Dynamic Hawkes Processes for Discovering Time-evolving Communities’ States behind Diffusion Processes

Maya Okawa1,3, Tomoharu Iwata2, Yusuke Tanaka2, Hiroyuki Toda1
Takeshi Kurashima1, Hisashi Kashima1
1NTT Service Evolution Labs., NTT Corporation
{maya.okawa.af,hiroyuki.toda.xb,takeshi.kurashima.uf}@hco.ntt.co.jp
2NTT Communication Science Labs., NTT Corporation
{tomoharu.iwata.gy,yusuke.tanaka.rh}@hco.ntt.co.jp
3Department of Intelligence Science and Technology, Kyoto University
kashima@i.kyoto-u.ac.jp

ABSTRACT
Sequences of events including infectious disease outbreaks, social network activities, and crimes are ubiquitous and the data on such events carry essential information about the underlying diffusion processes between communities (e.g., regions, online user groups). Modeling diffusion processes and predicting future events are crucial in many applications including epidemic control, viral marketing, and predictive policing. Hawkes processes offer a central tool for modeling the diffusion processes, in which the influence from the past events is described by the triggering kernel. However, the triggering kernel parameters, which govern how each community is influenced by the past events, are assumed to be static over time. In the real world, the diffusion processes depend not only on the influences from the past, but also the current (time-evolving) states of the communities, e.g., people’s awareness of the disease and people’s current interests. In this paper, we propose a novel Hawkes process model that is able to capture the underlying dynamics of community states behind the diffusion processes and predict the occurrences of events based on the dynamics. Specifically, we model the latent dynamic function that encodes these hidden dynamics by a mixture of neural networks. Then we design the triggering kernel using the latent dynamic function and its integral. The proposed method, termed DHP (Dynamic Hawkes Processes), offers a flexible way to learn complex representations of the time-evolving communities’ states, while at the same time it allows to computing the exact likelihood, which makes parameter learning tractable. Extensive experiments on four real-world event datasets show that DHP outperforms five widely adopted methods for event prediction.

CCS CONCEPTS
• Mathematics of computing → Time series analysis; • Human-centered computing → Social network analysis.

KEYWORDS
Hawkes processes; Event prediction; Neural networks

1 INTRODUCTION
Various social phenomena can be described by diffusion processes among multiple communities. For example, infectious diseases like COVID-19 are transmitted from one county to another, leading to a worldwide pandemic [35]. Information such as opinions, news, and articles are shared and disseminated among online communities, e.g., user groups in social networks, news websites, and blogs. Such diffusion phenomena are recorded as multiple sequences of events, which indicate when and in which community the event occurred. Understanding the diffusion mechanism and predicting future events are crucial for many practical applications across domains. For example, policymakers would be able to design prompt and appropriate interventions to curb the spread of disease given a better understanding of the mechanisms behind the transmission and more reliable predictions.

Temporal point processes provide an elegant mathematical framework for modeling event sequences. In these methods, the probability of event occurrences is determined by the intensity function. Hawkes process is an important class of point processes for modeling diffusion processes. These models use the triggering kernel to characterize diffusion processes and estimate its parameters via maximum likelihood. The triggering kernel encodes the magnitude and speed of influence from the past events, namely, how likely and quickly the past events in one community (i.e., “source” community) will affect the occurrence of a particular event in another community (i.e., “target” community). Hawkes process and its variants have been applied in diverse areas, from epidemic modeling [16] to social network analysis [12, 30, 41]. However, they have focused on learning the static influence of the past events on the current event, thereby largely overlooking the factor of time-evolution. In reality, the diffusion processes depend not only on the influences from the past but also on the current state of the target communities. For example, the outbreaks of infectious diseases in one community...
(e.g., country) can also be driven by people’s awareness of the disease in each community (country) and their preventive behaviors which can constantly change over time, on top of the past record of the disease occurrence. As another example, information diffusion heavily depends on ongoing peoples’ interests in the target community (e.g., online user group). In particular, the spread of information to one target community (user group) is strengthened when an topic deemed to be important by the target community emerges in the online space; while it is weakened in accordance with a gradual loss in peoples’ interest in the topic.

A few studies have considered the underlying dynamics of such “states” in communities. For instance, the SIR-Hawkes model [31] redesigned the triggering kernel of the Hawkes process by incorporating the recovered (immune) population dynamics over the course of the pandemic. Kobayashi et al. [18] proposed a time-dependent triggering kernel that varies periodically in time for modeling daily cycles of human activity. However, these approaches rely on hand-crafted functions for describing the latent dynamics of states and so demand expert domain knowledge. Moreover, they may not be flexible enough to accommodate the complexity and heterogeneity of the real world. In fact, in many practical applications the complete set of factors is largely unknown and thus difficult to model through restricted parametric forms. Taking information diffusion as an example, the time-evolution of peoples’ interest on a given topic is generally unknown and not directly observable.

A potential solution is to directly model the triggering kernel parameters using a flexible function of time (e.g., neural network). Alas, naively employing this approach makes parameter learning intractable since the log-likelihood of Hawkes processes involves the integral of the triggering kernel. Computing the integral of the triggering kernel in combination with the neural network is generally infeasible.

