Temporal Logic Point Processes

Shuang Li [1] **Lu Wang** [2] **Ruizhi Zhang** [3] **Xiaofu Chang** [4] **Xuqin Liu** [4] **Yao Xie** [5] **Yuan Qi** [4] **Le Song** [4 6]

Abstract

We propose a modeling framework for event data and aim to answer questions such as *when* and *why* the next event would happen. Our proposed model excels in *small data* regime with the ability to incorporate *domain knowledge* in terms of logic rules. We model the dynamics of the event starts and ends via intensity function with the structures informed by a set of first-order temporal logic rules. Using the softened representation of temporal relations, and a weighted combination of logic rules, our probabilistic model can deal with uncertainty in events. Furthermore, many well-known point processes (e.g., Hawkes process, self-correcting point process) can be interpreted as special cases of our model given simple temporal logic rules. Our model, therefore, riches the family of point processes. We derive a maximum likelihood estimation procedure for the proposed temporal logic model and show that it can lead to accurate predictions when data are sparse and domain knowledge is critical.

1. Introduction

A diverse range of application domains, such as healthcare (Reynaud-Bouret et al., 2010), finance (Bacry et al., 2015), smart city, and information networks (Zhao et al., 2015; Farajtabar et al., 2015; 2014), generate discrete events in continuous time. For instance, the occurrences of diseases on patients are event data; credit card uses are event data; the arrivals of passengers in subway systems are event data; the posting and sharing of articles in online social platforms are also event data. Modeling these continuous-time event data becomes increasingly important to understand the underlying systems, to make accurate predictions, and to regulate these systems towards desired states. Recently, sophisticated models such as recurrent Marked point processes (Du et al., 2016), neural Hawkes processes (Mei & Eisner, 2017) and policy-like generative point processes (Li et al., 2018) have been proposed, allowing us to model increasingly complex phenomena.

Although these models are flexible, they require lots of data to properly fit the models, making them perform poorly in the regime of small data. Furthermore, these models are notorious for their difficult-to-interpret prediction results and have been branded as “black boxes" (Doshi-Velez & Kim, 2017) – it is difficult to clearly explain or identify the logic behind the predictions. In some cases, interpretability is more important than predictions. For example, in medicine, people are more interested in understanding what treatments contribute to the occurrences and cures of diseases than merely predicting the patients' health status in real time (Lee & Yoon, 2017).

Very often, there already exists a rich collection of prior knowledge from a particular domain, and we want to incorporate them to improve the interpretability and generalizability of the event models. For instance, in medical areas and for patients with sepsis, we may leverage prior knowledge such as “if sepsis progresses to septic shock, the blood pressure may drop dramatically" or “if use drug Norepinephrine, the blood pressure may be controlled" to predict the appearance or disappearance of some symptoms in terms of when and why. We want to utilize knowledge like this rather than reinvent the wheel and purely relying on data to come up with the rules. When the amount of data is small and noisy, it will also be challenging to accurately recover these rules via data-driven models.

Our interest lies in interpretable event models. We aim to propose a unified framework for modeling the generative mechanism of the events by incorporating temporal logic rules (Smullyan, 2012). The occurrence time of events often exhibits complicated patterns. Our proposed temporal logic point processes explicitly model the occurrence rate of events in continuous time. More specifically, we use (transition) intensity functions to model the start and end of the events, and these intensity functions will be informed via a set of temporal logic rules involving both other types

[1]Department of Statistics, Harvard University; [2]Department of Computer Science, East China Normal University; [3]Department of Statistics, University of Nebraska-Lincoln; [4]Ant Group; [5]H. Milton Stewart School of Industrial & Systems Engineering, Georgia Institute of Technology; [6]School of Computational Science & Engineering, Georgia Institute of Technology. Correspondence to: Shuang Li <shuangli@fas.harvard.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

of events and temporal constraints.

In addition to the interpretability, our modeling framework has the following characteristics and advantages:

(*i*) **Tolerance of uncertainty.** Data are noisy in the real world, and time information is often imprecisely recorded. Our model uses a weighted combination of temporal logic rules as *soft constraints* rather than *hard constraints*. These designs allow for uncertainties in data and the impreciseness of the incorporated logic rules.

(*ii*) **Temporal relation constraints.** Our model can consider logic rules with *temporal relation constraints*, such as "A happens *before* B", "If A happens, and *after* 5 mins, B can happen", and "If A and B happen *simultaneously*, then *at the same time* C can happen". Our model uses a softened parametrization of the temporal relation constraints as part of the logic functions.

(*iii*) **Continuous-time reasoning process.** Our model captures the dynamics of a continuous-time reasoning process and can naturally deal with asynchronous events.

(*iv*) **Small data and knowledge transfer.** Our model better utilizes domain knowledge and, as a result, will work well on small datasets. Different datasets in similar concepts might share similar logic rules. We might leverage the learned logic weights in one dataset to warm-start the learning process on a different dataset. Our model makes it possible to transfer knowledge among different datasets.

Furthermore, we show that many existing point process models (Ogata, 1988; Ogata & Vere-Jones, 1984; Achab et al., 2017) can be recovered as special cases of our framework by specifying simple temporal logic rules. We derive a maximum likelihood estimation procedure for our model parameters (i.e., importance weights of logic rules), and show that the learned model can lead to interpretable and accurate predictions in the regime of small data.

2. Related Work

The seminal work Markov logic networks (MLNs) (Richardson & Domingos, 2006) and the extension (Singla & Domingos, 2008) propose to model the joint distribution of a set of predicates (i.e., variables) via Markov random field. The logic rules can be viewed as a template for constructing Markov networks and a weight is attached to each logic rule. This model elegantly incorporates domain knowledge and softens logic constraints. However, it is a static model and cannot be directly adapted to events or sequential data that usually exhibit complex temporal dependency.

To handle temporal relations, dynamic MLNs (Papai et al., 2012) adds a temporal dimension to MLNs and models influences between variables within a discretized time interval. The dynamic MLNs do not assume a causal direction (i.e., only the history can influence the future), in contrast with directed models, such as dynamic Bayesian networks (Kersting & De Raedt, 2001; Nachimuthu et al., 2010) and logic hidden Markov models (Kersting et al., 2006; Natarajan & Nevatia, 2007; Natarajan et al., 2008), which represent the conditional dependence in a directed graph. Specific for event data, probabilistic event logic for interval-based events (Tran & Davis, 2008; Brendel et al., 2011) has been proposed, which can allow for latent variables and model the joint probability of the events.

All these above-mentioned models are built upon the framework of probabilistic graphical models, which allow for partial and noisy observations. However, graphical models are also known for computationally expensive in inference. In contrast, our model is proposed based on two assumptions: (1) we assume a causal direction; and (2) we assume that the states of predicates are fully observed (we tolerate the noise in state transition times by introducing softened temporal relation predicates). Under these assumptions, we model temporal predicates as continuous-time stochastic processes and leverage the theory of point process (Jacobsen, 2006) to directly model the distribution of the inter-event time, which is characterized by the occurrence intensity function. We use the first-order logic rules to inform the intensity function design. A direct benefit is that the joint likelihood of the events has a much simpler expression in terms of the intensity function, which makes our model relatively easy in inference and more scalable with samples and variables.

