# Latent Logic Tree Extraction for Event Sequence Explanation from LLMs

Zitao Song [1]   Chao Yang [2]   Chaojie Wang [3]   Bo An [3,1]   Shuang Li [2]

## Abstract

Modern high-stakes systems, such as healthcare or robotics, often generate vast streaming event sequences. Our goal is to design an efficient, plug-and-play tool to elicit logic tree-based explanations from Large Language Models (LLMs) to provide customized insights into each observed event sequence. Built on the temporal point process model for events, our method employs the likelihood function as a score to evaluate generated logic trees. We propose an amortized Expectation-Maximization (EM) learning framework and treat the logic tree as latent variables. In the E-step, we evaluate the posterior distribution over the latent logic trees using an LLM prior and the likelihood of the observed event sequences. LLM provides a high-quality prior for the latent logic trees, however, since the posterior is built over a discrete combinatorial space, we cannot get the closed-form solution. We propose to generate logic tree samples from the posterior using a learnable GFlowNet, which is a diversity-seeking generator for structured discrete variables. The M-step employs the generated logic rules to approximate marginalization over the posterior, facilitating the learning of model parameters and refining the tunable LLM prior parameters. In the online setting, our locally built, lightweight model will iteratively extract the most relevant rules from LLMs for each sequence using only a few iterations. Empirical demonstrations showcase the promising performance and adaptability of our framework.

[1]School of Computer Science and Engineering, Nanyang Technological University, Singapore [2]School of Data Science, The Chinese University of Hong Kong, Shenzhen, China [3]Skywork AI, Singapore. Correspondence to: Shuang Li <lishuang@cuhk.edu.cn>.
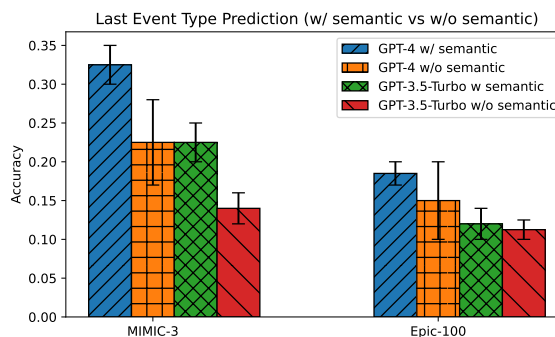
*Figure 1. GPT can help last event type prediction.* We found replacing the semantic meaningful event names by numerical event ids in event history degrades the performance of event prediction.

## 1. Introduction

Modern systems, such as healthcare, finance, and social media, produce voluminous data that are represented as discrete events with irregular timestamps. Generating concise and human-readable knowledge to explain this intricate event data is of great scientific and practical value. The distilled knowledge can be generalized to other contexts (Ullman et al., 2012; Campero et al., 2018).

For example, in healthcare, electronic health records (EHRs) are often represented as discrete event sequences, containing fine-grained time and type information on doctors' treatments, patients' measurements, and symptoms. It is desirable to generate concise medical knowledge such as the disease phenotypes and therapies, to shed light on these messy events. This will facilitate a deeper understanding of each patient's unique health journey and medical decisions, ultimately leading to more effective and individualized care.

However, the heterogeneity observed in each patient's data poses a challenge – each event sequence may exhibit diverse natures of medical histories, treatments, and conditions (Henrich & McElreath, 2003; Laland, 2004). Generating the most relevant and accurate knowledge to explain such heterogeneous data requires sophisticated methods.

Recently, Large Language Models (LLMs) have demonstrated promising human-like reasoning abilities as few-shot learners (Brown et al., 2020). When prompted with step-wise explanations of reasoning, these models excel in logical reasoning (Pan et al., 2023), abstract pattern in-

duction (Webb et al., 2023), and social learning (Leng & Yuan, 2023). Despite their success in text-based reasoning tasks, LLMs still face challenges in extending their reasoning capabilities to tabular data (Hegselmann et al., 2023) and discrete event sequences (Shi et al., 2023).

In this paper, we propose to leverage LLMs, trained on general-domain data, as a *prior* to generate human-readable knowledge. Specifically, we will encourage LLMs to generate logic trees, from their prior distribution $p(\mathcal{R})$. Given the observed discrete event data $\mathbf{X}$, the belief on the logic trees will be updated according to Bayes rule, i.e., $p(\mathcal{R}|\mathbf{X}) \propto p(\mathcal{R})p(\mathbf{X}|\mathcal{R})$ (Leng & Yuan, 2023; Acemoglu et al., 2011), where we will use a *temporal logic point process* (TL-PP) (Li et al., 2020) to model $p(\mathbf{X}|\mathcal{R})$. The inference procedure can be regarded as performing the reweighting of each logic tree from LLMs. The goal of our paper is *to perform the logic tree inference using LLM prior in a tractable and efficient manner*.

Performing inference on logic trees is challenging since the posterior distribution is intractable due to their discrete combinatorial space. Traditional solutions, such as MCMC, approximate intractable posteriors by sampling, yet is struggling with multi-modal distributions (Miao et al., 2019; Zhang et al., 2020; Lew et al., 2023). Reinforcement learning (RL) methods like proximal policy optimization (PPO) (Schulman et al., 2017), treating the sampling process as a policy, may also fail to capture the distribution's full diversity (Zhu et al., 2023). The problem becomes worse when the target distribution is incorrectly specified, leading to an overoptimized policy. In our paper, we will address the inference challenge using the GFlowNet (Bengio et al., 2023), a recently proposed sound diversity-seeking generative model for structured discrete variables. As a deep RL algorithm adept at managing unnormalized rewards, GFlowNet has shown effectiveness in fine-tuning Large Language Models with intractable thought posteriors (Hu et al., 2023a). We wish to extend its success to Tree-of-Thoughts (ToT) reasoning (Yao et al., 2023), such as generating logic trees to explain event sequences.

Our overall learning follows an amortized EM algorithm, where we treat the logic tree as latent variables. In the E-step, we train a GFlowNet to generate logic tree samples from their posterior distribution. The GFlowNet model parameters are shared across all training event sequences, which is the reason why we term it amortized EM. In the M-step, we use the generated logic tree samples to approximate marginalization over the posterior. This process provides an objective function for learning the TL-PP model parameters and refining some tunable LLM parameters (assuming the LLM priors are also learnable). The algorithm iterates between the E-step and M-step until convergence. During the testing stage, given a new event sequence, we employ the trained GFlowNet to efficiently perform inference

for explanatory logic trees by sampling from the posterior $p(\mathcal{R}|\mathbf{X}) \propto p(\mathcal{R})p(\mathbf{X}|\mathcal{R})$, leveraging well-trained priors and models from the training stage. This enables our method to efficiently and adaptively explain previously unseen event sequences. Our main contributions are:

(*i*) We introduce, **LaTee**, an amortized EM learning framework that can learn to infer and generate **La**tent logic **T**rees to **e**xplain observed **e**vent sequences, which leverages LLMs as prior;

(*ii*) In the E-step, we use GFlowNets to fine-tune LLMs and enable diverse logic tree generation, which better tackles the heterogeneity issue exhibited for event sequences;

(*iii*) Our method generates a relative margin of 20% over SOTA Attentioin-based temporal point process (TPP) models on future event prediction based on real-world behavior datasets. This shows that our interpretable and knowledge-driven TPP model is also flexible.

## 2. Related Works and Background

### 2.1. Knowledge Extraction from Event Sequences

Knowledge extraction refers to the process of refining, condensing, or summarizing large volumes of raw data to distill the most relevant and essential information. For noisy event sequences, we will represent our knowledge as a collection of symbolic logic trees, which is a hierarchical and structured representation of logical relationships among different elements or propositions (Campero et al., 2018). Our logic tree extraction from events is related to *symbolic rule induction* and *semantic cognition*.

**Symbolic Rule Induction.** Symbolic rule induction refers to the process of automatically discovering logical rules from observed data. Classic symbolic inductive logic programming (ILP) methods (Quinlan, 1990; Cropper & Tourret, 2020) mostly adopt discrete search in the space of logic programs and do very well at generalizing from just a few examples. Neuro-symbolic rule inductions (Evans & Grefenstette, 2018; Yang et al., 2017; Rocktäschel & Riedel, 2017; Campero et al., 2018) are *differentiable* ILP methods and are generally more robust to noisy input. In our approach, we take inspiration from a differentiable *backward chaining algorithm* (Rocktäschel & Riedel, 2017) and represent a symbolic logic tree starting from the target predicates. For instance, consider a *Put-into* task as our target predicate, in which we need to replace element $X$ from box $Y_1$, room $Z_1$ into box $Y_2$, room $Z_2$. We can represent this actionable strategy as a set of ordering logic rules as:

$$Put\text{-}into(X, Y) \leftarrow Open(Y) \wedge Pick\text{-}up(X), \quad (1)$$
$$Pick\text{-}up(X) \leftarrow Open(Y) \quad (2)$$
$$Open(Y) \leftarrow Move\text{-}to(Z) \quad (3)$$

which is a logic tree, with $Put\text{-}into(X, Y)$ being the root and other predicates being its children. Many classic or differentiable ILP methods can automatically learn such rules from data, however, they require carefully hand-crafted rule

templates for each ILP task in order to constrain and reduce the search space effectively (Glanois et al., 2022).

