that the *self-attention* here couples PRT and ConvEnc to improve the effectiveness of UT's modeling of the Hawkes point process. This will be introduced in Sects. 3.2 and 3.3. In addition, to increase the flexibility of the number of position-wise iterations, the Adaptive Computation Time (ACT) mechanism [33] is expanded in time-step iterations.

Output Layer The hidden historical state only delivers historical information and needs to be comprehensively transformed into the conditional intensity function to present the events' dynamic modal law. In the spirit of the work introduced by Qiang Zhang et al. [10], the conditional intensity function is defined through a nonlinear transformation of H^t as follows:

$$\lambda^*(t) = \text{softplus}\left(W_{i+1}\tanh\left(\mu_{i+1} + \left(\eta_{i+1} - \mu_{i+1}\right)\exp\left(-\gamma_{i+1}\left(t - t_i\right)\right)\right) + b_{i+1}\right)$$
$$\mu_{i+1} = \text{Gelu}\left(h_{i+1}\boldsymbol{W^{\mu}}\right)$$
$$\eta_{i+1} = \text{Gelu}\left(h_{i+1}\boldsymbol{W^{\eta}}\right)$$
$$\gamma_{i+1} = \text{softplus}\left(h_{i+1}\boldsymbol{W^{\gamma}}\right)$$

where $\boldsymbol{h_{i+1}} = \boldsymbol{H^t_{:,i+1}} \in R^{1\times d}$, $\boldsymbol{W^{\mu}} \in R^{d\times 1}$, $\boldsymbol{W^{\eta}} \in R^{d\times 1}$, $\boldsymbol{W^{\gamma}} \in R^{d\times 1}$, $W_{i+1} \in R^{1\times 1}$ and $b_{i+1} \in R^{1\times 1}$ for $t \in (t_i, t_{i+1}]$. The activation functions chosen above posse their respective functions: (1) Gelu activation function has been shown empirically to be superior to other activation functions in terms of self-attention; (2) Softplus activation function constrains the terms to be strictly positive; (3) Tanh activation function shrinks almost all values to $[-1, 1]$ to prevent problems from appearing too large or too small values. More importantly, the reasons for adopting this formal conditional intensity function are that the designed formula corresponds to the terms in the expression of Hawkes process conditional intensity function, i.e., the $\text{softplus}\left(W_{i+1}\tanh\left(\mu_{i+1}\right) + b_{i+1}\right)$ converges to *exogenous*

component μ as $t \to +\infty$, γ_{i+1} is the counterpart of the *decaying parameter,* and the *excitation parameter* is determined by $\left(\eta_{i+1} - \mu_{i+1}\right)$ [12].

In the aspect of calculating categorical probabilities, we set out straightforwardly from the history hidden state and pass a linear transformation followed by the softmax function.

$$\text{CategoricalProbabilities} = \text{softmax}\left(\boldsymbol{H^t W^p} + \boldsymbol{b^p}\right),$$

where $\boldsymbol{W^p} \in R^{d\times K}, \boldsymbol{b^p} \in R^{L\times K}$. Commonly, we choose the category with the highest probability as the event category prediction:

$$\text{Category} = \underset{i}{\text{argmax}}\, \text{Categorical Probabilities}_i,$$

where P_i is the predicted probability of each category i.

3.2 Relative positional encoding on the time step

The parallel structure of the self-attention mechanism does not explicitly model positional information, which is inseparable from the information integrity of the sequence. In the original UT framework, the information of sequence order is provided by a sinusoidal absolute positional encoding which adds representations of positional information $P_{1:L}$ to the vector representation of sequences and the time annotated by T_t with the same approach. If we invariably leverage absolute positional encoding without thinking, the hidden historical state would be computed schematically by

$$\begin{aligned} H^t &= f\left(H^t, E_{em} + P_{1:L} + T_t\right), \\ H^{t-1} &= f\left(H^{t-1}, E_{em} + P_{1:L} + T_{t-1}\right), \end{aligned}$$

where f represents a transformation function. However, in consideration of the iterative architecture of UT and the relative position involved in the point process intensity function, absolute positional encoding is not competent to provide information to learn the variation in where to attend over time steps since T_t shares the same positional encoding $P_{1:L}$ as T_{t-1} by simply element-wise plus. This will cause the model to lose this part of the information. To supplement the missing information, we are inspired by the idea of relative position to derive RPT.

The sequential information processing in the attention mechanism can essentially be boiled down to the normalized weight values of the attention weight matrix. Absolute position encoding is equivalent to adding the orders of the front and rear order to the weight. Therefore, the designed RPT only performs additional processing on the attention weight matrix, and the encoding task of the sequence is completed by the input layer. In the comparison, in the time and position encoding of the original UT, the attention weight between *ith* query vector q_i and *jth* key vector k_j within the same segment can be decomposed as follows:

$$\begin{aligned} A_{i,j}^{abs} &= \left(x_i + p_i + t\right) W_Q W_K^T \left(x_j^T + p_j^T + t^T\right) \\ &= \underbrace{x_i W_Q W_K^T x_i^T}_{(a)} + \underbrace{x_i W_Q W_K^T p_j^T}_{(b)} + \underbrace{x_i W_Q W_K^T t^T}_{(c)} \\ &+ \underbrace{p_i W_Q W_K^T x_j^T}_{(d)} + \underbrace{p_i W_Q W_K^T p_j^T}_{(e)} + \underbrace{p_i W_Q W_K^T t^T}_{(f)} \\ &+ \underbrace{t W_Q W_K^T x_j^T}_{(g)} + \underbrace{t W_Q W_K^T p_j^T}_{(h)} + \underbrace{t W_Q W_K^T t^T}_{(i)}, \end{aligned} \tag{1}$$

We modify the items in the above decomposition in accordance with the idea of relative position shifting with time. Note that RPT refers to the additional time steps, whereas the previous relative positional encoding can only deal with the position [34, 35]. This requires us to do extra work for the time step. Now we present the modification scheme:

$$\begin{aligned} A_{i,j}^{rel} &= \underbrace{x_i W_Q W_K^T x_i^T}_{(a)} + \underbrace{x_i W_Q W_K^T \boldsymbol{R_{i-j}^T}}_{(b)} + \underbrace{u W_Q W_K^T x_j^T}_{(d)} \\ &+ \underbrace{v W_Q W_K^T \boldsymbol{R_{i-j}^T}}_{(e)} + \underbrace{w W_Q W_K^T t^T}_{(f)} + \underbrace{t W_Q W_K^T \boldsymbol{R_{i-j}^T}}_{(h)}, \end{aligned} \tag{2}$$

- Firstly, we replace all symbols of key-based p_j for computing key vectors with a trainable relative counterpart $\boldsymbol{R_{i-j}^T}$ in terms (b), (e), (h) in Eq. (1). This reflects that the relative position is more important than where to attend, which is the embodiment of the notion that the intensity function $\lambda(t)$ can delicately be decomposed into the accumulative function of Δt. Notice that R is trainable there and clipped as follows [11]:
  $R = P_{1:L}\left[\text{clip}(i-j, k, -k)\right]$,

where k is the maximum absolute value of the clipping and $P_{1:L}$ is trainable. And t is extracted sequentially from a learnable T.

