Neuro-Symbolic Temporal Point Processes

Yang Yang [1]   Chao Yang [1]   Boyang Li [1]   Yinghao Fu [1]   Shuang Li [1]

Abstract

Our goal is to *efficiently* discover a compact set of temporal logic rules to explain irregular events of interest. We introduce a neural-symbolic rule induction framework within the temporal point process model. The negative log-likelihood is the loss that guides the learning, where the explanatory logic rules and their weights are learned end-to-end in a *differentiable* way. Specifically, predicates and logic rules are represented as *vector embeddings*, where the predicate embeddings are fixed and the rule embeddings are trained via gradient descent to obtain the most appropriate compositional representations of the predicate embeddings. To make the rule learning process more efficient and flexible, we adopt a *sequential covering algorithm*, which progressively adds rules to the model and removes the event sequences that have been explained until all event sequences have been covered. All the found rules will be fed back to the models for a final rule embedding and weight refinement. Our approach showcases notable efficiency and accuracy across synthetic and real datasets, surpassing state-of-the-art baselines by a wide margin in terms of efficiency.

1. Introduction

Explaining critical events, such as sudden health changes or unusual transactions, is essential in high-stakes domains like healthcare and finance. The dynamics of these events are typically governed by temporal logic rules, and automatically uncovering these rules from data holds significant scientific and practical value.

For example, in healthcare, it is desirable to compress and summarize medical knowledge or clinical experiences regarding disease phenotypes and therapies to a collection of temporal logic rules. The discovered rules can contribute to the sharing of clinical experiences and aid in the improvement of the treatment strategy. They can also provide specific explanations for the occurrence of an event. For example, the following clinical report

"*A 50-year-old patient, with a chronic lung disease since 5 years ago, took the booster vaccine shot on March 1st. The patient got exposed to the COVID-19 virus around May 12th, and afterward within a week began to have a mild cough and nasal congestion. The patient received treatment as soon as the symptoms appeared. After intravenous infusions at a healthcare facility for around 3 consecutive days, the patient recovered...*"

contains many clinical events with timestamps recorded. It sounds appealing to distill compact human-readable temporal logic rules from these noisy event data to aid diagnoses and treatment planning. In this paper, we present an *efficient* neural-symbolic rule induction algorithm capable of automatically learning *universal* rules from sequences of irregular event data. These universal rules act as summarized laws that effectively elucidate the dynamics of the events, offering valuable insights for clinical decision-making.

From modeling perspective, we design a *neural-symbolic temporal point process* (NS-TPP) that strikes the balance between model flexibility and interpretability. The occurrence rate (i.e., intensity) of events is a function of the neural predicate embeddings, where the functional form is determined by the logic rules that are uncovered from the data. Traditional parametric temporal point process (TPP) models like the Hawkes process offer interpretability, but their simplicity limits flexibility. Conversely, neural-based models, such as RMTPP (Du et al., 2016) and Transformer Hawkes (Zuo et al., 2020), provide expressiveness but are often criticized for their black-box nature and hinder their applications in high-stakes scenarios. Our NS-TPP strives to harness the strengths of both paradigms.

To enable *efficient* and *differentiable* rule learning, we propose a *neural-symbolic rule induction* framework for TPP, which aims to learn rule embeddings to identify the rule formula. In our model, predicates, or logic variables, are represented as fixed vector embeddings, either pre-trained or specified beforehand. Each rule embedding acts as a *learnable filter*, selecting the most relevant predicates and

---

[1]School of Data Science, The Chinese University of Hong Kong (Shenzhen). Correspondence to: Shuang Li <lishuang@cuhk.edu.cn>.

evidence from observational facts to form logical rules. During the forward pass, these filters scan predicate embeddings to find the best matches and combined with the observed events as fact, these filters generate logic-informed features. This forward pass can be thought of as using the rule content filter on the historical events to gather evidence, which will then be used to deduce the occurrence of the event of interest. In the backward pass, we calculate the loss as the negative log-likelihood based on a temporal point process. The rule embedding parameters are then optimized end-to-end through gradient descent.

Furthermore, to boost *flexibility* in rule learning, we utilize a *sequential covering* algorithm. This method involves progressively adding rules to the model and learning each rule embedding one by one. When a new rule is identified (i.e., the learning of the new rule embedding converges), the event sequences it explains are removed from the dataset. This rule learning process continues until all events are covered, which naturally eliminates the need to specify the total number of rules in advance. After identifying all rule embeddings and their weights using the sequential covering algorithm, we jointly refine the rule embeddings and weights by considering the full NS-TPP model. In this way, we further enhance the accuracy of rule embeddings and weights by maximizing the likelihood.

In summary, our contributions are as follows:

(*i*) Our NS-TPP model incorporates a neural-symbolic intensity function, striking a balance between flexibility and interpretability. By converting model structure learning into rule embedding learning, the discovered rule set automatically determines the model capacity and structure. All the rule embeddings and other model parameters will be learned in a differentiable way.

(*ii*) Our neural-symbolic rule induction algorithm naturally withstands input noise. This resilience is achieved through encoding rule content and predicates using embeddings. By computing features based on similarity scores among rule embedding, predicate embedding, and relevant facts, our approach ensures robustness to noisy inputs.

(*iii*) We improve rule discovery efficiency and flexibility by implementing a covering algorithm, dividing the complete learning problem into manageable sub-problems. Our algorithm's efficiency and accuracy are validated on both synthetic and real data, demonstrating approximately 100 times greater efficiency.

2. Related Work

We will compare our method with some existing works from the following aspects.

Temporal Point Process (TPP) Models   TPP models have emerged as an elegant framework for modeling event times and types in continuous time, directly treating the inter-event times as random variables. Advances in this field have largely concentrated on enhancing the flexibility of intensity functions to improve event prediction accuracy. Pioneering works such as the RMTPP (Du et al., 2016) and the continuous-time RNN further improved from RMTPP (Mei & Eisner, 2017) introduced recurrent neural network-based approaches to model the intensity functions. More recent studies by Zuo et al. (2020) and Zhang et al. (2020) have applied the self-attention mechanism to address long-term event dependencies, showcasing the potential of leveraging attention-based deep learning techniques for TPP. Despite these advancements, the reliance on black-box models raises significant interpretability issues, particularly in contexts requiring explanations for events, such as root cause analysis for abnormal events. This gap highlights the increasing agreement on the need for inherently interpretable models, as emphasized by Rudin (2019), to ensure the transparency of the decision-making in high-stakes systems.

In response to these challenges, Li et al. (2020; 2021) proposed integrating logic rules within the intensity function to foster interpretability. However, their methods either assume that the logic rules are prespecified or rely on a non-differentiable rule learning process, distinguishing our approach which offers a differentiable framework for rule learning.

Rule Mining   Discovering rules from data in an unsupervised manner has long been a challenging task. Unsupervised logic rule mining is about discovering inherent patterns in data without any prior labeling. Traditional approaches, such as Itemset Mining Methods like Apriori (Agrawal et al., 1994) and NEclatclosed (Aryabarzan & Minaei-Bidgoli, 2021), focus on identifying frequent itemsets. However, they cannot be directly adapted to events with recorded occurrence times, limiting their applicability in temporal datasets. On the other hand, Sequential Pattern Mining methods like CM-SPADE (Fournier-Viger et al., 2014a) and VGEN (Fournier-Viger et al., 2014b) aim to uncover temporal relationships in datasets. However, they only utilize the temporal ordering of events and are unable to effectively incorporate fine-grained timestamp information, which can lead to precision issues in rule mining.

Supervised logic rule mining requires labeled data, consisting of both positive and negative samples. The rules are mined usually under the principle that, for positive samples, at least one rule must be satisfied, and for negative samples, none of the rules should be satisfied. Among supervised rule mining methods, a notable example is Inductive Logic Programming (ILP) (Srinivasan, 2001), which provides a structured framework for rule learning. However,

ILP typically requires a balanced mix of positive and negative examples to achieve effective rule learning. The ILP methods can be categorized into forward-chaining methods (Campero et al., 2018; Payani & Fekri, 2019), which generate and test rules through iterative deductive reasoning, and backward-chaining methods (Minervini et al., 2018; Yang et al., 2022), which dynamically construct rules to satisfy specific queries or goals. These approaches, despite their innovative attempts at rule induction, often operate as opaque models. They lack the ability to clearly explain the reasoning behind their inferences, making them more like black boxes than interpretable systems.

