Inference and Learning for Latent Temporal Logic Point Processes

Mou Minghao*
Supervisor: Prof. Li Shuang†

Abstract

Temporal Logic Point Processes is a modeling framework which incorporate *domain knowledge* by introducing prespecified *first-order temporal logic rules*. The model achieves the SOTA in *small data* regime and is general enough to recover well-known point processes such as Hawkes process, self-correcting process. We base this model specifically on the context of *autonomous driving systems* and divide the processes into two different categories, i.e., mental processes and action processes. Each process can be viewed as a 0-1 two state stochastic process. This paper addresses a natural question: given the autonomous driving system is only able to observe the actions of other human drivers, how to recover the their mental states to enable the autonomous system to make further predictions? These latent Temporal Logic Point Processes pose a serious inferential challenge: we must perform inference in a model whose latent variables are temporal logic point processes. We derive two algorithms that allow joint inference for the latent processes and learning for the model. The first one is based on *Variational Inference* and the second one depends on *Monte-Carlo Expectation Maximization*, which leverages the autoregressive nature of the model. The code can be found at github.

Acknowledgement

This project is funded by the Undergraduate Research Award provided by The Chinese University of Hong Kong, Shenzhen.

1 Introduction

Most real application domains, such as finance [BMM15], natural disaster prediction [Ros21], social media [AS19], and health-care [RBS10] generate discrete events in the continuous-time stream. *Temporal Logic Point Processes* [LWZ+20] provides a modeling framework which allows to utilize the known domain knowledge and thus significantly reduces the amount of data needed to fit the model. Moreover, incorporating prior knowledge provides more explanability to the model, which is often much more important than the accuracy of prediction.

In this work, we consider the a *latent variable problem*. However, different from most common latent variable problems, the latent variables now are *temporal logic point processes*. The inferential problem is challenging since the number of events in the latent processes are undetermined. In the context, only incomplete data is provided, to learn the model parameter, one need to first impute the missing data (inference) and then use the imputed complete dataset to perform model learning. We will base our presentation in a specific context: *autonomous driving system*. The driving system can observe the actions made by human drivers, but it does not know the mental states that triggers the observed action according to some specified logic rules. The processes are divided into two categories: action and mental. All the action processes are fully observable while all the mental processes are latent. Our objective is to recover the mental processes using the observations.

*School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen
†School of Data Science, The Chinese University of Hong Kong, Shenzhen

To this end, we propose two different algorithms inspired by different works [LWB17, KW13]. The first algorithm is a direct application of *variational inference*. We try to find a variational proposal distribution which is close enough to the true posterior so that we can treat the variational proposal as the surrogate of the true posterior. Since the form of the variational proposal can be very flexible, we can easily sample from the it to obtain the events in those latent processes. After imputing the missing data, we maximize the ELBO to learn the model parameter. The second algorithm is based on *Monte-Carlo Expectation Maximization*, since the temporal logic point processes possess the autoregressive nature, i.e., the conditional intensity function at one time point only depends on the whole past history, we can use *Sequential Monte Carlo* to sequentially sample latent events while preserving the dependence between events. We adopt expecatation maximization to alternate between inference and learning. Therefore, the inference results can be viewed as a by-product of model learning.

2 Problem Statement and Formulation

2.1 Assumptions

- The actions $a_k, k \in \{1, ..., K\}$ (k stands for the type of the action) are fully observable.
- The mental states $m_i, i \in \{1, ..., I\}$ are completely hidden from observation.
- We are given a collection of prespecified logic rules $\mathcal{F}_{a_k}$ for deducing action $a_k, k \in \{1, ..., K\}$ and a collection of prespecified logic rules $\mathcal{F}_{m_i}$ for deducing mental states $m_i, i \in \{1, 2, ..., I\}$.
- At one particular time point, the action should be a one-hot vector in $\mathbb{R}^K$, i.e., during an infinitesimal time interval, we only allow one action.

2.2 Model Construction

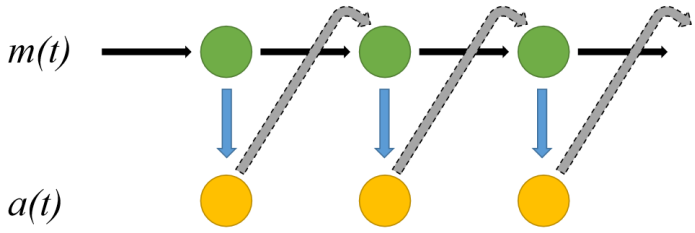

Figure 1: Discrete-time analogy of the model. $a(t)$ stands for the action process and $m(t)$ stands for the mental process.

We adopt the temporal logic point processes[LWZ+20] as the framework for our model. We need to model the dynamics of both the transition of mental states and the triggering of actions by mental processes, which are analogous to the counterparts in Hidden Markov Model. The transition of mental states corresponds to the transition of hidden states and the action triggering mechanism is like the emission in HMM. However, different from HMM, the past actions can also trigger mental states (as shown in Fig 1). We can capture both dynamics by specifying $\mathcal{F}_{a_k}$ and $\mathcal{F}_{m_i}$ for all k and i.

A Running Example. Consider a toy example where there are only one mental process $m(t)$ and one action process $a(t)$, namely, $K = I = 1$. Thus, the complete dataset in the time interval $[0, T)$ (from the God's perspective) is of the form

$$\mathcal{H}(T) := \{m(t_1), a(t_{1,1}), ..., a(t_{J_1,1}), m(t_2), ..., m(t_n), a(t_{1,n}), ..., a(t_{J_n,n})\},$$

where $0 \leq t_1 < t_2 < \cdots < t_n \leq T$ stands for the state transition time of mental states and $t_i < t_{1,i} < t_{2,i} < \cdots < t_{J_i,i}$ stands for the state transition time of actions.

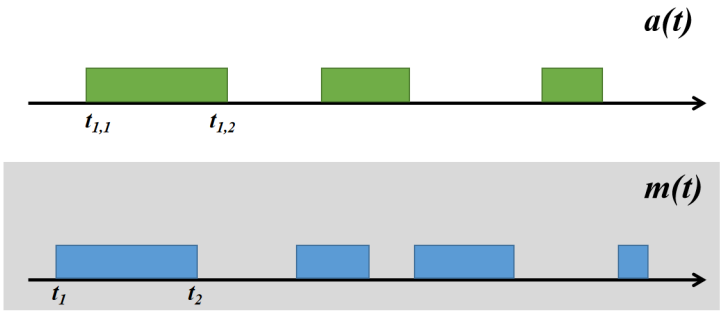

Figure 2: A running example with one mental process $m(t)$ and one action process $a(t)$.

Suppose two logic rules $f_a : (a \leftarrow m) \wedge (m \text{ before } a)$ and $f_m : (m' \leftarrow \neg a \wedge m) \wedge (m \text{ before } m')$ are given (i.e. $|\mathcal{F}_a| = 1$ and $|\mathcal{F}_m| = 1$). The equivalent *clausal forms* are written as $f_a(m(t_m), a(t_a), t_m \in \tau_m, t_a \in \tau_a) = (\neg m \vee a) \wedge (r_{be}(\tau_m, \tau_a))$ and $f_m(m(t_m), a(t_a), m(t'_m), t_m \in \tau_m, t_a \in \tau_a, t'_m \in \tau_{m'}) = (a \vee \neg m \vee m') \wedge (r_{be}(\tau_m, \tau_{m'}))$. Now, the *formula effect* corresponding to the rules f_a, f_m are

$$\delta_{f_a}(t|t_m \in \tau_m) := f(m(t_m), 1 - a(t_a), t_m \in \tau_m, t_a = t) - f(m(t_m), a(t_a), t_m \in \tau_m, t_a = t), \tag{1}$$

$$\begin{aligned}&\delta_{f_m}(t|t_m \in \tau_m, t_a \in \tau_a) := \\ &f_m(m(t_m), a(t_a), 1 - m(t'_m), t_m \in \tau_m, t_a \in \tau_a, t'_m = t) - f_m(m(t_m), a(t_a), m(t'_m), t_m \in \tau_m, t_a \in \tau_a, t'_m = t)\end{aligned} \tag{2}$$

Note that $\delta_{f_a}, \delta_{f_m} \in [-1, +1]$ since softened temporal constraints are used here.

the *conditional transition intensities* for $a(t)$ and $m(t)$ from state 0 to 1 and the intensities from state 1 to 0, contributed by the rule f (we only assume logic rules f_a, f_m) are modeled as

$$\lambda_a^*(t) := \exp\left(\theta_{f_a} \delta_{f_a}(t|t_m = t) + b_a(t)\right), \tag{3}$$

$$\mu_a^*(t) := \exp\left(\theta_{f_a} \delta_{f_a}(t|t_m = t) + b_a(t)\right), \tag{4}$$

and

$$\lambda_m^*(t) := \exp\left(\theta_{f_m} \cdot \sum_{\tau_m \in \mathcal{H}_m(t)} \sum_{\tau_a \in \mathcal{H}_a(t)} \delta_{f_m}(t|t_m \in \tau_m, t_a \in \tau_a) + b_m(t)\right), \tag{5}$$

$$\mu_m^*(t) := \exp\left(\theta_{f_m} \cdot \sum_{\tau_m \in \mathcal{H}_m(t)} \sum_{\tau_a \in \mathcal{H}_a(t)} \delta_{f_m}(t|t_m \in \tau_m, t_a \in \tau_a) + b_m(t)\right), \tag{6}$$

where $b_a(t)$ and $b_m(t)$ are base terms accounting for the spontaneous transition.