In comparison with the state-of-the-art point process models (Du et al., 2016; Mei & Eisner, 2017; Qian et al., 2020; Jarrett et al., 2019), our model constructs the intensity function in a more structured way – the introduced first-order temporal logic rules serve as templates to collectively gather evidence from history to infer future events, which is more data-efficient.

3. Temporal Logic

First-order Logic. A *predicate* such as $\texttt{Smokes}(c)$ or $\texttt{Friend}(c, c')$, denoted as $x(\cdot)$, is a logic function defined over a set of entities $\mathcal{C} = \{c_1, c_2, \ldots, c_{|\mathcal{C}|}\}$, i.e.,

$$x(\cdot) : \mathcal{C} \times \mathcal{C} \cdots \times \mathcal{C} \mapsto \{0, 1\}.$$

The predicates indicate the *property* or *relation* of entities. A first-order *logic rule* is a logical connectives of predicates, such as

$$f_1 : \quad \forall c \; \texttt{Cancer}(c) \leftarrow \texttt{Smokes}(c);$$
$$f_2 : \quad \forall c \, \forall c' \; \texttt{Smokes}(c') \leftarrow \texttt{Friend}(c, c') \wedge \texttt{Smokes}(c).$$

Commonly used logical connectives are: $\wedge$ for conjunction, $\vee$ for disjunction, $\leftarrow$ for implication, and $\neg$ for negation. It is often convenient to convert

Table 1. Logic Rule in Clausal Form.

x_A, x_B	$x_B \leftarrow x_A$	$\neg x_A \vee x_B$
0, 0	1	1
0, 1	1	1
1, 0	0	0
1, 1	1	1

logic rules to a *clausal form*, which is a conjunction or disjunction of predicates. Table 1 demonstrates the fact that logic rule $x_B \leftarrow x_A$ is logically equivalent to the clausal form $\neg x_A \vee x_B$.

Temporal logic predicate. A *temporal predicate* is a logic function $x(\cdot, \cdot)$ defined over the set of entities $\mathcal{C} = \{c_1, c_2, \ldots, c_{|\mathcal{C}|}\}$ and time $t \in [0, \infty)$, i.e.,

$$x(c, t): \quad \mathcal{C} \times \mathcal{C} \cdots \times \mathcal{C} \times [0, \infty) \mapsto \{0, 1\}.$$

The trajectory of a grounded temporal predicate $\{x(c,t)\}_{t\geq 0}$ can also be viewed as a *continuous-time two-state stochastic process*. For example, temporal predicate $\{\text{NormalBloodPressure}(c,t)\}_{t\geq 0}$ where c is the patient ID will take values 1 or 0 at any time t to indicate whether blood pressure is normal or abnormal, and the state transition time is stochastic.

For simplicity of notation, we will focus on the case with one entity, and drop the dependency of predicates on the entity. Hence, we will write $x(c,t)$ as $x(t)$ instead. Given a sample path (i.e., grounded temporal predicate) of $\{x(t)\}_{t\geq 0}$ up to time t, $\{x(t)\}_{t\geq 0}$ will stay in state 0 or state 1 for some *time interval*. For example, in Fig. 1 (top), the grounded predicate is recorded as $x(t) = 0$ for $t \in [0, t_1)$, $x(t) = 1$ for $t \in [t_1, t_2)$, and so on, where $t_1, t_2, \cdots$ are state transition times. In some cases, the grounded predicate $x(t)$ is instantaneous in state 0 or 1, and we will obtain a point-based predicate process. Here, we regard point as a degenerate time interval. As in Fig. 1 (bottom), the predicates will be triggered to be 1 at the event times, and we record $x(t_1) = 1$, $x(t_2) = 1$, and so on. For other time, $x(t) = 0$.

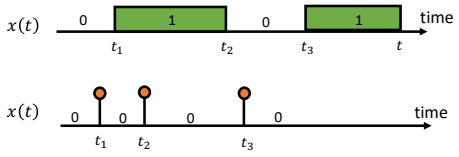


Figure 1. Illustration of grounded two-state (top) and point-based (bottom) temporal predicates.

Temporal relation. Allen's original paper (Allen, 1990) defined 13 types of *temporal relations between two time intervals*, denoted as $\{r_1, r_2, \ldots, r_{13}\}$, which are mutually exclusive. Specifically, let two time intervals for predicate x_A and predicate x_B be $\tau_A = [t_{A_1}, t_{A_2}]$ and $\tau_B = [t_{B_1}, t_{B_2}]$ respectively, where t_{A_1} and t_{B_1} are the interval starting times, and t_{A_2} and t_{B_2} are the interval ending times. Then a temporal relation is a logic function defined as

$$r(\cdot): (t_{A_1}, t_{A_2}, t_{B_1}, t_{B_2}) \mapsto \{0, 1\}$$

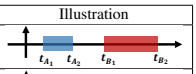
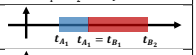
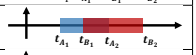
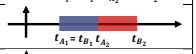
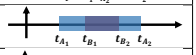
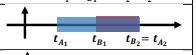
which can be mathematically evaluated by a step function $g(s)$ and an indicator function $\kappa(s)$ defined as

$$\text{step function: } g(s) = \begin{cases} 1 & s \geq 0 \\ 0 & s < 0 \end{cases}, \tag{1}$$

$$\text{indicator function: } \kappa(s) = \begin{cases} 1 & s = 0 \\ 0 & o.w. \end{cases}, \tag{2}$$

to enforce *hard* temporal constraints. We will discuss the softened approximation functions for step function $g(s)$ and delta function $\kappa(s)$ in section 4.3 to tolerate uncertainties in data, and using the softened temporal constraints, we will have $r(\cdot) \in [0, 1]$. Function forms of these 13 temporal relations can be founded in Table 2. Considering the inverses of relation $r_1 - r_6$ plus the symmetric relation r_7 "equal", there are a total of 13 relations. If there are *no* temporal relation constraints on x_A and x_B, then their temporal relations can take any of the 13 types, and $r_0 = r_{no}()$ returns the disjunction of these relations and is always "True" (i.e., 1).

Table 2. Interval-based temporal relation constraints and their illustrative figures.

Temporal Relation	Logic Function $r(\cdot)$	Illustration
r_b: A before B	$g(t_{B_1} - t_{A_2})$	
r_m: A meets B	$\kappa(t_{A_2} - t_{B_1})$	
r_o: A overlaps B	$g(t_{B_1} - t_{A_1}) \cdot g(t_{B_1} - t_{A_2}) \cdot g(t_{B_2} - t_{A_2})$	
r_s: A starts B	$\kappa(t_{A_1} - t_{B_1}) \cdot g(t_{B_2} - t_{A_2})$	
r_c: A contains B	$g(t_{B_1} - t_{A_1}) \cdot g(t_{A_2} - t_{B_2})$	
r_f: A finished-by B	$g(t_{B_1} - t_{A_1}) \cdot \kappa(t_{A_2} - t_{B_2})$	
r_e: A equals B	$\kappa(t_{A_1} - t_{B_1}) \cdot \kappa(t_{A_2} - t_{B_2})$	

More complex temporal relations can be decomposed as the composition of these 13 types of two way relations. For example, "(A and B) before C" can be decomposed as "(A before C) and (B before C)".