**Semantic Cognition.** Semantic cognition refers to the development of systems that can comprehend and manipulate meaning in a manner similar to human cognitive processes. It explores how knowledge is organized, represented, and utilized to derive semantic understanding from various forms of data. Previous research has described it as a process similar to reducing logical dimensions (Katz et al., 2008; Ullman et al., 2012) through employing probabilistic generative models. These models are capable of learning both logical rules and fundamental relationships that explain the data observed. Similar to ILP methods, they can perform deductive reasoning using logical rules. However, unlike traditional ILP methods, these models can also induce facts. While these approaches showed potential, they faced significant issues with scalability. The recent advancements in social learning in LLMs (Leng & Yuan, 2023) suggest that it might be beneficial to reexamine these concepts.

## 2.2. Knowledge-Driven Probabilistic Models for Event Sequences

Temporal point process (TPP) provides an elegant probabilistic model for continuous-time event sequences, which is characterized by an intensity function. The intensity function represents the occurrence rate of events, which is usually modeled as parametric, nonparametric, or deep neural network forms. Traditional parametric TPP models like the Hawkes process offer interpretability, but their simplicity limits flexibility. On the other hand, 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. In this paper, we aim to generate logic trees from a fine-tuned LLM to inform the functional form of the intensity, which strikes a balance between model flexibility and interpretability. The modeling idea takes inspiration from TL-PP (Li et al., 2020) , as detailed below.

**Rule-informed Event Sequences.** We will build a rule-informed conditional intensity function for the event sequences as:

$$\lambda(t; w, \mathcal{R}, \mathbf{X}_t) := \exp\Big\{ \sum_{f \in \mathcal{R}} w_f \phi_f(\mathbf{X}_t) + b(t) \Big\}, \quad (4)$$

where $f$ is a valid path from the symbolic logic tree $\mathcal{R}$, $\phi_f(\mathbf{X}_t)$ is the logic-informed feature derived from the number of ordered event combinations in event history $\mathbf{X}_t$ satisfying the path $f$ (with more details can be found in (Li et al., 2020)), and $w_f$ is the weight corresponding to rule $f$. Given this probabilistic model, we can use the negative log-likelihood of the temporal point process as a loss function to jointly learn weights $w$ and symbolic structure $\mathcal{R}$. Given a event sequence $\mathbf{X} = \{(t_i, e_i)\}_{i=1}^L$ observed over an interval

$[0, T]$, the negative log-likelihood of $\mathbf{X}$ is expressed as:

$$\mathcal{L}_{w,\mathcal{R}}(\mathbf{X}) = \quad (5)$$
$$-\sum_{j=1}^L \log \lambda(t_j; w, \mathcal{R}, \mathbf{X}_j) + \int_0^T \lambda(t; w, \mathcal{R}, \mathbf{X}_t) dt.$$

where each $t_j$ is the event trigger time and $\mathbf{X}_j$ refers to the event sequences up to $t_j$. Nevertheless, this learning problem is challenging because it requires learning the parameters $w$ in a continuous space as well as the symbolic structure $\mathcal{R}$ in a discrete space.

## 2.3. Human-Like Reasoning in LLMs

Recent developments in LLMs, such as GPT-4 (Achiam et al., 2023) and LlaMA 2 (Touvron et al., 2023), have extended the capabilities of AI beyond conventional predictive analytics to simulate sophisticated human-like interactions in various systems (Gao et al., 2023). In-context learning (ICL) within LLMs is a notable feature where the model performs tasks based on input-output examples without adjusting any parameters. Importantly, ICL can be understood through a Bayesian inference framework (Xie et al., 2021), where the augmented prompt serves as a semantic prior, guiding *latent* concepts acquired during pre-training for chain of thought reasoning and subsequent output (Wei et al., 2022; Kojima et al., 2022). Despite the transformative nature of ICL in LLMs, which allows them to adapt to new tasks without explicit retraining, challenges remain in explaining extrapolation to unseen tasks and understanding the impact of model architecture and optimization. Conversely, knowledge extraction from locally deployable LLMs, achieved through careful fine-tuning (Schick & Schütze, 2020) using Parameters Efficient Fine Tuning (PEFT) techniques (Hu et al., 2021; Dettmers et al., 2023), also provides valuable insights. We will adopt PEFT ideas in this paper.

## 2.4. Intractable Bayesian Inference in LLMs

The challenge in inferring the latent logical reasoning path from LLMs stems from the intractability of the posterior. Given question-answer pair $(X, Y)$, the posterior of the latent chain-of-thought $p_{LM}(Z|X, Y) = \frac{p_{LM}(X, Z, Y)}{\sum_{Z'} p_{LM}(X, Z', Y)}$ is intractable due to the discrete combinatorial space for thoughts $Z$ (Hu et al., 2023a). Existing approaches to address this intractable inference problem in language models often resort to tokenwise approximations using techniques like tempered and contrastive sampling (Malkin et al., 2021; Li et al., 2022), along with problem-specific strategies like beam search and local search techniques (Lu et al., 2021; Sha, 2020). In our paper, we will use GFlowNets to guide posterior sampling of logic trees and fine-tune LLMs.

**GFlowNets as Posterior Samplers in LLMs.** GFlowNets (Bengio et al., 2021; 2023) are originally introduced as a diversity-seeking probabilistic reinforcement learning algo-
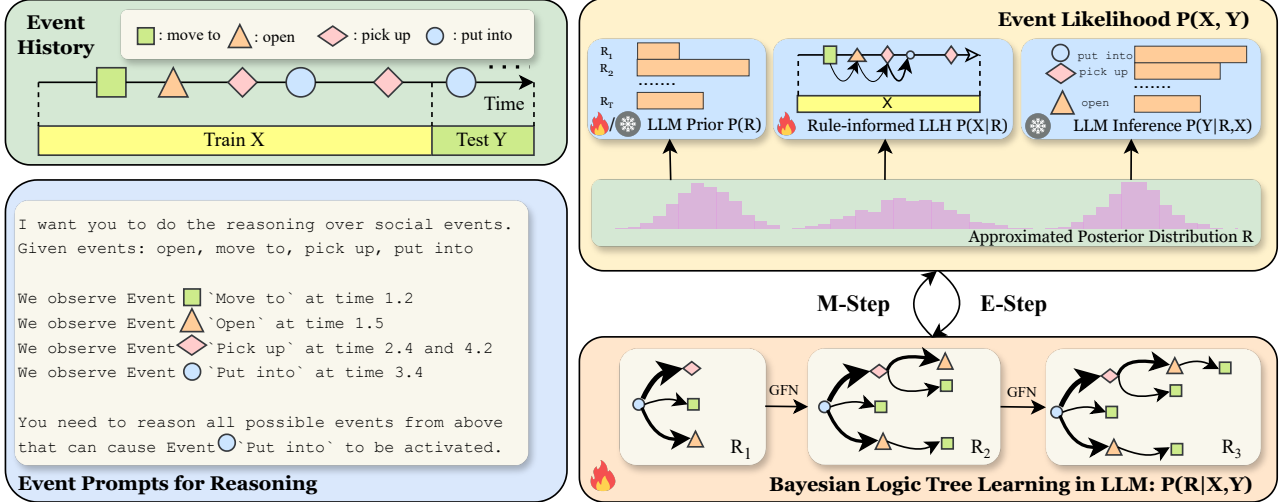
*Figure 2. The Architecture of Proposed Framework.* The presented event history represents a typical human trajectory: beginning with relocating to a new place, opening a box, picking up an object, and culminating in placing the retrieved item in a specific location. In the training phase, we first convert this explicit event history into a textual format. Subsequently, we employ a LLM to execute conditional sampling and perform backward reasoning, starting from the predetermined goal (E-Step). The resultant reasoning pathway is then transformed into a symbolic logic tree, which aids in updating the event probabilities (M-Step). In this context, the *campfire* icon signifies that the model is being updated, while the *snowflake* icon indicates that the model is in a 'frozen' state. We use the thickness of a path in the symbolic logic tree to represent its posterior probability.

rithm for molecular discovery. A recent work (Hu et al., 2023a) connects GFlowNets to Chain-of-Thoughts (CoT) generation, by leveraging the amortized inference ability of GFlowNets. In their work, given an unnormalized density (reward) $R : \mathcal{Z} \to \mathbb{R}_{>0}$, GFlowNets learn policies to sample sequences in a token-level, i.e., $\mathbf{Z}_t = z_1 z_2 \cdots z_t \mathtt{T} \in \mathcal{Z}$ (where $z_i$ is a language token and $\mathtt{T}$ denotes End of Sentence token), as if they were sampling from a target distribution. The goal of GFlowNets is to fine-tune a token-level language generation $q_{GFN}(\mathbf{Z}_t|\mathbf{Z}_{t-1}; \theta)$ initialized by LLM such that marginal $q_{GFN}(\mathbf{Z}_t) \propto r(\mathbf{Z}_t)$, i.e., driving the marginal like-lihood of generating a complete sequence is proportional to its reward. The learning objective for GFlowNets is defined by the subtrajectory balance (SubTB) objective, equiva-lent to the path consistency objective (Nachum et al., 2017; Deleu et al., 2024; Tiapkin et al., 2024) in Max-Entropy RL (Haarnoja et al., 2017).