- Secondly, corresponding to the query-based p_i in items (d), (e), and (f), respectively, we similarly replace them with trainable parameters u, v, and w, which are restricted from changing with the position on purpose. When the time step is determined, the query vector is identical for all the query positions, and the bias of the time step is demonstrated in the (f) term.
- Finally, we deliberately drop two bias terms (c) and (g) since the event embedding information maintains invariance throughout the recursive iteration and abandon one

bias term (i) because there is no time step bias information against t step.

Under the modification, we are conscious of the idea proposed above, yielding an appealing formula that each term has a dedicated role: the term (a) manifests the attention to the content of the event, and the term (b) signifies the content-based position bias, the term (d) captures the global content-based query bias, the term (e) commands relative position bias, the term (f) represents the position coherent with the time step bias, and term (h) addresses global time step query bias designed to represent the consistency between time step and position. This provides a feasible and straightforward solution to the issue that positional attention variates over the time step. Moreover, relative positional encoding has been empirically demonstrated to better express positional information relative to absolute positional encoding. Yet, it has not been favored in point process modeling.

After the RPT provides the order information, the attention weight matrix will perform a secondary distribution of weights through ConvEnc (next section) operation and then perform matrix multiplication with the encoded sequence vector matrix to obtain the hidden historical vectors that the output layer can decode.

3.3 Convolution enhancing perceptual attention

Notice that the strong correlation of events at adjacent time points has not been well-matched captured in the model, which is evident in the interdependency of event occurrence. To deal with this challenge, we propose a simple method that employs a two-dimensional convolution operation on the attention weight matrix. The attention weight matrix is referred to as a single-channel image. We just adjust the adjacent weights, so keep the number of channels the same. Firstly, we fill the edge of the attention weight matrix with the padding of size one, then use a 2×2 convolution kernel to move on it with a stride of 1, and trim the last row and last column of the matrix after convolution. Note that future information will not be attended since having masked out all values that correspond to future events. We name this operation Convolution Enhancing Perceptual Attention and schematically formulate it as follows. Below we briefly illustrate the improvements brought about by this simple design through a mathematical description.

Firstly, suppose the weight matrix of the convolution kernel is

$$W_c = \begin{bmatrix} w_{00} & w_{01} \\ w_{10} & w_{11} \end{bmatrix}.$$

Then, define the operation:

$$q_i \to k_j := w_{(1+i-m),(1+j-l)} \cdot Attn_{i,j},$$

where m and l are the subscript of the query vector $\dot{q}_m$ and the key vector $\dot{k}_l$ of ConvEnc,$i \le m$, $j \le l$,and $[Attn_{i,j}]_{L\times L}$ is the score weight matrix of attention before convolution enhancement.

Finally, define convolution enhancement operation $< \dot{q}_i, \dot{k}_j >$:

$$\left\langle \dot{q}_n, \dot{k}_n \right\rangle = q_n \to k_n, n \ge 1, \tag{3}$$

$$\left\langle \dot{q}_n, \dot{k}_{n-1} \right\rangle = q_{n-1} \to k_{n-1} + q_n \to k_{n-1} + q_n \to k_n, n \ge 2, \tag{4}$$

$$\begin{aligned} \left\langle \dot{q}_n, \dot{k}_{n-l} \right\rangle = & \underbrace{q_{n-1} \to k_{n-l}}_{(a)} + \underbrace{q_{n-1} \to k_{n-l+1}}_{(b)} \\ & + \underbrace{q_n \to k_{n-l}}_{(c)} + \underbrace{q_n \to k_{n-l+1}}_{(d)}, n-1 \ge l \ge 2, \end{aligned} \tag{5}$$

Each term on the right-hand of the Eq. (5) has an intuitive meaning: term (a) represents that the last event queries the current key event, the term (b) reflects that the last queries the next key, the term (c) shows that the current queries the current key and (d) embody that the current queries the next key. Equations (3) and (4) apply the same principle, but some counterpart items are equal to zero due to the mask for future information, i.e., $Attn_{n-1,n-1}$, $Attn_{n,n-1}$ and $Attn_{n-1,n}$ equal to zero in Eq. (3) and $Attn_{n-1,n}$ equals zero in Eq. (4). With the addition of such a convolution operation, the attention weight will be composited by the current and previous time query vectors' attention to the current and next time. The model can learn the correlation between adjacent times with the relative positional encoding. After getting the convolution-enhanced attention matrix, we continuously perform the Gelu activation function, mask, and softmax operation to get the final attention weight. We visualize this operation and subsequent detail processing in the following Fig. 2.

In general, ConvEnc can be summarized as a secondary distribution of the self-attention weight. Specific to the issue of modeling the Hawkes process, this approach adds the information of the positional receptive field in the neighboring area based on the global receptive field obtained by the self-attention mechanism. On the one hand, this processing improves the flexibility of the attention weight, and on the other hand, the distribution of the receptive field is more in line with the logic of the Hawks process.

3.4 Parameter learning and prediction approach

For a sequence $e = \{(t_i, m_i)\}_{i=1}^{L}$ in an observation interval [0, T], given the intensity function $\lambda(t|\mathcal{H}_t)$, model parameters can be learned by maximizing the log-likelihood of observing e.

$$\ell(e) = \sum_{i=1}^{L} \left(\log f\left(t_i|\mathcal{H}_i\right) + \log P\left(m_i|\mathcal{H}_i\right)\right)$$
$$= \sum_{i=1}^{L} \left[\log \lambda\left(t_i|\mathcal{H}_i\right) + \log P\left(m_i|\mathcal{H}_i\right)\right] - \int_0^T \lambda\left(t|\mathcal{H}_t\right)dt.$$

To estimate the next time point, we compute the expected time under the predicted distribution $f^*(t)$:

$$\widehat{t}_{j+1} = \int_{t_j}^{\infty} t \cdot f^*(t)dt.$$

Because the integral of the softplus function is not evaluated in closed form. In this work, we use the numerical integral method: first, sampling from the distribution $f^*(t)$, and then computing the sample mean as an unbiased estimate of the expectation $\widehat{t}_{j+1}$. And as for calculating the integral $\int_0^T \lambda\left(t|\mathcal{H}_t\right)dt$, we adopt Monte Carlo simulation: first sampling t_i from Uniform distribution $U(0, T)$, then averaging $\lambda\left(t_i|\mathcal{H}_t\right)$, which is evaluated as capable in Zhang et al. [12].

4 Experiments

In this section, we evaluate the performance of our proposed network on two synthetic datasets and three real-world datasets and then compare it with the state-of-the-art RNN-based models and self-attention-based models. Subsequently, we conduct ablation studies to show the impact of the proposed modules UT, RPT, and ConvEnc in our network. Furthermore, we demonstrate the interpretability of the learned attention weights.