For degenerate point-based predicate process, where $t_{A_1} = t_{A_2} = t_A$, and $t_{B_1} = t_{B_2} = t_B$, we will have a total of 3 types of temporal relations with function forms,

$$\begin{aligned} \texttt{Before}(t_A, t_B)&: \ g(t_B - t_A), \\ \texttt{After}(t_A, t_B)&: \ g(t_A - t_B), \\ \texttt{Equal}(t_A, t_B)&: \ \kappa(t_A - t_B). \end{aligned} \tag{3}$$

Temporal logic formula. A *temporal logic formula* is a logical composition of temporal logic predicates and temporal relations, $f(\mathcal{X}_f, \mathcal{T}_f) \in \{0, 1\}$, where

- $\mathcal{X}_f = \{x_u(t)\}$ is a set of temporal predicates used to define the formula f,
- $\mathcal{T}_f = \{\tau_u\}$ is a set of time intervals, with each $x_u \in \mathcal{X}_f$ associated with a time interval $\tau_u = [t_{u_1}, t_{u_2}]$ (1 and 2 in the subscript indicate interval start and end respectively). We require that within time interval τ_u, the value of the temporal logic predicate $x_u(t)$ remains fixed.

Then a temporal logic formula has a generic form

$$f(\mathcal{X}_f, \mathcal{T}_f) := \left(\left(\bigvee_{x_u \in \mathcal{X}_f^+} x_u(t_u)\right) \bigvee \left(\bigvee_{x_v \in \mathcal{X}_f^-} \neg x_v(t_v)\right)\right) \bigwedge \left(\bigwedge_{x_u, x_v \in \mathcal{X}_f} r_i(\tau_u, \tau_v)\right) \quad (4)$$

where $\mathcal{X}_f^-$ is the set of predicates used as negation in the formula f, $\mathcal{X}_f^+ = \mathcal{X}_f \setminus \mathcal{X}_f^-$, $t_u \in \tau_u$, $t_v \in \tau_v$, and $\{r_i(\tau_u, \tau_v)\}$ is the set of prespecified temporal relations between pairs of predicates where r_i can take any of the 13 types of temporal relation. Note that if we use the soft temporal relation constraints as in section 4.3, then the temporal logic formula $f(\mathcal{X}_f, \mathcal{T}_f) \in [0, 1]$.

4. Temporal Logic Point Processes

Suppose we define a collection of d temporal logic predicates $\boldsymbol{X}(t) = \{x_1(t), x_2(t), \ldots, x_d(t)\}$, which is a compact representation of temporal knowledge base and can be grounded at any time t. An example of $\{\boldsymbol{X}(t)\}_{t \geq 0}$ in healthcare context is illustrated in Fig. 2. Each temporal predicate $x_u(t) \in \{0, 1\}$, such as UseDrug1(t), NormalBloodPressure(t) and so on, represent the properties, medical treatments, and health status of a patient at time t and can be grounded from data.

A set of *prespecified* temporal logic formulae $\mathcal{F} = \{f_1, f_2, \cdots\}$ express our *prior belief* on how these temporal predicates are related. For example in Fig. 2, a first-order temporal logic rule is defined as "(GoodSurvivalCondition$(t'') \leftarrow$ NormalBloodPressure$(t) \wedge$ NormalHeartBeat$(t')) \wedge$ Before$(t, t'') \wedge$ Before(t', t'')". We want to incorporate these temporal logic formulae to our point process model to explain the generative mechanism of the events. More details are provided as follows.

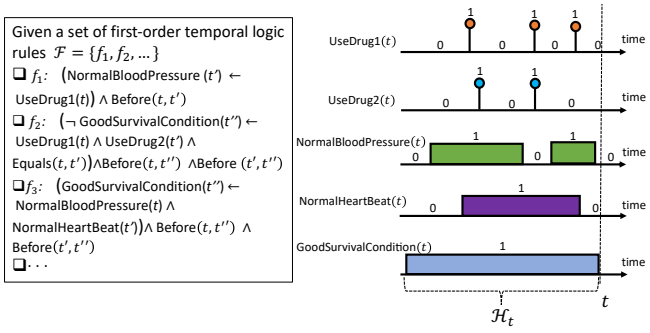


Figure 2. Running example in healthcare context.

4.1. Intensity model for temporal predicate

We note that, for a temporal predicate, the positive and negative values will occur in an alternating fashion, dividing the time axis into segments. To facilitate later exposition, we will denote $\mathcal{H}_u(t)$ as the sequence of time intervals for each temporal predicate $x_u(t)$. More specifically, if we observe a sequence of transition times $\{t_1, t_2, \ldots, t_n\}$ and the states at transition times between $[0, t)$, then we define

$$\mathcal{H}_u(t) := \{x_u(0), x_u(t_1), x_u(t_2), \ldots, x_u(t_n), x_u(t)\} \quad (5)$$

where values of the temporal predicate remain fixed within each time interval $[0, t_1), [t_1, t_2), \ldots, [t_{n-1}, t_n), [t_n, t)$. And the length of each interval $t_{i+1} - t_i \geqslant 0$ is the dwell time of a particular fixed state.

Given the set of $\mathcal{H} = \{\mathcal{H}_u\}_{u=1,\ldots,d}$ for all temporal predicates, we can model the sequence of events for a particular temporal predicate using two intensity functions as illustrated in Fig. 3(a). More specifically, define $\lambda_u^*(t) := \lambda(t|\mathcal{H}(t))$ as the conditional transition intensity for "$x_u(t)$ transits from 0 to 1", and $\mu_u^*(t) := \mu(t|\mathcal{H}(t))$ as the conditional transition intensity for "$x_u(t)$ transits from 1 to 0".

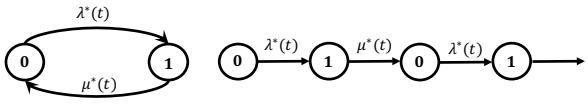


Figure 3. (a) Two-state transition diagram of a temporal predicate. (b) Unrolled conditional process.

We can unroll the transition diagram and obtain a conditional process, with a unique sample path. All the transition intensities are time and history dependent. Suppose $x_u(t) = 0$ at $t = 0$, we will have the conditional process as displayed in Fig. 3(b).

4.2. Intensity design guided by temporal logic rules

We now discuss how to design the conditional transition intensity for temporal predicates by fusing a set of temporal logic formulae $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$ from domain knowledge. We especially focus on how to utilize first-order temporal logic rules to construct features from history.

We will take a simple first-order temporal logic rule with temporal relation constraints as our running example. In plain language, a temporal reasoning rule for deducing event of type C is

$$f_1: \quad (C \leftarrow A \wedge B) \wedge (A \text{ before } B) \wedge (A \text{ and } B \text{ before } C), \quad (6)$$

which has the corresponding logical form as "if predicate x_A is true and predicate x_B is true, then predicate x_C is true; furthermore, x_A has to occur before x_B, and both have to occur before x_C".