## 3. Our Proposed LaTee using LLM prior

Instead of eliciting linear thoughts from LLMs, our focus is to extract and reweight symbolic logic trees generated from LLMs to explain the dynamics of the observed event sequences. We hope the obtained logic trees will not only offer personalized explanations for each event sequence but also enable accurate future events prediction.

### 3.1. LLM-Symbolic Integration by Latent Variables

Given event sequences $X$ and next event type $Y$, where $X$ records explicit event sequences with $k$ events $X =$

$\{(t_i, e_i)\}_{i=1}^{k}$, where $t_i$ is the $i$-th event time, $e_i$ is the $i$-th event type, and $Y = e_{k+1}$ is the next event type after the $k$-th event. We are interested in finding a collection of latent symbolic logic trees $\mathcal{R}$, which are composed of various event types that trigger subsequent events and best explain the likelihood of the observed event sequence:

$$p(X, Y) = \sum_{\mathcal{R}} p(X, Y|\mathcal{R})p(\mathcal{R}). \tag{6}$$

For this mixture latent variable model (LVM), we treat $\mathcal{R}$ as latent variables; $p(\mathcal{R})$ is the prior distribution for the latent logic tress; and the joint likelihood of the event se-quences $p(X, Y|\mathcal{R})$ can be derived from temporal point process framework, as shown in Eq. (5).

We will employ LLMs as the prior $p(\mathcal{R})$ for logic trees. Additionally, if we aim to leverage the powerful reason-ing and generation capabilities of LLMs to predict $Y$ — for instance, in the context of symptom-treatment pairs or question-answer pairs for $(X, Y)$ — it becomes intriguing to explore the recommendations of $Y$ given $X$ provided by LLMs. Consequently, we further decompose the mixture LVM (as shown in Eq. (6)) as:

$$p(X, Y) = \sum_{\mathcal{R}} p(X, Y|\mathcal{R})p_{LM}(\mathcal{R}), \tag{7}$$

$$= p_{LM}(Y|X, \mathcal{R}) \sum_{\mathcal{R}} p_w(X|\mathcal{R})p_{LM}(\mathcal{R}; \phi). \tag{8}$$

We aim to jointly optimize the event likelihood parameter $w$ and the tunable parameters $\phi$ in the prior language model. The challenge in learning arises from the latent variables $\mathcal{R}$. Fortunately, the EM algorithm provides an effective tool for learning mixture models with latent variables. However,

in the E-step, we need to analytically evaluate the current posterior $p(\mathcal{R}|X,Y) \propto p_{LM}(Y|X,\mathcal{R})p_w(X|\mathcal{R})p_{LM}(\mathcal{R})$, which is intractable due to that the partition function requires the summation over the discrete space of $\mathcal{R}$. To tackle this intractability, variational-EM algorithm (Dempster et al., 1977; Beal, 2003; Koller & Friedman, 2009) can be used to approximate the posterior by optimization. We will address this issue by introducing an amortized EM, where in the E-step we learn GFlowNets to sample from $p(\mathcal{R}|X,Y)$ without the need to calculate the partition function.

### 3.2. Amortized EM framework for Logic Tree Inference

The derivation of the Evidence Lower Bound (ELBO) for Eq. (7) is presented in Appendix E. It explains the rationale for analytically evaluating the posterior in the E-step to achieve a tight ELBO.

Specifically, in the E-step, we will draw samples from the posterior over the latent symbolic logic tree, denoted as $p_{LM}(\mathcal{R}|X,Y)$, which comes from an amortized sampler of $\mathcal{R}$ with an LLM as its policy. In the M-step, we maximize the log-likelihood of the joint probability of the sampled latent variables $\mathbb{E}_{\mathcal{R}\sim p(\mathcal{R}|X,Y)}[\log p_{LM}(Y|X,\mathcal{R})p_w(X|\mathcal{R})p_{LM}(\mathcal{R})]$ with respect to the parameters of $w$. This combination of amortized inference (learning to sample the symbolic logic tree from the language model) and supervised learning (optimizing the likelihood model with the 'supervision' involving $\mathcal{R}$ sampled from the amortized posterior) is presented in Fig. 2. We illustrate them in detail in the sections below.

**E-Step: Amortized Inference with GFlowNets.** For inference in the high-dimension discrete latent space, we leverage the probabilistic framework of GFlowNets (Bengio et al., 2021; 2023). Consider a symbolic logic tree $\mathcal{R}$, we start from the root $\mathcal{R}_0 := \{z_0\}$, in which $z_0$ is the target predicate (can be composed by multiple language tokens). We follow *backward chaining* (Rocktäschel & Riedel, 2017) to form a symbolic proof tree in a top-down fashion by prompting LLMs. We grow the logic tree one level deeper at a time based on the previous paths. Concretely, suppose $\mathcal{R}_t$ can be represented by $m$ paths, *i.e.*, $\mathcal{R}_t := \{z_0^{(i)} z_1^{(i)} \cdots z_j^{(i)}\}_{i=1}^m$, where $z_j^{(i)} \in \mathcal{Z}$ is the $j$-th predicate presented in the $i$-th path from the predefined predicate space $\mathcal{Z}$. If the maximum number of nodes for each path to expand is constrained to $W$, the generative process from a symbolic logic tree $\mathcal{R}_t$ to $\mathcal{R}_{t+1}$ can be represented as:

$$\log q_{GFN}(\mathcal{R}_{t+1}|\mathcal{R}_t) := \sum_{i=1}^m \sum_{k=1}^{W+1} \log q_{LM}(z_{j+1}^{(i),k}|z_0^{(i)} \cdots z_j^{(i)}),$$
(9)

where $q_{LM}$ is the autoregressive sequence generation model and $z_{j+1}^{(i),k}$ is the next level predicates chosen from $\mathcal{Z}_W \cup \{\mathtt{T}\}$, $\mathcal{Z}_W \subseteq \mathcal{Z}$, $|\mathcal{Z}_W| = W$, and $\mathtt{T}$ denotes a stop symbol. The nodes in the symbolic logic tree thus grow in $O(W^n)$

and will not stop expanding until all the paths reach the termination state $\mathtt{T}$, i.e.,

$$\log q_{GFN}(\mathtt{T}|\mathcal{R}_t) := \sum_{i=1}^m \log q_{LM}(\mathtt{T}|z_0^{(i)} \cdots z_j^{(i)}).$$
(10)

The marginal likelihood of sampling a terminal logic tree $\mathcal{R}_t$ is given by

$$q_{GFN}(\mathcal{R}_t \to \mathtt{T}) = \int_{\tau=(\mathcal{R}_0 \rightsquigarrow \mathcal{R}_t)} \Pi_{i=1}^t q_{GFN}(\mathcal{R}_i|\mathcal{R}_{i-1})q_{GFN}(\mathtt{T}|\mathcal{R}_t)d\tau$$

over trajectories $\tau$ starting at $\mathcal{R}_0$ and ends at $\mathcal{R}_t$. Notably, the goal of GFlowNet training is to fit the parametric policy $q_{GFN}(\cdot|\cdot;\theta)$ such that its terminating probability $q_{GFN}(\mathcal{R}_t \to \mathtt{T})$ is proportional to a predefined reward $r$. In our case, GFlowNet's reward $r$ is defined as the posterior of logic trees $p(\mathcal{R}|X,Y)$, i.e., $r(\mathcal{R}|X,Y) \propto p_{LM}(Y|X,\mathcal{R})p_w(X|\mathcal{R})p_{LM}(\mathcal{R})$. By construction, GFlowNet's marginal terminating distribution is proportional to its reward function $r(\mathcal{R}|X,Y)$, thus we will have the final samples $\mathcal{R}$ given by the GFlowNet's policy $q_{GFN}(\cdot|\cdot,\theta)$ following the distribution of unnormalized true posterior of $p(\mathcal{R}|X,Y)$. Here, the given reward function $r$ can be decomposed as a product of likelihood terms that accumulated over steps of the sampling sequence. In this case, a forward-looking SubTB loss (Madan et al., 2023) for GFlowNet can help local credit assignment (Hu et al., 2023a;b). The SubTB learning objective for trajectory $\tau = (\mathcal{R}_0, \mathcal{R}_1, \cdots, \mathcal{R}_t)$ is:

$$\mathcal{L}_{SubTB}(\theta) = \qquad\qquad\qquad\qquad (11)$$
$$\sum_{0 \le i < j \le t} \left[ \log \frac{r(\mathcal{R}_i^{\mathtt{T}})\Pi_{k=i+1}^j q_\theta(\mathcal{R}_k|\mathcal{R}_{k-1})q_\theta(\mathtt{T}|\mathcal{R}_j)}{r(\mathcal{R}_j^{\mathtt{T}})q_\theta(\mathtt{T}|\mathcal{R}_i)} \right]^2,$$

where $q_\theta$ is the conditional GFlowNet policy initialized by a language model $p_{LM}$ conditioned on prefix $X$ and $Y$. The detailed derivation of the SubTB loss is given in Appendix F. In practice, this loss can be minimized by gradient descent on $\theta$ sampled either *on-policy* or *off-policy*, just as in reinforcement learning. To predict the event type $Y$ for an unseen event sequence $X$, one can draw samples of $\mathcal{R}$ from $q_\theta$ followed by sampling from $p_{LM}(Y|X,\mathcal{R})$.