Indeed, we can write the conditional intensity compactly as

$$\lambda_a(t) = \mathbb{I}(a(t) = 0)\lambda_a^*(t) + (1 - \mathbb{I}(a(t) = 0))\mu_a^*(t).$$
$$\lambda_m(t) = \mathbb{I}(m(t) = 0)\lambda_m^*(t) + (1 - \mathbb{I}(m(t) = 0))\mu_m^*(t).$$

Then, based on the compact representations of the intensity, we can write the likelihood as

$$\mathcal{L}\{a(t); \boldsymbol{\theta}_a\} = \prod \lambda_a(t_{i,j}) \exp\left(-\int_0^T \lambda_a(s)ds\right).$$

$$\mathcal{L}\{m(t); \boldsymbol{\theta}_m\} = \prod \lambda_m(t_i) \exp\left(-\int_0^T \lambda_m(s)ds\right).$$

The complete likelihood is

$$\mathcal{L}\{\boldsymbol{\theta}\} = \left(\prod \lambda_a(t_{i,j})\right)\left(\prod \lambda_m(t_i)\right)\exp\left(-\int_0^T [\lambda_a(s)+\lambda_m(s)]ds\right),$$

where $\mathcal{H}(t) := \{\{t_{i,j}\}_{i,j}, \{t_i\}_i\}$ consists of the whole past history and $(\lambda_a(s)+\lambda_m(s))$ can be viewed as the rate of the whole process.

By definition of conditional intensity, for the predicate $a(t)$, given a realization of the process up to time T, the likelihood $\mathcal{L}\{a(t)\}_{t\geq 0}$ is

$$\mathcal{L}\{a(t);\boldsymbol{\theta}_a\} := \lambda_a^*(t_{1,1})\exp\left(-\int_0^{t_{1,1}}\lambda_a^*(s)ds\right)\cdots\exp\left(-\int_{t_{J_n,n}}^T \mu_a^*(s)ds\right). \tag{7}$$

provided $a(0)=0$ and $a(t)=1, \forall t\in[t_{J_n,n},T)$, which is a reasonable assumption since at time $t=0$, there should be no action. The likelihood for $m(t)$ is

$$\mathcal{L}\{m(t);\boldsymbol{\theta}_m\} := \lambda_m^*(t_1)\exp\left(-\int_0^{t_1}\lambda_m^*(s)ds\right)\cdots\exp\left(-\int_{t_n}^T \mu_m^*(s)ds\right), \tag{8}$$

provided that $m(0)=0$ and $m(t)=1, \forall t\in[t_n,T)$. One should also notice that the likelihoods depend on the whole past history $\mathcal{H}(t) := \mathcal{H}_m(t)\bigcup\mathcal{H}_a(t)$.

By superposition, we know that the likelihood for the complete data is

$$\mathcal{L}[\boldsymbol{\theta}] := \mathcal{L}\{a(t);\boldsymbol{\theta}_a\}\cdot\mathcal{L}\{m(t);\boldsymbol{\theta}_m\}, \tag{9}$$

where $\boldsymbol{\theta} := [\boldsymbol{\theta}_a, \boldsymbol{\theta}_m]$ is the concatenation of model parameters.

The goal is to learn the model parameter $\boldsymbol{\theta}$. If $\boldsymbol{\theta}$ is available, we can make prediction on the action by sampling from the action process. The natural idea is to directly maximize the log-likelihood, i.e. $\boldsymbol{\theta}^* := arg\max_{\boldsymbol{\theta}}\log\mathcal{L}\{\boldsymbol{\theta}\}$. However, the past history $\mathcal{H}(t)$ is incomplete since the mental process is hidden from observations. Therefore, we need to jointly perform inference on the hidden processes and learn the model parameters $\boldsymbol{\theta}$.

2.3 Generalization to K Action Processes and I Mental Processes

Now, we have K action processes $a_k(t), k\in\{1,2,...,K\}$ and I mental processes $m_i(t), i\in\{1,2,...,I\}$, as show in Fig 3. It is intuitive that the action processes are not independent since the mental process are not independent. However, given all the mental states $\boldsymbol{z}$, all the action processes should become conditionally independent. We further assume that there are, for each action process $a_k(t)$, a collection of prespecified temporal logic rules $\mathcal{F}_k := \{f_{1,k}, f_{2,k}, ..., f_{n_k,k}\}$ and for each mental process $m_i(t)$, a collection of temporal logic rules $\mathcal{F}_i := \{f_{1,i}, f_{2,i}, ..., f_{n_i,i}\}, i=1,2,...,I$.

The complete dataset in the interval $[0,T)$ is given as

$$\mathcal{H}(t) := \left(\bigcup_{i=1}^I \{m_i(t_{1,i}),...,m_i(t_{I_i,i})\}\right)\bigcup\left(\bigcup_{k=1}^K \{a_k(t_{1,k}),...,a_k(t_{J_k,k})\}\right), \tag{10}$$

where $0\leq t_{1,i}<\cdots<t_{I_i,i}<T$ stands for the mental states transition time for the i-th mental process and $t_{1,k}<\cdots<t_{J_k,k}$ stands for the action transition time for the k-th action process. ($<$ because according to our assumption, the event that two mental or two actions happen at the same time is with probability zero.) For an arbitrary logic rule f, we can rewrite it equivalently as its *clausal form*:

$$f(m_1(t_{1,m}),...,m_I(t_{I,m}),a_1(t_{1,a}),...,a_K(t_{K,a}),t_{1,m}\in\tau_{m_1},...,t_{I,m}\in\tau_{m_I},t_{1,a}\in\tau_{a_1},...,t_{K,a}\in\tau_{a_K}). \tag{11}$$

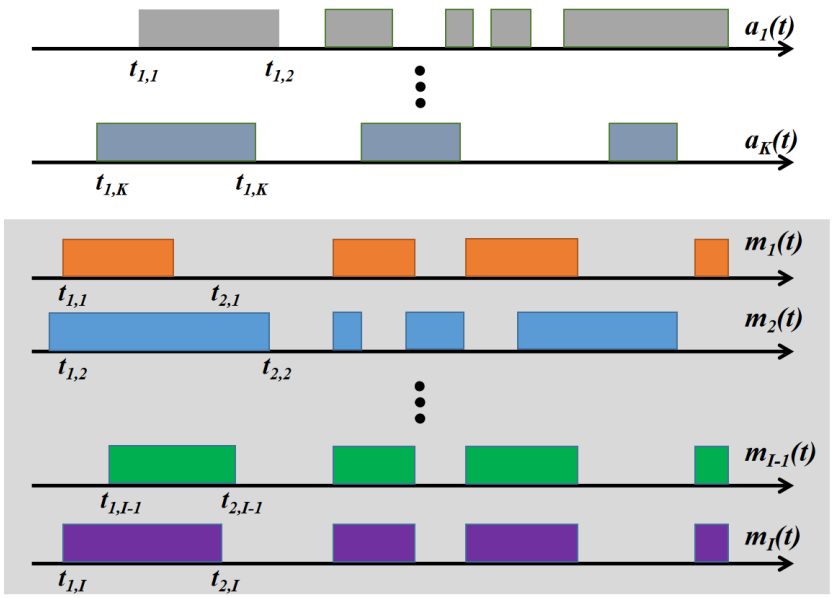

Figure 3: A general example with I mental processes and K action processes.