In this paper, we propose a novel Hawkes process model referred to as DHP (Dynamic Hawkes Process) which automatically learns the underlying dynamics of the communities’ states behind the diffusion processes in a manner that allows tractable learning. We introduce the latent dynamics function for each community that represents its hidden dynamic states. Our core idea is to extend the triggering kernel by combining it with the latent dynamics function and its integral. Specifically, we model the magnitude of diffusion by the latent dynamics function and the speed of diffusion by the integral of the latent dynamics function. This design choice offers two benefits. First, the resulting triggering kernel can be expressed as a product of two components: composite function with the “inner” function being the integral of the latent dynamics function and the “outer” function being the basic triggering kernel; and the derivative of the inner function of that composite function (i.e., the latent dynamics function). Hence, by applying the substitution rule for definite integrals (i.e., the chain rule in reverse), we can obtain a closed-form solution for the integral of the triggering kernel involved in the log-likelihood. Second, it allows capture of the simultaneous changes of magnitude and speed of the diffusion as they are related through the latent dynamics function and its integral, which is desirable for many applications. For example, in the context of disease spread, active preventative measures can reduce both the magnitude and the speed of the infection. To model the integral of the latent dynamics function, we utilize and extend a monotonic neural network [6, 33]. This formulation enables DHP to learn flexible representations of the community state dynamics that underlie the diffusion processes. It should be noted that DHP can be easily extended to capture the time-evolving relationships between communities, by introducing the latent dynamics function for pairs of communities. In this work, we adopt DHP to demonstrate the hidden state dynamics of individual communities.

The main contributions of this paper are as follows:

- We propose a novel Hawkes process framework, DHP (Dynamic Hawkes Process) for modeling diffusion processes and predicting future events. The proposal, DHP, is able to learn the time-evolving dynamics of community states behind the diffusion processes.
- We introduce latent dynamics function; it reflects the hidden community dynamics and design the triggering kernel of the Hawkes process intensity using the latent dynamics function and its integral. The resulting model is computationally tractable and flexible enough to approximate the true evolution of the community states underlying the diffusion processes.
- We carry out extensive experiments using four real-world event datasets: Reddit, News, Protest, and Crime. The results show that DHP outperforms the existing works. Case studies demonstrate that DHP uncovers the hidden state dynamics of communities which underlie the diffusion processes by the latent dynamic function (See Figure 1).
2 RELATED WORK

With the evolution of data collection technology, extensive event sequences with precise timestamps are becoming available in an array of fields such as public health and safety [20, 22, 39], economics and finance [2, 5, 14], communications [12], reliability [3, 34, 36], and seismology [25–27]. Temporal point processes provide a principled theoretical framework for modeling such event sequences, in which the occurrences of events are determined by the intensity function.

Classical examples of temporal point processes include reinforced Poisson process [29], self-correcting point process [15], and Hawkes process [13]. The Reinforced Poisson process [29] considers the cumulative count of past events and a time-decreasing trend, and has been recently applied for predicting online popularity [32]. The intensity of the self-correcting point process [15] increases steadily and this trend is corrected by past observed events. Although these models have been widely used, they are not suitable for modeling diffusion processes between communities as they cannot explicitly model the influence of the past events underlying diffusion processes. Hawkes process [13] explicitly models the influence of the past events and captures triggering patterns between events (i.e., diffusion processes). Hawkes processes have been proven effective for modeling diffusion processes, including earthquakes and aftershocks [22], near-repeat patterns of crimes [22], financial transactions [2, 11, 14], online purchases [8, 10, 38, 40], and information cascades [12, 30, 41].

Recent studies employ neural network architectures to model point process intensity. In [9], the authors design the intensity using RNN. Omi et al. [28] extended our work by combining it with a monotonic neural network. Compared to classical point process methods, RNN-based models provide a more flexible way to handle the complex dependencies between events. However, the above methods focus on learning the triggering patterns of diffusion processes, i.e., influences from past events, and disregard the current (time-evolving) states of the communities. In the real world, event occurrences largely depend on the current community states (e.g., people’s awareness of the disease, ongoing people’s interest), which can evolve over time, as well as the past.

Several studies incorporate the time-variant dynamics of the community states behind diffusion processes into the Hawkes process formulation. For instance, the SIR-Hawkes model [31] considers recovered (immune) population dynamics to enhance the prediction of infectious disease events over the course of pandemic. Kobayashi et al. [18] proposed a time-dependent Hawkes process that accounts for the circadian and weekly cycles of human activity. Navaroli et al. [23] used nonparametric estimation to learn cyclic human activities underlying digital communications. All of the above methods, unfortunately, rely on a domain expert’s knowledge to elucidate the dynamics of the communities states behind diffusion processes. Such dynamics are often quite complex and remain unexplored in many practical applications.

Different from the existing methods, our proposed method both incorporates the temporal dynamics of communities’ states and the past influences.

3 PRELIMINARIES

This section provides the general framework of point processes on which our work is built, and the formal definition of the event prediction problem studied in this paper.

3.1 Hawkes Processes

Point process is a random sequence of events occurring in continuous time \( \{t_1, t_2, \ldots, t_I\} \), with \( t_i \in [0, T] \). Point processes are fully determined by “intensity” function \( \lambda(t) \). Given the history of events \( \mathcal{H}(t) \) up to time \( t \), the intensity is defined as

\[
\lambda(t) \equiv \lim_{\Delta t \to 0} \frac{\mathbb{E}[N(t + \Delta t) - N(t) | \mathcal{H}(t)]}{\Delta t},
\]

where \( N(t) \) is a number of events falling in \([0, t]\), \( \Delta t \) is a small time interval, and \( \mathbb{E} \) is an expectation. The intensity value \( \lambda(t) \) at time \( t \) measures the probability that an event occurs in the infinitesimal time interval \([t, t + \Delta t] \) given past events \( \mathcal{H}(t) \).