**M-Step: Model Parameter Updating.** The marginal terminal distribution of GFlowNet is used as a variational approximation to the intractable posterior $p(\mathcal{R}|X,Y)$ to perform updates to the generative model's parameters. Thus, for the event demonstrations $X$ and next event type $Y$, we can uncover its underlying symbolic logic tree $\mathcal{R}$ from the policy of the conditional GFlowNet $q_{GFN}$ and perform in expectation gradient update on the parameters $w$ of event likelihood and tunable parameters $\phi$ for structure prior learning:

$$\mathcal{L}_{llh}(w,\phi) = \mathbb{E}_{\mathcal{R}\sim q_{GFN}(\mathcal{R}\to\mathtt{T})}[\log p_w(X|\mathcal{R})$$
$$+ \log p_{LM}(Y|X,\mathcal{R}) + \log p_\phi(\mathcal{R})]. \quad (12)$$

It should be noted that the evolving nature of the generative models $p_w$, $p_{LM}$, and $p_\phi$ during joint optimization leads to a dynamic reward system for the GFlowNets. The
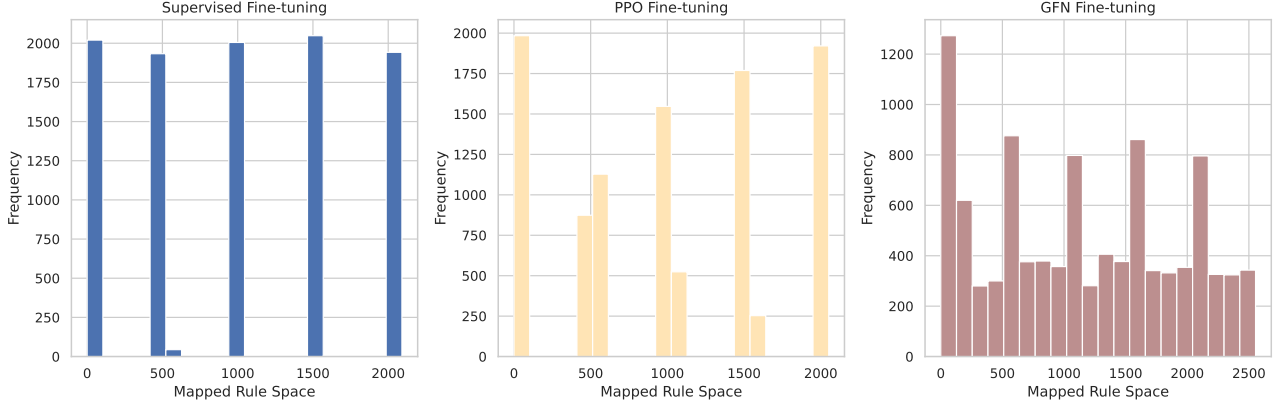
*Figure 3.* Empirical rule distributions sampled from the Language Model fine-tuned by three different approaches. We use 10,000 samples to depict the frequency distribution of a complete logic rules search space with support $|\mathcal{R}| = 2500$. The x-axis represents these logic rules in a nominal 1-D format, where each point corresponds to a specific rule. The ordering of these points is not indicative of any inherent sequence.

---

**Algorithm 1** Bayesian Logic Tree Learning for Events

---

**Input:** data pool $\{\mathcal{X}, \mathcal{Y}\}$, rule weights $w$, tunable parameters $\theta$ for LLM as the GFlowNet policy, tunable parameters $\phi$ for LLM as the prior policy, optimization and exploration hyperparameters, threshold $\alpha$
**repeat**
    sample batch data pair $(X, Y) \sim \{\mathcal{X}, \mathcal{Y}\}$
    sample $\tau \sim q_\theta(\tau|X, Y); \tau = (\mathcal{R}_0, \cdots, \mathcal{R}_T)$
    $r_t \leftarrow p_w(X|\mathcal{R}_t)p(Y|X, \mathcal{R}_t)p_\phi(\mathcal{R}_t), t = 0, \cdots, T$
    $\mathcal{L}_{SubTB} \leftarrow$ SubTB loss in Eq. (11) along $\tau$ with reward $r_t$
    E-step: GD on $\theta$ with $\nabla_\theta \mathcal{L}_{SubTB}$
    **if** $\mathcal{L} < \alpha$ **then**
        Sample $\tau \sim q_\theta(\tau|X, Y)$
        M-step: GD on $w$ and $\phi$ with $\nabla_{w,\phi} \mathcal{L}_{llh}$ in Eq. (12)
    **end if**
**until** some convergence criteria

---

training process involves alternating between E-steps and M-steps, with the frequency of GFlowNet updates between successive M-steps being a variable parameter that can be either predetermined or adaptively chosen. Following the approach outlined in (Hu et al., 2023b), adaptive E-steps are implemented through loss thresholding. This method uses a moving average of the GFlowNet's training loss as a measure of the accuracy in approximating the true posterior. An M-step gradient update is executed following a GFlowNet update only if this moving average drops below a set loss threshold. The overall algorithm is presented in Alg. 1.

### 3.3. Discussion

**Comparison with ILP systems.** In our approach, we harness LLMs to generate latent logic trees, replacing traditional symbolic ILP systems. While traditional ILP systems that are based on discrete space search excel in rule learning

from minimal examples, they are sensitive to noisy input, and a single error can lead to malfunction. Neural-symbolic rule induction systems, on the other hand, are more robust to noise but will struggle with few-shot learning and may face the risk of overfitting. Symbolic reasoning through LLMs integrates pretrained knowledge, addressing the challenge of learning rules from limited data. Additionally, our approach employs a step-by-step process reflecting human cognitive functions (Wei et al., 2022), enhancing semantic understanding from event sequences.

**Comparison with GFlowNets-CoT.** We share some similarities with GFlowNets-CoT (Hu et al., 2023a) in the Bayesian inference stage in which LLM is used as a probabilistic generative model that simultaneously generates logical rules and a set of core relations underlying them. However, the distinction between our approach and GFlowNets-CoT is that we extend GFlowNets fine-tuning to a sentence-level symbolic logic tree $\mathcal{R}$, which is similar to sentence-level Tree-of-Thoughts (Yao et al., 2023), through directly querying sentence probabilities within a confined 'sentence space' for backward learning, as opposed to ToT's forward-only inference approach. GFlowNets-CoT's likelihood model relies only on $p_{LM}(X, Y, \mathcal{R})$ while ours is built upon a *modular* likelihood model by decomposing $p(X, Y|\mathcal{R})$ into event likelihood $L_w$ along with the language likelihood $p_{LM}$, *i.e.,* $p_{LM}(\mathcal{R})p_w(X|\mathcal{R})p_{LM}(Y|X, \mathcal{R})$.

## 4. Experiments

### 4.1. Experimental Setup

**Datasets and Evaluation Setup.** Our study involves one synthetic and three real-world event sequence datasets, containing both semantic and non-semantic information. We view events in these datasets as predicates that can form a symbolic logic tree. For each sequence, we focus on

*Table 1.* Last event prediction performance on three real-world behavior datasets using both attention-based TPP models and Language Model *Opt-1.5B* as predictors. ER stands for Error Rate and MR stands for Mean Rank. The performance is averaged over three different seeds and the standard deviation is stored in the parenthesis. The best performance is in bold and also highlighted in gray.

| METHOD | DATASET METRICS | MIMIC-3 ER (%) ↓ | MR ↓ | EPIC-100 ER (%) ↓ | MR ↓ | STACKOVERFLOW ER (%) ↓ | MR ↓ |
|---|---|---|---|---|---|---|---|
| ATTNHP (YANG ET AL., 2021) | | 36.66(13.76) | 1.25(0.00) | 67.53(0.00) | 2.45(0.00) | **33.33**(0.00) | **1.95**(0.10) |
| PT-ATTNHP (XUE ET AL., 2023B) | | 77.50(0.00) | 1.75(0.00) | 68.33(1.44) | 2.27(0.16) | 71.11(32.71) | 3.40(0.81) |
| | $k = 0$ | 100.00(0.00) | 2.24(0.02) | 78.75(1.76) | 4.75(0.00) | 100.00(0.00) | 3.33(0.00) |
| $k$-SHOT COT | $k = 1$ | 100.00(0.00) | 2.23(0.00) | 76.25(1.57) | 4.66(0.02) | 100.00(0.00) | 3.33(0.00) |
| | $k = 3$ | 100.00(0.00) | 2.23(0.00) | 76.25(0.02) | 4.63(0.00) | 100.00(0.00) | 3.13(0.00) |
| TOT (DEPTH= 3, WIDTH= 3) | | 100.00(0.00) | 2.23(0.00) | 71.25(1.76) | 4.69(0.12) | 96.67(4.71) | 3.07(0.09) |
| SFT FINE-TUNING | | 82.50(2.50) | 2.14(0.03) | 75.83(1.44) | 4.38(0.11) | 93.33(6.67) | 4.44(0.30) |
| PPO FINE-TUNING | | 77.50(0.00) | 2.55(0.00) | 77.50(0.00) | 3.99(0.03) | 73.33(6.67) | 3.29(0.32) |
| GFN FINE-TUNING | | **27.50**(8.66) | **1.14**(0.05) | **55.25**(9.01) | **2.15**(0.48) | 33.45(5.12) | 2.23(0.23) |

predicting the final event. For the synthetic dataset, we create sequences of event predicates sampled from a pre-specified TL-PP using the thinning algorithm (Ogata, 1981). The functional form of the intensity is informed by the predefined logic rules. Regarding the real-world datasets, one is the **MIMIC-III** (Johnson et al., 2016), an electronic health record dataset from intensive care unit patients. We use various lab measurements and treatment approaches as event predicates. The other is **EPIC-KITCHENS-100** (EPIC-100) (Damen et al., 2021), which documents every-day kitchen activities from a first-person perspective over several days, with actions labeled. We analyze these labeled actions in sequence to predict the human's next action based on their past activities. The final one is **StackOverflow** (SO) (Leskovec & Krevl, 2014), which records a sequence of reward history with badges from the question-answering website *StackOverflow* to promote the engagement among its users. Each event in the sequence signifies the receipt of a particular metal. For all the datasets, We consider each sequence as a record pertaining to a single individual and partition each dataset into 80%, 10%, 10% train/dev/test splits by the total population. More details about these datasets can be found in the Appendix G.1.