By taking the above clausal form, we can define the *formula effect* corresponding to the logic rule f. (WLOG, Suppose f is used to deduce action a_1. Moreover, it is assumed that the actions depend on previous actions only through the mental states, so we omit the actions in the clausal form.)

$$\begin{aligned}
&\delta_f(t|t_{1,m} \in \tau_{m_1}, ..., t_{I,m} \in \tau_{m_I}) \\
&= f(m_1(t_{1,m}), ..., m_I(t_{I,m}), 1 - a_1(t_{1,a}), t_{1,m} \in \tau_{m_1}, ..., t_{I,m} \in \tau_{m_I}, t_{1,a} = t) \\
&- f(m_1(t_{1,m}), ..., m_I(t_{I,m}), a_1(t_{1,a}), t_{1,m} \in \tau_{m_1}, ..., t_{I,m} \in \tau_{m_I}, t_{1,a} = t).
\end{aligned} \tag{12}$$

Note that $\delta_f \in [-1, +1]$ since softened temporal relations are used here.
Also, we can write the *formula effect* corresponding to a logic rule f that is responsible for deducing a mental process m_1.

$$\begin{aligned}
&\delta_f(t|t_{1,m} \in \tau_{m_1}, ..., t_{I,m} \in \tau_{m_I}, t_{1,a} \in \tau_{a_1} ..., t_{K,a} \in \tau_{a_K}) \\
&= f(1 - m_1(t_{1,m}), ..., m_I(t_{I,m}), a_1(t_{1,a}), ..., a_K(t_{K,a}), t_{1,m} = t, ..., t_{I,m} \in \tau_{m_I}, t_{1,a} \in \tau_{a_1}, ..., t_{K,a} \in \tau_{a_K}) \\
&- f(m_1(t_{1,m}), ..., m_I(t_{I,m}), a_1(t_{1,a}), ..., a_K(t_{K,a}), t_{1,m} = t, ..., t_{I,m} \in \tau_{m_I}, t_{1,a} \in \tau_{a_1}, ..., t_{K,a} \in \tau_{a_K})
\end{aligned} \tag{13}$$

The *conditional transition intensity* for $a_k(t)$ from state 0 to state 1, contributed by the rules $\mathcal{F}_k$ is modeled as

$$\lambda^*_{a_k}(t) := \exp\left(\sum_{j=1}^{n_k} \omega_{f_{j,k}} \left(\delta_{f_{j,k}}(t|t_{1,m} = t, ..., t_{I,m} = t)\right) + b_k(t)\right). \tag{14}$$

where $b_k(t)$ is the spontaneous rate for the k-th action process. The intensity from state 1 to state 0 is defined similarly, the only difference is that $a_k(t_{k,a}) = 0$.
Similarly, the *conditional transition intensity* for $m_i(t)$ from state 0 to state, contributed by the rules $\mathcal{F}_i$ is modeled as

$$\lambda^*_{m_i}(t) := \exp\left(\sum_{j=1}^{n_i} \omega_{f_{j,i}} \left(\sum_{\tau_{m_{-i}} \in \mathcal{H}_{m_{-i}}(t)} \sum_{\tau_{a_1} \in \mathcal{H}_{a_1}(t)} \cdots \sum_{\tau_{a_K} \in \mathcal{H}_{a_K}(t)} \delta_{f_{j,i}}(t|t_{1,m} \in \tau_{m_1}, ..., t_{I,m} \in \tau_{m_I}, ...)\right) + b_i(t)\right). \tag{15}$$

where $b_i(t)$ is the spontaneous rate for the i-th mental process. The intensity from state 1 to state 0 $\mu^*_{m_i}(t)$ is defined similarly, the only difference is that $m_i(t_{i,m}) = 0$.

By definition of conditional intensity, for the predicate $a_k(t)$, given a realization of the process up to time t, the *likelihood* $\mathcal{L}\{a_k(t)\}_{t\geq 0}$ is

$$\mathcal{L}\{a_k(t); \boldsymbol{\omega}\} := \lambda^*_{a_k}(t_{1,k}) \exp\left(\int_0^{t_{1,k}} \lambda^*_{a_k}(s)ds\right) \cdots \exp\left(-\int_{t_{J_k,k}}^{t} \mu^*_{a_k}(s)ds\right). \tag{16}$$

provided $a_k(0) = 0$ and $a_k(t) = 1$, which is a reasonable assumption since at time $t = 0$, there should be no action.
The *likelihood* $\mathcal{L}\{m_i(t)\}_{t\geq 0}$ is

$$\mathcal{L}\{m_i(t); \boldsymbol{\omega}\} := \lambda^*_{m_i}(t_{1,i}) \exp\left(\int_0^{t_{1,i}} \lambda^*_{m_i}(s)ds\right) \cdots \exp\left(-\int_{t_{I_i,i}}^{t} \mu^*_{m_i}(s)ds\right). \tag{17}$$

provided $m_i(0) = 0$ and $m_i(t) = 1$. One should also notice that the likelihoods depends on the whole past history $\mathcal{H}(t)$.

The *joint likelihood*, by superposition, is given as

$$\mathcal{L}\{\boldsymbol{\omega}\} := \prod_{i=1}^{I} \mathcal{L}\{m_i(t); \boldsymbol{\omega}\} \cdot \prod_{k=1}^{K} \mathcal{L}\{a_k(t); \boldsymbol{\omega}\}. \tag{18}$$

3 Intermediate Results

3.1 Auto-Encoder Variational Bayes + LSTM

The goal of inference is to compute the *posterior distribution*

$$p_{\boldsymbol{\theta}}(\mathcal{H}_m(T)|\mathcal{H}_a(T)),$$

where $\mathcal{H}_m(T)$ denotes the collection of all the mental transition times in the time interval $[0, T)$ and $\mathcal{H}_a(T)$ denotes the collection of all the action transition times in the time interval $[0, T)$. Analytically compute the true posterior is in general intractable. Therefore, we consider a proposal distribution parametrized by variational parameter ψ

$$q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)).$$

We discretize the time interval into fine scaffolds $\{[s_{i-1}, s_i)\}_{i=1}^M$ where $s_0 = 0, s_M = T$ and $s_{i-1} < s_i, \forall i = 1, 2, ..., M$. We choose $\max_{i\in\{1,2,...,M\}}(s_i - s_{i-1})$ to be small enough so that inside each interval, there is at most one mental transition. (i.e., we discretize the mental process.) According to chain rule we have

$$q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) = q_\psi(\{z_m\}_{m=1}^M|\mathcal{H}_a(T)) = q_\psi(z_1|\mathcal{H}_a(T))q_\psi(z_2|z_1, \mathcal{H}_a(T)) \cdots q_\psi(z_M|z_{1:M-1}, \mathcal{H}_a(T)),$$

where z_m stands for the mental transition (time and type; there can be no mental transition) in $[s_{m-1}, s_m)$. We can use two LSTMs [MQE19] to encode the information $\mathcal{H}_a(T)$ and $z_{1:i}$, say $f_1(\mathcal{H}_a(T)) = \boldsymbol{h}^a$ and $f_2(z_{1:i}) = \boldsymbol{h}^m_i$. Once we have a new sample z_{i+1}, we update the embedding by $\boldsymbol{h}^m_{i+1} := f_2(\boldsymbol{h}^m_i, z_{i+1})$. We summarize all the learnable parameters of f_1 and f_2 as ψ. Then,

$$\begin{aligned} q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) &= q_\psi(\{z_m\}_{m=1}^M|\mathcal{H}_a(T)) = q_\psi(z_1|\mathcal{H}_a(T))q_\psi(z_2|z_1, \mathcal{H}_a(T)) \cdots q_\psi(z_M|z_{1:M-1}, \mathcal{H}_a(T)) \\ &= q_\psi(z_1|\boldsymbol{h}^a)q_\psi(z_2|\boldsymbol{h}^a, \boldsymbol{h}^m_1) \cdots q_\psi(z_M|\boldsymbol{h}^a, \boldsymbol{h}^m_{M-1}). \end{aligned} \tag{19}$$

We can train the LSTMs by minimizing the KL-divergence between the true posterior and the proposal

$$\min_\psi \mathcal{L}(\boldsymbol{\theta}, \psi, \mathcal{H}_a(T)) := \min_\psi KL(q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T))||p_{\boldsymbol{\theta}}(\mathcal{H}_m(T)|\mathcal{H}_a(T))),$$

which is equivalent to maximize the Evidence Lower Bound (ELBO) [KW13]

$$\max_{\psi} \left(\mathbb{E}_{q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T))} \left[\log p_{\boldsymbol{\theta}}(\mathcal{H}_m(T), \mathcal{H}_a(T)) \right] + \mathbb{H} \left[q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) \right] \right).$$