Write the temporal logic formula in clausal form as Eq. (4) and we have

$$\begin{aligned} & f_1(x_A, x_B, x_C, t_A \in \tau_A, t_B \in \tau_B, t_C \in \tau_C) \quad (7) \\ = \; & (\neg x_A(t_A) \vee \neg x_B(t_B) \vee x_C(t_C)) \\ & \wedge r_{be}(\tau_A, \tau_B) \wedge r_{be}(\tau_B, \tau_C) \end{aligned}$$

where the temporal logic function f_1 can be evaluated based on the states of $x_A(t_A)$ at time t_A, $x_B(t_B)$ at time t_B, and $x_C(t_C)$ at time t_C, where t_A, t_B and t_C can be any time. The temporal relation constraints, which are part of the logic formula, will be evaluated based on the corresponding time intervals τ_A, τ_B and τ_C, with $t_A \in \tau_A$, $t_B \in \tau_B$ and $t_C \in \tau_C$. Within these time intervals, predicates $x_A(t), x_B(t)$

and $x_C(t)$ all maintain fixed values, which may be different from each other.

We are interested in forward reasoning where we model the conditional transition intensity of deduced predicate x_C and treat the histories of x_A and x_B as evidence. For predicate $x_C(t)$, at any time t, it has two potential outcomes 0 or 1. One can observe only one, but not both, of the two potential outcomes. The unobserved outcome is called the "*counterfactual*" outcome. The grounded predicate state $x_C(t)$ is always the "*observed*" outcome at time t, and $1 - x_C(t)$ is always the "*counterfactual*" outcome at time t.

To incorporate the knowledge from formula f_1 in constructing the transition intensity for x_C at any time t, we define a **formula effect** (FE) term, denoted as δ_{f_1} as

$$\begin{aligned} \text{FE} &= \delta_{f_1}(t \mid t_A \in \tau_A, t_B \in \tau_B) \\ &:= f_1(x_A, x_B, 1 - x_C, t_A \in \tau_A, t_B \in \tau_B, t_C = t) \\ &\quad - f_1(x_A, x_B, x_C, t_A \in \tau_A, t_B \in \tau_B, t_C = t) \end{aligned} \tag{8}$$

where the logic formula f_1 is evaluated as in Eq. (7), t_A and t_B can take any time (but only before t will make the logic function non-zero), and x_C is evaluated at time t.

FE answers the question "what would happen if x_C transits its state given logic formula f_1 is satisfied". Note that the sign of FE can be 1, -1 or 0, which can be interpreted as

$$\text{sgn(FE)} = \begin{cases} 1 & \text{Positive effect to transit,} \\ -1 & \text{Negative effect to transit,} \\ 0 & \text{No effect to transit.} \end{cases}$$

If we use the soft temporal constraints as in section 4.3, then $f_1 \in [0, 1]$ and therefore $\delta_{f_1} \in [-1, 1]$. The magnitude of the formula effect will quantify the strength of the influence that is time-dependent. We can aggregate the formula effect from history to construct features. These features further play a role to construct the conditional intensity functions.

Inspired by this, the *conditional transition intensity* for x_C from state 0 to 1, contributed by logic formula f_1 is modeled as

$$\lambda_C^*(t) = \exp\{w_{f_1} \cdot \underbrace{\sum_{\tau_A \in \mathcal{H}_A(t)} \sum_{\tau_B \in \mathcal{H}_B(t)} \delta_{f_1}(t \mid t_{A_1} \in \tau_A, t_{B_1} \in \tau_B)}_{\text{feature } \phi_{f_1}(t)}\} \tag{9}$$

where the sign of the formula effect $\delta_{f_1}(t \mid t_A, t_B)$ indicates whether logic f_1 exerts a positive or negative effect provided the history $\mathcal{H}_A(t)$ and $\mathcal{H}_B(t)$, and the magnitude of $\delta_{f_1}(t_A, t_B, t)$ quantifies the strength of the influence. The double summation takes into account all combinations of temporal intervals in $\mathcal{H}_A(t)$ and $\mathcal{H}_B(t)$. Since within each time interval τ_A and τ_B, predicates $x_A(t)$ and $x_B(t)$ maintain fixed values, we can simply evaluate the states of x_A and x_B at the starting point of each intervals t_{A_1} and t_{B_1}. The valid combinations (i.e., combination of the evidence that induces a nonzero formula effect) defined by f_1 are illustrated in Fig. 4(a). We also have a weight parameter, w_{f_1}, associated with the logic rule f_1. One can think of the formula weight as the confidence level on the formula. The higher the weight, the more influence the formula has on the intensity of $\lambda_C^*(t)$.

The conditional transition intensity $\mu_C^*(t)$ has the same expression as Eq. (9). The only difference is that when we compute $\lambda_C^*(t)$, we have $x_C(t) = 0$, whereas when we compute $\mu_C^*(t)$, we have $x_C(t) = 1$. As illustrated in Fig. 4(b), the valid combinations that yield nonzero formula effects correspond to $x_A(t_A) = 1, x_B(t_B) = 1$, and $x_A(t_A)$ happens before $x_B(t_B)$ and both before t. The feature can be evaluated from grounding $\delta_{f_1}(t|t_A, t_B)$ using Eq. (7) and (8).

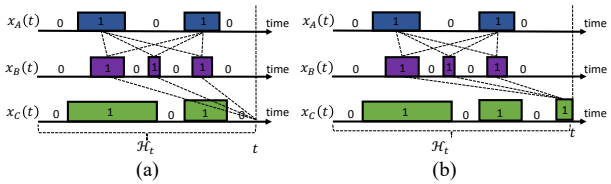


Figure 4. Combinations of A, B informed by f_1 to infer transition intensity of C. Different combinations can have different weights due to the temporal relation kernel introduced by the soft temporal relation constraints.

Predicate x_C can be deduced from more than one logic formulae. For example, x_C can belong to multiple rules $f_1(x_A, x_B, x_C)$ and $f_2(x_C, x_D)$. We assume effect of temporal logic formula f_1 and f_2 are *additive* in designing the transition intensity for x_C. In general, given a set of temporal logic formulae $\mathcal{F}_C = \{f_1, \ldots, f_n\}$ for deducing $x_C(t)$, the conditional transition intensity for predicate x_C, is designed as

$$\lambda_C^*(t) = \exp\left\{\sum_{f \in \mathcal{F}_C} w_f \cdot \phi_f(t) + b(t)\right\}, \tag{10}$$

where we introduce a base temporal function $b(t)$ to always allow for spontaneous transition without influence from the logic. For instance, $b(t)$ can either be a constant $b(t) = b$, or a deterministic function of t, or a regularized but flexible model. The intensity $\mu_C^*(t)$ has the same expression as in Eq. (10).

4.3. Softened temporal constraints

In practice, the temporal information usually cannot be accurately recorded. It makes more sense to introduce **soft constraints** for the temporal relations. We introduce softened approximation functions for step function $g(s)$ and delta function $\kappa(s)$ in replacement of those used in the definitions of temporal relations in Table 2.

Step function $g(s)$ can be softened as a triangular function with area one or a logistic function,

$$\begin{aligned} g(s) &= \min(1, \max(0, \beta s + \tfrac{1}{2})), \\ \text{or} \quad g(s) &= \frac{1}{1 + \exp(-\beta s)}. \end{aligned} \tag{11}$$

Delta function $\kappa(s)$ can be softened as a triangular function

with area one, or a Laplace distribution,

$$\kappa(s) = \max(0, \min(\tfrac{s}{\gamma^2} + \tfrac{1}{\gamma}, -\tfrac{s}{\gamma^2} + \tfrac{1}{\gamma})),$$
$$\text{or} \quad \kappa(s) = \frac{\exp(-|s|/\gamma)}{\gamma}. \qquad (12)$$

Parameters β and $\gamma \geq 1$ can be either specified or treated as unknown parameters that will be learned from data.