4.1 Datasets

We first introduce the two synthetic datasets. The datasets are generated from the Hawkes processes with different parameters in the conditional intensity function $\lambda^*(t) = \mu + \sum_{i=1}^{M} \sum_{t_j<t} \alpha_i \exp(-\gamma_i t)$. We refer to these two synthetic datasets as Hawkes 1 and Hawkes 2. Specifically, we use the thinning algorithm [36] to generate Hawkes 1 and Hawkes 2 on the time interval from 0 to 100 s in which the parameters are set as $\{\mu = 0.2, M = 1, \alpha_1 = 0.8, \gamma_1 = 1\}$ and $\{\mu = 0.2, M = 2, \alpha_1 = 0.4, \gamma_1 = 1, \alpha_2 = 8, \gamma_2 = 20\}$, respectively. Using these two sets of parameters can generate synthetic datasets with different characteristics. Figure 3 visualizes the condition intensity functions of the Hawkes processes with different parameters. From Fig. 3, it can be seen that the intensity function of Hawkes 1 fluctuates with

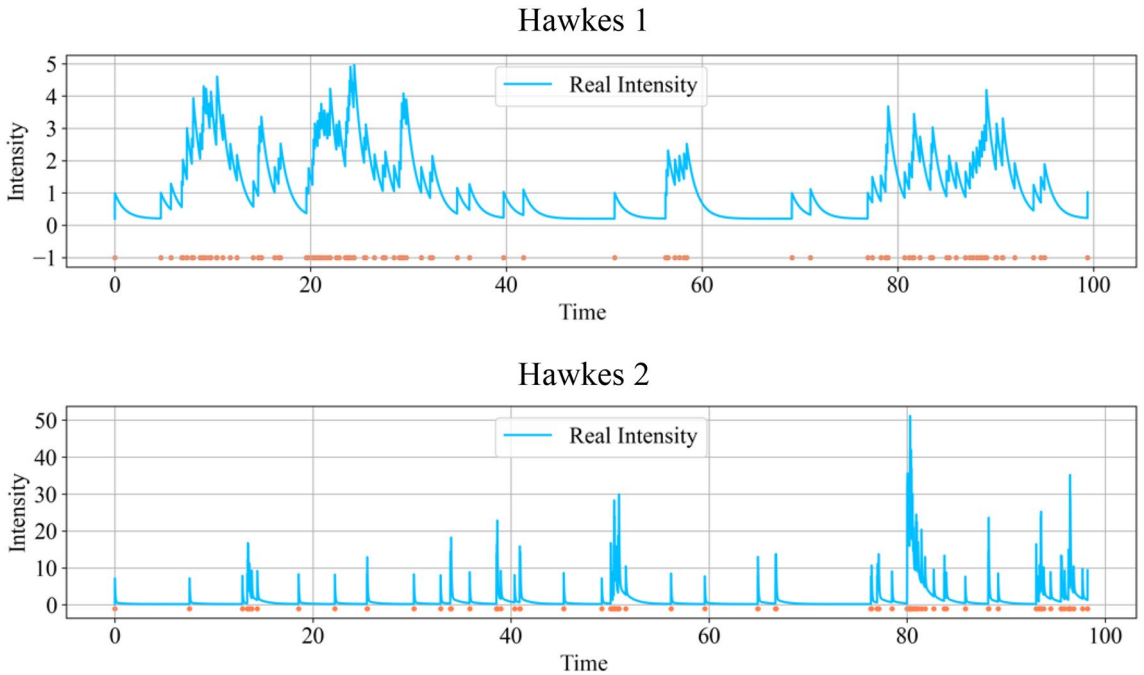

Fig. 3 The intensities of the two Hawkes processes over the synthetic datasets, respectively

a significant frequency, while the intensity function of Hawkes 2 is relatively gentle.

We now introduce the three real-world datasets, Mooc [37], Wikipedia [37], and Yelp Toronto [11]. Mooc dataset is an open-source dataset that consists of interactions (videos, answers, etc.) of students enrolling in a Mooc online course. A total of 672,447 interaction records are collected in this dataset. Each interaction is an event with various types (97 unique types) of 7,047 users. Wikipedia dataset is sequential data edited on Wikipedia pages, selecting the 1,000 most edited pages as items that users (a total of 8227 users) edited at least five times a month. This generates 157,474 interactions of users studied as types of the occurrence time of the event. Yelp Toronto dataset comes from reviews of the 300 most frequented restaurants in Toronto over time. Each record expresses a sequence of time of visits by customers of a specific restaurant. These datasets are selected purposefully to cover variations in data characteristics, i.e., the number of event types ranges from 97 to 8227, and the average sequence length ranges from 56 to 717. The statistics of the above three real-world datasets are summarized in Table 1.

4.2 Metrics

To evaluate the fitting performance of the proposed method, we use four metrics, i.e., negative log-likelihood (NLL), deviation from the real negative log-likelihood (D-NLL), root mean square error (RMSE) for the time interval, and accuracy of event type prediction (ACC). NLL and D-NLL show the model's goodness-of-fit and can be calculated as

$$\mathrm{NLL} = -\frac{1}{N}\sum_{i=1}^{N}\ell\left(e_i\right)$$

$$\mathrm{D\_NLL} = \mathrm{NLL}_{fit} - \mathrm{NLL}_{real}$$

where $\{e_i\}_{i=1}^{N}$ is the set of N event sequences, NLL_{fit} is the fitted negative log-likelihood and NLL_{real} is the true log-likelihood. The D-NLL closer to zero indicates better predictive performance. When the D-NLL cannot be calculated, the smaller the NLL, the better. Based on the maximum likelihood principle, the smaller the NLL, the higher the fitting performance of the model.

Further, we use the other three metrics to test the predictive performance. The RMSE and ACC are calculated as

$$\mathrm{NLL} = \frac{1}{N}\sum_{j=1}^{N}\sqrt{\frac{1}{L_j}\sum_{i=1}^{L_j}\left(t_i-\hat{t}_i\right)^2}$$

$$\mathrm{ACC} = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{L_j}\sum_{i=1}^{L_j}\alpha\left(c_i,\hat{c}_i\right)$$

where $\alpha\left(c_i,\hat{c}_i\right) = \begin{cases} 1, c_i = \hat{c}_i \\ 0, c_i \neq \hat{c}_i \end{cases}$, t_i is the moment of the event, $\hat{t}_i$ is the predicted value at the moment of the event, c_i is the true category and $\hat{c}_i$ is the predicted category. The RMSE closer to zero indicates better predictive performance, and the ACC closer to one indicates better predictive performance. Since the real models of the three real-world datasets are unknown, one cannot obtain the D-NLL. We, therefore, do not test the D-NLL on the real-world datasets.