Proof.

$$\begin{aligned}
&KL(q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T))||p_{\boldsymbol{\theta}}(\mathcal{H}_m(T)|\mathcal{H}_a(T))) = \int q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) \log \frac{q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) p_{\boldsymbol{\theta}}(\mathcal{H}_a(T))}{p_{\boldsymbol{\theta}}(\mathcal{H}_m(T), \mathcal{H}_a(T))} \\
&= -\left(-\int q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) \log q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) + \int q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) \log p_{\boldsymbol{\theta}}(\mathcal{H}_m(T), \mathcal{H}_a(T)) \right) \\
&+ \int q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) \log p_{\boldsymbol{\theta}}(\mathcal{H}_a(T)) \\
&= -\left(\mathbb{H}(q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T))) + \mathbb{E}_{q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T))} [\log p_{\boldsymbol{\theta}}(\mathcal{H}_m(T), \mathcal{H}_a(T))] \right) + \log p_{\boldsymbol{\theta}}(\mathcal{H}_a(T)).
\end{aligned}$$

or, assume $q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T))$ is reparameterizable, i.e., there exists a differentiable mapping $g(\epsilon; \psi) : \epsilon \mapsto \mathcal{H}_m(T)$ and $\epsilon \sim s(\epsilon)$ has a tractable distribution, then the objective can be rewritten as

$$\max_{\psi} \left(\mathbb{E}_{s(\epsilon)} \left[\log p_{\boldsymbol{\theta}}(\mathcal{H}_a(T), g(\epsilon; \psi)) \right] + \mathbb{H} \left[q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) \right] \right).$$

Since we can draw samples from the proposal $q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T))$, say, we have drawn L samples

$$\{\mathcal{H}_m^{(l)}(T)\}_{l=1}^L \sim q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)),$$

$$\mathcal{L}(\boldsymbol{\theta}, \psi, \mathcal{H}_a(T)) \approx \frac{1}{L} \sum_{l=1}^{L} \left(\log p_{\boldsymbol{\theta}}(\mathcal{H}_m^{(l)}(T), \mathcal{H}_a(T)) - \log q_\psi(\mathcal{H}_m^{(l)}(T)|\mathcal{H}_a(T)) \right).$$

or,

$$\mathcal{L}(\boldsymbol{\theta}, \psi, \mathcal{H}_a(T)) \approx \frac{1}{L} \sum_{l=1}^{L} \left(\log p_{\boldsymbol{\theta}}(\mathcal{H}_m^{(l)}(T), \mathcal{H}_a(T)) \right) + \mathbb{H} \left[q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) \right],$$

since we can explicitly compute the entropy.

Differentiate the RHS w.r.t. $\boldsymbol{\theta}$ and ψ, we can perform gradient ascent to jointly optimize the model parameter and the variational parameter. The pseudo code is as follows:

Initialize $\boldsymbol{\theta}$ and ψ, repeat

- $\{\mathcal{H}_a^{(r)}(T)\}_{r=1}^R \leftarrow$ Random minibatch of R action realizations.
- (if reparametrized) $\boldsymbol{\epsilon} \leftarrow$ Random samples from the noise distribution $p(\boldsymbol{\epsilon})$.
- $\boldsymbol{g} \leftarrow \nabla_{\boldsymbol{\theta}, \psi} \frac{N}{R} \sum_{r=1}^{R} \mathcal{L}(\boldsymbol{\theta}, \psi, \mathcal{H}_a^{(r)}(T))$. ($N$ is the total number of action realizations (i.e. training data) we have)
- $\boldsymbol{\theta}, \psi \leftarrow$ Update parameters using the gradients $\boldsymbol{g}$.

Until convergence of parameters $(\boldsymbol{\theta}, \psi)$.

Gumbel-Max Trick. Recall that

$$q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) = q_\psi(z_1|\boldsymbol{h}^a) q_\psi(z_2|\boldsymbol{h}^a, \boldsymbol{h}_1^m) \cdots q_\psi(z_M|\boldsymbol{h}^a, \boldsymbol{h}_{M-1}^m) = \prod_{j=1}^{M} q_\psi(z_j|\boldsymbol{h}^a, \boldsymbol{h}_{j-1}^m).$$

For each $j \in \{1, 2, ..., M\}$, the distribution $q_\psi(z_j|\boldsymbol{h}^a, \boldsymbol{h}^m_{j-1})$ is discrete. Therefore, the variational objective function is NOT differentiable with respect to ψ. We consider relaxing the discrete distribution to a continuous distribution defined on the simplex $\Delta^I \subset \mathbb{R}^I$, where I is the total number of possible mental states, by *Gumbel-Max trick* [JGP16]. For a fixed choice of j, the sample space of z_j is

$$\left\{ \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}, \cdots, \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix} \right\} \subset \Delta^I,$$

with corresponding probabilities

$$\{\pi_0, \pi_1, \pi_2, ..., \pi_I\}, \quad \sum_{k=0}^{I} \pi_k = 1.$$

each of the probability π_j depends on the variational parameter ψ. Let $g_0, g_1, g_2, ..., g_I \sim_{i.i.d.} Gumbel(0, 1)$, then

$$\boldsymbol{y} \sim \text{Gumbel-Softmax Distribution}, \quad y_j := \frac{1}{\tau} softmax(\log \pi_0 + g_0, \log \pi_1 + g_1, ..., \log \pi_I + g_I), j = 1, 2, ..., I,$$

where τ is the temperature parameter. As $\tau \to 0^+$, the distribution of $\boldsymbol{y}$ approaches the distribution of z_j. Thus, $q_\psi(z_j|\boldsymbol{h}^a, \boldsymbol{h}^m_{j-1})$ can be reparameterized as

$$q_\psi(z_j|\boldsymbol{h}^a, \boldsymbol{h}^m_{j-1}) = q(f_j(\boldsymbol{g}; \psi, \tau)|\boldsymbol{h}^a, \boldsymbol{h}^m_{i-1}).$$

Since f is a differentiable mapping w.r.t. to ψ, we can take the gradient

$$\begin{aligned}
&\nabla_\psi \left(\mathbb{E}_{q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T))} \left[\log p_{\boldsymbol{\theta}}(\mathcal{H}_m(T), \mathcal{H}_a(T)) \right] + \mathbb{H} \left[q_\psi(\mathcal{H}_m(T)|\mathcal{H}_a(T)) \right] \right) \\
&= \nabla_\psi \left(\mathbb{E}_{\boldsymbol{g}_1, \boldsymbol{g}_2, ..., \boldsymbol{g}_M \sim s(\boldsymbol{g})} \left[\log p_{\boldsymbol{\theta}} \left(\{f_j(\boldsymbol{g}_j; \psi, \tau)\}_{j=1}^M, \mathcal{H}_a(T) \right) \right] + \nabla_\psi \mathbb{H} \left[q \left(\{f_j(\boldsymbol{g}_j; \psi, \tau)\}_{j=1}^M \right) \right] \right) \\
&= \mathbb{E}_{\boldsymbol{g}_1, \boldsymbol{g}_2, ..., \boldsymbol{g}_M \sim s(\boldsymbol{g})} \left[\sum_{j=1}^M \frac{\partial \log p_{\boldsymbol{\theta}}(\{f_j(\boldsymbol{g}_j; \psi, \tau)\}_{j=1}^M, \mathcal{H}_a(T))}{\partial f_j(\boldsymbol{g}_j; \psi, \tau)} \nabla_\psi f_j(\boldsymbol{g}_j; \psi, \tau) \right] + \nabla_\psi \mathbb{H} \left[q \left(\{f_j(\boldsymbol{g}_j; \psi, \tau)\}_{j=1}^M \right) \right].
\end{aligned}$$

Note that the term in red has a closed-form expression [AZNB22]. Therefore, we can sample from the distribution $s(\boldsymbol{g})$ to get a Monte-Carlo estimate of the gradient w.r.t. ψ of the objective function.

3.2 Monte-Carlo Expectation Maxmization

For latent variable problem, we need to jointly perform inference for hidden variables and learn the model parameters. *Expectation-Maximization* algorithm is a good choice [LWB17]. EM alternates between inference: computing the expected log-likelihood for the complete data w.r.t. the posterior:

$$\mathcal{L}(\boldsymbol{\theta}) := \mathbb{E}_{p(H_m(T)|\mathcal{H}_a(T), \boldsymbol{\theta}_{curr})} [\log p(\mathcal{H}_m(T), \mathcal{H}_a(T)|\boldsymbol{\theta})]$$

and learning: taking gradients $\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$ with respect to the model parameters.