4.4. Likelihood

By the definition of transition intensity in Eq. (10), we can write out the likelihood. For predicate C, given a realization of the process up to time t, as in Fig. 3(b), the likelihood $\mathcal{L}(\{x_C(t)\}_{t\geq 0})$ is

$$\lambda_C^*(t_1)\exp\left(-\int_0^{t_1}\lambda_C^*(s)ds\right)\cdot\mu_C^*(t_2)\exp\left(-\int_{t_1}^{t_2}\mu_C^*(s)ds\right)$$
$$\cdots\exp\left(-\int_{t_n}^{t}\mu_C^*(s)ds\right), \qquad (13)$$

provided predicate x_C starts in state 0 and stays in state 1 up to time t. We give the proof of the likelihood in Appendix C. By considering all the predicates, the likelihood for the dataset is $\mathcal{L} = \prod_{u\in\{1,\ldots,d\}}\mathcal{L}(\{x_u(t)\}_{t\geq 0})$.

4.5. Inference

All the unknown parameters regarding the logic weights $(\boldsymbol{w}_f, b)$ and the temporal relations β and γ will be jointly learned by maximizing the likelihood. The likelihood function as in Eq. (13) has no closed-form expression in most cases and the integral term involved requires numerical approximation. But this numerical approximation is not difficult for one dimension. Stochastic gradient descent type of optimization algorithm can be utilized to learn model parameters. More specifically, if we want to ensure $w_f > 0$ for better interpretation of the results, projected gradient descent (Boyd & Vandenberghe, 2004; Chorowski & Zurada, 2014) methods can be adopted to satisfy the constraints.

5. Experiments

We empirically demonstrated the interpretability, prediction accuracy, and flexibility of our proposed temporal logic point processes (TLPP) on both synthetic (including Hawkes processes and self-correcting point process) and real data (including healthcare application about sepsis patients mortality prediction and finance application about credit card fraud event prediction).

5.1. Recover Well-known Temporal Point Processes.

We focused on Hawkes processes and self-correcting processes and showed they are special cases of our model via using one logic rule to completely recover the dynamics of these parametric point processes.

The experiments were set up as follows. We fixed the parameters of a chosen parametric point process and produced training samples from it. Then we provided a temporal logic rule to our model as prior knowledge. Our temporal logic point process model was constructed by encoding the prior knowledge in its intensity structures, and the model parameters were learned using the training samples. For evaluation, new samples were generated from the learned model and were compared with the training samples. We visually demonstrated their similarities.

(*i*) **Nonlinear Hawkes** is a self-exciting point process with intensity function of the form

$$\lambda(t) = \exp(b + \alpha\sum_{t_i<t}\exp(-(t-t_i))), \qquad (14)$$

where $b > 0$ and $\alpha > 0$, and it models the mechanism that previous events will boost the occurrence rate of new events. This temporal pattern can be described as a logic rule such as "If A happens, then A will happen again afterwards", which can be expressed as

$$f_{\text{Hawkes}}: \quad X_A(t) \leftarrow X_A(t') \wedge \texttt{Before}(t', t), \qquad (15)$$

where $X_A(t)$ is a degenerate temporal predicate which will be triggered to be 1 at time t. This correctness of the rule can be evaluated as a logic function $f_{\text{Hawkes}}: \ (X_A(t)\vee\neg X_A(t'))\cdot g(t-t')$ where $g(s)$ is a softened temporal predicate such as a logistic function in Eq. (11).

(*ii*) **Self-Correcting** or self-regulating point process is with intensity function of the form

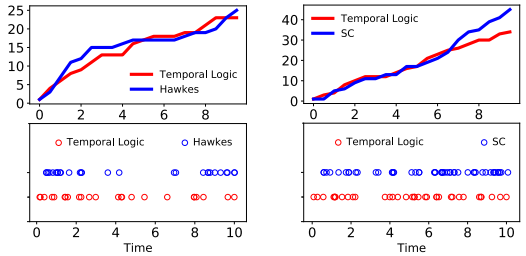
$$\lambda(t) = \exp\left(bt - \sum_{t_i<t}\alpha\right), \qquad (16)$$

where $b > 0$ and $\alpha > 0$, and it models the fact that previous events will inhibit the occurrence rate of new events. Using the logic language, it corresponds to "If A happens, then A will not happen again", which can be expressed as

$$f_{\text{self-correcting}}: \neg X_A(t) \leftarrow X_A(t') \wedge \texttt{Before}(t', t). \qquad (17)$$

Similarly, this rule can be evaluated as a logic function $f_{\text{self-correcting}}: \ (\neg X_A(t)\vee\neg X_A(t'))\cdot g(t-t')$.

Utilizing the hypothesized logic rules as (15) and (17), our model successfully recovered the clustering patterns of the targeting processes defined as Eq. (14) and Eq. (16) only using one short sequence of events, and the cumulative counts of the generated events and the training event are almost the same up to any time t, as displayed in Fig. 5.



(a): Hawkes. (b) Self-correcting.

Figure 5. Illustration of the generated events (in red) and the training events (in blue). Top: cumulative counts of the events. Bottom: realizations of the events that occur irregularly over time.

5.2. MIMIC-III: Predict Patient's Survival Rate

MIMIC-III is an electronic health record dataset of patients admitted to the intensive care unit (Johnson et al., 2016). We gleaned patients diagnosed as sepsis (8,284 patients) from it to construct our dataset in this experiment.

Predicates and Logic Rules. We had human experts (i.e., doctors) help us define the logic rules according to the pathogenesis of sepsis (Hotchkiss & Karl, 2003). A set of first-order temporal logic rules were defined over a collection of predicates, which were summarized in Table 3. The predicates refer to drug usage, symptoms of patients (mainly vital signs), survival condition, and temporal relations respectively. Three categories of drugs, Antibacterial, Vasoactive drugs, and Diuretics, were included. Sepsis is caused by the body's response to an infection, and Antibacterials are directly targeting the source of the infection. In our list of Antibacterials, three drugs (i.e., Meropenem, Tobramycin, and Vancomycin) are for severe sepsis, and the remaining are for mild sepsis. They are required to be delivered to the same patient following the order — Antibacterials for severe sepsis cannot be applied before those for mild sepsis, due to antibiotic resistance. Vasoactive drugs and Diuretics are needed to alleviate severe clinical conditions when the septic shock (a complication of sepsis) appears, which will reflect in the state changes of vital signs. Among all symptoms, low blood pressure is closely related to the mortality rate. The complete logic rules were summarized in Appendix A, Table 9, and they were prior knowledge of our model.

Table 3. Defined Predicates for Sepsis Patients in MIMIC-III.