Hawkes process [13] is an important class of point processes, and can describe self-exciting phenomena. The intensity of Hawkes process is defined as

\[
\lambda(t) = \mu + \sum_{j: t_j < t} g(t - t_j),
\]

where \( \mu \geq 0 \) is a background rate and \( g(\cdot) \geq 0 \) is a triggering kernel encoding the augmenting or attenuating effect of past events on current events. Intuitively, each event at time \( t_j \) elevates the occurrence rate of events at time \( t \) by the amount \( g(t-t_j) \) for \( t > t_j \).

The univariate Hawkes process can be extended to multivariate Hawkes Process (MHP) to handle the mutual excitation of events (i.e., diffusion) among different communities (denoted by dimensions). Suppose we have \( I \) historical observations \( \mathcal{D} = \{(t_i, m_i)\}_{i=1}^I \) with time \( t_i \in [0, T] \) and community \( m_i \in \{1, \ldots, M\} \). In our setting, the communities indicate countries, city districts, online user groups or news websites. For an \( M \)-dimensional multivariate Hawkes process, the intensity of the \( m \)-th dimension takes the following form:

\[
\lambda_m(t) = \mu_m + \sum_{j: t_j < t} g_{m,m_j}(t-t_j),
\]

where \( \mu_m \) is the background rate of dimension \( m \) and \( g_{m,m_j}(\cdot) \geq 0 \) is the triggering kernel that captures the impact of an event in community \( m_j \) on the occurrence of an event in community \( m \). The typical choice for the triggering kernel is the exponential memory kernel, which is defined by \( g_{m,m_j}(\Delta_j) = \alpha_{m,m_j} \exp(-\beta_{m,m_j} \Delta_j) \), where \( \Delta_j \) represents the time interval \( \Delta_j = t - t_j \), \( \alpha_{m,m_j} \) quantifies the magnitude of the influence from community \( m_j \) on the event occurrence in community \( m \), and \( \beta_{m,m_j} \) controls how quickly its effect decays in time (i.e., speed of the diffusion). Other candidates include power law kernel [26], Raleigh kernel [37], and log-normal distribution [23].

The negative log-likelihood function of a multivariate Hawkes process over time interval \([0, T]\) is given by:

\[
\mathcal{L} = \sum_{i=1}^I \log \lambda_{m_i}(t_i) - \sum_{m=1}^M \int_0^T \lambda_m(t) dt.
\]
3.2 Problem Definition

The key notations used in the paper are listed in Table 3 of Appendix A. An event is represented by the pair \((t, m)\), where \(t\) and \(m\) denote time and community (e.g., country, news website) where the event happened, respectively. An event sequence is defined as the set of events \(\mathcal{D} = \{(t_i, m_i)\}_{i=1}^{I}\) with \(t_i \in [0, T]\), where \(I\) denotes the number of events that have occurred up to time \(T\).

Event Prediction Problem. Given the event sequence \(\mathcal{D}\) in the observation time window \([0, T]\), we aim to leverage \(\mathcal{D}\) to predict the number of events within any given time period; and the event times in the future time window \([T, T+\Delta T]\).

4 DYNAMIC HAWKES PROCESSES

In this section, we present DHP (Dynamic Hawkes Process), a novel multivariate Hawkes process framework for event prediction; it can learn the time-evolution of the communities underlying the diffusion processes. Figure 2 illustrates DHP. We design the triggering kernel of DHP intensity (panel A in Figure 2) as the product of two components: the triggering kernel with the input of time-rescaled events (panel B in Figure 2), which learns the decay influence from the past events; and the latent dynamics function (panel C) to adjust the magnitude of the influence from the past events. The latent dynamics function describes the time-evolving states of the communities (indicated by dimensions). In the context of disease spread, the latent dynamics function represents the dynamics of people’s awareness of the disease in each country. For information diffusion, it characterizes the temporal evolution of readers’ interests in news websites. We elaborate on the formulation of DHP in §4.1, followed by parameter learning (§4.2). The prediction procedure is described in Appendix C.

4.1 Model Formulation

The proposed model specifies the intensity of Hawkes process for dimension \(m\) as

\[
\lambda_m(t) = \mu_m + \sum_{j:j \preceq t} g_{m,m_j}(\hat{\lambda}_j) f_m(t),
\]

where \(\mu_m\) is the background rate for the \(m\)-th dimension (i.e., community), \(g_{m,m_j}(\cdot)\) is any chosen triggering kernel between dimension \(m\) and dimension \(m_j\) such as exponential memory kernel or log-normal distribution, and \(\hat{\lambda}_j\) is the time-rescaled or transformed time interval between the current time \(t\) and the time of \(j\)-th event \(t_j\). \(f_m(t) \geq 0\) represents the dynamics of the \(m\)-th community underlying the diffusion processes at time \(t\), which controls the magnitude of diffusion. The transformed time interval \(\Delta_j\) is defined by the integral of the latent dynamics function between \(t_j\) and \(t\) as follows:

\[
\hat{\lambda}_j = \int_{t_j}^{t} f_m(\tau)d\tau = F_m(t) - F_m(t_j),
\]

where \(F_m(t)\) denotes the integral function of the continuous-time dynamics \(f_m(t)\), that is \(F_m(t) = \int_{0}^{t} f_m(\tau)d\tau\). The above formulation can be understood by considering an analogy drawn from the time-scaling theorem [4]. Intuitively, this transformation adjusts the influence of each event by stretching or shrinking time based on the value of the latent dynamics function \(f_m(t)\). When \(f_m(t) < 1\), the interval times are lengthened so that event times are further separated. Likewise, when \(f_m(t) > 1\), the interval times are compressed so that events are drawn closer together. If \(f_m(t) = 1\) for all \(t > 0\), \(\hat{\lambda}_j = \int_{t_j}^{t} 1d\tau = t - t_j\), and Equations 5 and 6 reduce to a simple multivariate Hawkes process (Equation 3). This formulation assumes that the speed of diffusion varies according to the temporal dynamics of the target community \(m\), which is captured by \(f_m(t)\). This assumption is realistic; for instance, disease spread is controlled by people’s awareness of the disease in each country and their preventive behaviors. Information diffusion is largely influenced by the reader’s interest in each news website. It is worth mentioning that the latent dynamic function can be easily extended to consider the dynamics of pairwise interactions between dimensions, by redefining the latent dynamics function as \(f_{m,m_j}(t)\). The following discussion holds even under this extension.

Our formulation allows considering the latent state of each dimension \(m\) at current time \(t\) as well as the influence from the past events. Also, it can capture the simultaneous changes in diffusion magnitude and speed, which is desirable for many applications (as discussed in the following paragraph). Most importantly, it enables us to compute the analytic integral of the intensity, which is required for evaluating the log-likelihood (further discussion can be found in §4.2) and predicting the number of future events (See Appendix C).

Triggering Kernel. The triggering kernel can have many forms. For example, we can assume the exponential memory kernel for \(g_{m,m_j}(\cdot)\), i.e.,

\[
\lambda_m(t) = \mu_m + \sum_{j:j \preceq t} f_m(t)\alpha_{m,m_j} \exp\left(-\beta_{m,m_j}(F_m(t) - F_m(t_j))\right),
\]

where \(\alpha_{m,m_j}\) encompasses the magnitude of the static interaction between the \(m\)-th and \(m_j\)-th dimension; and \(\beta_{m,m_j}\) weights the
decay of the influence over time. Notice that, the above formulation relies on the implicit assumption that the magnitude and speed of diffusion are related through the latent dynamics function $f_m(t)$, which controls the magnitude of diffusion; its integral $F_m(t)$ governs the speed of diffusion. For example, when $f_m(t) = 2$ for every $t$, the second term of Equation 7 is $2\alpha \exp(-2\beta \Lambda_j)$, where $\Lambda_j = t_j - t_j$. When $f_m(t) = 0.5$ for every $t$, it is $0.5\alpha \exp(-0.5\beta \Lambda_j)$. This assumption is reasonable since the magnitude and speed of diffusion vary simultaneously in many cases. Taking disease transmission as an example, active prevention measures can both reduce the magnitude and the speed of the infection. The influence on the magnitude of diffusion from the latent dynamics function is tuned by $\alpha$, and the influence on the speed of diffusion from its integral is tuned by $\beta$.

**Latent dynamics function.** The design of $f_m(\cdot)$ is flexible to so any non-negative function can be used. Inspired by [28], we utilize and extend a monotonic neural network [6, 33] that learns a strictly monotonic function to design the latent dynamics function. Concretely, we model the integral function $F_m(t)$ using the monotonic neural network. This guarantees that its derivative (i.e., the latent dynamics function $f_m(t)$) is strictly non-negative, so intensity $\lambda_m(t)$ results in a non-negative function. In describing the integral function we propose to further enhance the expressiveness of the monotonic neural network by using a mixture of monotonic neural networks. Formally,

$$F_m(t) = \sum_{c=1}^{C} \pi_c \Phi_m^c(t) + b_0 t,$$

where $C$ is the number of mixture components, $\Phi_m^c(\cdot)$ is the $c$-th monotonic neural network, $\pi_c$ is the mixture weight of the $c$-th component, and $b_0$ is a bias parameter for the output layer. To preserve monotonicity of the integral of the latent dynamics function $F_m(t)$, we impose non-negative constraints on the mixture weights $\{\pi_1, \ldots, \pi_C\}$ and parameter $b_0$. For each dimension, we construct $L$ fully connected neural layers with monotonic activation functions. Whenever the context is clear, we simplify notation $\Phi_m^c(\cdot)$ to $\Phi(\cdot)$. At each layer $l \in \{1, 2, \ldots, L\}$ of the monotonic neural network, the hidden-state vector $h^{(l)}$ is given by