Our NS-TPP learns temporal logic rules from data in an unsupervised manner, utilizing fine-grained temporal information without requiring positive or negative labeling. This extends unsupervised temporal logic rule discovery methods, broadening the scope of rule learning without relying on labeled data.

3. Background

Predicates and Temporal Logic Rules Define a set of predicates as $\mathcal{X}$, where each variable $X_u \in \mathcal{X}$ is a boolean logic variable. Denote the target predicate we aim to explain as $Y \in \mathcal{X}$. For example, Y could represent a sudden change in a patient's health, an unusually large transaction, or an alarm in manufacturing. We assume that the target predicate can be explained by a set of Horn rules (i.e., if-then rules) with temporal ordering constraints, each having the general form:

$$f : \quad Y \leftarrow \left(\bigwedge_{X_u \in \mathcal{X}_f} X_u \right) \bigwedge \left(\bigwedge_{X_u, X_v \in \mathcal{X}_f} R\left(X_u, X_v\right) \right) \quad (1)$$

where $\mathcal{X}_f$ is the set of body predicates associated with rule f, and $R\left(X_u, X_v\right)$ represents temporal relations between each paired predicates X_u and X_v. These relations, categorized as "Before", "Equal", "After" or "None", define the temporal constraints between X_u and X_v, with "None" specifying the absence of any temporal relation.

Temporal Point Process (TPP) Consider adding a temporal dimension to the previously defined static predicates, and the grounded predicates by observed data (i.e., fact) results in a list of spiked events, denoted as $\{X_u(t)\}_{t \geq 0}$, where each $X_u(t) \in \{0, 1\}$ at any time $t \geq 0$. Specifically, $X_u(t)$ transitions instantaneously from 0 (False) to 1 (True) at the timestamp when the event occurs. In our context, each event sequence sample represents a $|\mathcal{X}|$-dimensional multivariate temporal point process. We use $\mathcal{H}_{t^-} = \{X_u(t)\}_{u=1,\ldots,|\mathcal{X}|}$ to denote all the observed events up to but not including t.

We are interested in modeling and learning logical explanations to the occurrence of the target event sequence $\{Y(t)\}_{t \geq 0}$ with event time recorded as $\{t_1, t_2, \ldots\}$. We treat the inter-event time intervals as random variables and

the duration until the next event Y is characterized by the conditional intensity function, denoted as $\lambda\left(t \mid \mathcal{H}_{t^-}\right)$. By definition,

$$\lambda(t \mid \mathcal{H}_{t^-})dt = \mathbb{E}\left[N([t, t + dt]) \mid \mathcal{H}_{t^-}\right]$$

where $N([t, t + dt])$ denotes the number of events occurring in the interval $[t, t + dt]$. Given the occurrence time of event Y, such as $(t_1, \ldots, t_n)$, the joint likelihood function of the data is computed by $p(t_1, \ldots, t_n) = \prod_{i=1}^{n} p^*(t_i)$ using the chain rule, where the conditional probability

$$p^*(t_i) = \lambda^*(t_i) \exp\left(-\int_{t_{i-1}}^{t_i} \lambda^*(s)ds\right).$$

Here, to simplify the notation, we denote $p^*(t) := p\left(t \mid \mathcal{H}_{t^-}\right)$ and $\lambda^*(t) := \lambda\left(t \mid \mathcal{H}_{t^-}\right)$.

In this paper, we will model $\lambda^*(t)$ using neural-symbolic features, and we name our model as NS-TPP. Moreover, we aim to design a neural-symbolic rule induction algorithm to efficiently uncover the logic rule set $\mathcal{F} := \{f_1, f_2, \ldots, f_H\}$ and learn other continuous model parameters jointly through maximizing the likelihood by gradient descent. Given the learned NS-TPP, we can deduce and explain the occurrence of target events in a *probabilistic* and *continuous-time* manner.

4. Neural-Symbolic Temporal Point Process (NS-TPP)

4.1. Neural-Symbolic Feature Construction

The core idea of our proposed NS-TPP is to formulate the neural-symbolic features to construct the intensity function for $\{Y(t)\}_{t \geq 0}$. Let's temporarily assume that all rules are known, denoted as $\mathcal{F}$, and we model the intensity function as:

$$\lambda^*\left(t \mid \mathcal{F}\right) = b_0 + \sum_{f \in \mathcal{F}} \gamma_f \phi_f\left(\mathcal{H}_{t^-}\right) \quad (2)$$

where b_0 represents the base term independent of rules, γ_f denotes the impact weight of each rule $f \in \mathcal{F}$, and $\phi_f(\cdot)$ is the neural-symbolic feature that depends on rules and data. We will elaborate on how to compute the neural-symbolic feature $\phi_f\left(\mathcal{H}_{t^-}\right)$ below and the overall framework is illustrated in Figure 1.

Predicate Embedding For each predicate $X \in \mathcal{X}$, we represent it as a row embedding vector. All the predicate embeddings are denoted as $\boldsymbol{k}_1, \boldsymbol{k}_2, \ldots, \boldsymbol{k}_{|\mathcal{X}|}$, each of dimension d, i.e., $\boldsymbol{k}_i \in \mathbb{R}^{1 \times d}$. These predicate embeddings can be obtained through pretraining or prespecification. These embeddings can take various forms, such as one-hot representations, which are simple binary vectors, or dense vector
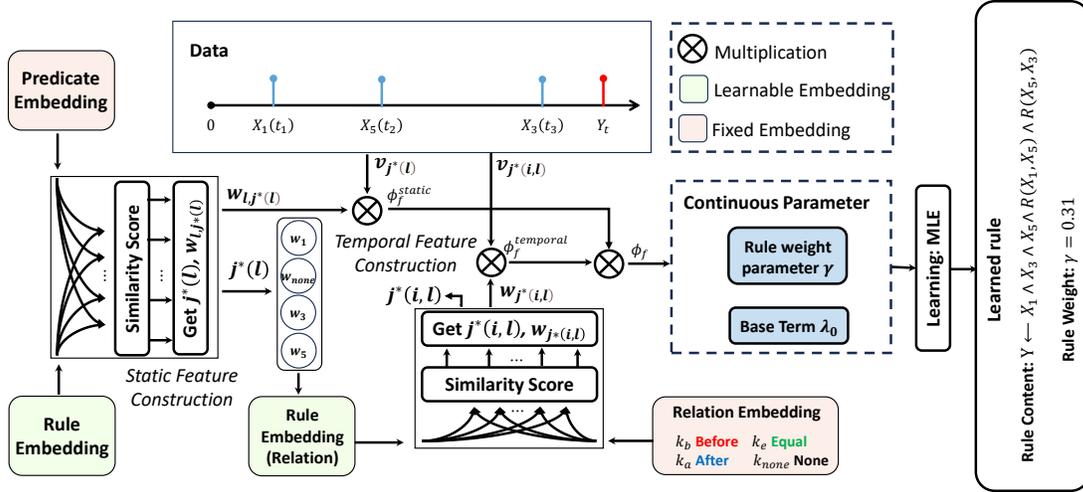
Figure 1. Overview of our neural-symbolic framework for temporal logic induction. The framework begins with preparing fixed predicate embeddings. During the forward pass, rule embeddings scan these predicate embeddings to identify optimal compositional matches. A modified attention-like block then integrates these matches with observed events to generate static neural-symbolic features. Temporal relations are then incorporated to create neural-symbolic temporal features given the already selected predicate pairs. We will use the same matching idea to obtain the temporal neural-symbolic features. The final neural-symbolic features by combing static and temporal parts will be used to compute the intensity function and the likelihood. We will adopt MLE to learn all the rule embedding parameters and other continuous model parameters (rule weights and base term) in a differentiable way. The overall rule learning scheme employs the sequential covering algorithm to learn rules progressively until no further rules can be added. Finally, all the rule embeddings and other model parameters will be refined to optimize the likelihood.

embeddings extracted from pretrained models like neural TPP (e.g., Transformer Hawkes), which may capture the semantic dependency between predicates. We also introduce and specify a dummy predicate embedding (e.g., as a zero vector), denoted as k_0, to signify a predicate with no semantic meaning. We will show later that the introduction of the dummy predicate embedding is to accommodate various rule lengths in rule learning.