Antibacterials	`Use-Levofloxacin`(t), `Use-Ceftriaxone`(t), `Use-Meropenem`(t), `Use-Ceftazidime`(t), `Use-Tobramycin`(t), `Use-Metronidazole`(t), `Use-Vancomycin`(t), `Use-Azithromycin`(t), `Use-Ciprofloxacin`(t), `Use-Piperacillin`(t)
Vasoactive Drugs	`Use-Metoprolol`(t), `Use-Diltiazem`(t), `Use-Norepinephrine`(t)
Diuretics	`Use-Furosemide`(t)
Symptoms	`NormalBloodPressure`(t), `NormalHeartRate`(t), `NormalRespiratoryRate`(t), `NormalTemperature`(t), `NormalUrineOutput`(t)
Suvival Condition	`GoodSurvivalCondition`(t)
Temporal Relation	`Before`(t,t'), `Equal`(t,t')

Data Pre-Processing. We extracted trajectories of the predicates from raw data. All predicates were grounded sequentially with state transition times recorded. Specifically, we distilled the following information from data, such as

$$([\texttt{patient ID}], [\texttt{predicate ID}], [\texttt{time}], [\texttt{state}]).$$

For drug-use predicates, treatment times were recorded with state 1; for symptom predicates, state transition times (i.e., from normal state 1 to abnormal state 0 or vice versa) were recorded; and for survival, since the predicate can only transit its states at most once, patients' admission and discharge times (if provided), and their final states (0 or 1) and observation times (if provided) were recorded. All these predicates took values 0 or 1; the temporal relation predicates were grounded by recorded times and took values in between 0 and 1 (see Appendix D for more details).

Prediction Accuracy. We first *predict the survival rate of each patient given her last recorded time in the dataset*. We compared the prediction accuracy of our temporal logic model with four classic baseline methods, RNN (Ge et al., 2018), LSTM (Suresh et al., 2017; Xu et al., 2018), Logistic Regression (LR), and dynamic Bayesian Networks (BN) (Loghmanpour et al., 2015) using the same test data consisting 100 patients. All models were trained on training data containing 50, 500, and 4,000 patients' trajectories, respectively, with the comparison results displayed in Table 4.

The inputs of all models are preprocessed predicate states (binary) and their recorded times[1]. Baseline models were established as follows: We considered a one-layer RNN and LSTM with 128 hidden units. One-hot embeddings of predicates were fed into the RNN and LSTM sequentially to predict the next-time symptoms and survival. For Logistic Regression and dynamic Bayesian Networks, time was discretized and the features were aggregated within a time interval to predict the current survival status. We used the same prior knowledge (logic rules) to construct the dependency structures for Bayesian Networks for a fair comparison. For our model, weights of logic rules were initialized as a small number, say, .001 (like the standard initialization for neural networks). To ensure non-negative weights, projected gradient descent was used in training.

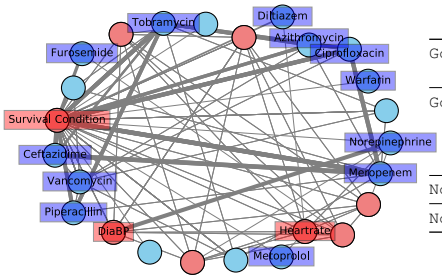
Our model performed fairly well and was consistently superior to the baselines on the considered small- and medium-sized problems, mainly due to better utilization of prior knowledge and the ability to capture complex temporal dependence in data. RNN and LSTM require more data to have a better fit. Logistic Regression is a parsimonious model, yet it aggregates the features in a linear and less-structured fashion. Bayesian Networks are ready to incorporate prior knowledge by representing conditional dependence in a directed graph, but it was time-consuming in inference.

Table 4. Predication Results of Survival Rate

Method	Train/Test: 50/100	500/100	4000/100
LSTM	0.405	0.420	0.436
RNN	0.439	0.442	0.424
LR	0.506	0.507	0.518
BN	0.530	0.570	0.540
Temp Logic	**0.584**	**0.647**	**0.682**

To validate continuous-time reasoning, we predicted the time to survival, and compared with a state-of-the-art point

[1]We also tried to construct inputs using one-hot embeddings of medications combined with real symptom values, but the performance degraded



Effective First-order Logic Rules for MIMIC-III

$\texttt{GoodSurvivalCondition}(t) \leftarrow \texttt{Use-Drug}(t') \wedge \texttt{Before}(t', t)$, where $\texttt{Drug} = \{\texttt{Ceftazidime, Tobramycin, Vancomycin, Diltiazem, Furosemide Warfarin, Ciprofloxacin}\}$

$\texttt{GoodSurvivalCondition}(t) \leftarrow \texttt{Use-Drug1}(t') \wedge \texttt{Use-Drug2}(t'') \wedge \texttt{Before}(t', t) \wedge \texttt{Before}(t'', t) \wedge \texttt{Before}(t', t'')$, where $(\texttt{Drug1}, \texttt{Drug2}) = \{(\texttt{Ceftriaxone, Meropenem}), (\texttt{Ciprofloxacin, Meropenem}), (\texttt{Ceftriaxone, Tobramycin}), (\texttt{Azithromycin, Tobramycin}) (\texttt{Ciprofloxacin, Tobramycin}), (\texttt{Piperacillin, Tobramycin})\}$

$\texttt{NormalHeartRate}(t) \leftarrow \texttt{Use-Metoprolol}(t') \wedge \texttt{Before}(t', t)$

$\texttt{NormalBloodPressure}(t) \leftarrow \texttt{Use-Norepinephrine}(t') \wedge \texttt{Before}(t', t)$

Figure 6. MIMIC-III. Left: Learned formula graph (width of the lines indicates the value of the weights). Right: Effective logic rules.

process model RMTPP (Du et al., 2016). RMTPP was implemented by using LSTM to extract all predicates' historical events (i.e., symptoms and drugs) as feature embeddings (size = 36) and formed the conditional intensity via nonlinear mappings (ReLU). All models were trained on 50, 500, and 4,000 patients, and were tested on 100 patients. The Mean Absolute Errors (MAE) of time to survival with the unit is the day are displayed in Table 5.

Table 5. Prediction Results of Time to Survival: Mean Absolute Errors (MAE) in Days

Method	Train/Test: 50/100	500/100	4000/100
RMTPP	0.464	0.398	0.399
Temp Logic	**0.233**	**0.252**	**0.227**

Our model yields smaller MAEs than RMTPP, and the results showcase the strength of our model in reasoning about the occurrence time for events in small data, due to incorporating domain knowledge to form conditional intensity, whereas the LSTM-based conditional intensities are data-hungry by nature.

Our model yields smaller MAEs than RMTPP, and the results showcase the strength of our model in reasoning about the occurrence time for events in small data, due to incorporating domain knowledge to form conditional intensity, whereas the LSTM-based conditional intensities are data-hungry by nature.

Knowledge Transfer Ability. We evaluated the knowledge transfer ability via splitting MIMIC dataset based on age information of patients. We tested whether the knowledge learned from a large dataset D1 (patient age: 19-44, sample size: 300) can be transferred to a small dataset D2 (patient age: 65-90, sample size: 100) with slightly different patterns. Consider three scenarios to set up the experiments: Benchmark (no transfer), train on dataset D3 (patient age: 65-90, sample size: 50) and test on D2; Transfer 1 (freeze the weights), train on D1 and test on D2; and Transfer 2 (fine-tune the weights), train on D1, warm start the model on D3, and test on D2. Table 6 shows the "prediction accuracy of the survival rate" of the models under three scenarios.