$$h^{(l)} = \sigma(W^{(l)} h^{(l-1)} + b^{(l)}),$$

where $W^{(l)}$ and $b^{(l)}$ are parameter matrix and vector to learn for $l$-th layer, respectively. The input of the first layer is time $t$: $h^{(0)} = t$. $\sigma(\cdot)$ is a monotonic non-linear function. Following the previous work [6, 33], we use tanh activation for hidden layers and softplus for the last layer. The output of the monotonic neural network is $\Phi(t) = B h^{(L)}$, where $B$ is a learnable weight matrix. The weight parameter matrices $W^{(l)}$ and $B$ are imposed to be non-negative. The latent dynamics function $f_m(t)$, which is the derivative of the monotonic neural network $F_m(t)$, takes the following form,

$$f_m(t) = \sum_{c} \pi_c \Phi_m^c(t) + b_0,$$

where $\Phi_m^c(t)$ is the gradient of the monotonic neural network $\Phi_m^c(\cdot)$ with respect to time $t$, namely $\Phi_m^c(t) = d\Phi_m^c(t)/dt$. The gradient $\Phi_m^c(t)$ can be obtained by applying the automatic differentiation implemented in deep learning frameworks such as TensorFlow. As we place no restriction on the parametric forms of the community dynamics underlying the diffusion processes, our model can fit various complex dynamics of each community’s state.

This design choice enables us to automatically learn unknown complex dynamics of the communities’ states behind the diffusion processes, while at the same time allowing us to compute the exact log-likelihood for training as described in § 4.2.

### 4.2 Parameter Learning

Given the history of events up to but not including $T$, $\mathcal{D} = \{(t_i, m_i)\}_{i=1}^T$, we learn all the parameters of DHP by minimizing the negative log-likelihood of the observed event sequences. Specifically, we simultaneously estimate the neural network weights and the kernel parameters $\{\mu, A, \beta\}$: the background rates $\mu = \{\mu_1, \ldots, \mu_M\}$, the matrix of interactions $A = (\alpha_{ij}) \in \mathbb{R}^{M \times M}$, and the decay rates $\beta = \{\beta_1, \ldots, \beta_M\}$. The negative log-likelihood function of DHP over time interval of $[0, T)$ is obtained by substituting Equation 5 into Equation 4:

$$L = \sum_{i=1}^{T} \log \left( \mu_{i} + \sum_{j \neq i} g_{m_i, m_j} (f_m(t_i) - F_m(t_j)) f_m(t_j) \right)$$

$$- \sum_{m=1}^{M} \left[ \mu_{m} (t_i - t_{i-1}) + \int_{t_{i-1}}^{t_i} g_{m, m_j} (f_m(t) - F_m(t_j)) f_m(t) dt \right].$$

The problem here is to obtain integral $A_i$ in the last term, which reduces to

$$A_i = \sum_{j \neq i} \int_{t_{i-1}}^{t_i} g_{m, m_j} (f_m(t) - F_m(t_j)) f_m(t) dt.$$

Notice that the integrand of the above integral can be regarded as the product of composite function $g_{m, m_j} (f_m(t))$ and the derivative $f_m(t)$ of the “inner” part of that composite function. Hence, we can solve the integral of Equation 12 in closed form by applying a substitution (also called ”the reverse chain rule“) technique. Making the substitution $x = f_m(t) - F_m(t_j)$ gives $dx = f_m(t) dt$. Changing variables from $t$ to $x$, the integral of Equation 12 becomes,

$$\int_{t_{i-1}}^{t_i} g_{m, m_j} (f_m(t) - F_m(t_j)) f_m(t) dt$$

$$= \int_{F(t_{i-1}) - F(t_j)}^{F(t_i) - F(t_j)} g_{m, m_j}(x) dx = \int_{F(t_{i-1}) - F(t_j)}^{F(t_i) - F(t_j)} \left[ G_{m, m_j}(x) (F(t_i) - F(t_j)) - G_{m, m_j}(x) (F(t_{i-1}) - F(t_j)) \right]$$

where $G_{m, m_j}(t)$ denotes the integral of $g_{m, m_j}(t)$. This can be computed analytically for many common kernels (see Appendix B).

Given the exact log-likelihood, we back-propagate the gradients of the loss function $L$. In the experiment, we employ mini-batch optimization.

### 5 EXPERIMENTS

We start by setting up the qualitative and quantitative experiments, and then report their results.

#### 5.1 Datasets

We used four real-world event datasets from different domains.
We compare DHP against five widely used point process methods that incorporate the influence of the past events:

- HPP (Homogeneous Poisson Process): It is the simplest point process where the intensity is assumed to be constant over time.
- RPP (Reinforced Poisson Processes) [29, 32]: RPP accounts for the aging effect and the cumulative count of past events.
- SelfCorrecting (Self-correcting Point Process) [15]: Its intensity is assumed to increase linearly over time and this tendency is corrected by the historical events.
- Hawkes (Hawkes Process): Its intensity is parameterized by Equation 3, which explicitly models the influence of the past events by using the static triggering kernel.
- RMTPP (Recurrent Marked Temporal Point Process) [9]: It employs RNN to encode the non-linear effects of past events. The event sequences are first embedded by RNN and then used as the input of the intensity.

See Appendix D.3 for details of the comparison methods.