Regardless of how the predicate embeddings are obtained, it is essential that each predicate embedding is distinct and carries concrete semantic meaning. This is crucial for interpreting the rule formula from the learned rule embeddings. We denote the stacked predicate embedding matrix as $\boldsymbol{K} = [\boldsymbol{k}_0; \boldsymbol{k}_1; \dots, \boldsymbol{k}_{|\mathcal{X}|}] \in \mathbb{R}^{(|\mathcal{X}|+1) \times d}$.

Rule Embedding   Now, let's introduce the *rule embedding* that will be learned from data to indicate a rule formula f. Each rule embedding will act as a learnable filter to compute the similarity score with the predicate embeddings, selecting predicates to form a rule and gather evidence from data to construct the neural-symbolic feature.

Each rule embedding encodes one rule. Suppose we aim to learn a rule with length L, we will initialize the rule embedding as $\boldsymbol{Q}_f = [\boldsymbol{q}_1; \boldsymbol{q}_2; \dots; \boldsymbol{q}_L] \in \mathbb{R}^{L \times d}$, where each row vector $\boldsymbol{q}_l \in \mathbb{R}^{1 \times d}$, sharing the same dimension with the predicate embedding, and L indicates the *maximum* rule length.

Neural-Symbolic Feature   The fundamental concept behind the proposed rule induction is that the rule embedding $\boldsymbol{Q}_f$ can be regarded as L slots to be filled in by predicate embeddings. Learning the rule embedding will dynamically decide which predicates to select to form the rule, and the selection score is based on the similarity between each current rule embedding vector and all the predicate embedding vectors. Written in matrix form, we can determine which predicate embeddings to fill in each rule embedding slot by computing the similarity score:

$$\boldsymbol{W} = \text{softmax}(\boldsymbol{Q}_f \boldsymbol{K}^\top / \tau) \tag{3}$$

where softmax is applied row-wise. The similarity score will serve as the selection probability.

For example, suppose we aim to fill in $\boldsymbol{q}_l$, we will compute the similarity score of current $\boldsymbol{q}_l$ with all candidate predicate embeddings $[\boldsymbol{k}_0; \boldsymbol{k}_1; \dots, \boldsymbol{k}_{|\mathcal{X}|}]$ to find the best match to fill in. This is realized by first computing the (soft) selection score as

$$w_{lj} = \frac{\exp\left(\boldsymbol{q}_l \boldsymbol{k}_j^\top / \tau\right)}{\sum_{j'=0,1,\dots,|\mathcal{X}|} \exp\left(\boldsymbol{q}_l \boldsymbol{k}_{j'}^\top / \tau\right)}, \quad j = 0, \dots, |\mathcal{X}| \tag{4}$$

where τ is the temperature (hyperparameter) that controls the approximation error of the softmax function with the (hard) max function. Each element satisfies $0 < w_{lj} < 1$,

and each row ensures $\sum_j w_{lj} = 1$. Therefore, w_{lj} can be interpreted as the selection probability of predicate j to slot l. For each row l, the highest score index, denoted as $j^*(l) = \arg\max_j\{w_{lj}\}$, yields the predicate (embedding) to be selected to fill in l. In practice, however, we will choose to sample the best-matching predicate index according to the softmax function to introduce randomness. The additional noise may aid the rule embedding learning, preventing convergence to (very bad) suboptimal rule embeddings. This sampling from the softmax can be achieved by injecting Gumbel noise, i.e.,

$$j^*(l) = \underset{j \in \{0,\ldots,|\mathcal{X}|\}}{\text{argmax}} \{\boldsymbol{q}_l \boldsymbol{k}_j^\top / \tau + \epsilon_j\} \qquad (5)$$

where each $\epsilon_j \sim \text{Gumbel}(0,1)$.

It is worth mentioning that $j^*(l)$ can be equal to 0, meaning that the best-matching predicate for slot l in the rule embedding is the dummy predicate embedding. By filling in the slot by dummy predicate embedding, we thereby have the flexibility to learn the rules with lengths smaller than L.

We have discussed how to determine a rule formula by selecting predicate embeddings to fill in the rule embedding. Next, we will discuss how to ground the rule using data to construct the neural-symbolic feature. Let's temporarily ignore any temporal relations in each rule and consider a general static horn rule, such as $f : \quad Y \leftarrow \left(\bigwedge_{X_u \in \mathcal{X}_f} X_u\right)$, where $\mathcal{X}_f$ is formed by the selected predicates and $|\mathcal{X}_f| = L$. The neural-symbolic feature associated with this static rule can be represented as:

$$\phi_f^{\text{static}}(\mathcal{H}_{t^-}) = \underbrace{\prod_{l=1,\ldots,L} w_{l,j^*(l)}}_{\text{similarity score}} \underbrace{\prod_{l=1,\ldots,L} v_{j^*(l)}}_{\text{fact}}. \qquad (6)$$

Here, each $j^*(l)$, where $l = 1,\ldots,L$, is determined by sampling, and each element $0 < w_{l,j^*(l)} < 1$ is the corresponding similarity score. $v_{j^*(l)}$, which indicates the fact, is queried from the historical events $\mathcal{H}_{t^-}$. If the corresponding event has ever occurred (i.e., the temporal predicate $X_{j^*(l)}$ has been once grounded as True), then $v_{j^*(l)} = 1$; otherwise, $v_{j^*(l)} = 0$.

Connection to Attention Let's pause here to draw an analogy of our neural-symbolic feature construction (as shown in Eq. (6)) to the Attention mechanism (Vaswani et al., 2017). Recall that Attention is defined based on queries $\boldsymbol{Q} \in \mathbb{R}^{n \times d}$, keys $\boldsymbol{K} \in \mathbb{R}^{m \times d}$, and values $\boldsymbol{V} \in \mathbb{R}^{m \times v}$, and the output is computed by a weighted sum:

$$\text{Attn}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}}\right)\boldsymbol{V} \in \mathbb{R}^{n \times v}.$$

We see that the way that we construct the neural-symbolic feature is similar to the attention mechanism. During the forward pass, the rule embedding (serving as query) scans across predicate embeddings (serving as keys) to find the best compositional match. Combined with observed events (serving as values), these filters produce logic-informed features.

However, our mechanism is a stricter form of attention. Instead of using all the similarity (attention) scores to compute a weighted sum output, our module approximately obtains the highest similarity score by sampling and discards all the remaining similarity weights. We use multiplication instead of summation, reflecting the nature of logic rules where all body conditions must be satisfied simultaneously for the rule to trigger. Additionally, our keys are fixed predicate embeddings with prespecified semantic meanings, which remain frozen during training.

Adding Temporal Relations Until now, we have not taken into account any temporal relation constraints in the rule. Nevertheless, the neural-symbolic rule induction framework described above can be readily expanded to incorporate the learning of temporal relations. Building on the same idea, we can introduce a prespecified or pre-trained predicate embedding to signify temporal relations "Before", "Equal", "After" and "None". This yields a matrix embedding $\boldsymbol{K}_r := [\boldsymbol{k}_b; \boldsymbol{k}_e; \boldsymbol{k}_a; \boldsymbol{k}_{none}] \in \mathbb{R}^{4 \times d}$. We can learn the rule embedding $\boldsymbol{Q}_f^r := [\boldsymbol{q}_{12}; \boldsymbol{q}_{13}; \ldots; \boldsymbol{q}_{L-1,L}] \in \mathbb{R}^{\frac{L(L-1)}{2} \times d}$ to specify what temporal relation constraints should be included to the rule. Specifically, $\boldsymbol{q}_{l-1,l}$ indicates the temporal relation types of the selected predicate in slot $l-1$ and l. The rule embedding $\boldsymbol{Q}_f^r$ will also be filled in by the temporal predicate embedding, with the similarity scores (i.e. selection probabilities) $\boldsymbol{W}$ computed as Eq. (3).