The unique challenge of latent temporal logic point processes is that the latent variables take the form of a temporal logic point process. **Since the number of latent events is undetermined**, we propose a variety of methods for performing inference over sets of unknown cardinality. Monte Carlo methods are well-suited to this challenge, stochastically simulating sets of size. In particular, the causal, temporal structure of the latent temporal logic point process naturally suggests sequential Monte Carlo (SMC) methods.

Data-Driven Sequential Monte Carlo. Sequential Monte Carlo leverages the autoregressive nature of temporal logic point processes (the instantaneous rate is only a function of preceding events) to sequentially propose and resample particles. First, let $\{s_m\}_{m=1}^M, s_0 = 0, s_i < s_{i+1}, s_M = T$, define a *scaffold* that partitions the time range $[0, T)$ into M disjoint intervals. Moreover, we need to require that the time interval

to be *sufficiently narrow* so that based on our assumption, we could assume that the particle sampled from the proposal distribution consists of *only one* hidden mental state. The value of the p-th particle in the i-th interval, $z_i^{(p)}$, is a latent events: $z_i^{(p)} := (k_n^{(p)}, t_n^{(p)})$, where $k_n^{(p)}$ takes value in the collection of one-hot vectors and the zero vector in $\mathbb{R}^I$, is the type of the mental states and $t_n^{(p)}$ is the transition time. Contrast this with the set of observed events x_i for the same interval. We only propose latent events for vertices whose data is missing in that interval. We generate a candidate set of latent events for interval i by sampling from a proposal distribution, $z_i^{(p)} \sim r(z_i|\mathcal{H}_a(s_i), \mathcal{H}_m^{(p)}(s_{i-1}), \psi)$, and weighting the newly generated particle with the funciton,

$$\omega(\mathcal{H}_m^{(p)}(s_i)) = \frac{p(\mathcal{H}_a(s_i), \mathcal{H}_m^{(p)}(s_i)|\boldsymbol{\theta})}{p(\mathcal{H}_a(s_{i-1}), \mathcal{H}_m^{(p)}(s_{i-1})|\boldsymbol{\theta}) r(\mathcal{H}_m^{(p)}(s_{i-1})|\mathcal{H}_a(s_i), \mathcal{H}_m^{(p)}(s_{i-1}), \psi)}. \tag{20}$$

The key design choice is the proposal distribution. Intuitively, we need to choose the proposal distribution $r(z_i|\mathcal{H}_a(s_i), \mathcal{H}_m(s_{i-1}); \psi)$ to resemble $p(z_i^{(p)}|\mathcal{H}_a(s_i), \mathcal{H}_m(s_{i-1}); \boldsymbol{\theta})$.

The Choice of Proposal r. We can adopt the idea of smoothing in HMM [Mur23]. Note that in the paragraph above, we consider a proposal of the form $r(z_i^{(p)}|\mathcal{H}_a(s_i), \mathcal{H}_m(s_{i-1}); \psi)$, which only takes into account the past actions. This resembles the *forward filtering* in HMM. In the problem, because all of the observations are available, we could instead consider a proposal of the form

$$r(z_i^{(p)}|\mathcal{H}_a(T), \mathcal{H}_m(s_{i-1}); \psi), \tag{21}$$

which conditions on all the observations. Intuitively, it would provide more accurate inference results.

Monte Carlo Expectation Maximization. We jointly infer the latent mental states and learn the model parameter $\boldsymbol{\theta}$ (recall that $\boldsymbol{\theta}$ is the weights of all the logic rules) by EM algorithm. We begin with initializing the model parameter as $\boldsymbol{\theta}^{(0)} \leftarrow \mathbf{0}$ (or some other random vector). Then repeat the following steps:

1. for each interval $[s_{i-1}, s_i)$, we train the proposal by minimizing the KL-divergence

$$\psi^* := arg\min_{\psi} KL(p(z_i|\mathcal{H}_a(s_i), \mathcal{H}_m(s_{i-1}); \boldsymbol{\theta}_{curr})||r(z_i|\mathcal{H}_a(s_i), \mathcal{H}_m(s_{i-1}); \psi)). \tag{22}$$

   This makes the proposal distribution resemble the true posterior.

2. we sample M particles $\{z_i^{(p)}\}_{p=1}^M \sim r(z_i^{(p)}|\mathcal{H}_a(s_i), \mathcal{H}_m(s_{i-1}); \psi^*)$. Weight the particles by

$$\omega(\mathcal{H}_m^{(p)}(s_i)) = \frac{p(\mathcal{H}_a(s_i), \mathcal{H}_m^{(p)}(s_i)|\boldsymbol{\theta}_{curr})}{p(\mathcal{H}_a(s_{i-1}), \mathcal{H}_m^{(p)}(s_{i-1})|\boldsymbol{\theta}_{curr}) r(\mathcal{H}_m^{(p)}(s_{i-1})|\mathcal{H}_a(s_i), \mathcal{H}_m^{(p)}(s_{i-1}), \psi)}. \tag{23}$$

   Possibly, we need to resample the particles according to their weights. This replicates particles with large weights and removes particles that are inconsistent. Note that we can compute explicitly both of the p terms since the model is specified and the complete data up to time i and $i-1$ are given. (They are just two likelihoods)

3. repeat step 1 and 2 till the last time interval. We then obtain M particles $\{\mathcal{H}_m^{(p)}(T)\}_{p=1}^M$. Each of the particle could be viewed as a sample (with different weights) from the posterior distribution $p(\mathcal{H}_m(T)|\mathcal{H}_a(T); \boldsymbol{\omega}_{curr})$.

4. We maximize the expected log-likelihood of the complete data by stochastic gradient ascent. Note that

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} \mathbb{E}_{p(\mathcal{H}_m(T)|\mathcal{H}_a(T); \boldsymbol{\theta}_{curr})}[\log p(\mathcal{H}_m(T), \mathcal{H}_a(T)|\boldsymbol{\theta})] &= \mathbb{E}_{p(\mathcal{H}_m(T)|\mathcal{H}_a(T); \boldsymbol{\theta}_{curr})}[\nabla_{\boldsymbol{\theta}} \log p(\mathcal{H}_m(T), \mathcal{H}_a(T)|\boldsymbol{\theta})] \\ &\approx \sum_{p=1}^M \frac{\omega(\mathcal{H}_m^{(p)}(T))}{\sum_{p=1}^M \omega(\mathcal{H}_m^{(p)}(T))} \nabla_{\boldsymbol{\theta}} \log p(\mathcal{H}_m^{(p)}(T), \mathcal{H}_a(T)|\boldsymbol{\theta}).\end{aligned} \tag{24}$$

Then,

$$\boldsymbol{\theta}_{curr} := arg\max_{\boldsymbol{\theta}} \mathbb{E}_{p(\mathcal{H}_m(T)|\mathcal{H}_a(T);\boldsymbol{\theta}_{curr})}[\log p(\mathcal{H}_a(T), \mathcal{H}_m(T)|\boldsymbol{\theta})] \tag{25}$$

we perform stochastic gradient ascent to do the optimization.