Table 6. Prediction Results of Survival Rate via Knowledge Transfer.

Method	Benchmark	Transfer1	Transfer2
RNN	0.511	0.507	0.517
LSTM	0.516	0.502	0.527
LP	0.519	0.514	0.522
BN	0.501	0.530	0.531
TempLogic	**0.615**	**0.637**	**0.659**

We also split the dataset by gender, and obtained similar trends. The above results demonstrate a better generalization and knowledge transfer power of our model — the learned relatively important logic rules can be transferred across datasets with similar tasks.

Interpretability. It is of value to understand what medical treatments are more effective than others. Although all introduced logic rules are likely to be true, some rules might be satisfied (or violated) more often than others, which will reflect in bigger (or smaller) values of the learned formula weights. Using the entire dataset, the learned formula graph and the picked effective logic rules (i.e., whose weights were significantly greater than 0) were shown in Fig. 6. From the results, we noticed that high frequent drug-usage did not necessarily yield high logic weights. For example, drug *Heparin* was of a relatively high use frequency (account for $0.128/1$ of all drug usage), but the associated logic rule was not effective. This means, when doctors prescribed these drugs to patients, they believed the drugs will insert positive effects; however based on patients' data, the drug effects were not as effective as expected. On the other hand, drug *Ceftazidime* was rarely used $(0.01/1)$ but the related rule was effective. These discoveries are interesting.

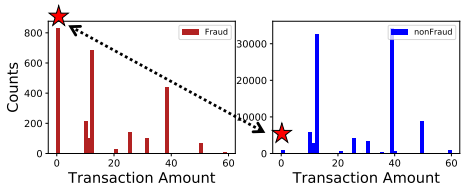
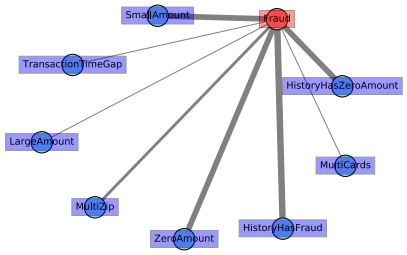


Figure 7. Transaction amount distributions: Fraud v.s. Normal.

5.3. Credit Card Fraud Detection

We used a credit card dataset from the "UCSD-FICO Data Mining Contest"(FICO-UCSD., 2009) to detect fraud trans-



Effective First-order Logic Rules for Fraud Transaction Detection

$\texttt{Is-Fraud}(t) \leftarrow \texttt{Has-FraudHistory}(t') \wedge \texttt{Before}(t', t)$
$\texttt{Is-Fraud}(t) \leftarrow \texttt{Is-ZeroAmount}(t') \wedge \texttt{Equal}(t', t)$
$\texttt{Is-Fraud}(t) \leftarrow \texttt{Is-SmallAmount}(t') \wedge \texttt{Equal}(t', t)$
$\texttt{Is-Fraud}(t) \leftarrow \texttt{Has-ZeroAmountHistory}(t') \wedge \texttt{Before}(t', t)$
$\texttt{Is-Fraud}(t) \leftarrow \texttt{Has-MultiZip}(t') \wedge \texttt{Before}(t', t)$

Figure 8. Fraud detection. Left: Learned formula graph (edge width is proportional to formula weight). Right: Effective logic rules.

Table 7. Defined Predicates for Credit Card Fraud Detection.

Signs of Fraud	$\texttt{Is-ZeroAmount}(t)$, $\texttt{Has-MultiCards}(t)$, $\texttt{Has-ZeroAmountHistory}(t)$, $\texttt{Is-LargeAmount}(t)$, $\texttt{Is-SmallAmount}(t)$, $\texttt{Has-LargeTransactionTimeGap}(t)$, $\texttt{Has-MultiZip}(t)$, $\texttt{Has-FraudHistory}(t)$
Fraud Condition	$\texttt{Is-Fraud}(t)$
Temporal Relation	$\texttt{Before}(t, t')$, $\texttt{Equal}(t, t')$

actions. The dataset is labeled, anonymous and imbalanced. It contains 100,000 (#Fraud: 2,654, #Normal: 97,346) transactions of 73,729 customers over 98 days. Each transaction has information such as account ID, transaction amount, transaction time, corresponding email address, zip-code, and location.

Table 8. Fraud Detection

Method	Train/Test: 500/1000	5000/1000	50000/1000
LSTM	0.632	0.633	0.637
RNN	0.634	0.621	0.633
LR	0.650	0.652	0.641
BN	0.636	0.649	0.649
Temp Logic	**0.728**	**0.754**	**0.762**

Predicates and Logic Rules. Logic rules were defined from domain knowledge and exploratory data analyses. Some typical (but not absolute) signs of credit card fraud, such as "declined purchases followed by small ones" or "one card but multiple different addresses" (Delamaire et al., 2009) guided us design the rules. We also did preliminary analyses of the data, and especially focused on the distinct properties of fraud transactions. For example, as demonstrated in Fig. 7, fraud transactions yield an evident higher proportion of extremely small transactions (e.g. zero transaction amount) and this observation was summarized as a rule. See Table 7 for the defined predicates. The set of first-order logic rules can be found in the Appendix B.

Data Pre-Processing. Predicates were sequentially grounded given credit card's transaction trajectory, with state transition times recorded. We thus obtained a list of events, i.e.,

$$([\texttt{account ID}], [\texttt{predicate ID}], [\texttt{time}], [\texttt{state}]).$$

Predicates, such as Is-Fraud, or Is-ZeroAmount, are about transaction properties; they were grounded simultaneously once a transaction occurred. Other predicates that record the history information, such as Has-FraudHistory, were triggered to be 1 the first time the condition was satisfied. Thresholds that determined small amount (i.e., >0 and <5) and large amount (i.e., >50) were set from the exploratory data analyses.

Prediction Accuracy. Our goal is to *detect fraud credit transactions on the fly*. Similarly, we compared our model with LSTM, RNN, Logistic Regression, and dynamic Bayesian Networks. The baselines were set up similar to the MIMIC-III dataset. Since fraud transactions are rare events, we found if we trained LSTM and RNN using the original imbalanced training datasets, they had a very low prediction accuracy in testing; we oversampled the fraud samples in training with ratio $1/2$ at each batch which improved the performance. The inputs features for all models were binary predicates for a fair comparison. We noticed that this feature construction has already incorporated prior knowledge and it empirically increased the model performance than directly using the raw attributes of transactions as features. We created a more balanced test dataset to make the prediction accuracy reasonable, with fraud ratio 0.4303. The prediction accuracy results were demonstrated in Table 8. The performance of our model is fairly well even using a handful of training samples. For this imbalanced dataset, detecting fraud is like looking for a needle in a haystack, and the temporal logic model provides a method to incorporate prior knowledge to effectively search for evidence which aids the prediction.

Interpretability. The learned formula graph and discovered effective logic rules were demonstrated in Fig. 8, which showed the key factors in reasoning about fraud transactions. Interestingly, the fraud transactions are more likely to be frequent and consecutive small amount transactions, rather than a transaction of extremely large amount.