### 5.5 Performance Evaluation

In this section, we first compare DHP with existing methods on event prediction. Table 2 presents the negative log-likelihood (NLL) of the test data and Mean Absolute Percentage Error (MAPE) for different methods on the real-world event datasets. In this table, we omit the result of RMTPP since its log-likelihood function differs from those used in the other methods, (it is defined for the whole event sequence from all communities, not for the separate

| Source       | Time span          | # Events | Communities |
|--------------|--------------------|----------|-------------|
| Reddit       | Reddit API         | Mar 1 - 31 Aug, 2020 | 23,059 | 25 subreddits |
| News         | GDELT              | Jan 20 - Mar, 2020  | 19,541 | 40 news websites |
| Protest      | ACLED              | Mar 1 - 21 Nov, 2020 | 22,313 | 35 countries |
| Crime        | Chicago Data Portal| Mar 1 - Dec, 2020   | 29,318 | 13 community areas |
sequences of the individual communities, which precludes fair comparison. As shown in the table, our proposal, DHP, outperforms the four existing methods across all the datasets in terms of NLL. HPP has the worst NLL in most cases since it does not explore the temporal variation of the event occurrences. RPP and SelfCorrecting cannot achieve good results as they encode strong assumptions on the functional forms of the intensity, which limits the expressivity of the model. Hawkes surpasses HPP, RPP, and SelfCorrecting, which explicitly models the dependencies between past and current events. However, it still falls short for modeling the dynamic changes of the community states in the diffusion process. Our DHP achieves even better NLL than Hawkes. This verifies that incorporating latent community dynamics is essential for event prediction and that DHP can learn effective representations of the time-evolving dynamics of community states.

DHP achieves the best MAPE for all datasets. RMTPP performs the second best in terms of MAPE for Reddit and News datasets, which is probably because RMTPP exploits the power of RNN for learning non-linear dependencies between events. But RMTPP performs poorly for Protest and Crime datasets since it cannot capture changes in the event occurrences due to the temporal evolution of communities’ states, e.g., a large reduction in protest events due to the COVID-19. DHP outperforms all other methods across the datasets on the two metrics. The above result reveals the effectiveness of encoding the community state dynamics governing the diffusion process for event prediction. It also suggests that the assumption of DHP, i.e., the magnitude and speed of diffusion are related, holds for real diffusion processes.

5.6 Sensitivity Study

In this section, we analyze the impacts of hyperparameters or experimental settings. We report the prediction performance of DHP under different settings for the four datasets.

Number of mixtures. We examine how the number of mixture components, \( C \), determines the prediction performance of DHP. Figure 3a shows the negative log-likelihood (NLL) on the test data with respect to different numbers of mixtures \{1, 2, 3, 4, 5\}. In this experiment, we fixed the number of layers as 3 and used the power-law kernel. The NLL performance tends to be stable for all the datasets. It slightly increases as the number of mixture components becomes larger for Protest and Crime dataset. The results indicate that increasing the number of mixtures can improve the expressiveness of the model.

Figure 3: Sensitivity Study: NLL performance of DynamicHawkes on different settings for four datasets.

Kernel functions. We investigate the effect of three kernel functions: exponential kernel, power-law kernel, and Raleigh kernel, where the number of mixtures and the number of layers are set to 3. As shown in Figure 3b, the power-law kernel yields the best performance on all datasets.

Number of layers. Figure 3c evaluates the sensitivity of our neural network \( \Phi_m^c(t) \) to the number of layers \( L \in \{1, 2, 3, 4, 5\} \) by fixing the number of mixtures as 3 and using the power-law kernel. We observe that DHP yields better NLL results for the Protest dataset with larger numbers of layers. For the other three datasets, it has little effect on the performance.

In general, DHP shows stable and robust prediction performance across different settings.

5.7 Case Studies

In order to further verify the capability of DHP, we analyze the temporal dynamics of the community states behind the diffusion process learned by DHP from each dataset.

Figure 4 visualizes the interactions between selected 8 Reddit communities (i.e., subreddits) learned from Reddit dataset. In each row, we compare the estimated interactions between subreddits by DHP (middle) and Hawkes (bottom), and the ground truth (top). For the ground truth, node size corresponds to the aggregated number of hyperlinks for each “target” community in the default 5-day interval; the weight of each edge represents the number of hyperlinks between source community \( m' \) and target community \( m \).
For DHP, node size is proportional to $\sum_{m'} a_{m,m'} f_m(t)$; edge width is $a_{m,m'} f_m(t)$. For Hawkes, node size is $\sum_{m'} a_{m,m'}$; edge width is $a_{m,m'}$. Note that Hawkes produces the same results across times since it assumes the triggering kernel is static over time. We can see that the interactions learned by DHP are more consistent with the true evolution of the interactions between online user communities compared to Hawkes.

The top panel of Figure 1 shows the intensity $\lambda_m(t)$ and estimated dynamics function $f_m(t)$ learned for Reddit dataset along with the observed event sequences for the two subreddits. The latent dynamics function increases up to the end of May, rapidly for news and slowly for space. This is probably due to the COVID-19 lockdown. These results demonstrate that our DHP learns a reasonable representation of the latent temporal dynamics of the online communities.