To determine the best-matching temporal predicate embedding to fill in the rule embedding, similarly, one can sample an index, $\boldsymbol{j}^* \sim \text{softmax}(\boldsymbol{Q}_f^r \boldsymbol{K}_r^\top / \tau)$. The selected relation type is interpretable based on the sampled index for each row. The neural-symbolic feature focusing solely on temporal relations can be expressed as:

$$\phi_f^{\text{temporal}}(\mathcal{H}_{t^-}) = \underbrace{\prod_{i,l=1,\ldots,L; i<l} w_{j^*(i,l)}}_{\text{similarity score}} \underbrace{\prod_{i,l=1,\ldots,L; i<l} v_{j^*(i,l)}}_{\text{fact}}. \qquad (7)$$

Here, each fact $v_{j^*(i,l)}$ is queried from the historical events $\mathcal{H}_{t^-}$ by checking their temporal relations. Specifically, for $j^* \in \{before, equal, after, none\}$

$$v_{j^*(i,l)} = \begin{cases} \mathbb{1}\{t_i - t_l < -\delta\} & j^* = before \\ \mathbb{1}\{|t_i - t_l| \le \delta\} & j^* = equal \\ \mathbb{1}\{t_i - t_l > \delta\} & j^* = after \\ 1 & j^* = none \end{cases} \qquad (8)$$

Here, $\delta \geq 0$ is specified as the tolerance to accommodate data noise. Considering the general logic rule defined in Eq. (1), the neural-symbolic feature by combining the static and temporal parts is computed as:

$$\phi_f(\mathcal{H}_{t-}) = \phi_f^{\text{static}}(\mathcal{H}_{t-}) \cdot \phi_f^{\text{temporal}}(\mathcal{H}_{t-}). \qquad (9)$$

The calculation is to reflect that the body conditions are satisfied only when both the static part and the temporal relation part are simultaneously true.

4.2. More Robust Feature Construction

In the feature construction process (as detailed in Eq. (6), (7), and (9)), the product of terms, though each close to 1, tends to decrease significantly as the number of terms increases. To maintain numerical stability, we opt for the minimum function over the product, replacing $x_1 x_2 \ldots x_N$ with $\min\{x_1, x_2, \ldots, x_N\}$. This choice ensures stability and aligns with the logical interpretation that a true rule requires each condition within it to be true. Although the minimum function is not differentiable, we address this by employing a differentiable approximation known as the soft-min function, represented as:

$$f(x) = -\frac{1}{\rho} \log \frac{1}{N} \sum_{i=1}^{N} e^{-\rho x_i}, \qquad (10)$$

which approaches $\min_i |x_i|$ as $\rho \to +\infty$. This function is used to compute $\phi_f(\mathcal{H}_{t-})$, where each x_i takes values of w and v.

5. Learning

We've discussed how to construct the NS-TPP intensity using a differentiable feedforward computational graph, which allows for the learning of rules (rule embeddings $\boldsymbol{Q}$) and other continuous model parameters (such as b_0 and $[\gamma_f]_{f \in \mathcal{F}}$ as detailed in Eq. (2)) through (stochastic) gradient descent to maximize data likelihood. To learn the entire rule set, we propose a more flexible learning strategy using the sequential covering algorithm.

This involves learning rules one by one progressively. We start with an empty set $\mathcal{F} = \emptyset$. We will learn the first rule by optimizing its rule embedding and weight using the following intensity model (constructed by a single rule) by stochastic gradient descent to maximize the likelihood:

$$\lambda^*(t \mid \mathcal{F}) = b_0 + \gamma_f \phi_f(\mathcal{H}_{t-}). \qquad (11)$$

Once the optimization converges, we store the rule embedding and weight, and remove the event sequences that have been explained by this discovered rule. We update $\mathcal{F} = \{f_1\}$ and continue this process for a subsequent rule, assuming the same model as shown in Eq. (11). This procedure continues until no new rules can be added (i.e., all

the event sequences have been covered). Or more often in practice we can terminate the procedure when the new discovered rule yields weight becoming smaller than some threshold. As a last step, we use the stored rule embeddings and weights to build a full model, and continue to refine the rule embeddings and weights for more accurate global model learning.

Our proposed dynamic approach eliminates the need to predefine the total number of rules, allowing the data to guide the model growing process. Additionally, we break down the overall rule problem into manageable subproblems, which simplifies the learning.

Model Interpretation For each temporal rule, the final rule formula can be directly obtained by checking the final matching score, i.e.,

$$\begin{cases} j^*(l) = \underset{j}{\operatorname{argmax}}\{w_{lj}\} \\ j^*(i,l) = \underset{j}{\operatorname{argmax}}\{w_{j(i,l)}\} \end{cases} \qquad (12)$$

where the semantic meaning of each predicate embedding has been pre-labeled.

6. Experiment

6.1. Synthetic Data Experiments

6.1.1. EXPERIMENT SETUP

This study utilizes a meticulously structured experimental framework that includes 30 body predicates (X_1 to X_{30}) to quantitatively assess the effectiveness of our proposed method. The framework consists of three distinct rule groups, each encompassing 1 to 3 rules to simulate varying degrees of decision logic complexity. To maintain clarity of results, each sample adheres to no more than one rule. Rule weights range from 0.40 to 1.20, indicating the differing significance of each rule within the model. The "Ratio" metric conveys the proportion of samples in the dataset that conform to a specific rule, offering an intuitive understanding of the rule's coverage.

Notably, samples not conforming to any rule are influenced solely by a baseline impact, "base", uniformly set to 0.02 across all rule groups, allowing us to control for baseline effects when assessing the model's ability to learn the importance of each rule.

The experimental datasets vary in size with 5,000, 10,000, and 20,000 instances respectively, ensuring a comprehensive evaluation of the model's performance across data scales. Results for all data sizes are presented to guarantee the integrity of the analysis and the transparency of findings. Configuration details can be found in Table 1.

Table 1. Ground Truth Rules and Ratios of Synthetic Dataset

Group	Rule	Weight	Ratio
1	$Y \leftarrow X_1 \wedge X_2 \wedge X_3$ $\wedge(X_1 \text{ before } X_2)$	0.40	0.20
2	$Y \leftarrow X_1 \wedge X_2 \wedge X_3$ $\wedge(X_1 \text{ before } X_2)$	0.40	0.10
	$Y \leftarrow X_4 \wedge X_5$ $\wedge(X_4 \text{ after } X_5)$	0.80	0.15
3	$Y \leftarrow X_1 \wedge X_2 \wedge X_3$	0.40	0.10
	$Y \leftarrow X_4 \wedge X_5$ $\wedge(X_4 \text{ after } X_5)$	0.80	0.15
	$Y \leftarrow X_6 \wedge X_7$ $\wedge(X_6 \text{ before } X_7)$	1.20	0.15

6.1.2. ACCURACY AND EFFICIENCY

We conducted experiments on nine datasets with sample sizes of 5000, 10000, and 20000, corresponding to Groups 1, 2, and 3, respectively. The aim was to evaluate the accuracy and efficiency of our model.

Given the inherent randomness in rule searching, we executed multiple runs for each rule search on all datasets, varying the number of runs from 1 to 4. The rule with the minimum loss was selected as the optimal rule, ensuring consideration of different rule search iterations and identifying the top-performing logical rule. To ensure result stability and credibility, we repeated each experimental configuration ten times, reporting the average accuracy and time results in Figure 2.
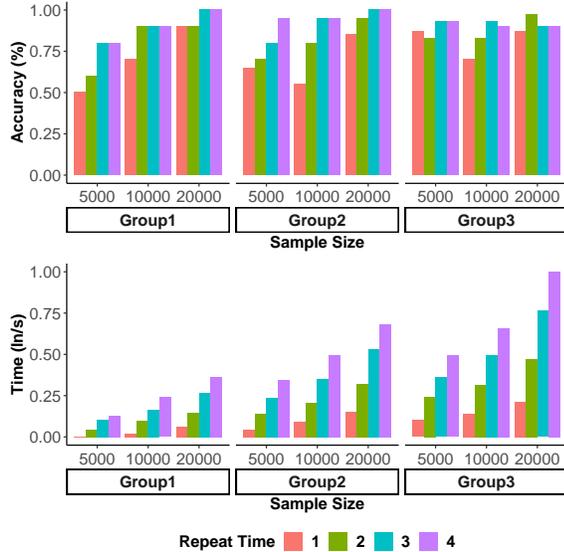

Figure 2. Performance on Different Datasets at Various Repetition Times. The upper panel showcases the accuracy achieved by different groups within varying sample sizes for each repetition. The lower panel details the time efficiency across datasets, noting that time measurements, presented in seconds, have been log-normalized for clarity.