**Metrics.** We follow the common next-event prediction task in TPPs (Du et al., 2016; Mei & Eisner, 2017) and emphasize the performance of last event type prediction $k$ from its history $\mathcal{H}$ output by the language model. We evaluate the prediction $\hat{k}$ by *Error Rate* (ER) and *Mean Rank* (MR) that measures the average rank of the ground-truth type in the list; a smaller MR means a higher rank, and thus a better result.

**Base models.** In this study, we utilize three distinct sizes of language models from the OPT family (Zhang et al., 2022): *Opt-125M* (small), *Opt-1.5B* (medium), and *Opt-6.7B* (large), as our foundational language model backbones for latent logic tree extraction $p_{LM}(\mathcal{R}|X)$ in the **E-steps**. These models are fine-tuned for logic tree learning using the

LoRA adaptation layer and further optimized through quantization (Dettmers et al., 2023) to minimize GPU memory consumption during both forward and backward processing stages. We use *Zephyr-3B* (Tunstall et al., 2023) and *Mistral-7B-Instruct* (Jiang et al., 2023) as frozen inference models for $p_{LM}(Y|X, \mathcal{R})$ in the **M-steps**. Detailed methodologies and specifics regarding the fine-tuning of these Large Language Models (LLMs) can be found in the Appendix G.4.

**Competitors.** In our study, we categorize competitors into three distinct types. The first category includes prompt-based approaches applied to language models, such as $k$-shot Chain-of-Thought (**CoT**) (Wei et al., 2022) and Tree-of-Thoughts (**ToT**) (Yao et al., 2023), which are utilized to generate reasoning chains and make prediction of the last event. The second category involves fine-tuning methods for language models, notably supervised fine-tuning (**SFT**) and Proximal Policy Optimization (**PPO**) (Schulman et al., 2017) fine-tuning. The final category consists of advanced neural Temporal Point Process (TPP) models specifically designed for event prediction. Within this group, we focus on **AttNHP** (Yang et al., 2021), an attention-based TPP whose performance is either on par with or superior to the Neural Hawkes Process (NHP) (Mei & Eisner, 2017) and other attention-based models (Xue et al., 2023a). Additionally, we consider PromptTPP (Xue et al., 2023b), a prompting model based on AttNHP (abbreviated as **Pt-AttNHP**), tailored for processing streaming events with a retrieval memory mechanism.

### 4.2. Results and Analysis

The primary findings are summarized in Table 1. Notably, only using the local LLM for event prediction $p_{LM}(Y|X)$ (0-shot CoT) yields the least effective results across all three datasets. Intriguingly, incorporating examples and adopting a tree-like reasoning structure (ToT) in the prompt do help enhance performance on the EPIC-100 and StackOverflow datasets to some extend. Furthermore, Supervised Fine-
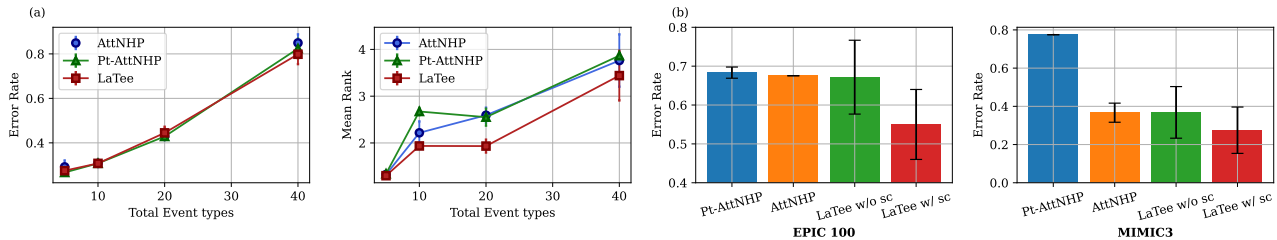
*Figure 4.* (a) Illustration of Scalability on the number of event types on four synthetic datasets. (b) Illustration of the performance of using semantic and not using semantic information on two real-world datasets.

*Table 2. Scalability of the proposed model on two real-world behavior datasets.* The tree traversal depth $d$ and tree expansion width $w$ is fixed to 3 without further clarification in the Table. We use *Opt-1.3B* for Epic-100 and Stackoverflow as base model in E-step. Error Rate (%) is used as the evaluation metric for both Epic-100 and StackOverflow. The performance is averaged over three different seeds and the standard deviation is stored in the parenthesis.

| METHOD | DATASET | EPIC-100 W/ SC | SO W/O SC |
|---|---|---|---|
| LATEE (*Tree Depth*) | $d = 2$ | 69.23(9.43) | 73.31(23.29) |
| | $d = 3$ | 69.40(9.21) | 34.50(5.23) |
| | $d = 4$ | 68.21(8.13) | 34.41(5.08) |
| LATEE (*Tree Width*) | $w = 3$ | 69.40(9.21) | 34.50(5.23) |
| | $w = 5$ | 61.31(9.24) | 33.53(5.10) |
| | $w = 7$ | 55.25(9.01) | 34.37(5.42) |
| LATEE (*Model Size*) | OPT-350M | 69.40(9.21) | 72.87(16.35) |
| | OPT-1.3B | 66.40(9.52) | 34.50(5.23) |
| | OPT-6.7B | 61.40(8.36) | 33.45(5.12) |

Tuning (SFT) exhibits similarly weak performance, while Proximal Policy Optimization (PPO) Fine-Tuning on the LLM shows marginal improvement but still lags behind attention-based Temporal Point Process (TPP) models. We hypothesize this underperformance is due to SFT's limited generalization capabilities and the shifted distribution mapping inherent in PPO's reward signals (refer to Analysis 2). It is noteworthy that Pt-AttNHP consistently falls short of AttNHP's performance across all datasets. This may be attributed to Pt-AttNHP's reliance on a prompt-like retrieval memory for time-horizon generalization, which potentially leads to overlooking individual-level characteristics in unseen event histories. Lastly, the proposed LaTee, fine-tuned with GFN objectives, matches AttNHP's accuracy on Stack-Overflow by focusing solely on latent structure learning. Remarkably, it surpasses AttNHP by a relative margin of 25% on MIMIC-3 and 18% on EPIC-100 containing semantic content (as detailed in Analysis 1).

**Analysis 1: The Role of LLM in Enhancing Event Logic Discovery through Semantic Cognition.** From the data presented in Fig. 1, it is evident that GPT enhances next events type prediction by substituting semantically meaningless numerical event IDs with meaningful event names. This study aims to explore whether the semantic content embedded in event history can bolster structure learning

and, consequently, improve event prediction accuracy on a local deployable LLM. As shown in Fig. 4(b), we observe a noteworthy reduction in error rate (approximately 25%) for both EPIC-100 and MIMIC-3 datasets when employing semantic event names for reasoning and inference. This decrease is significantly more pronounced compared to the improvement seen when transitioning from attention-based TPP models to LaTee models that do not apply semantic information. Moreover, we illustrate two examples of semantic tree structures learned by LaTee in Appendix A.

**Analysis 2: The Necessity of GFN Fine-Tuning in LLMs for Logic Tree Discovery and the Role of Prompts in Rule Discovery.** As indicated by the baselines in Table 1, approaches such as zero-shot Chain-of-Thought (CoT) prompting, $k$-shot prompting, and Tree-of-Thoughts (ToT) prompting demonstrate limited efficacy in yielding meaningful results. Similarly, Supervised Fine-Tuning (SFT) and Proximal Policy Optimization (PPO) fine-tuning on Large Language Models (LLMs) for next events prediction are outperformed by attention-based Temporal Point Process (TPP) models. However, GFN fine-tuning, which focuses on teaching models how to reason rather than predict, enables LLMs to match and even exceed the prediction accuracy of attention-based TPP models, particularly when integrating semantic information. To understand this improvement, Fig. 3 offers a visualization of the diverse rule distributions generated by fine-tuned LLMs. We notice that rule distributions in both SFT and PPO fine-tuning are predominantly concentrated in five regions, whereas GFN fine-tuning exhibits a more diverse spread across the entire rule space.