Figure 5 shows inferred interactions among news websites from 15 countries learned for News dataset. In these figures, the node size denotes the value of the latent dynamics function $\sum_{m'} a_{m,m'} f_m(t)$ for each news website; the edge width denotes the strength of interactions between them $a_{m,m'} f_m(t)$. East Asian and South-East Asian countries (denoted by blue and yellow) rise to their peaks around late February (See Figure 5a) and then decrease until mid-March (Figure 5b), while the other countries are peaked around or after March 15 (Figure 5b), not in February (Figure 5a). We can also see in Figure 6 that the dynamic function peaks around mid-February for China (left), followed by the United Kingdom with the peak of mid-March (right). These trends are synchronized to the growth of the pandemic in each country. East Asian and South-East Asian countries experienced their first peak in COVID-19 cases ahead of the other countries, which would trigger the people’s early interest on COVID-19 related topics and accelerate the spread of COVID-19 related news early on. This confirms that our proposal, DHP, well reproduces the complex evolution in news website activities.

Figure 7 shows the intensity and latent dynamic function learned from the Protest dataset. According to a previous study, in contrast to the online events, the pandemic initially leads to a reduction in protest events and the trend was corrected after several weeks. DHP well characterizes this trend. In China (left), the dynamic function decreased following the onset of the coronavirus around the beginning of March and returned to a moderate level by mid-June. For Russia, it declined gradually from March until the beginning of July, where the first peak of the pandemic occurred around May 11. In conclusion, DHP uncovers the latent community dynamics.

---

5https://acleddata.com/2020/09/03/demonstrations-political-violence-in-america-new-data-for-summer-2020/
underlying the diffusion processes, and so provides meaningful insights about the diffusion mechanism.

6 CONCLUSION AND FUTURE WORK

Modeling and predicting diffusion processes are important tasks in many applications. We presented a novel Hawkes process framework, DHP (Dynamic Hawkes Process), that can learn the temporal dynamics of the community state underlying diffusion processes. The proposed DHP allows for the automatic discovery of the community state dynamics underlying the diffusion processes as well as offering tractable learning. By conducting extensive experiments on four real event datasets, we demonstrate that DHP provides better performance for modeling and predicting diffusion processes than several existing methods.

For future work, we plan to explore the following two directions. First, DHP can be extended to capture the pairwise dynamics of the interactions among communities by introducing the latent dynamics function for pairs of communities. We will extend DHP to this case and conduct experiments to evaluate the performance of the extended DHP in capturing the time-evolving dynamics of the pairwise interactions between communities. Secondely, DHP is built on the assumption that the magnitude and speed of the diffusion are pairwise interactions between communities. Secondly, DHP is extended in this case and conduct experiments to evaluate the performance of diffusion processes.

REFERENCES

[1] Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. TensorFlow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467 (2016).

[2] Emmanuel Bacry, Jacopo Maitromatteo, and Jean-François Muzy. 2015. Hawkes processes in finance. Market Microstructure and Liquidity 1, 01 (2015), 1550005.

[3] Laurence A Baxter, Masaaki Kijima, and Michael Tortorella. 1996. A point process model for the reliability of a maintained system subject to general repair. Stochastic models 12, 1 (1996), 12–1.

[4] Emery N Brown, Riccardo Barbieri, Valérie Ventura, Robert E Kass, and Loren M Frank. 2002. The time-rescaling theorem and its application to neural spike train data analysis. Neural computation 14, 2 (2002), 325–346.

[5] Valerio Chavez-Demoulin, Anthony C Davison, and Alexander J McNeil. 2005. Estimating value-at-risk: a point process approach. Quantitative Finance 5, 2 (2005), 227–234.

[6] Pavel Chilinski and Ricardo Silva. 2020. Neural likelihoods via cumulative distribution functions. In Conference on Uncertainty in Artificial Intelligence. PMLR, 420–429.

[7] François Chollet et al. 2015. Keras. https://github.com/fchollet/keras.

[8] Hanjun Dai, Yichen Wang, Rakshit Trivedi, and Le Song. 2016. Deep coevolutionary network: Embedding user and item features for recommendation. arXiv preprint arXiv:1609.03675 (2016).

[9] Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. 2016. Recurrent marked temporal point processes: Embedding event history to vector. In KDD. 1550 (2014), 1392–1412.

[10] Nan Du, Yichen Wang, Xiao He, Jiming Sun, and Le Song. 2015. Time-sensitive recommendation from recurrent user activities. In NeurIPS. 3492–3500.

[11] Paul Embrechts, Thomas Liniger, and Lu Lin. 2011. Multivariate Hawkes processes: an application to financial data. Journal of Applied Probability 48, A (2011), 367–378.
Table 3: Table of symbols.

| Symbol | Definition |
|--------|------------|
| \(H(t)\) | event sequence up to time \(t\) |
| \(N(t)\) | total number of events up to time \(t\) |
| \(N^m(t)\) | number of events of dimension \(m\) up to time \(t\) |
| \(\lambda_m(t)\) | intensity function for dimension \(m\) |
| \(\mu_m\) | background rate for dimension \(m\) |
| \(g_{m,m'}(t)\) | triggering kernel between dimension \(m\) and dimension \(m'\) |
| \(a_{m,m'}\) | interactions between dimension \(m\) and dimension \(m'\) |
| \(f_m(t)\) | dynamic function for dimension \(m\) |
| \(F_m(t)\) | integral of dynamic function for dimension \(m\) |
| \(\Phi_m(t)\) | neural network function of dimension \(m\) for component \(c\) |
| \(M\) | number of dimensions |
| \(C\) | number of mixture components |
| \(L\) | number of layers of neural network |
| \(S\) | number of time intervals in test time period |

Table 4: Integral for some common kernels.

| Type | Equation \(g(t)\) | Integral \(G(t)\) |
|------|-----------------|-----------------|
| Exponential (EXP) | \(\alpha \exp(-\beta t)\) | \(-\frac{\alpha}{\beta} \exp(-\beta t)\) |
| Power-law (PWL) | \(\frac{\alpha^p}{(\alpha + \beta t)^{p+1}}\) | \(-\frac{\alpha}{p(\alpha + \beta t)^p}\) |
| Raleigh (RAY) | \(\alpha t \exp(-\beta t^2)\) | \(-\frac{\alpha}{2\beta} \exp(-\beta t^2)\) |

SUPPLEMENTAL MATERIAL

A NOTATION REFERENCE

For readers’ convenience, we list the important notations used throughout the paper in Table 3.