We must emphasize that our standards for calculating accuracy are extremely stringent; a learned rule is considered correct only if it is completely learned and aligns exactly with the Ground Truth. For instance, While correct rule is $Y \leftarrow X_1 \wedge X_2 \wedge X_3 \wedge (X_1 \text{before} X_2)$, in cases of incorrect learning, we may often derive rule like $Y \leftarrow X_1 \wedge X_2 \wedge X_3 \wedge (X_1 \text{before} X_2) \wedge (X_1 \text{before} X_3)$, which is not going too far from the ground truth.

Even under strict evaluation standards, our model demonstrates promising accuracy even with small sample sizes and a single run, with further significant improvements observed as the number of repetitions or sample size increases. Also, with increasing sample sizes or repetition counts, we observe a linear increase in time, which remains well within acceptable limits, showcasing our model's excellent scalability and practicality. It is notable that in practice, our method can perform parallel rule searches (every time we search for rules, instead of searching for a single rule, we can search for multiple rules), providing a substantial speed advantage that sets it apart from other algorithms.

TELLER (Li et al., 2021) and CLUSTER (Kuang et al., 2024) are two other algorithms capable of learning first-order temporal logic rules to explain the mechanisms behind event occurrences. We compared our method to them under the condition of searching each rule four times, and the results in terms of accuracy and time are shown in Figure 3. It is evident that our algorithm significantly enhances accuracy while reducing training time when compared to the previous SOTA algorithm. On average, NS-TPP achieves a 112-fold speedup, with its accuracy significantly increasing from 49% to 93%.
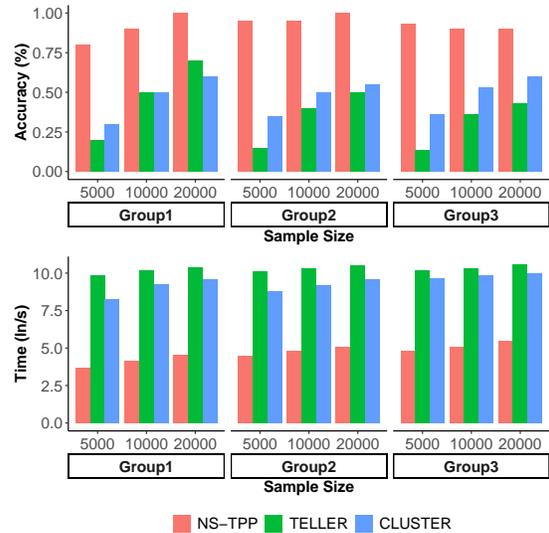

Figure 3. Comparison Results of Running Time and Accuracy with TELLER and CLUSTER.

Building on the high accuracy of rule learning, we are able to easily obtain more precise rule weights. The Mean Absolute Error (MAE) between the rule weights calculated by our algorithm and the true values across various datasets is illustrated in Figure 4.
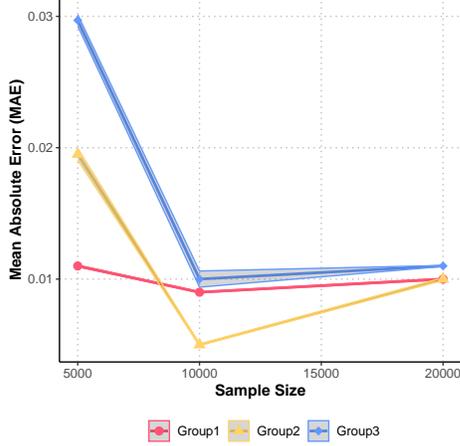

Figure 4. The MAE and Variance of learned rules' weight

In order to further demonstrate the superiority of our approach, we showcased the specific temporal logic rules learned by different methods on the Group-2 dataset with 10,000 samples. More result can be seen in Appendix A. CLNN (Yan et al., 2023), a method that is capable of learning fuzzy temporal logic rules, is also included in the comparison. The results of the rule learning are shown in Table 2. It is evident that NS-TPP can accurately learn the rules along with their corresponding weights, whereas other baseline methods encounter difficulties in the rule-learning phase.

Table 2. Learned rules and corresponding weights on the Group-2 dataset under different models.For each model, we report the best result of four runs as evaluated by log-likelihood.

Model	Learned Rules (Group2)	Weight
Ground Truth	$Y \leftarrow X_1 \wedge X_2 \wedge X_3 \wedge (X_1 \text{ before } X_2)$	0.40
	$Y \leftarrow X_4 \wedge X_5 \wedge (X_4 \text{ after } X_5)$	0.80
NS-TPP	$Y \leftarrow X_1 \wedge X_2 \wedge X_3 \wedge (X_1 \text{ before } X_2)$	0.40
	$Y \leftarrow X_4 \wedge X_5 \wedge (X_4 \text{ after } X_5)$	0.79
CLUSTER	$Y \leftarrow X_2 \wedge X_4 \wedge X_5 \wedge (X_5 \text{ before } X_4)$	0.58
	$Y \leftarrow X_1$	0.40
TELLER	$Y \leftarrow X_4 \wedge X_5 \wedge (X_4 \text{ after } X_5)$	0.76
	$Y \leftarrow X_5$	0.73
CLNN	$Y \leftarrow (X_3 \text{after} X_{13})^{2.79} \wedge (X_{15} \text{after} X_{21})^{2.71} \wedge (X_{24})^{2.46}$	0.87
	$Y \leftarrow (X_1 \text{before} X_{18})^{4.17} \wedge (X_1 \text{after} X_9)^{2.21} \wedge (X_9 \text{before} X_{17})^{2.17}$	0.01

6.1.3. EVENT PREDICTION

In addition to the aforementioned baseline methods capable of learning temporal logic rules, we also compared our model with some neural network-based methods specialized in event prediction, forecasting the occurrence of target events, and using MAE as the evaluation metric for the event prediction task. For baseline descriptions and environment configuration, refer to Appendices C and D. As shown in Table 3, our model consistently excels in all metrics, matching or surpassing the baseline performance.

Table 3. Event time prediction MAE on synthetic dataset (Group-3, 20000 samples)

METHOD	GROUP-3 (20000 SAMPLES)
THP (ZUO ET AL., 2020)	26.1545
RMTPP (DU ET AL., 2016)	31.0179
ERPP (XIAO ET AL., 2017)	28.8209
GCH (XU ET AL., 2016)	26.7682
LG-NPP (ZHANG ET AL., 2021)	32.9013
GM-NLF (EICHLER ET AL., 2017)	26.8176
TELLER (LI ET AL., 2021)	27.8301
CLNN (YAN ET AL., 2023)	29.7430
CLUSTER (KUANG ET AL., 2024)	26.0351
NS-TPP	**24.8616**

6.2. Real Data Experiments

6.2.1. EXPERIMENT SETUP

Our research involved the study of two datasets: the Car-Following dataset for assessing autonomous vehicle behavior, and the LowUrine dataset, which encompasses a wealth of medical records from ICU patients.

Within the Car-Following dataset, we gleaned five key driving behavior features from over 460 hours of driving data, leading to the documentation of 10,042 sequences. Our endeavor in this dataset is to analyze these sequences to mine for vehicle-following patterns and to deduce the underlying temporal logic rules that govern such dynamics.

The LowUrine dataset, derived from the MIMIC-IV[1], focuses on the electronic health records of 4074 ICU patients diagnosed with sepsis, capturing the physiological changes that occur leading up to the critical juncture of septic shock. A thorough analysis was conducted on 29 vital signs and laboratory tests, selected based on recommendations from previous validated studies (Komorowski et al., 2018). Special attention was given to recording the first abnormal values within the 48 hours prior to an abnormal urine output event. The analysis of this dataset aims to identify early warning signals and reveal logical patterns that may indicate the onset of septic shock, offering practical significance for clinical intervention. Details on data processing can be found in Appendix B.