5. Repeat step 1,2,3,4 until $\boldsymbol{\theta}$ converges.

3.3 Forwards-Backwards Algorithm

This section presents the inference algorithm for latent temporal logic point process, which imitates the *forwards-backwards* algorithm [Mur23] in HMM. In this algorithm, we compute forwards the posterior

$$p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}), \mathcal{H}_a(s_i))$$

and backwards the distribution

$$p(\mathcal{H}_a(s_{i+1:I})|\mathcal{H}_m(s_{i-1:i})).$$

Finally, we merge the results in forward passing and backward passing together to get the smoothing result $p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}), \mathcal{H}_a(s_I))$. In this way, unlike the Monte Carlo Expectation Maximization, we do not minimize the KL-divergence

$$KL(p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}), \mathcal{H}_a(s_i); \boldsymbol{\theta}_{curr})||r(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}), \mathcal{H}_a(s_i); \boldsymbol{\psi})),$$

instead, we minimize

$$KL(p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}), \mathcal{H}_a(s_I); \boldsymbol{\theta}_{curr})||r(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}), \mathcal{H}_a(s_I); \boldsymbol{\psi})).$$

Since we are dealing with the *offline* setting, all the actions $\mathcal{H}_a(s_I)$ are available. It would be sensible to consider the distribution $p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}), \mathcal{H}_a(s_I); \boldsymbol{\theta})$. To compute this posterior, we adopt an idea similar to the *forwards-backwards algorithm* in Hidden Markov Model. In forward passing, we compute the distribution $p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}), \mathcal{H}_a(s_i); \boldsymbol{\theta})$ recursively with the base case $p(\mathcal{H}_m(s_1)|\mathcal{H}_a(s_1)) = p(\mathcal{H}_a(s_1)|\mathcal{H}_m(s_1))p(\mathcal{H}_a(s_1))$ (we omit $\boldsymbol{\theta}$ in expressions for notational simplicity). In backward passing, we start from $p(\mathcal{H}_a(s_{I+1:I})|\mathcal{H}_m(s_{I-1:I})) = 1$, compute backwards $p(\mathcal{H}_a(s_{i+1:I})|\mathcal{H}_m(s_{i-1:i}))$ recursively. Note that the *transition* distribution

$$p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}), \mathcal{H}_a(s_{i-1}))$$

and the *emission* distribution

$$p(\mathcal{H}_a(s_{i-1:i})|\mathcal{H}_m(s_{i-1:i}))$$

are known.

Forward Passing. Note first that

$$\begin{aligned}
&p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}), \mathcal{H}_a(s_i)) \propto p(\mathcal{H}_m(s_i), \mathcal{H}_a(s_i)) = p(\mathcal{H}_a(s_{i-1:i})|\mathcal{H}_a(s_{i-1}), \mathcal{H}_m(s_i)) \\
&= p(\mathcal{H}_a(s_{i-1:i})|\mathcal{H}_m(s_{i-1:i}))p(\mathcal{H}_m(s_i), \mathcal{H}_a(s_{i-1})) \\
&= p(\mathcal{H}_a(s_{i-1:i})|\mathcal{H}_m(s_{i-1:i}))p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{1:i-1}), \mathcal{H}_a(s_{1:i-1}))p(\mathcal{H}_m(s_{i-1}), \mathcal{H}_a(s_{i-1}))) \\
&= p(\mathcal{H}_a(s_{i-1:i})|\mathcal{H}_m(s_{i-1:i}))p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{1:i-1}), \mathcal{H}_a(s_{1:i-1})) \\
&\cdot p(\mathcal{H}_a(s_{i-2:i-1})|\mathcal{H}_m(s_{i-2:i-1}))p(\mathcal{H}_m(s_{i-2:i-1})|\mathcal{H}_m(s_{i-2}), \mathcal{H}_a(s_{i-2}))p(\mathcal{H}_m(s_{i-2}), \mathcal{H}_a(s_{i-2}))) \\
&= \cdots \\
&= \prod_{t=1}^{i} p(\mathcal{H}_a(s_{t-1:t})|\mathcal{H}_m(s_{t-1:t}))p(\mathcal{H}_m(s_{t-1:t})|\mathcal{H}_m(s_{t-1}), \mathcal{H}_a(s_{t-1})).
\end{aligned} \tag{26}$$

Equivalently, we can write down the recursive formula

$$\begin{aligned}
&p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}),\mathcal{H}_a(s_i)) \\
&\propto p(\mathcal{H}_a(s_{i-1:i})|\mathcal{H}_m(s_{i-1:i}))p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{1:i-1}),\mathcal{H}_a(s_{1:i-1})) \\
&\cdot\prod_{t=1}^{i-1} p(\mathcal{H}_a(s_{t-1:t})|\mathcal{H}_m(s_{t-1:t}))p(\mathcal{H}_m(s_{t-1:t})|\mathcal{H}_m(s_{t-1}),\mathcal{H}_a(s_{t-1})) \\
&= p(\mathcal{H}_a(s_{i-1:i})|\mathcal{H}_m(s_{i-1:i}))p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{1:i-1}),\mathcal{H}_a(s_{1:i-1}))p(\mathcal{H}_m(s_{i-2:i-1})|\mathcal{H}_m(s_{i-2}),\mathcal{H}_a(s_{i-1}))
\end{aligned} \tag{27}$$

The base case is $p(\mathcal{H}_m(s_1)|\mathcal{H}_a(s_1)) = p(\mathcal{H}_a(s_1)|\mathcal{H}_m(s_1))p(\mathcal{H}_a(s_1))$. This term can be calculated analytically since $p(\mathcal{H}_a(s_1)|\mathcal{H}_m(s_1))$ is just the sampling model (since $\mathcal{H}_m(s_1)$ is already sampled, we know at every time point, the rate of the action process, hence we can sample x_1 using sampling methods for point processes) while $p(\mathcal{H}_m(s_1))$ is triggered only by the spontaneous rate (Thus, in the first time interval, we are sampling from a (possibly inhomogeneous, determined by the choice of the rate $b(t)$) Possion process). Therefore, continuely apply (27), we final have $p(\mathcal{H}_m(s_{I-1:I})|\mathcal{H}_m(s_{I-1}),\mathcal{H}_a(s_I))$.

Backward Passing. We start from the base case $p(x_{I+1:I}|z_I) = 1$ and compute backwards recursively by the following formula

$$\begin{aligned}
&p(\mathcal{H}_a(s_{i:I})|\mathcal{H}_m(s_{i-2:i-1})) = \int p(\mathcal{H}_m(s_{i-1:i}),\mathcal{H}_a(s_{i+1:I})|\mathcal{H}_m(s_{i-1:i}))d\mathcal{H}_m(s_{i-1:i}) \\
&= \int p(\mathcal{H}_a(s_{i-1:i})|\mathcal{H}_m(s_{i-1:i}),\mathcal{H}_m(s_{i:I}),\mathcal{H}_m(s_{i-2:i-1}))p(\mathcal{H}_m(s_{i-1:i}),\mathcal{H}_a(s_{i:I})|\mathcal{H}_m(s_{i-2:i-1}))d\mathcal{H}_m(s_{i-1:i}) \\
&= \int p(\mathcal{H}_a(s_{i-1:i})|\mathcal{H}_m(s_{i-1:i}))p(\mathcal{H}_m(s_{i-1:i}),\mathcal{H}_a(s_{i:I})|\mathcal{H}_m(s_{i-2:i-1}))d\mathcal{H}_m(s_{i-1:i}) \\
&\propto \int p(\mathcal{H}_a(s_{i-1:i})|\mathcal{H}_m(s_{i-1:i}))p(\mathcal{H}_a(s_{i+1:I})|\mathcal{H}_m(s_{i-2:i}))d\mathcal{H}_m(s_{i-1:i}) \\
&= \int p(\mathcal{H}_a(s_{i-1:i})|\mathcal{H}_m(s_{i-1:i}))p(\mathcal{H}_a(s_{i+1:I})|\mathcal{H}_m(s_{i-1:i}))d\mathcal{H}_m(s_{i-1:i})
\end{aligned} \tag{28}$$

Merging. Finally, we have

$$\begin{aligned}
&p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}),\mathcal{H}_a(s_I)) = p(\mathcal{H}_m(s_{i-1:i}))|\mathcal{H}_m(s_{i-1}),\mathcal{H}_a(s_i),\mathcal{H}_a(s_{i:I})) \\
&\propto p(\mathcal{H}_m(s_{i-1:i}),\mathcal{H}_m(s_{i-1}),\mathcal{H}_a(s_{i:I})|\mathcal{H}_a(s_i)) \\
&= p(\mathcal{H}_a(s_{i:I})|\mathcal{H}_m(s_{i-1:i}),\mathcal{H}_m(s_{i-1}),\mathcal{H}_a(s_i))p(\mathcal{H}_m(s_{i-1:i}),\mathcal{H}_m(s_{i-1})|\mathcal{H}_a s_i)) \\
&\propto p(\mathcal{H}_a(s_{i:I})|\mathcal{H}_m(s_{i-1:i}))p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}),\mathcal{H}_a(s_i)).
\end{aligned} \tag{29}$$

Therefore, we can sample from $p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}),\mathcal{H}_a(s_I))$ to get particles, which should be more accurate than $p(\mathcal{H}_m(s_{i-1:i})|\mathcal{H}_m(s_{i-1}),\mathcal{H}_a(s_i))$.

3.4 Experiments

3.4.1 Synthetic Data

Generation. We consider generate synthetic data with *three* mental processes, *four* action processes (predicate 0-2: mental; predicate 3-6: action.), and *eight* prespecified logic rules. We simulate these processes up to a preset time horizon T. Use the simulated actions as the observations to jointly perform posterior inference and model learning. See fig. 4, 5, 6, 7, 8, 9 for the generation results.