B KERNEL INTEGRAL

In our experiment, we used three types of triggering kernel: Exponential, Power-law and Raleigh. Table 4 presents their equations and integrals, where \(\alpha\) and \(\beta\) are parameters of the triggering kernel. \(p\) is the scaling exponent of the power-law (\(p > 1\)) and we fix \(p = 2\) in the experiment.

C PREDICTION

For each time interval \((t_s, t_{s+1}]\) and each dimension \(m\), given the history of events up to time \(t_s\), we calculate the expected number of events in \((t_s, t_{s+1}]\) by \(\int_{t_s}^{t_{s+1}} \lambda_m(\tau) d\tau\). As discussed in Section 4.2, this integral takes analytic form for the proposed method. Similarly to Equation 13, we obtain

\[
\tilde{N}^m((t_s, t_{s+1}]) = \int_{t_s}^{t_{s+1}} \lambda_m(\tau) d\tau + \sum_{j:t_j\in[t_s,t_{s+1}]} G_{m,m_j} \left( F(t_{s+1}) - F(t_j) \right) - G_{m,m_j} \left( F(t_s) - F(t_j) \right),
\]

where \(\tilde{N}^m((t_s, t_{s+1}])\) is the predicted number of events in the given time interval \((t_{s+1}, t_s]\) for dimension \(m\).

D EXPERIMENT

D.1 Datasets

- **Reddit**: The data collection procedure followed the one used in [19]. During crawling we selected the 25 most popular subreddits, and retrieved hyperlinks among those subreddits: we identified and recorded posts in one source subreddit that contain links to different target subreddits. This process finally yielded a total of roughly 23,000 posts, each of which had submission time, source subreddit, and target subreddit. We treated a list of hyperlinks to each target subreddit as a separate sequence and considered target subreddits as communities (i.e., dimensions). The source subreddit was not used for training but for qualitative evaluation (Figure 4).

- **News**: The original dataset contains over a million of news articles related to COVID-19. Each piece of news had a timestamp and a URL. We extracted the domain of news websites from a URL and obtained more than 1,000 unique domains. We filtered out 40 country-specific domains and used them as communities. The granularity of time is one second.

- **Protest**: We sampled 35 popular countries and retrieved events from those countries. Each event was associated with two attributes: timestamp and country. The dataset was recorded at minute level.

- **Crime**: Each event recorded the time and community area where a crime happened. The time granularity is one minute.

D.2 Implementation details

All code was implemented using Python 3.9 and Keras [7] with a TensorFlow backend [1]. We conducted all experiments on a machine with four 2.8GHz Intel Cores and 16GB memory.

The model parameters were trained using the ADAM optimizer [17] with \(\beta_1 = 0.9, \beta_2 = 0.999\) and a learning rate of 0.002. For the neural networks-based models (i.e., RMTPP and DHP), the number of hidden units in each layer is fixed as 8. In our experiment, the number of mixtures is set to 3 for Reddit, News and Protest datasets, and to 5 for Crime dataset. In all experiments, we used the power-law kernel. The number of layers is set to 2 for Reddit and Protest datasets, 1 for News dataset, 3 for Crime dataset, respectively.

D.3 Comparison methods

- **HPP (Homogeneous Poisson Process)**: The simplest point process whose intensity is constant over time: \(\lambda_m(t) = \lambda_m\).

- **RPP (Reinforced Poisson Processes)** [29, 32]: For each dimension \(m\), the intensity of RPP is characterized by

\[
\lambda_m(t) = \gamma_m(t) N^m(t),
\]

where \(\gamma_m(t)\) is the relaxation function that characterizes the aging effect, and \(N^m(t)\) is the number of events of dimension \(m\) that have occurred up to \(t\). Following the prior work [32], we define \(\gamma_m(t)\) by the following relaxation log-normal function:

\[
\gamma_m(t) = \exp\left(-\frac{(\log t - \alpha_m)^2}{2\beta_m^2} \right),
\]

where \(\alpha_m\) and \(\beta_m\) are parameters, which are local to the dimension.
• SelfCorrecting (Self-correcting Point Process) [15]: The intensity function of SelfCorrecting is assumed to increase steadily over time with the rate $\beta_m > 0$; this trend is corrected by constant $\rho_m > 0$ every time an event arrives. Its intensity function associated with dimension $m$ is given by

$$\lambda_m(t) = \exp\left(\alpha_m + \beta_m(t - \rho_m N^m(t))\right),$$

(18)

where $\alpha_m, \beta_m,$ and $\rho_m$ are parameters, and $N^m(t)$ is the number of events of dimension $m$ in $(0, t]$. 