---

[1] https://mimic.mit.edu/

6.2.2. DISCOVERED LOGIC RULES

In the Car-Following dataset, we explored temporal logic rules influencing vehicle dynamics by sequentially treating different events as the target event. While this dataset is relatively simpler, it is still crucial for understanding vehicle behavior patterns. In contrast, the LowUrine dataset is more complex and of greater importance, where our focus was on mining rules leading to sudden abnormal decreases in urine output. Urine output, being a significant health status indicator, especially when low urine output may signal impending septic shock, is critical for monitoring in ICU settings. Therefore, in this dataset, particular attention was paid to instances where urine output becomes abnormal after maintaining normal levels for at least 48 hours, as these events are more meaningful for prediction and explanation.

In Table 4, we showcase a selection of key logic rules discovered using our methodology, along with their corresponding weights. Notably, for the medical logic rules identified within the LowUrine dataset, our findings align with conclusions from various existing studies and are substantiated by a wealth of medical literature. For a detailed discussion of how these literatures corroborate our findings, refer to Appendix F.

Table 4. Learned Rule[3] for different Dataset.

Dataset	Rule	Weight
Car-Following	$A \leftarrow C$	0.58
	$C \leftarrow Fa$	0.66
	$F \leftarrow A \wedge D \wedge (A \text{ before } D)$	0.62
LowUrine	$LowUrine \leftarrow VO2P$	0.37
	$LowUrine \leftarrow RRate \wedge He$	0.49
	$LowUrine \leftarrow BUN \wedge LA$	0.41
	$LowUrine \leftarrow RRate \wedge LA$ $\wedge(RRate \text{ after } LA)$	0.32
	$LowUrine \leftarrow Ma \wedge VO2P \wedge LA$ $\wedge(Ma \text{ before } VO2P)$ $\wedge(Ma \text{ before } LA)$ $\wedge(VO2P \text{ before } LA)$	0.38

6.2.3. EVENT PREDICTION

In our experimental section, we employed the same baseline models as in the synthetic data experiments, using Mean Absolute Error (MAE) as the evaluation metric to predict "Low Urine" events in the LowUrine dataset and "Constant Speed Following" events in the Car-Following dataset. The performance of our model across these two datasets is presented in Table 5. The results indicate that our model outperforms all the baseline models in predicting both types of events.

---

[3]For detailed explanations of the predicates across all rules, refer to Appendix E.

Table 5. Event time prediction MAE on real datasets

METHOD	CAR-FOLLOWING	LOWURINE
THP	3.8920	2.4234
RMTPPP	4.5575	2.4643
ERPP	4.0947	2.6122
GCH	3.9819	2.5367
LG-NPP	4.2787	2.5672
GM-NLFF	4.7195	2.6925
TELLER	4.6012	2.4401
CLNN	4.3842	2.4371
CLUSTER	3.7255	2.3675
NS-TPP	**3.1614**	**2.3262**

7. Conclusion

In this paper, we introduce a new approach that integrates neural-symbolic rule induction with temporal point process models, focused on efficiently mining temporal logic rules to better understand anomalies in complex event sequences. This method not only enhances the efficiency of the rule learning process but also ensures the interpretability of the results. Extensive testing on both synthetic and real datasets has revealed significant advantages of this approach in terms of efficiency and accuracy in rule mining, demonstrating its practicality and effectiveness in complex data analysis.

Acknowledgements

Impact Statement

Our research introduces a novel neuro-symbolic framework for temporal logic induction, marking a significant advancement in machine learning's capability to process and interpret complex temporal data. By seamlessly integrating neural networks with symbolic reasoning, our approach not only enhances model interpretability but also improves predictive accuracy across diverse datasets. This work opens new avenues for developing AI systems that can better understand and predict temporal sequences, with broad implications for fields such as autonomous systems, healthcare monitoring, and financial forecasting. Our framework's flexibility and efficiency showcase its potential to foster AI solutions that not only mimic human behavior and cognition but also enhance decision-making with ethical and transparent attributes.

References

Agrawal, R., Srikant, R., et al. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pp. 487–499. Santiago, Chile, 1994.

Aprilia, N. and Januarto, O. Hubungan kebugaran jasmani dengan prestasi belajar siswa smp: Literature review. *Sport Science and Health*, 2022. doi: 10.17977/um062v4i62022p495-507.

Aryabarzan, N. and Minaei-Bidgoli, B. Neclatclosed: A vertical algorithm for mining frequent closed itemsets. *Expert Systems with Applications*, 174:114738, 2021.

Campero, A., Pareja, A., Klinger, T., Tenenbaum, J. B., and Riedel, S. Logical rule induction and theory learning using neural theorem proving. *ArXiv*, abs/1809.02193, 2018. URL https://api.semanticscholar.org/CorpusID:52176194.

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, 2016.

Eichler, M., Dahlhaus, R., and Dueck, J. Graphical modeling for multivariate hawkes processes with nonparametric link functions. *Journal of Time Series Analysis*, 38(2):225–242, 2017.

Fournier-Viger, P., Gomariz, A., Campos, M., and Thomas, R. Fast vertical mining of sequential patterns using co-occurrence information. In *Advances in Knowledge Discovery and Data Mining: 18th Pacific-Asia Conference, PAKDD 2014, Tainan, Taiwan, May 13-16, 2014. Proceedings, Part I 18*, pp. 40–52. Springer, 2014a.

Fournier-Viger, P., Gomariz, A., Šebek, M., and Hlosta, M. Vgen: fast vertical mining of sequential generator patterns. In *Data Warehousing and Knowledge Discovery: 16th International Conference, DaWaK 2014, Munich, Germany, September 2-4, 2014. Proceedings 16*, pp. 476–488. Springer, 2014b.

Inzucchi, S., Lipska, K., Mayo, H., Bailey, C., and Mcguire, D. K. Metformin in patients with type 2 diabetes and kidney disease: a systematic review. *JAMA*, 312 24:2668–75, 2014. doi: 10.1001/jama.2014.15298.

Johnson, A. E., Bulgarelli, L., Shen, L., Gayles, A., Shammout, A., Horng, S., Pollard, T. J., Moody, B., Gow, B., Lehman, L.-w. H., et al. Mimic-iv, a freely accessible electronic health record dataset. *Scientific Data*, 10(1):1–9, 2023.

Kallet, R. and Diaz, J. The physiologic effects of noninvasive ventilation. *Respiratory care*, 54 1:102–15, 2009.

Komorowski, M., Celi, L. A., Badawi, O., Gordon, A. C., and Faisal, A. A. The artificial intelligence clinician learns optimal treatment strategies for sepsis in intensive care. *Nature medicine*, 24(11):1716–1720, 2018.

Kuang, Y., Yang, C., Yang, Y., and Li, S. Unveiling latent causal rules: A temporal point process approach for abnormal event explanation. In *AISTATS*, 2024.

Landow, L. Splanchnic lactate production in cardiac surgery patients. *Critical Care Medicine*, 21:S84–91, 1993. doi: 10.1097/00003246-199302001-00015.

Li, S., Wang, L., Zhang, R., Chang, X., Liu, X., Xie, Y., Qi, Y., and Song, L. Temporal logic point processes. In *International Conference on Machine Learning*, pp. 5990–6000. PMLR, 2020.

Li, S., Feng, M., Wang, L., Essofi, A., Cao, Y., Yan, J., and Song, L. Explaining point processes by learning interpretable temporal logic rules. In *International Conference on Learning Representations*, 2021.

Mei, H. and Eisner, J. M. The neural hawkes process: A neurally self-modulating multivariate point process. *Advances in neural information processing systems*, 30, 2017.

Minervini, P., Bosnjak, M., Rocktäschel, T., and Riedel, S. Towards neural theorem proving at scale. *arXiv preprint arXiv:1807.08204*, 2018.

Mohsenin, V. Practical approach to detection and management of acute kidney injury in critically ill patient. *Journal of Intensive Care*, 5, 2017. doi: 10.1186/s40560-017-0251-y.