**Analysis 3: The Scalability of the Proposed Method and the Impact of LLM and Symbolic Logic Tree Sizes on Performance.** This analysis explores the scalability of our proposed method by examining the effect of an increased number of event types across four synthetic datasets without any semantic information. As shown in Fig. 4, LaTee demonstrates comparable scaling abilities in Error Rates and Mean Rank to those of attention-based TPP models. Notably, LaTee consistently achieves a lower Mean Rank, likely due to the additional confidence imparted by the learned structure information in making predictions. Additionally, we analyze the impact of varying tree sizes and

*Table 3.* Performance Evaluation for Alternate EM-loops Frequencies on Synthetic@5 (Earlystop was made at the fifth epoch).

| | NLL $\downarrow$ | ER (%) $\downarrow$ | MR $\downarrow$ |
|---|---|---|---|
| E-STEPS ONLY (WITH GROUNDTRUTH LIKELIHOOD) | 1389.31 | 62.5 | 2.025 |
| EM-LOOPS (ALTERNATE FREQ = 1) | 117.58 | 70.0 | 2.075 |
| EM-LOOPS (ALTERNATE FREQ = 20) | 108.44 | 67.5 | 2.000 |
| EM-LOOPS (ALTERNATE FREQ = 50) | 106.35 | 70.0 | 1.900 |

*Table 4.* Performance Evaluation for using different LMs for Inference (E-steps) and Generation (M-steps) on Synthetic@5 (Earlystop was made at the fifth epoch).

| E-STEPS LM (FINE-TUNED) | M-STEPS LM (FROZEN) | NLL $\downarrow$ | ER (%) $\downarrow$ | MR $\downarrow$ |
|---|---|---|---|---|
| OPT-1.3B | OPT-1.3B | 128.64 | 97.5 | 2.23 |
| OPT-1.3B | ZEPHYR-3B | 149.25 | 87.5 | 2.50 |
| OPT-1.3B | MISTRAL-7B-INSTRUCT | 117.62 | 70.0 | 2.08 |
| ZEPHYR-3B | MISTRAL-7B-INSTRUCT | 116.21 | 70.0 | 1.95 |

LLM sizes. Assuming the predefined predicate space $\mathcal{Z}$ has a cardinality $|\mathcal{Z}| = N$, the maximum allowable depth and width of the logic tree are restricted to $d$ and $w$ ($w << N$), respectively, then the entirety of the search space can be approximated as $O(N^{w^d})$. In Table 2, we restrict depth $d$ and width $w$ below 4 and 7 and the empirical findings suggest that increasing the tree widths has a more beneficial effect than increasing tree depth or model size on semantic event sequences. This could be attributed to the fact that ground-truth rules often consist of multiple short rules, and a wider tree is better equipped to encompass more semantically similar predicate events at the same level. It's also important to note that for non-semantic event sequences, enlarging the model size tends to be more advantageous than increasing tree sizes.

**Analysis 4: Ablating E-M Update Steps in LaTee.** Unlike traditional EM algorithms where the E-step typically has a closed-form solution, E-step in GFlowNet-EM progressively moves closer to the target distribution $p$. This requires sufficient gradient steps in the 'approximate E-step' to closely align the approximate distribution with the target while it also should regularly switch to M-steps for updating likelihood functions using the new sampled latent variables in E-steps. This non-stationary update thus gives us a challenge of scheduling E-M steps for a better convergence rate.

Consequently, we added experiments comparing the convergence speed of both SubTB loss (E-steps) and NLL loss (M-steps) under varying frequencies of alternation. We provide the plot of convergence analysis for EM in the Appendix G.3 Fig. 8 and 9 and report final performance in Table 3. Interestingly, we observe that more frequent alternations of E-M loops lead to a faster convergence of the SubTB loss (E-steps) but a slower rate for M-step. Additionally, the frequency of alternation appears to have minimal impact on the overall evaluation performance.

**Analysis 5: Ablating LLMs for E-M Steps.** To investigate whether the world knowledge in the LM is most useful in the generation model (M-steps LM), the inference model (E-steps LM), or both, we compared the effects of using different sizes/versions of LMs for inference (E-steps) and generation (M-steps). In our experiment, we used Opt-1.3B as the base inference model (which has a minor language understanding ability on LM benchmark task), and used three different estimation (generation) models to make the event prediction, i.e., Opt-1.3B, Zephyr-3B, Mistral-7B-Instruct. The results are shown in Table 4.

Our evaluation strategy in Table 4 focused exclusively on altering the model size to guarantee fairness in comparison. We observe that employing larger language models (LMs) for both inference (E-steps) and generation (M-steps) phases can enhance event prediction performance. Notably, an increase in the size of the LM used for generation (M-steps) exhibited a more pronounced positive impact compared to enlarging the LM for inference (E-steps). The results suggest that the extensive world knowledge encoded in larger LMs is more beneficial for generation tasks (M-steps). This finding encourages future improvements in reasoning abilities in the M-steps by calling API-based LLMs like GPT-4 and Claude-3 with an extracted logic tree from a fine-tuned local lightweight LLMs as the prompt.

## 5. Conclusion

The incorporation of general knowledge from Large Language Models (LLMs) is key to deciphering complex structures in noisy event sequences. To facilitate this, we present LaTee, an amortized EM-style framework that leverages LLMs' prior knowledge for latent tree structure learning for event sequence explanation. We simplify the complex posterior with GFlowNets and perform inference based on the learned structure without further gradient updates. Empirical results show that this method notably enhances generalization in event histories with semantic information.

## Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

## Acknowledgement

## References

Acemoglu, D., Dahleh, M. A., Lobel, I., and Ozdaglar, A. Bayesian learning in social networks. The Review of Economic Studies, 78(4):1201–1236, 2011.

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. arXiv preprint arXiv:2303.08774, 2023.

Beal, M. J. Variational algorithms for approximate Bayesian inference. University of London, University College London (United Kingdom), 2003.

Bengio, E., Jain, M., Korablyov, M., Precup, D., and Bengio, Y. Flow network based generative models for non-iterative diverse candidate generation. Advances in Neural Information Processing Systems, 34:27381–27394, 2021.

Bengio, Y., Lahlou, S., Deleu, T., Hu, E. J., Tiwari, M., and Bengio, E. Gflownet foundations. Journal of Machine Learning Research, 24(210):1–55, 2023.

Boyd, A., Bamler, R., Mandt, S., and Smyth, P. User-dependent neural sequence models for continuous-time event data. Advances in Neural Information Processing Systems, 33:21488–21499, 2020.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. Advances in neural information processing systems, 33:1877–1901, 2020.

Campero, A., Pareja, A., Klinger, T., Tenenbaum, J., and Riedel, S. Logical rule induction and theory learning using neural theorem proving. arXiv preprint arXiv:1809.02193, 2018.

Chen, R. T., Amos, B., and Nickel, M. Neural spatio-temporal point processes. arXiv preprint arXiv:2011.04583, 2020.

Cropper, A. and Tourret, S. Logical reduction of metarules. Machine Learning, 109:1323–1369, 2020.

Damen, D., Doughty, H., Farinella, G. M., Fidler, S., Furnari, A., Kazakos, E., Moltisanti, D., Munro, J., Perrett, T., Price, W., and Wray, M. The epic-kitchens dataset: Collection, challenges and baselines. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 43(11):4125–4141, 2021. doi: 10.1109/TPAMI.2020. 2991965.

Deleu, T., Nouri, P., Malkin, N., Precup, D., and Bengio, Y. Discrete probabilistic inference as control in multi-path environments. arXiv preprint arXiv:2402.10309, 2024.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. Journal of the royal statistical society: series B (methodological), 39(1):1–22, 1977.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. Qlora: Efficient finetuning of quantized llms. arXiv preprint arXiv:2305.14314, 2023.

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.

Evans, R. and Grefenstette, E. Learning explanatory rules from noisy data. Journal of Artificial Intelligence Research, 61:1–64, 2018.

Feng, X., Wan, Z., Wen, M., Wen, Y., Zhang, W., and Wang, J. Alphazero-like tree-search can guide large language model decoding and training. arXiv preprint arXiv:2309.17179, 2023.

Gao, C., Lan, X., Lu, Z., Mao, J., Piao, J., Wang, H., Jin, D., and Li, Y. S3: Social-network simulation system with large language model-empowered agents. arXiv preprint arXiv:2307.14984, 2023.

Glanois, C., Jiang, Z., Feng, X., Weng, P., Zimmer, M., Li, D., Liu, W., and Hao, J. Neuro-symbolic hierarchical rule induction. In International Conference on Machine Learning, pp. 7583–7615. PMLR, 2022.

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. Reinforcement learning with deep energy-based policies. In International conference on machine learning, pp. 1352–1361. PMLR, 2017.

Hao, S., Gu, Y., Ma, H., Hong, J. J., Wang, Z., Wang, D. Z., and Hu, Z. Reasoning with language model is planning with world model. arXiv preprint arXiv:2305.14992, 2023.

Hegselmann, S., Buendia, A., Lang, H., Agrawal, M., Jiang, X., and Sontag, D. Tabllm: Few-shot classification of tabular data with large language models. In International Conference on Artificial Intelligence and Statistics, pp. 5549–5581. PMLR, 2023.

Henrich, J. and McElreath, R. The evolution of cultural evolution. Evolutionary Anthropology: Issues, News, and Reviews: Issues, News, and Reviews, 12(3):123–135, 2003.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685, 2021.

Hu, E. J., Jain, M., Elmoznino, E., Kaddar, Y., Lajoie, G., Bengio, Y., and Malkin, N. Amortizing intractable inference in large language models. arXiv preprint arXiv:2310.04363, 2023a.