Learning from Complete Data. If the complete data is observed, we can treat the log-likelihood as the loss function and perform standard gradient ascent method to learn the model parameter $\boldsymbol{\theta}$, which is the weights of the prespecified logic rules and base terms. However, standard gradient method suffers from the problem that the rule weights might become negative when the model parameter converges. For the constrained problem, one might need to utilize the *gradient projection method* instead to take the constraints of the problem into consideration.

Learning from Incomplete Data. We use the algorithm described in section 3.1 to jointly perform inference and learning. 48 samples are generated by TLPP_Generation. The batch size is set to be 16. Only the action data is used. See fig. 10, 12, 11, 13 for the results of learning form incomplete dataset.

4 Research Plan and Expected Outcome

4.1 Research Plan

1. Debugging the currently implemented code
2. Do experiments on the synthetic data and check if the model parameter $\boldsymbol{\theta}$ and the variational parameter $\boldsymbol{\psi}$ converge
3. Do experiments on real world dataset

4.2 Expected Outcome

1. After the ELBO $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\psi}, \mathcal{H}_a(T))$ converges, the model parameter $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$ converge as well
2. The algorithm for joint inference and learning can be scaled to large dataset
3. The algorithm performs well on both the synthetic data and the real-world dataset and the running speed is relatively fast

References

[AS19] Hamidreza Alvari and Paulo Shakarian. Hawkes process for understanding the influence of pathogenic social media accounts. In *2019 2nd International Conference on Data Intelligence and Security (ICDIS)*, pages 36–42. IEEE, 2019.

[AZNB22] Kareem Ahmed, Zhe Zeng, Mathias Niepert, and Guy Van den Broeck. Simple: A gradient estimator for k-subset sampling. *arXiv preprint arXiv:2210.01941*, 2022.

[BMM15] Emmanuel Bacry, Iacopo Mastromatteo, and Jean-François Muzy. Hawkes processes in finance. *Market Microstructure and Liquidity*, 1(01):1550005, 2015.

[JGP16] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[KW13] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[LWB17] Scott W Linderman, Yixin Wang, and David M Blei. Bayesian inference for latent hawkes processes. *Advances in Neural Information Processing Systems*, 2017.

[LWZ+20] Shuang Li, Lu Wang, Ruizhi Zhang, Xiaofu Chang, Xuqin Liu, Yao Xie, Yuan Qi, and Le Song. Temporal logic point processes. In *International Conference on Machine Learning*, pages 5990–6000. PMLR, 2020.

[MQE19] Hongyuan Mei, Guanghui Qin, and Jason Eisner. Imputing missing events in continuous-time event streams. In *International Conference on Machine Learning*, pages 4475–4485. PMLR, 2019.

[Mur23] Kevin P. Murphy. *Probabilistic Machine Learning: Advanced Topics*. MIT Press, 2023.

[RBS10] Patricia Reynaud-Bouret and Sophie Schbath. Adaptive estimation for hawkes processes; application to genome analysis. *The Annals of Statistics*, 38(5):2781–2822, 2010.

[Ros21] Gordon J Ross. Bayesian estimation of the etas model for earthquake occurrences. *Bulletin of the Seismological Society of America*, 111(3):1473–1480, 2021.

5 Appendix

5.1 Generation Results

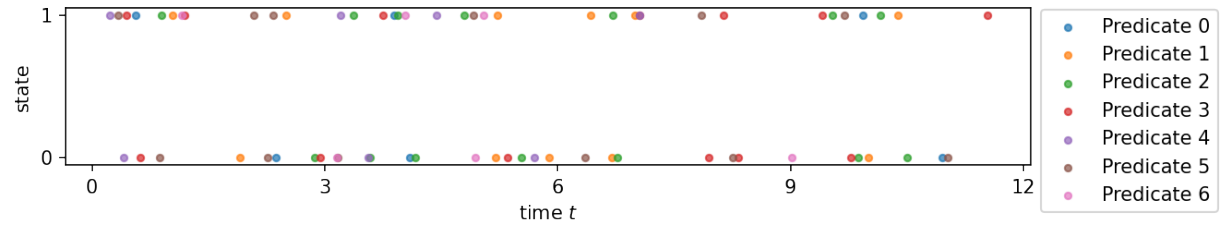

Figure 4: Simulation result: the x-axis represents time t and the y-axis represents the state of the corresponding predicate.

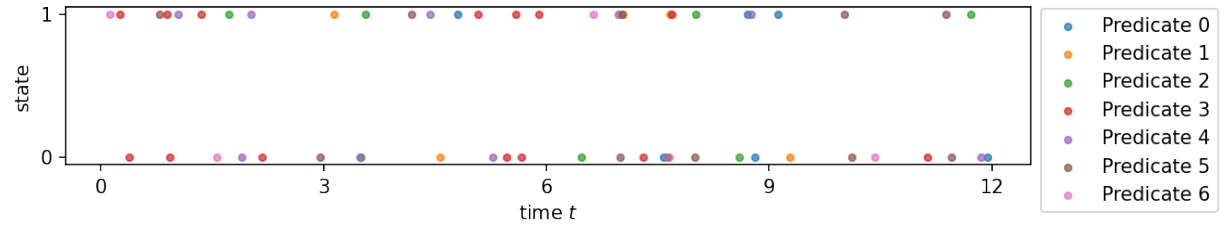

Figure 5: Simulation result: the x-axis represents time t and the y-axis represents the state of the corresponding predicate.

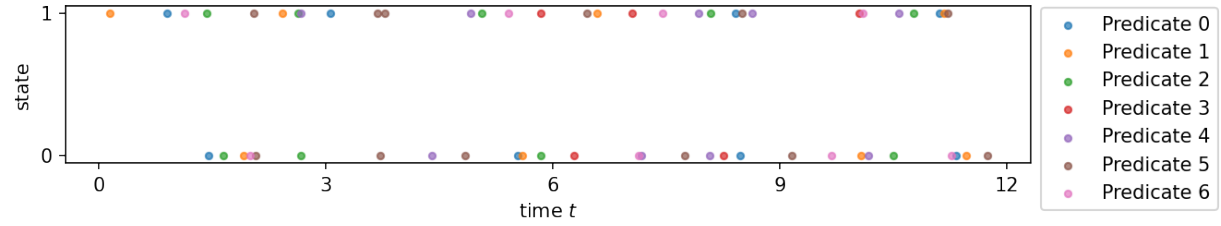

Figure 6: Simulation result: the x-axis represents time t and the y-axis represents the state of the corresponding predicate.

5.2 Learning Results

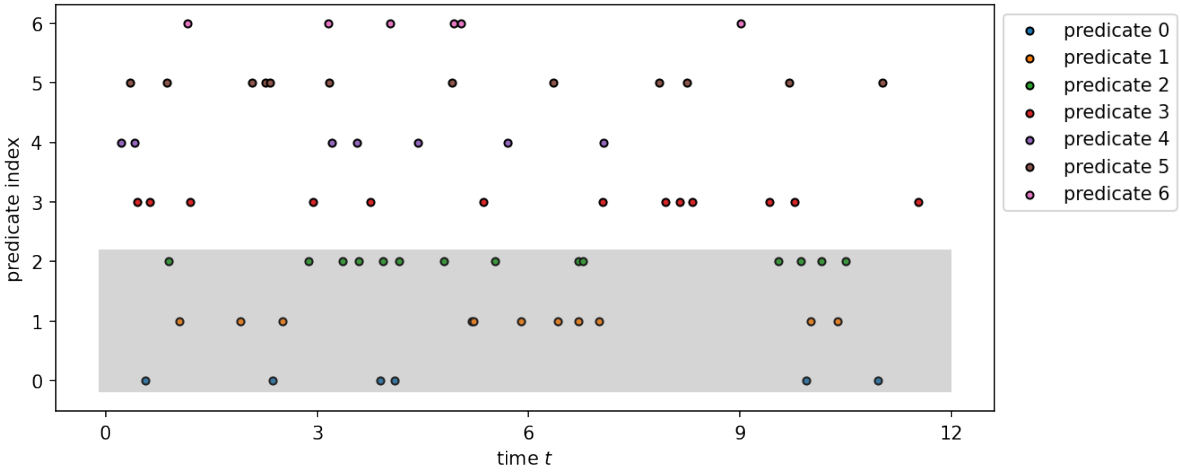

Figure 7: Simulation result: the x-axis represents time t and the y-axis represents the index of the predicate.