Payani, A. and Fekri, F. Inductive logic programming via differentiable deep neural logic networks. *arXiv preprint arXiv:1906.03523*, 2019.

Rhodes, A. and Bennett, E. Early goal-directed therapy: An evidence-based review. *Critical Care Medicine*, 32:S448–S450, 2004. doi: 10.1097/01.CCM.0000145945.39002.8D.

Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019.

Saria, S. Individualized sepsis treatment using reinforcement learning. *Nature medicine*, 24(11):1641–1642, 2018.

Shekarriz, B. and Stoller, M. Uric acid nephrolithiasis: current concepts and controversies. *The Journal of urology*, 168 4 Pt 1:1307–14, 2002. doi: 10.1016/S0022-5347(05) 64439-4.

Srinivasan, A. The aleph manual, 2001.

Suetrong, B. and Walley, K. R. Lactic acidosis in sepsis: it's not all anaerobic: implications for diagnosis and management. *Chest*, 149(1):252–261, 2016.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wei, M., Esbaei, K., Bargman, J., and Oreopoulos, D. Relationship between serum magnesium, parathyroid hormone, and vascular calcification in patients on dialysis: A literature review. *Peritoneal Dialysis International*, 26: 366 – 373, 2006. doi: 10.1177/089686080602600315.

Xiao, S., Yan, J., Yang, X., Zha, H., and Chu, S. M. Modeling the intensity function of point process via recurrent neural networks. *ArXiv*, abs/1705.08982, 2017. URL https://api.semanticscholar.org/CorpusID:7487003.

Xu, H., Farajtabar, M., and Zha, H. Learning granger causality for hawkes processes. In *International conference on machine learning*, pp. 1717–1726. PMLR, 2016.

Yan, R., Wen, Y., Bhattacharjya, D., Luss, R., Ma, T., Fokoue, A., and Julius, A. A. Weighted clock logic point process. In *The Eleventh International Conference on Learning Representations*, 2023.

Yang, Y., Xiong, S., Kerce, J. C., and Fekri, F. Temporal inductive logic reasoning. *arXiv preprint arXiv:2206.05051*, 2022.

Zhang, Q., Lipani, A., Kirnap, O., and Yilmaz, E. Self-attentive hawkes process. In *International conference on machine learning*, pp. 11183–11193. PMLR, 2020.

Zhang, Q., Lipani, A., and Yilmaz, E. Learning neural point processes with latent graphs. In *Proceedings of the Web Conference 2021*, pp. 1495–1505, 2021.

Zuo, S., Jiang, H., Li, Z., Zhao, T., and Zha, H. Transformer hawkes process. In *International conference on machine learning*, pp. 11692–11702. PMLR, 2020.

A. Result On Other Datasets

Table A.1. Learned rules and corresponding weights on the Group-1 dataset under different models. For each model, we report the best result of four runs as evaluated by log-likelihood.

Model	Learned Rules (Group1)	Weight
Ground Truth	$Y \leftarrow X_1 \wedge X_2 \wedge X_3 \wedge (X_1 \text{ before } X_2)$	0.40
NS-TPP	$Y \leftarrow X_1 \wedge X_2 \wedge X_3 \wedge (X_1 \text{ before } X_2)$	0.41
CLUSTER	$Y \leftarrow X_1 \wedge X_2 \wedge X_3 \wedge (X_1 \text{ before } X_2)$	0.37
TELLER	$Y \leftarrow X_1$	0.68
CLNN	$Y \leftarrow X_{27}^{0.981} \wedge X_3^{0.484} \wedge X_9^{0.215}$	0.29

Table A.2. Learned rules and corresponding weights on the Group-3 dataset under different models. For each model, we report the best result of four runs as evaluated by log-likelihood.

Model	Learned Rules (Group3)	Weight
Ground Truth	$Y \leftarrow X_1 \wedge X_2 \wedge X_3$	0.40
	$Y \leftarrow X_4 \wedge X_5 \wedge (X_4 \text{ after } X_5)$	0.80
	$Y \leftarrow X_6 \wedge X_7 \wedge (X_6 \text{ before } X_7)$	1.20
NS-TPP	$Y \leftarrow X_1 \wedge X_2 \wedge X_3$	0.39
	$Y \leftarrow X_4 \wedge X_5 \wedge (X_4 \text{ after } X_5)$	0.79
	$Y \leftarrow X_6 \wedge X_7 \wedge (X_6 \text{ before } X_7)$	1.20
CLUSTER	$Y \leftarrow X_{16} \wedge X_{23} \wedge X_{24}$	0.46
	$Y \leftarrow X_1 \wedge X_2 \wedge X_3 \wedge (X_1 \text{ before } X_2)$	0.47
	$Y \leftarrow X_{12}$	0.55
TELLER	$Y \leftarrow X_7$	1.28
	$Y \leftarrow X_6 \wedge X_7$	1.24
	$Y \leftarrow X_6$	1.11
CLNN	$Y \leftarrow (X_1 \text{before} X_{25})^{1.02} \wedge X_3^{0.740} \wedge X_2^{0.571}$	0.58
	$Y \leftarrow X_2^{0.791} \wedge X_7^{0.491} \wedge X_{18}^{0.347}$	0.13
	$Y \leftarrow (X_{20} \text{before} X_8)^{0.631} \wedge X_4^{0.61} \wedge (X_{13} \text{before} X_{10})^{0.417}$	0.62

B. MIMIC-IV Dataset Preprocessing Details

MIMIC-IV[4] is a publicly available database sourced from the electronic health record of the Beth Israel Deaconess Medical Center (Johnson et al., 2023). The information available includes patient measurements, orders, diagnoses, procedures, treatments, and deidentified free-text clinical notes. Sepsis is a leading cause of mortality in the ICU, particularly when it progresses to septic shock. Septic shocks are critical medical emergencies, and timely recognition and treatment are crucial for improving survival rates. In the real-world experiments on the MIMIC-IV dataset, we aim to find logic rules related to septic shocks for the whole patient samples and infer the most likely rule reasons for specific patients, which would be potential early alarms when some abnormal indicators occur.

Patients We select 4074 patients that satisfied the following criteria from the dataset: (1) The patients are diagnosed with sepsis (Saria, 2018). (2) Patients, if diagnosed with sepsis, the timestamps of any clinical testing, specific lab values, timestamps of medication administration, and corresponding dosage were not missing.

---

[4]https://mimic.mit.edu/

Outcome   Real-time urine output was treated as the outcome indicator since low urine output signals directly indicate a poor circulatory system and is a warning sign of septic shock.

Data Preprocessing   In our experiment, we focus on the electronic health records of 4,074 ICU patients diagnosed with sepsis, capturing the physiological changes that occur leading up to the critical juncture of septic shock. A thorough analysis was conducted on 29 vital signs and laboratory tests, selected based on recommendations from previous validated studies (Komorowski et al., 2018). Special attention was given to recording the first abnormal values within the 48 hours prior to an abnormal urine output event. The analysis of this dataset aims to identify early warning signals and reveal logical patterns that may indicate the onset of septic shock, offering practical significance for clinical intervention.

These risk factors are commonly assessed in sepsis patients to monitor their clinical status and guide appropriate interventions. The interpretation of these factors requires clinical judgment and consideration of the patient's overall condition. Appendix C shows the categories of some variables extracted from the MIMIC-IV dataset and their reference range.

C. About Baselines

We consider the following baselines through synthetic data experiments and healthcare data experiments to compare the rule learning ability and event prediction with our proposed model:

Neural-based (black-box) models for irregular event data

- Transformer Hawkes Process (THP) (Zuo et al., 2020): It is a sophisticated model that combines the Transformer's sequence modeling capabilities with the Hawkes process for handling irregularly timed events. This innovative approach allows for effective forecasting and understanding of complex temporal event dependencies.

- Recurrent Marked Temporal Point Processes (RMTPP) (Du et al., 2016): It is a model that utilizes recurrent neural networks to analyze and predict the timing and types of events in sequences. It excels in handling complex temporal relationships in data, making it valuable for applications requiring a detailed understanding of event sequences and their dynamics.