Hu, E. J., Malkin, N., Jain, M., Everett, K. E., Graikos, A., and Bengio, Y. Gflownet-em for learning compositional latent variable models. In International Conference on Machine Learning, pp. 13528–13549. PMLR, 2023b.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al. Mistral 7b. arXiv preprint arXiv:2310.06825, 2023.

Johnson, A. E., Pollard, T. J., Shen, L., Lehman, L.-w. H., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Anthony Celi, L., and Mark, R. G. Mimic-iii, a freely accessible critical care database. Scientific data, 3(1):1–9, 2016.

Katz, Y., Goodman, N. D., Kersting, K., Kemp, C., and Tenenbaum, J. B. Modeling semantic cognition as logical dimensionality reduction. In Proceedings of the annual meeting of the cognitive science society, volume 30, 2008.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. Advances in neural information processing systems, 35: 22199–22213, 2022.

Koller, D. and Friedman, N. Probabilistic graphical models: principles and techniques. MIT press, 2009.

Laland, K. N. Social learning strategies. Animal Learning & Behavior, 32:4–14, 2004.

Leng, Y. and Yuan, Y. Do llm agents exhibit social behavior? arXiv preprint arXiv:2312.15198, 2023.

Leskovec, J. and Krevl, A. Snap datasets: Stanford large network dataset collection, 2014.

Lew, A. K., Zhi-Xuan, T., Grand, G., and Mansinghka, V. K. Sequential monte carlo steering of large language models using probabilistic programs. arXiv preprint arXiv:2306.03081, 2023.

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, X. L., Holtzman, A., Fried, D., Liang, P., Eisner, J., Hashimoto, T., Zettlemoyer, L., and Lewis, M. Contrastive decoding: Open-ended text generation as optimization. arXiv preprint arXiv:2210.15097, 2022.

Lu, X., Welleck, S., West, P., Jiang, L., Kasai, J., Khashabi, D., Bras, R. L., Qin, L., Yu, Y., Zellers, R., et al. Neurologic a* esque decoding: Constrained text generation with lookahead heuristics. arXiv preprint arXiv:2112.08726, 2021.

Madan, K., Rector-Brooks, J., Korablyov, M., Bengio, E., Jain, M., Nica, A. C., Bosc, T., Bengio, Y., and Malkin, N. Learning gflownets from partial episodes for improved convergence and stability. In International Conference on Machine Learning, pp. 23467–23483. PMLR, 2023.

Malkin, N., Wang, Z., and Jojic, N. Coherence boosting: When your pretrained language model is not paying enough attention. arXiv preprint arXiv:2110.08294, 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.

Miao, N., Zhou, H., Mou, L., Yan, R., and Li, L. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pp. 6834–6842, 2019.

Nachum, O., Norouzi, M., Xu, K., and Schuurmans, D. Bridging the gap between value and policy based reinforcement learning. Advances in neural information processing systems, 30, 2017.

Ogata, Y. On lewis' simulation method for point processes. IEEE transactions on information theory, 27(1):23–31, 1981.

Pan, L., Albalak, A., Wang, X., and Wang, W. Y. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. arXiv preprint arXiv:2305.12295, 2023.

Quinlan, J. R. Learning logical definitions from relations. Machine learning, 5:239–266, 1990.

Rocktäschel, T. and Riedel, S. End-to-end differentiable proving. Advances in neural information processing systems, 30, 2017.

Schick, T. and Schütze, H. It's not just size that matters: Small language models are also few-shot learners. arXiv preprint arXiv:2009.07118, 2020.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017.

Sha, L. Gradient-guided unsupervised lexically constrained text generation. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 8692–8703, 2020.

Shchur, O., Biloš, M., and Günnemann, S. Intensity-free learning of temporal point processes. arXiv preprint arXiv:1909.12127, 2019.

Shi, X., Xue, S., Wang, K., Zhou, F., Zhang, J. Y., Zhou, J., Tan, C., and Mei, H. Language models can improve event prediction by few-shot abductive reasoning. arXiv preprint arXiv:2305.16646, 2023.

Tiapkin, D., Morozov, N., Naumov, A., and Vetrov, D. P. Generative flow networks as entropy-regularized rl. In International Conference on Artificial Intelligence and Statistics, pp. 4213–4221. PMLR, 2024.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288, 2023.

Tunstall, L., Beeching, E., Lambert, N., Rajani, N., Rasul, K., Belkada, Y., Huang, S., von Werra, L., Fourrier, C., Habib, N., et al. Zephyr: Direct distillation of lm alignment. arXiv preprint arXiv:2310.16944, 2023.

Ullman, T. D., Goodman, N. D., and Tenenbaum, J. B. Theory learning as stochastic search in the language of thought. Cognitive Development, 27(4):455–480, 2012.

van Krieken, E., Thanapalasingam, T., Tomczak, J., Van Harmelen, F., and Ten Teije, A. A-nesi: A scalable approximate method for probabilistic neurosymbolic inference. Advances in Neural Information Processing Systems, 36:24586–24609, 2023.

Wang, X., Zhu, W., Saxon, M., Steyvers, M., and Wang, W. Y. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. Advances in Neural Information Processing Systems, 36, 2024.

Webb, T., Holyoak, K. J., and Lu, H. Emergent analogical reasoning in large language models. Nature Human Behaviour, 7(9):1526–1541, 2023.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. Advances in Neural Information Processing Systems, 35: 24824–24837, 2022.

Xiao, S., Yan, J., Yang, X., Zha, H., and Chu, S. Modeling the intensity function of point process via recurrent neural networks. In Proceedings of the AAAI conference on artificial intelligence, volume 31, 2017.

Xie, S. M., Raghunathan, A., Liang, P., and Ma, T. An explanation of in-context learning as implicit bayesian inference. arXiv preprint arXiv:2111.02080, 2021.

Xie, Y., Kawaguchi, K., Zhao, Y., Zhao, X., Kan, M.-Y., He, J., and Xie, Q. Decomposition enhances reasoning via self-evaluation guided decoding. arXiv preprint arXiv:2305.00633, 2023.

Xu, H. No train still gain. unleash mathematical reasoning of large language models with monte carlo tree search guided by energy function. arXiv preprint arXiv:2309.03224, 2023.

Xue, S., Shi, X., Chu, Z., Wang, Y., Zhou, F., Hao, H., Jiang, C., Pan, C., Xu, Y., Zhang, J. Y., et al. Easytpp: Towards open benchmarking the temporal point processes. arXiv preprint arXiv:2307.08097, 2023a.

Xue, S., Wang, Y., Chu, Z., Shi, X., Jiang, C., Hao, H., Jiang, G., Feng, X., Zhang, J. Y., and Zhou, J. Prompt-augmented temporal point process for streaming event sequence. arXiv preprint arXiv:2310.04993, 2023b.

Yang, C., Mei, H., and Eisner, J. Transformer embeddings of irregularly spaced events and their participants. arXiv preprint arXiv:2201.00044, 2021.

Yang, F., Yang, Z., and Cohen, W. W. Differentiable learning of logical rules for knowledge base completion. arXiv preprint arXiv:1702.08367, 2017.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. Tree of thoughts: Deliberate problem solving with large language models. arXiv preprint arXiv:2305.10601, 2023.

Zhang, M., Jiang, N., Li, L., and Xue, Y. Language generation via combinatorial constraint satisfaction: A tree search enhanced monte-carlo approach. arXiv preprint arXiv:2011.12334, 2020.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., et al. Opt: Open pre-trained transformer language models. arXiv preprint arXiv:2205.01068, 2022.

Zhu, B., Sharma, H., Frujeri, F. V., Dong, S., Zhu, C., Jordan, M. I., and Jiao, J. Fine-tuning language models with advantage-induced policy alignment. arXiv preprint arXiv:2306.02231, 2023.

Zhu, X., Wang, J., Zhang, L., Zhang, Y., Gan, R., Zhang, J., and Yang, Y. Solving math word problem via cooperative reasoning induced language models. arXiv preprint arXiv:2210.16257, 2022.