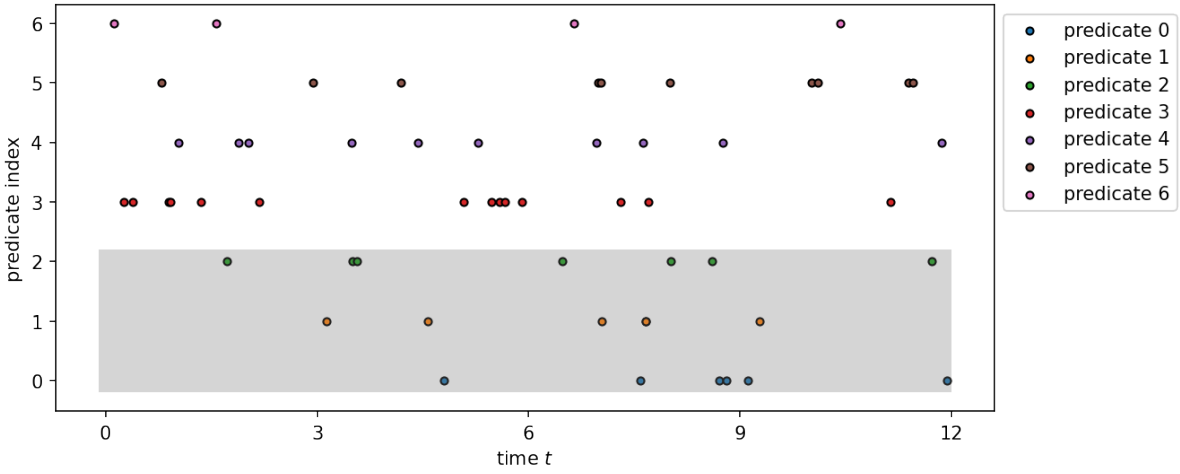

Figure 8: Simulation result: the x-axis represents time t and the y-axis represents the index of the predicate.

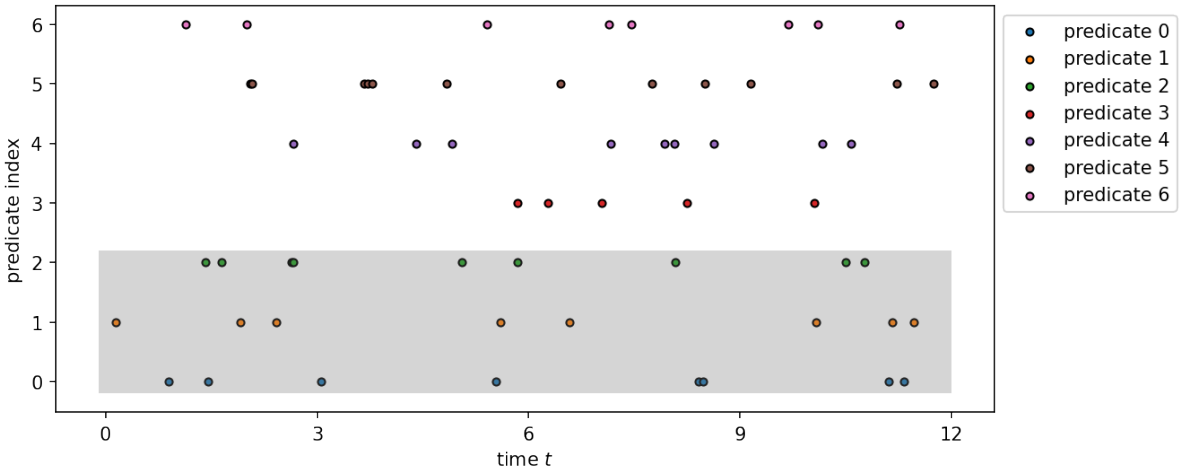

Figure 9: Simulation result: the x-axis represents time t and the y-axis represents the index of the predicate.

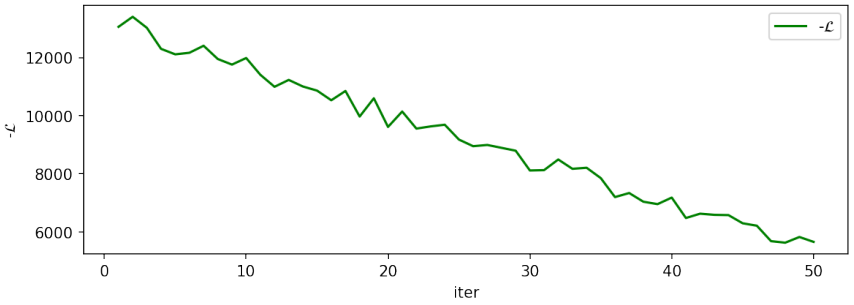

Figure 10: Negative Evidence Lower Bounded (ELBO) $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\psi}, \mathcal{H}_a(T))$.

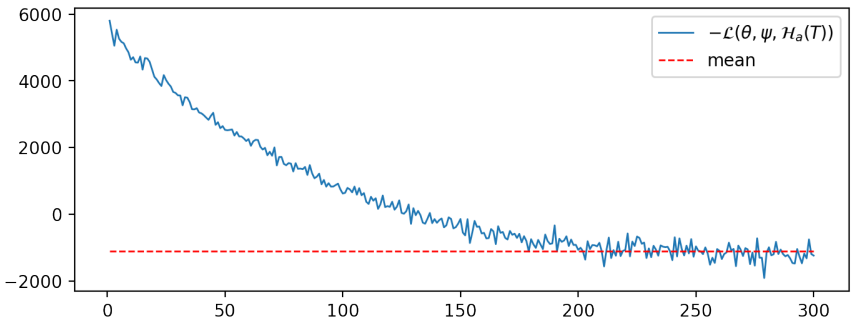

Figure 11: After around 200 iterations, the ELBO $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\psi}, \mathcal{H}_a(T))$ converges.

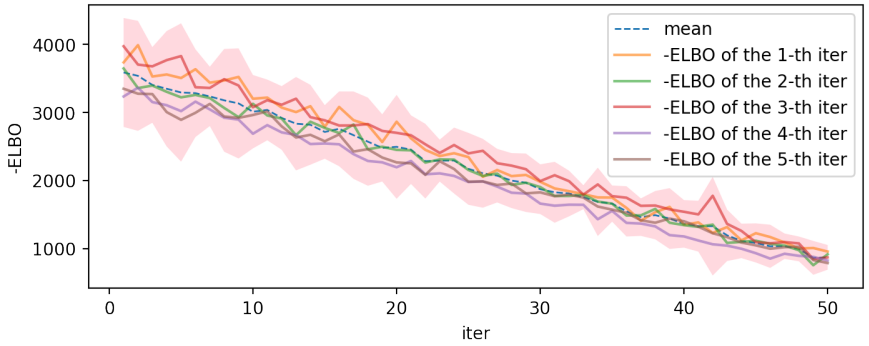

Figure 12: the pink shaded area represents the confidence interval [mean-3*std,mean+3*std].

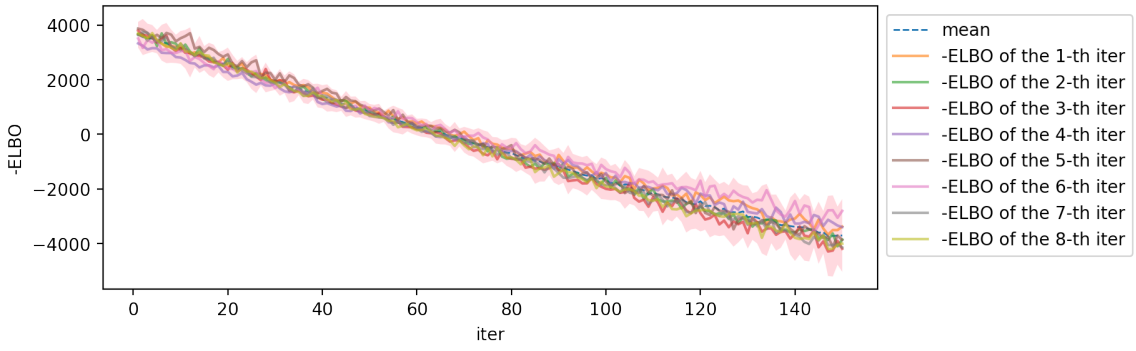

Figure 13: the pink shaded area represents the confidence interval [mean-3*std,mean+3*std].