- ERPP (Xiao et al., 2017): It is a neural network approach for modeling event sequences, focusing on capturing the complex temporal patterns and dependencies between events. This model is notable for its ability to effectively handle a wide range of event-based datasets, providing insights into the underlying structure and dynamics of temporal data.

- LG-NPP algorithm (Zhang et al., 2021): It is an innovative neural process model designed for learning and predicting the intricate patterns in event sequences. This algorithm stands out for its effectiveness in capturing the long-term dependencies and subtle nuances within sequential data, making it highly applicable in complex temporal analysis tasks.

Simple parametric/nonparametric models for irregular event data

- Granger Causal Hawkes (GCH) (Xu et al., 2016): It is a statistical approach that combines Granger causality analysis with the Hawkes process to understand the influence of past events on future occurrences. It excels in identifying causal relationships in temporal data, making it particularly useful in fields where understanding the impact of past events on future dynamics is crucial.

- GM-NLF algorithm (Eichler et al., 2017): This is a sophisticated algorithm designed for analyzing complex nonlinear relationships in time series data. It is particularly notable for its ability to model and predict intricate patterns and dependencies, enhancing the understanding of dynamic systems in various domains.

Logical models for irregular event data

- Clock Logic Neural Networks (CLNN) (Yan et al., 2023): It represents a novel approach in neural network design, integrating time-aware mechanisms to better handle temporal data. This model is particularly effective in capturing both the sequential and timing aspects of events, offering enhanced performance in tasks requiring precise temporal understanding and prediction.

- TELLER (Li et al., 2021): This is a cutting-edge neural network model designed for temporal and event-based data analysis. It stands out for its ability to intricately model and predict complex patterns in sequential data, making it highly effective in applications requiring deep temporal understanding and forecasting.

- CLUSTER (Kuang et al., 2024): This is an automated method for uncovering "if-then" logic rules to explain observational events. This approach demonstrates accurate performance in both discovering rules and identifying root causes.

We compared our model with some models from previous studies on the same dataset, finding that not only does it run in a shorter time, but it also achieves higher accuracy.

D. Experimental Environment Configuration

For our proposed method, all experiments were conducted on a Linux server with an Intel(R) Xeon(R) Gold 6248R CPU @ 3.00GHz and 30Gi of memory, running Ubuntu 20.04.5 LTS. Due to the modest size of our model parameters, CPU execution was found to be more efficient. Hence, while all baseline methods except TELLER were run on GPU, we opted to perform our experiments on the CPU. The coding environment utilized was Python 3.9.12, with PyTorch 2.0.1 serving as the primary machine-learning framework.

E. Glossary

E.1. Car-Following Dataset

Predicates	Explanation
Fa	Free acceleration
C	Cruising at a desired speed
A	Acceleration following a leading vehicle
D	Deceleration following a leading vehicle
F	Constant speed following

E.2. LowUrine Dataset

The 29 extracted predicates can be categorized into the following five groups:

- Vital Signs:

  - Heart Rate: The number of times the heart beats per minute. An elevated or abnormal heart rate may indicate physiological stress or an underlying condition.
  - Arterial Blood Pressure (systolic, mean, diastolic): Measures the force exerted by the blood against the arterial walls during different phases of the cardiac cycle. Abnormal blood pressure values may indicate cardiovascular dysfunction or organ perfusion issues.
  - Temperature (Celsius): Body temperature is a measure of the body's internal heat. Abnormal temperatures may indicate infection, inflammation, or other systemic disorders.
  - Respiratory Rate(RRate): The number of breaths taken per minute. Abnormal respiratory rates may suggest respiratory distress or dysfunction.
  - SpO_2: Oxygen saturation level in the blood. Decreased SpO_2 levels may indicate inadequate oxygenation.

- Biochemical Parameters:

  - Potassium, Sodium, Chloride, Glucose: Electrolytes and blood sugar levels that help maintain essential bodily functions. Abnormal levels may indicate electrolyte imbalances, metabolic disorders, or organ dysfunction.
  - Blood Urea Nitrogen (BUN), Creatinine: Indicators of renal function. Elevated levels may suggest impaired kidney function.
  - Magnesium(Ma), Ionized Calcium: Important minerals involved in various physiological processes. Abnormal levels may indicate electrolyte imbalances or organ dysfunction.

- Total Bilirubin: A byproduct of red blood cell breakdown. Elevated levels may indicate liver dysfunction.
- Albumin: A protein produced by the liver. Abnormal levels may indicate malnutrition, liver disease, or kidney dysfunction.

- Hematological Parameters

  - Hemoglobin(He): A protein in red blood cells that carries oxygen. Abnormal levels may indicate anemia or oxygen-carrying capacity issues.
  - White Blood Cell (WBC): Cells of the immune system involved in fighting infections. Abnormal levels may indicate infection or inflammation.
  - Platelet Count: Blood cells responsible for clotting. Abnormal levels may suggest bleeding disorders or impaired clotting ability.
  - Partial Thromboplastin Time (PTT), Prothrombin Time (PT), INR: Tests that assess blood clotting function. Abnormal results may indicate bleeding disorders or coagulation abnormalities.

- Blood Gas Analysis

  - pH (Arterial): A measure of blood acidity or alkalinity. Abnormal pH values may indicate acid-base imbalances or respiratory/metabolic disorders.
  - Arterial Base Excess: Measures the amount of excess or deficit of base in arterial blood. Abnormal levels may indicate acid-base imbalances or metabolic disturbances.
  - Arterial CO2 Pressure(AO2P), Venous O2 Pressure(VO2P): Parameters that assess respiratory and metabolic function. Abnormal values may indicate respiratory failure or metabolic disturbances.

- Metabolic Parameter

  - Lactic Acid(LA): An indicator of tissue perfusion and oxygenation. Elevated levels may suggest tissue hypoxia or impaired cellular metabolism.

F. Medical References

In our clinical research employing the MIMIC-IV dataset, we strengthened our findings with corroborative evidence from medical literature, demonstrating the robustness and clinical applicability of our methodology. This integrative process ensures our discovered rules not only align with expert insights but are also grounded in established medical knowledge, enhancing the interpretability and real-world applicability of temporal logic rules in healthcare analytics.

- **Rule 1: LowUrine ← VO2P**: These rules involve venous O2 pressure, it linked to cardiac output and tissue hypoxia in septic shock (Mohsenin, 2017; Rhodes & Bennett, 2004).

- **Rule 2: LowUrine ← RRate ∧ He**: The studies indicate that effective management of respiratory function and maintaining adequate hemoglobin levels are crucial for ensuring efficient oxygen delivery and preventing complications like low urine output. This highlights the interconnectedness of respiratory health, oxygen transport capacity, and kidney function in maintaining overall systemic health (Kallet & Diaz, 2009; Aprilia & Januarto, 2022).

- **Rule 3: LowUrine ← BUN ∧ LA**:Research highlights the significant impact of metabolic disturbances, such as hyperuricemia and the risk of lactic acidosis from medications like metformin, on renal function and urine output. Managing these conditions through urinary alkalization and careful medication management is crucial for preventing renal complications and maintaining adequate urine output (Shekarriz & Stoller, 2002; Inzucchi et al., 2014).

- **Rule 4: LowUrine ← RRate ∧ LA ∧ (RRate after LA)**: Abnormal levels of lactate are typically induced by tissue hypoxia or metabolic disturbances, while subsequent abnormalities in respiratory rate may represent the body's compensatory effort to eliminate excess acid metabolites through respiration. Together, these symptoms may indicate a deteriorating clinical condition, progressing towards sepsis (Suetrong & Walley, 2016).

- **Rule 5: LowUrine ← Ma ∧ VO2P ∧ LA ∧ (Ma before VO2P) ∧ (Ma before LA) ∧ (VO2P before LA)**: Research indicates that hypomagnesemia is associated with increased cardiovascular risk, which may indirectly impact kidney function and urine output (Wei et al., 2006). Additionally, inadequate oxygen delivery and elevated lactate levels signal

systemic hypoperfusion, including renal hypoperfusion, potentially leading to reduced urine output (Landow, 1993). These findings underscore the importance of magnesium levels, venous O2 pressure, and lactate in maintaining kidney health and appropriate urine output.