4.3 Experiment settings

The proposed ICAHP-TR consists of the input layer, UT layer, and output layer. The input layer uses a fully connected layer with a hidden unit of 64 as the event embedding in the experiments. The UT layer is based on the open-source UT network, whose main hyperparameters include the batch size, hidden state size, number of self-attention heads, and number of time steps are listed in Table 2 and two dropout layers are added after the masked multi-head dot-product self-attention and the transition function layer with the probability of random discard set to 20%. In the proposed UT-embedded component, the maximum relative position of RPT proposed in this paper is set to 25 and the size of the ConvEnc convolution kernel is 2×2 with the initialization value set to one. The

Table 2 Hyper-parameters for training each dataset

Dataset	Batch size	Hidden state size	Num. heads	Num. time steps
Hawkes 1	32	32	8	6
Hawkes 2	32	32	8	6
Mooc	16	64	4	4
Wikipedia	4	32	4	4
Yelp Tornonto	2	16	4	2

Table 1 Datasets statistics. "# Category" means the number of categories

Dataset	# Category	Min length	Average length	Max length	Total sequence
Mooc	97	4	56	493	7047
Wikipedia	8,227	84	157	1936	1000
Yelp Toronto	1	424	717	2868	300

Table 3 The real NLLs of the two Hawkes processes

	Hawkes process 1	Hawkes process 2
Real-NLL	0.4231	− 0.0455

Table 4 NLL, D-NLL, and RMSE comparison in the synthetic datasets

Dataset	Model	NLL	D-NLL	RMSE
Hawkes 1	ICAHP-TR	**0.4279**	**0.0048**	**2.31**
	THP	− 0.1781	0.6012	2.7538
	UTHP	0.4347	0.0116	2.53
	SAHP	0.4412	0.0181	18.75
	LogNormMix	0.4512	0.0281	2.96
	FullyNN	0.4788	0.0557	2.48
	RMTPP	0.6538	0.2307	2.43
Hawkes 2	ICAHP-TR	**− 0.0362**	**0.0093**	2.54
	THP	− 0.8034	− 0.7579	2.76
	UTHP	− 0.0214	0.0241	2.65
	SAHP	− 0.3396	− 0.2941	23.83
	LogNormMix	− 0.0087	0.0368	2.56
	FullyNN	− 0.0058	0.0397	**2.48**
	RMTPP	0.6148	0.6603	25.85

In the table, the bold font indicates the best performance on all models

output layer consists of two fully connected layers. The number of units is determined by the number of likelihood function parameters and the number of categories respectively. The parameters of the model are initialized using the Kaiming method. We chose Adam as the optimizer, whose learning rate is based on the warm-up schedule with an initial learning rate of 5e–5. When the validation loss declining quantity is less than 1e−4 and more than 30 times, an early stop is adopted. The control of the above parameters is obtained by comparing the first 20 epochs of model training with the grid search method. Each dataset is split into a training set, validation set, and test set according to the ratio of 3/1/1. All the experiments are executed on a computer equipped with an Intel Xeon-5317 CPU @ 3.6 GHz, 64 GB of memory, and an NVIDIA RTX2080Ti GPU with 11 GB of memory. The model is the Pytorch version.

4.4 Comparisons and convergence evaluation

We compared our proposed approach in terms of fitting, prediction, and convergence with five benchmark models: RMTPP, FullyNN, LogNormMix, SAHP, THP, and UTHP. To make a fair comparison, we tried different hyperparameter configurations for the baseline models and chose the one with the best validation performance for all models. Below we will successfully present and analyze the results of comparative experiments on synthetic and real-world datasets.

(1) *Comparison in Synthetic Datasets* Table 3 shows the real NLLs of the two synthetic datasets. The experimental results of the test sets on the synthetic datasets are summarized in Table 4, and the visualization of D-NLLs is shown in Fig. 4. The D-NLL of our method in Table 4 is closest to zero, that is, the light blue point in Fig. 4 is closest to the red dashed line. The results show that our approach is inherently superior to the existing method using RNN and self-attention. Observing the distance of the point in Fig. 4 from the red dotted line, we can intuitively see that our approach has the best fitting performance (NLL, D-NLL) to the intensity function. It outperforms the second-best method by 74.10% on average, which significantly improves. The other benchmark models have worse fitting performance and are even under-fitting. This indicates that the benchmark models based on RNN or self-attention are hard to capture long-term dependencies and recursive dependencies simultaneously. In contrast, our model is competent because we achieved the best fitting performance on the synthetic dataset.

In testing the prediction of future arrival times, we use the RMSE. It can be seen from Table 4 that the prediction error (RMSE) of our method for the Hawkes1 dataset is lower than the benchmark models. The only regret is that the prediction effect on the Hawkes 2 dataset is slightly lower than that of FullyNN (an increase of 2.41%) due to the insufficient accuracy of the Monte Carlo simulation. This phenomenon

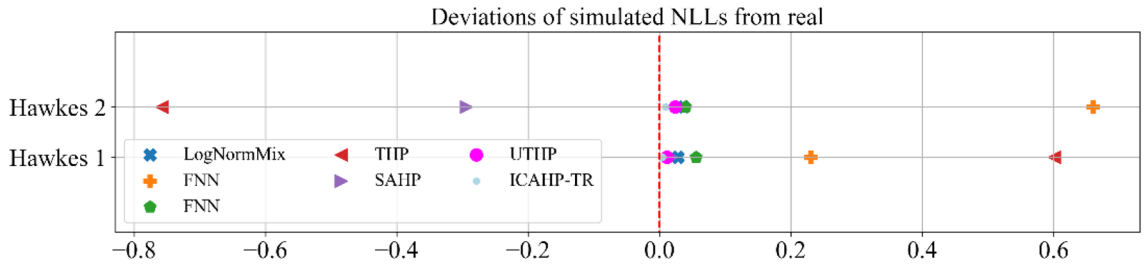

Fig. 4 Deviations of estimated NLLs. The abscissa of the vertical red line indicates zero

also exists in real-world datasets, and we will discuss possible reasons later. We, however, obtained suboptimal results. THP performs poorly in both Hawkes1 and Hawkes2 due to the inclusion of RMSE loss in its loss function, resulting in underfitting. RMTPP performs worse in Hawkes2 than Hawkes1 because large fluctuations have lasting effects, but RNN cannot capture long-term dependencies.

(2) *Comparison in real-world Datasets* The results of the test metrics on the test sets for the three real-world datasets are summarized in Table 5. Compared with benchmark models, our model achieves the best effectiveness on NLL and ACC. In particular, on Wikipedia dataset, we gain 26 times high accuracy in predicting event categories than any other method, which is an excellent result. This is because we have learned the hidden representation of historical information superior to the benchmark models. This is consistent with the results of the synthetic datasets. It shows that the goodness of fit of the proposed model is successfully satisfactory for different simulated or natural scenarios. Note that SAHP uses the method of calculating the sum of rectangular areas in a large range to calculate the integral, so when the data magnitude is large, the time prediction will have serious distortion.