6. Conclusion

In this paper, we proposed a unified framework that integrates first-order temporal logic rules as prior knowledge into point processes. Our model is easy to interpret, works well on small data, and thus has wide applications in real domains such as healthcare and finance.

Acknowledgments

This work was supported in part by NSF CAREER CCF-1650913, DMS-1938106, and DMS-1830210 to Y.X. This work was supported in part by NSF grants CDS&E-1900017 D3SC, CCF-1836936 FMitF, IIS-1841351, CAREER IIS-1350983 to L.S. This work was supported by Ant Group through Ant Financial Research Program.

References

Achab, M., Bacry, E., Gaïffas, S., Mastromatteo, I., and Muzy, J.-F. Uncovering causality from multivariate hawkes integrated cumulants. *The Journal of Machine Learning Research*, 18(1):6998–7025, 2017.

Allen, J. F. Maintaining knowledge about temporal intervals. In *Readings in qualitative reasoning about physical systems*, pp. 361–372. Elsevier, 1990.

Bacry, E., Mastromatteo, I., and Muzy, J.-F. Hawkes processes in finance. *Market Microstructure and Liquidity*, 1 (01):1550005, 2015.

Boyd, S. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Brendel, W., Fern, A., and Todorovic, S. Probabilistic event logic for interval-based event recognition. In *CVPR 2011*, pp. 3329–3336. IEEE, 2011.

Chorowski, J. and Zurada, J. M. Learning understandable neural networks with nonnegative weight constraints. *IEEE transactions on neural networks and learning systems*, 26(1):62–69, 2014.

Delamaire, L., Abdou, H., and Pointon, J. Credit card fraud and detection techniques: a review. *Banks and Bank systems*, 4(2):57–68, 2009.

Doshi-Velez, F. and Kim, B. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

Du, N., Dai, H., Trivedi, R., Upadhyay, U., Gomez-Rodriguez, M., and Song, L. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1555–1564. ACM, 2016.

Farajtabar, M., Du, N., Rodriguez, M. G., Valera, I., Zha, H., and Song, L. Shaping social activity by incentivizing users. In *Advances in neural information processing systems*, pp. 2474–2482, 2014.

Farajtabar, M., Wang, Y., Rodriguez, M. G., Li, S., Zha, H., and Song, L. Coevolve: A joint point process model for information diffusion and network co-evolution. In *Advances in Neural Information Processing Systems*, pp. 1954–1962, 2015.

FICO-UCSD. *FICO Credit Card Dataset*, 2009. URL https://ebiquity.umbc.edu/blogger/2009/05/24/ucsd-data-mining-contest/.

Ge, W., Huh, J.-W., Park, Y. R., Lee, J.-H., Kim, Y.-H., and Turchin, A. An interpretable icu mortality prediction model based on logistic regression and recurrent neural networks with lstm units. In *AMIA Annual Symposium Proceedings*, volume 2018, pp. 460. American Medical Informatics Association, 2018.

Hotchkiss, R. S. and Karl, I. E. The pathophysiology and treatment of sepsis. *New England Journal of Medicine*, 348(2):138–150, 2003.

Jacobsen, M. *Point process theory and applications: marked point and piecewise deterministic processes*. Springer Science & Business Media, 2006.

Jarrett, D., Yoon, J., and van der Schaar, M. Dynamic prediction in clinical survival analysis using temporal convolutional networks. *IEEE journal of biomedical and health informatics*, 2019.

Johnson, A. E., Pollard, T. J., Shen, L., Li-wei, H. L., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L. A., and Mark, R. G. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

Kersting, K. and De Raedt, L. Towards combining inductive logic programming with bayesian networks. In *International Conference on Inductive Logic Programming*, pp. 118–131. Springer, 2001.

Kersting, K., De Raedt, L., and Raiko, T. Logical hidden markov models. *Journal of Artificial Intelligence Research*, 25:425–456, 2006.

Lee, C. H. and Yoon, H.-J. Medical big data: promise and challenges. *Kidney research and clinical practice*, 36(1): 3, 2017.

Li, S., Xiao, S., Zhu, S., Du, N., Xie, Y., and Song, L. Learning temporal point processes via reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 10781–10791, 2018.

Loghmanpour, N. A., Kanwar, M. K., Druzdzel, M. J., Benza, R. L., Murali, S., and Antaki, J. F. A new bayesian network-based risk stratification model for prediction of short-term and long-term lvad mortality. *ASAIO journal (American Society for Artificial Internal Organs: 1992)*, 61(3):313, 2015.

Mei, H. and Eisner, J. M. The neural hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems*, pp. 6754–6764, 2017.

Nachimuthu, S. K., Wong, A., and Haug, P. J. Modeling glucose homeostasis and insulin dosing in an intensive care unit using dynamic bayesian networks. In *AMIA Annual Symposium Proceedings*, volume 2010, pp. 532. American Medical Informatics Association, 2010.

Natarajan, P. and Nevatia, R. Hierarchical multi-channel hidden semi markov models. In *IJCAI*, volume 7, pp. 2562–2567, 2007.

Natarajan, S., Bui, H. H., Tadepalli, P., Kersting, K., and Wong, W.-K. Logical hierarchical hidden markov models for modeling user activities. In *International Conference on Inductive Logic Programming*, pp. 192–209. Springer, 2008.

Ogata, Y. Statistical models for earthquake occurrences and residual analysis for point processes. *Journal of the American Statistical association*, 83(401):9–27, 1988.

Ogata, Y. and Vere-Jones, D. Inference for earthquake models: a self-correcting model. *Stochastic processes and their applications*, 17(2):337–347, 1984.

Papai, T., Kautz, H., and Stefankovic, D. Slice normalized dynamic markov logic networks. In *Advances in Neural Information Processing Systems*, pp. 1907–1915, 2012.

Qian, Z., Alaa, A. M., Bellot, A., Rashbass, J., and van der Schaar, M. Learning dynamic and personalized comorbidity networks from event data using deep diffusion processes. *arXiv preprint arXiv:2001.02585*, 2020.

Reynaud-Bouret, P., Schbath, S., et al. Adaptive estimation for hawkes processes; application to genome analysis. *The Annals of Statistics*, 38(5):2781–2822, 2010.

Richardson, M. and Domingos, P. Markov logic networks. *Machine learning*, 62(1-2):107–136, 2006.

Singla, P. and Domingos, P. M. Lifted first-order belief propagation. In *AAAI*, volume 8, pp. 1094–1099, 2008.

Smullyan, R. R. *First-order logic*, volume 43. Springer Science & Business Media, 2012.

Suresh, H., Hunt, N., Johnson, A., Celi, L. A., Szolovits, P., and Ghassemi, M. Clinical intervention prediction and understanding using deep networks. *Machine Learning for healthcare Conference*, 2017.

Tran, S. D. and Davis, L. S. Event modeling and recognition using markov logic networks. In *European Conference on Computer Vision*, pp. 610–623. Springer, 2008.

Xu, Y., Biswal, S., Deshpande, S. R., Maher, K. O., and Sun, J. Raim: Recurrent attentive and intensive model of multimodal patient monitoring data. *KDD*, 2018.

Zhao, Q., Erdogdu, M. A., He, H. Y., Rajaraman, A., and Leskovec, J. Seismic: A self-exciting point process model for predicting tweet popularity. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1513–1522. ACM, 2015.