Table 5 NLL, ACC, and RMSE comparison in the real-world datasets

Dataset	Model	NLL	ACC	RMSE
Mooc	ICAHP-TR	**8.0028**	**0.4179**	204,233
	THP	12.5117	0.0533	252,723
	UTHP	8.0924	0.3516	232,761
	SAHP	11.1188	0.1776	–
	LogNormMix	8.1848	0.3863	6,519,344
	FullyNN	8.0514	0.4067	**160,293**
	RMTPP	12.2653	0.3812	1,169,489
Wikipedia	ICAHP-TR	**17.7072**	**0.2044**	105,241
	THP	4787.8539	0.0076	210,151
	UTHP	19.6219	0.0123	176,512
	SAHP	18.5213	0.0000	–
	LogNormMix	18.0170	0.0075	161,017
	FullyNN	20.0090	0.0072	**85,785**
	RMTPP	19.5873	0.0075	955,220
Yelp Toronto	ICAHP-TR	**13.0543**		**545,139**
	THP	26.8383		6,814,144
	UTHP	13.2017		651,296
	SAHP	13.2263		–
	LogNormMix	13.0842		635,484
	FullyNN	15.7130		595,918
	RMTPP	13.3252		16,015,129

In the table, the bold font indicates the best performance on all models

The symbol '–' stands for RMSE results with numerical problems of SAHP, and whitespaces mean that these items have no results in the Wikipedia dataset

Our model performs best on Yelp Toronto but gets the second-best result of RMSE on the Mooc and Wikipedia datasets (The best results are all from FullyNN). Two reasons may cause this deficiency: (1) ICAHP-TR uses two numerical methods for time prediction, resulting in a decrease in accuracy; (2) since FullyNN uses the median to predict, while ICAHP-TR uses the expectation, this difference may lead to differences in the effectiveness; (3) FullyNN has a lower numerical error in finding the root of equations for the prediction. Further work will be considered to improve the accuracy of the time prediction.

(3) *Convergence Evaluation* To monitor training loss changes, we compare the NLL convergence curves with benchmarks in the two datasets (Hawkes 1, Mooc). Since the performance of the NLL convergence curves is similar in all datasets, we chose these two representative datasets to reduce redundancy. Figure 5 shows that the convergence epoch and descent rate of ICAHP-TR in the entire training process has a significant advantage over benchmarks. The reasonable explanation for this phenomenon lies in the superior ability of ICAHP-TR to learn the modal dynamics of sequential data information.

From the upper right of Fig. 5 (corresponding to the Hawkes1 dataset), the proposed model decreases from relatively minor losses and converges to the real NLL after less than 50 epochs. However, the benchmarks do not reach a stable fluctuation until after 50 epochs later. The remaining three graphs in Fig. 5 show the convergence of the Mooc dataset. Obviously, the proposed model's convergence curve is higher than that of other models (except SAHP and THP). Although SAHP and THP converge fast, they both fall into suboptimal solutions due to the defect that self-attention cannot recursively process.

4.5 Ablation study

We use NLL as an evaluation metric to conduct ablation experiments on the proposed model to quantify the contribution of each structure to performance improvement. We quantify the percentage increase in NLL versus the original architecture according to

$$\text{Percentage Increase} = \frac{NLL_{removed} - NLL_{original}}{NLL_{original}} \times 100\%,$$

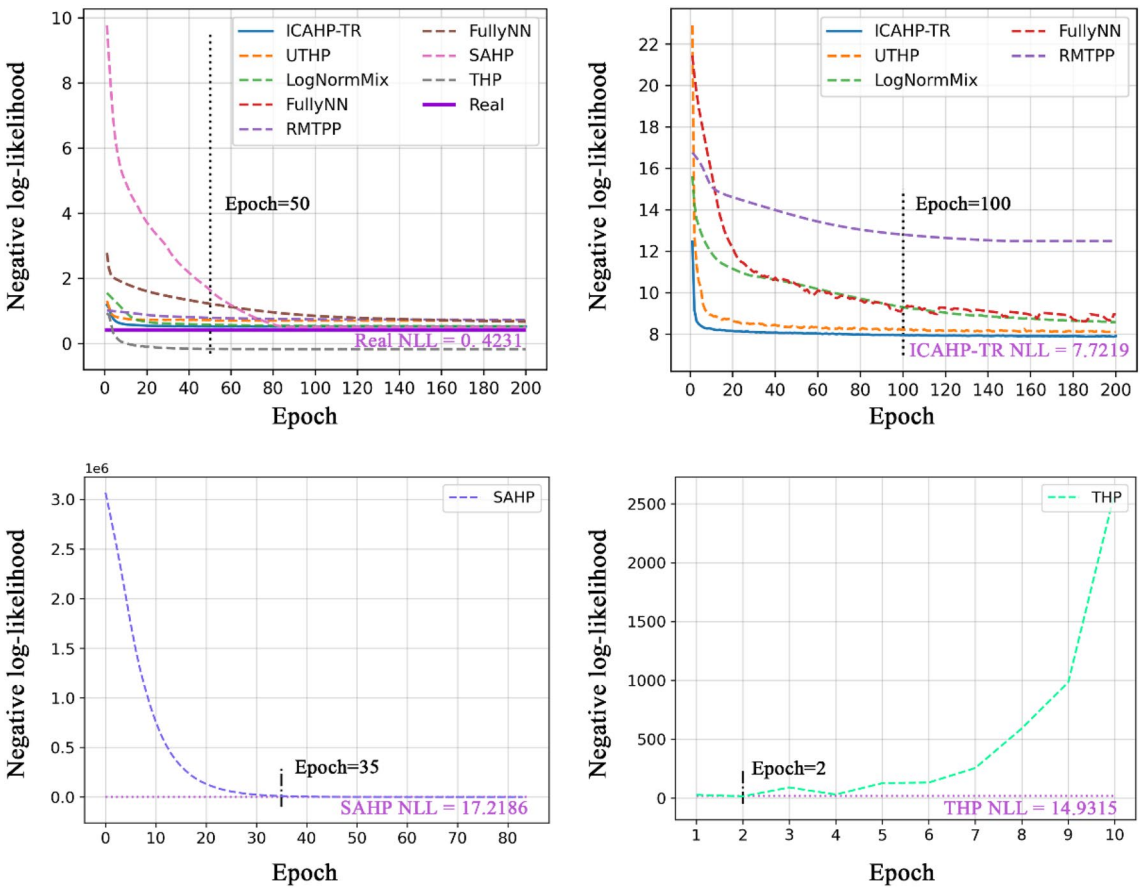

Fig. 5 Training curves of different methods fitted on Hawkes 1(upper left of the Fig) and Mooc (upper right, lower left, lower right of the Fig)

Table 6 Changes of NLL across ablation tests

	Hawkes 1	Hawkes 2	Mooc	Wikipedia	Yelp Toronto
UT + RPT + ConvEnc	0.4279	− 0.0362	7.7751	17.7071	13.0543
UT + RPT	0.4431	− 0.0243	7.8381	19.4396	13.1021
UT	0.4521	− 0.0191	8.0787	19.5411	13.2017
Self-attention	0.4797	− 0.0144	7.6431	19.5436	13.2197

as UT, RPT, and ConvEnc are removed from the original model one by one (as shown in Table 6 and Fig. 6). The smaller the percentage increase, the greater the contribution of the structure to the improvement of model performance. Note that when removing one component, we keep the other component hyperparameters invariable compared to the original model. The experimental results are summarized in Table 7.

Overall, the three components have improved the learning ability of the dynamic model. ConvEnc increases 0.83% on average, RPT increases 11.80% on average, and UT increases 5.09% on average. Specifically, while ConvEnc is critical in Wikipedia and Yelp Toronto, lower and negative ablation promotions in Hawkes1, Hawkes2 and Mooc show that it can be harmful on the short sequence datasets, i.e., RPT plays a more important role. One possible explanation is that RPT can learn simple stimulus laws, and ConvEnc has become an unnecessary burden. Compared with self-attention, UT has a greater effect on synthetic datasets, while its performance in real-world datasets is mediocre. This may be because there is an obvious recursive relationship between the synthetic datasets, indistinguishable from the three real-world datasets.

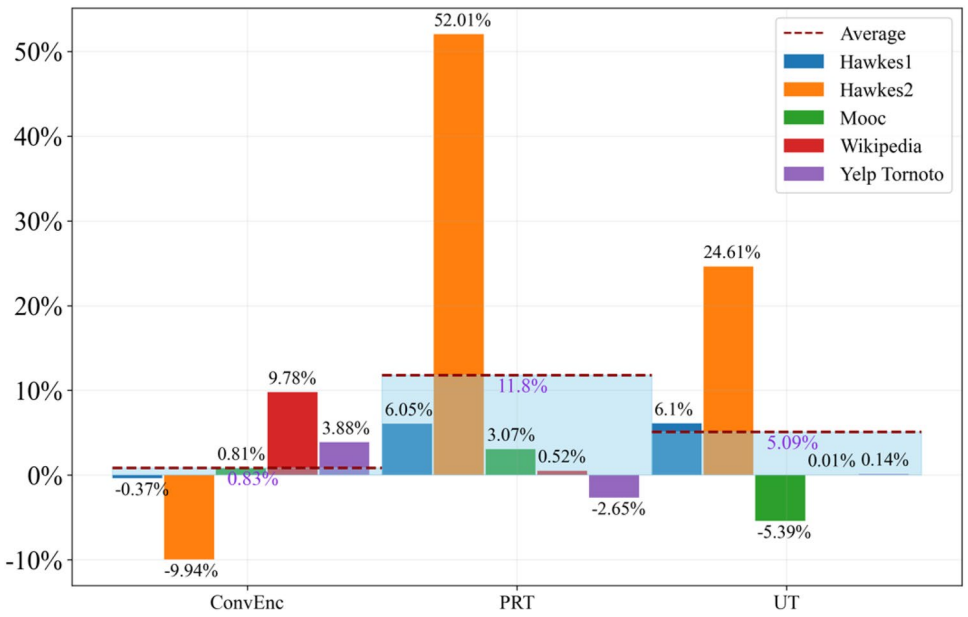

Fig. 6 The percentage increase in NLL after removing the components

Table 7 Percentage changes of NLL across ablation tests

	Hawkes 1 (%)	Hawkes 2 (%)	Mooc (%)	Wikipedia (%)	Yelp Toronto (%)	Average (%)
ConvEnc	– 0.37	– 9.94	0.81	9.78	3.88	0.83
RPT	6.05	52.01	3.07	0.52	– 2.65	11.80
UT	6.10	24.61	– 5.39	0.01	0.14	5.09

4.6 Interpretability analysis

The weight matrix of the self-attention mechanism provides a reference tool for the interpretability of the model. In this section, we visually illustrate the interpretability advantages of the weight matrix of ICAHP-TR from its heat map.

Figure 7 visualizes attention patterns of ICAHP-TR on synthetic datasets via the attention weight heat maps. Combining two datasets, we can find two commonalities: 1. the information at the initial time point gets more attention (Head 1 and Head 4 of Hawkes 1, Head 2 of Hawkes 2), which is reflected by the initial intensity; 2. the attention degree is high near the diagonal line, i.e., the successive time points because the impact of the event at the far time points on the intensity function at the current time points is exponentially attenuated. Moreover, the heat map also reflects the characteristics of both datasets: (1) the diagonals of Head 1 and Head 4 of Hawkes 1 are sparse, which means that when the occurrence frequency of events is high, certain events suffer from a sustaining impact on the follow-up, i.e., the excitation of this event on the intensity function does not have time to diminish; (2) events of Hawkes 2 occur at a low frequency, so previous historical time has a relatively uniform influence on subsequent events (the intensity function is shown as a relatively low value that is similar to *endogenous component*), which can be seen from Head 3 and Head 4. These situations are consistent with our intuitive understanding of the synthetic datasets. This manifests that we have indeed learned the dynamic pattern of the data.

The heat maps of learned attention patterns of ICAHP-TR on real-world datasets are shown in Fig. 8. In the scenario of the Mooc dataset, online class behavior is a series of continuous operations in a specific period, so the recent behaviors significantly impact the current actions. What is reflected in the attention weight is that the area to the left of the diagonal has a greater weight. Therefore, the learned weight matrix shown in the upper left of Fig. 8 fully meets the feature of the actual scene.

As for Wikipedia page editors, it is often necessary to correct or add content whenever online experts discover that past edits are wrong or that content needs to be updated. Moreover, there are delays in content updates or errors. For these cases, the model needs to learn about the medium- and long-term dependency, making the attention weight heat map relatively

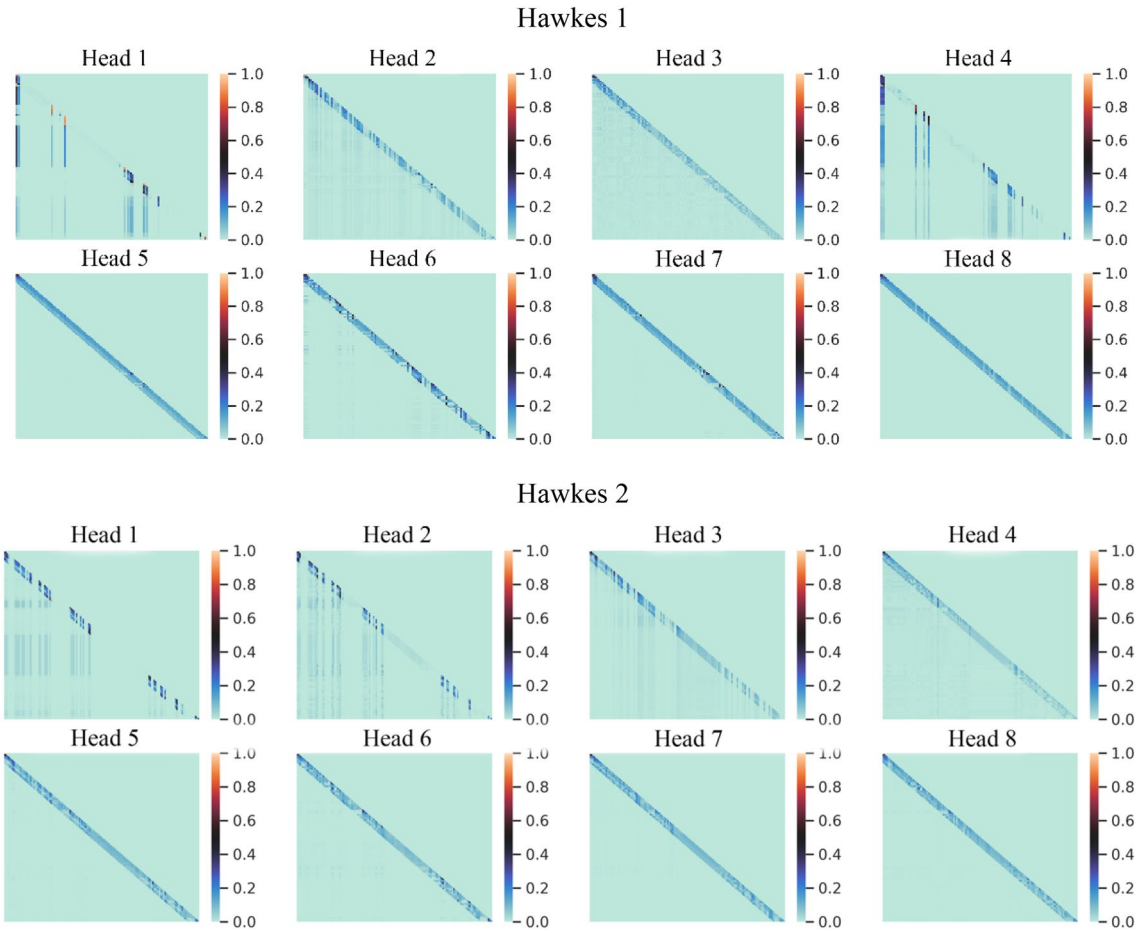

Fig. 7 Attention weight heat maps on synthetic datasets

sparse. This coincides with the embodiment shown in the upper right of Fig. 8.

Regarding restaurant traffic, returned customers tend to prefer the restaurant they consider satisfactory. Customers who regularly visit the restaurant are more likely to return later. In this way, the weight in the attention matrix will tend to be uniform, i.e., each patronage time point has the same impact on the current. This corresponds to Head 1, Head 2, and Head 3 in the lower of Fig. 8. Of course, the recent visits can continue strengthening the restaurant's love. This is reflected in Head 4.

The above analysis of the three real-world datasets shows that our model can be adapted to different actual scenarios and can reverse valuable conclusions from the heat map of the attention matrix.

5 Conclusion

This paper proposes the ICAHP-TR, which captures the global receptive field on historical information considering the recursive induction. We devise two tools for collaborative work, RPT, and ConvEnc, to achieve the input of sequence position on the time step and a more reasonable attention score weight. The former indicates that the time interval sequence information can be captured more efficiently and accurately. The latter enhances the influence of adjacent positions based on the relative position idea. Extensive experimental results suggest ICAHP-TR's superior fitting performance and excellent interpretability in synthetic and real-world datasets. Also, the effectiveness of the components has been verified in ablation experiments.

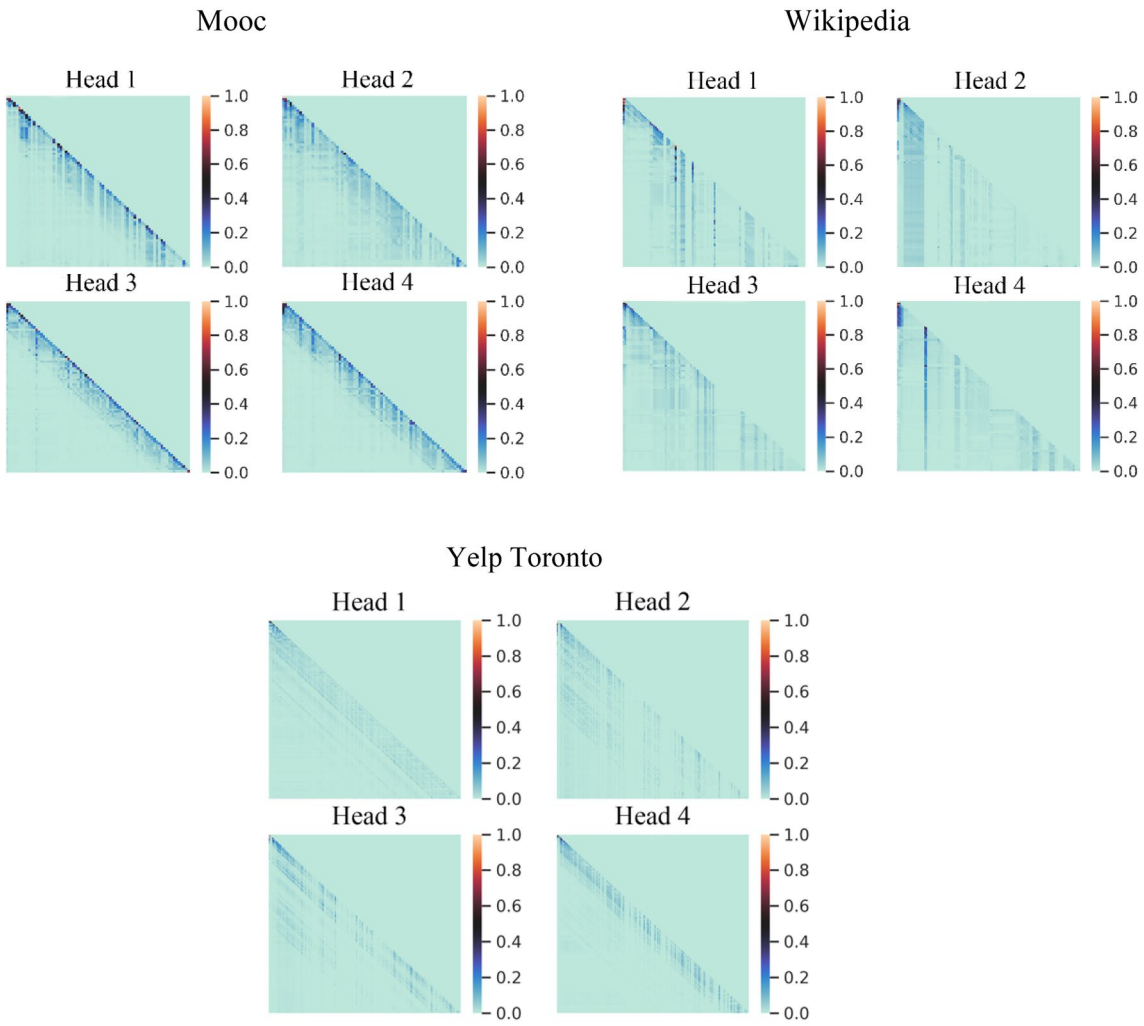

Fig. 8 Attention weight heat maps on real-world datasets

In future work, we will explore lowering the time prediction by reducing the numerical error caused by calculating the non-closed formal integral or selecting other reasonable statistics. In addition, avoiding the parameter growth problem caused by RPT and ConvEnc is a research route to achieve both effectiveness and fast calculation.

Acknowledgements This work was supported by the Applied Basic Research Programs of Shanxi Province (Grant no. 201901D211105).

Data availability The data that support the findings of this study are openly available in a public repository ifl-tpp at https://github.com/shchur/ifl-tpp.

References

1. Du N, Dai H, Trivedi R, Upadhyay U, Gomez-Rodriguez M, and Song L (2016) Recurrent marked temporal point processes: Embedding event history to vector. In: Proceedings of ACM SIGKDD International Conference on knowledge discovery and data mining, pp 1555–1564. https://doi.org/10.1145/2939672.2939875
2. Gao H, Huang T, Liu Y, Yin Y, Li Y (2022) PPO2: location privacy-oriented task offloading to edge computing using reinforcement learning for intelligent autonomous transport systems. IEEE Trans Neural Netw Learn Syst. https://doi.org/10.1109/TITS.2022.316-9421
3. Dai Z, Zhou H, Dong X (2020) Forecasting stock market volatility: the role of gold and exchange rate. AIMS Math 5(5):5094–5105. https://doi.org/10.3934/math.2020327
4. Gao H, Qiu B, Duran Barroso RJ, Hussain W, Xu Y, Wang X (2022) TSMAE: a novel anomaly detection approach for internet of things time series data using memory-augmented autoencoder. IEEE Trans Netw Sci Eng. https://doi.org/10.1109/TNSE.2022.3163144
5. Tang L, Zheng S, Zhou Z (2018) Estimation and test of restricted linear EV model with nonignorable missing covariates. Appl Math-A J Chin Univ 33(3):344–358. https://doi.org/10.1007/s11766-018-3550-8
6. Zhou Z, Tang L (2019) Testing for parametric component of partially linear models with missing covariates. Stat Pap 60(3):747–760. https://doi.org/10.1007/s00362-016-0848-6

7. Tan Z, Zheng S (2020) Extremes of a type of locally stationary Gaussian random fields with applications to Shepp statistics. J Theor Probab 33(4):2258–2279. https://doi.org/10.1007/s10959-019-00953-6
8. Chen Y, Tan Z (2019) Almost sure limit theorem for the order statistics of stationary Gaussian sequences. Filomat 32(9):3355–3364. https://doi.org/10.2298/FIL1809355C
9. Alan GH (1971) Spectra of some self-exciting and mutually exciting point processes. Biometrika 58:83–90. https://academic.oup.com/biomet/article/58/1/83/224809. Accessed 28 May 2021
10. Omi T, Ueda N, Aihara K (2019) Fully neural network-based model for general temporal point processes. arXiv preprint arXiv:1905.09690
11. Shchur O, Biloš M, Günnemann S (2019) Intensity-free learning of temporal point processes. arXiv preprint arXiv:1909.12127
12. Zhang Q, Lipani A, Kirnap O, and Yilmaz E (2019) Self-attentive Hawkes processes. arXiv preprint arXiv:1907.07561, 2019.
13. Zuo S, Jiang H, Li Z, Zhao T, Zha H (2020) Transformer Hawkes process. In: Proceedings of the 37th International Conference on machine learning, vol 119, pp 11692–11702
14. Gao H, Xu K, Cao M, Xiao J, Xu Q, Yin Y (2021) The deep features and attention mechanism-based method to dish healthcare under social IoT systems: an empirical study with a hand-deep local-global net. IEEE Trans Netw Sci Eng 9(1):336–347. https://doi.org/10.1109/TCSS.2021.3102591
15. Xiao J, Xu H, Gao H, Bian M, Li Y (2021) A weakly supervised semantic segmentation network by aggregating seed cues: the multi-object proposal generation perspective. ACM Trans Multimed Comput Commun Appl 17(1s):1–19. https://doi.org/10.1145/3-419842
16. Huang C, Liu B, Qian C, Cao J (2021) Stability on positive pseudo almost periodic solutions of HPDCNNs incorporating D operator. Math Comput Simul 190(2021):1150–1163. https://doi.org/10.1016/j.matcom.2021.06.027
17. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, and Polosukhin I (2017) Attention is all you need. In: Proceedings of the 31st lnternational Conference on Neural lnformation Processing Systems, pp 6000–6010
18. Bengio Y (1994) Learning long-term dependencies with gradient descent difficult. IEEE Trans Neural Netw 5(2):157–166. https://ieeexplore.ieee.org/document/279181. Accessed 3 Apr 2021
19. Dehghani M, Gouws S, Vinyals O, Uszkoreit J, Kaiser U (2019) Universal transformers. In: Proceedings of the International Conference on learning representations, OpenReview.net. https://openreview.net/forum?id=HyzdRiR9Y7. Accessed 1 Apr 2021
20. Shaw P, Uszkoreit J, Vaswani A (2018) Self-attention with relative position representations. arXiv preprint arXiv:1803.02155
21. Su J, Wang Z, Chen M (2020) Orthogonal exponential functions of the planar self-affine measures with four digits. Fractals. https://doi.org/10.1142/S0218348X20500164
22. Li J, Li P (2018) Inverse elastic scattering for a random source. SIAM J Math Anal 51(6):4570–4603. https://doi.org/10.1137/18M1235119
23. Mei H, Eisner JM (2017) The neural Hawkes process: a neurally self-modulating multivariate point process. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp 6754–6764
24. Lin Z, Feng M, Santos C, Yu M, Xiang B, Zhou B, Bengio Y (2017) A structured self-attentive sentence embedding. arXiv preprint arXiv:1703.03130
25. Gao H, Xiao J, Yin Y, Liu T, Shi J (2022) A mutually supervised graph attention network for few-shot segmentation: the perspective of fully utilizing limited samples. IEEE Trans Neural Netw Learn Syst. https://doi.org/10.1109/TNNLS.2022.3155486
26. Zhang L, Liu J, Song Z, Xin Z (2021) Universal Transformer Hawkes process with adaptive recursive iteration. Eng Appl Artif Intell. https://doi.org/10.1016/j.engappai.2021.104416
27. Guo R, Li J, Liu H (2018) Initiator: noise-contrastive estimation for marked temporal point process. In Proceedings of the International Joint Conference on artificial intelligence, pp 2191–2197. https://doi.org/10.24963/ijcai.2018/303
28. Xiao S, Farajtabar M, Ye X, Yan J, Song L, and Zha H (2017) Wasserstein learning of deep generative point process models. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp 3247–3257
29. Xiao S, Xu H, Yan J, Farajtabar M, Yang X, Song L, Zha H (2018) Learning conditional generative models for temporal point processes. In: Proceedings of the 32nd AAAI Conference on artificial intelligence, vol 32(1), pp 6302–6310
30. Li S, Xiao S, Zhu S, Du N, Xie Y, Song L (2018) Learning temporal point processes via reinforcement learning. In: Proceedings of the 32nd Conference on Neural Information Processing Systems, pp 10781–10791
31. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
32. Ba JL, Kiros JR, and Hinton GE (2014). Layer normalization. arXiv preprint arXiv:1607.06450, 2016
33. Graves A, Wayne G, Danihelka I (2014) Neural turing machines. arXiv preprint arXiv:1410.5401
34. Liu Z, Zhou Y, Zhang Y (2020) On inexact alternating direction implicit iteration for continuous Sylvester equations. Numer Linear Algebra Appl. https://doi.org/10.1002/nla.2320
35. Zhou W, Zhang L (2020) A modified Broyden-like quasi-Newton method for nonlinear equations. J Comput Appl Math. https://doi.org/10.1016/j.cam.-2020.112744
36. Ogata Y (1981) On Lewis' simulation method for point processes. IEEE Trans Inf Theory IT-27(1):23–31
37. Kumar S, Zhang X, Leskovec J (2019) Predicting dynamic embedding trajectory in Temporal Interaction Networks. In: Proceedings of the ACM SIGKDD International Conference, pp 1269–1278. https://doi.org/10.1145/3292500.3330895.