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.
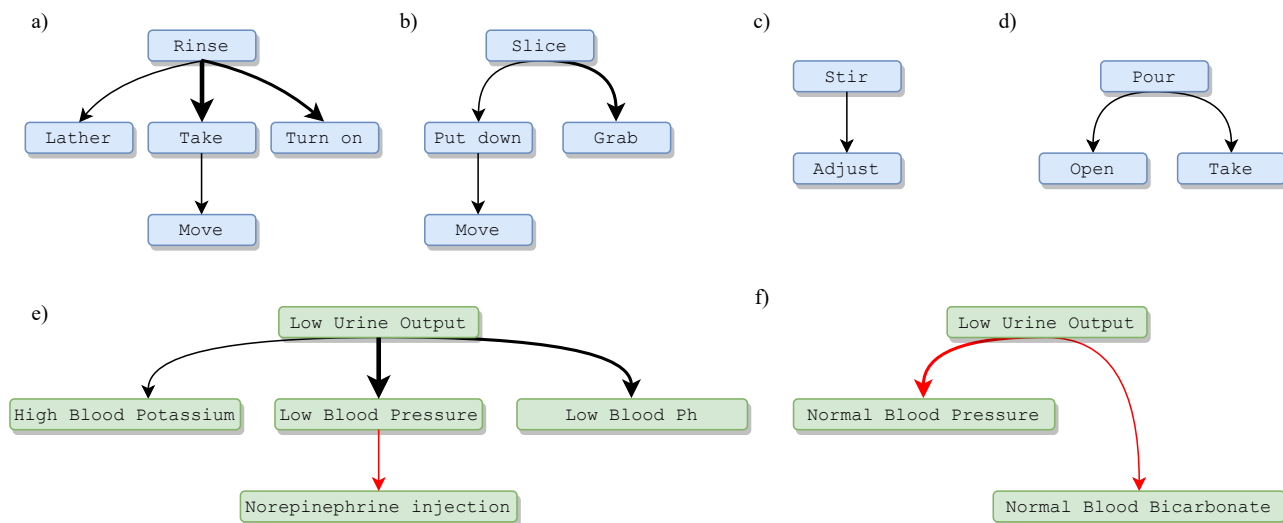
## A. Learned Logic Tree Examples



*Figure 5.* *Illustration of learned symbolic logic tree structures from event histories on two real-world datasets containing semantic information.* Fig. (a)-(d) are learned from EPIC-100 and Fig. (e)-(f) are learned from MIMIC-3. We use the thickness of the edges to represent posterior probability and the color to represent the weights corresponding to each rule (Black color stands for activation and Red color stands for inhibition, *i.e.*, low blood pressure will facilitate low urine while normal blood pressure suppresses low urine).

## B. Broader Impact

**Differentiable Extraction of Non-Linear Structures from LLMs**: Our approach extends the application of in-context learning in large language models (LLMs) beyond traditional posterior inference (Xie et al., 2021) and independent demonstrations (Wang et al., 2024). We focus on non-linear prompt structures, enabling the extraction of complex entities like symbolic proof trees and latent positions of political figures. This differentiable method enhances the versatility of LLMs in handling diverse, non-linear structures.

**Advancing Neuro-Symbolic Inference with Foundation Models**: Foundational models, including Vision-and-Language Models (VLMs) and LLMs, serve as informative belief priors for world modeling and latent concept understanding. Our work augments posterior inference capabilities, moving past models like A-NESI (van Krieken et al., 2023) that rely on uninformative Dirichlet priors. This progression is pivotal for tackling more intricate, multimodal, and scalable neurosymbolic challenges.

**Enhancing Data Privacy in Event Sequence Explanation and Prediction**: By fine-tuning locally accessible, lightweight LLMs (under 7B parameters) while maintaining data privacy, our model offers wide applications in sensitive areas like healthcare and credit card fraud detection. The logic trees extracted from local LLMs can be integrated with public LLMs for prediction tasks. This aspect also paves the way for exploring improvements in reasoning abilities for API-based LLMs like GPT-4 and Claude-3 using these extracted logic trees.

These facets of our research not only contribute to the evolution of language model applications but also pave the way for new advancements in privacy-sensitive areas and neurosymbolic computing.

## C. More Related Works

**Temporal Point Processes.** In recent decades, a diverse range of Neural Temporal Point Processes (TPPs) have been proposed to model event sequences with various properties. Many of these TPPs are based on a parametric intensity function that evolves through a series of latent states (Du et al., 2016; Xiao et al., 2017; Boyd et al., 2020; Chen et al., 2020). To effectively capture long-range dependencies within these sequences, the attention mechanism has been adapted for TPPs (Zuo et al., 2020; Yang et al., 2021; Mei & Eisner, 2017). Moreover, intensity-free TPP models have also shown promising results, particularly in the EasyTPP framework (Shchur et al., 2019; Xue et al., 2023a). However, the application of Large Language Models (LLMs) in learning event sequences remains largely unexplored. Recent research, such as LAMP (Shi et al., 2023), introduces a GPT-based abductive reasoning approach built upon attention-based TPP models for

event prediction. This approach, however, necessitates additional textual data for event description and relies on costly API services. Our focus, instead, is on harnessing the reasoning capabilities of local LLMs for event prediction.

**Non-Linear Reasoning in LLMs.** Recent research has focused on exploring complex, non-linear reasoning paths such as tree structures within Large Language Models (LLMs) (Zhu et al., 2022; Xu, 2023; Yao et al., 2023; Hao et al., 2023; Xie et al., 2023). Various methods, including beam search (Xie et al., 2023), depth-/breadth-first search (Yao et al., 2023), Monte Carlo Tree Search (Hao et al., 2023), and MCTS with an enhanced value function (Feng et al., 2023), have been implemented to navigate these tree structures effectively using LLMs' self-assessment capabilities to identify more effective reasoning pathways. Nonetheless, research on differentiable learning for non-linear reasoning within LLMs remains scarce. Recent studies, such as by Hu et al. (2023a), suggest fine-tuning LLMs using GFlowNets objectives to augment the diversity of reasoning chains. In our research, applying LLM-based tree search to discern the inherent structure in event sequences presents challenges due to the limited event data available for fine-tuning LLMs for specific event prediction tasks. Therefore, our focus shifts towards the development of differentiable logic trees to facilitate non-linear reasoning in LLMs, achieved by iteratively expanding and refining the logic tree structure.

## D. Limitations

Resource constraints limited our experiments to models with up to 6.7B parameters and event sequences of a maximum of 40 events. This limited the capacity of input events because of the constraints on the maximum number of input tokens for a language model. However, we anticipate our findings to be applicable to larger models and longer sequences. Notably, optimizing larger models with limited data presents challenges, and exploring more complex latent problems is an ongoing challenge.

## E. ELBO Derivation

Given data pair $(X, Y)$, we can represent write the joint likelihood $\log p(X, Y)$ as

$$\log p(X, Y) = \log \frac{p(X, Y, \mathcal{R})}{p(\mathcal{R}|X, Y)} \tag{13}$$

$$= \log \frac{p(X, Y, \mathcal{R})q(\mathcal{R}|X, Y)}{p(\mathcal{R}|X, Y)q(\mathcal{R}|X, Y)} \tag{14}$$

$$\sum_R q(\mathcal{R}|X, Y) \log p(X, Y) = \sum_R q(\mathcal{R}|X, Y) \log \frac{p(X, Y, \mathcal{R})q(\mathcal{R}|X, Y)}{p(\mathcal{R}|X, Y)q(\mathcal{R}|X, Y)} \tag{15}$$

$$\log p(X, Y) = \sum_R q(\mathcal{R}|X, Y) \log \frac{q(\mathcal{R}|X, Y)}{p(\mathcal{R}|X, Y)} + \sum_R q(\mathcal{R}|X, Y) \log \frac{p(X, Y, \mathcal{R})}{q(\mathcal{R}|X, Y)} \tag{16}$$

$$= D_{\text{KL}}(q||p) + \underbrace{\mathbb{E}_{\mathcal{R} \sim q(\mathcal{R}|X, Y)}[\log \frac{p(X, Y|\mathcal{R})p(\mathcal{R})}{q(\mathcal{R}|X, Y)}]}_{\text{ELBO}} \tag{17}$$

$$\geq \mathbb{E}_{\mathcal{R} \sim q(\mathcal{R}|X, Y)}[\log \frac{p(X, Y|\mathcal{R})p(\mathcal{R})}{q(\mathcal{R}|X, Y)}] \tag{18}$$

Thus, the ELBO $\mathcal{L}$ for the joint likelihood of $p(X, Y)$ is $\mathbb{E}_{\mathcal{R} \sim q(\mathcal{R}|X, Y)}[\log \frac{p(X, Y|\mathcal{R})p(\mathcal{R})}{q(\mathcal{R}|X, Y)}]$.

## F. GFlowNets Learning Objective

We learn the amortized sampler of posterior distribution $p(\mathcal{R}|X, Y)$ by a Sub-Trajectory Balance Objective (Madan et al., 2023) of GFlowNet. The original Sub Trajectory objective is given by:

$$\mathcal{L}_{SubTB}(\tau_{m:n}) = \left( \log \frac{F(s_m; \theta)\Pi_{i=m}^{n-1}p_F(s_{i+1}|s_i; \theta)}{F(s_n; \theta)\Pi_{i=m}^{n-1}p_B(s_i|s_{i+1;\theta})} \right)^2 \tag{19}$$

$$\mathcal{L}(\tau) = \frac{\sum_{0 \leq i < j \leq n} \lambda^{j-i} L_{SubTB}(\tau_{i:j})}{\sum_{0 \leq i < j \leq n} \lambda^{j-i}} \tag{20}$$

In our case, we enforce $F(s_n; \theta) = R(s_n)$ if $s_n$ is terminal, so we have $R(s_n^\top) = F(s_n)p_F(\top|s_n)$. Since we are generating

a tree structure level by level, thus the backward probability is one, *i.e.,* $p_B(s|s') = 1$, and $\lambda = 1$, we have

$$\mathcal{L}_{SubTB}(\mathcal{R}_{0:n}) = \sum_{0 \leq i < j \leq n} \left( \log \frac{F(\mathcal{R}_i; \theta) \Pi_{k=i+1}^j p_F(\mathcal{R}_k|\mathcal{R}_{k-1})}{F(\mathcal{R}_j; \theta) \Pi_{k=i+1}^j p_F(\mathcal{R}_{k-1}|\mathcal{R}_k)} \right)^2 \tag{21}$$

$$= \sum_{0 \leq i < j \leq n} \left( \log \frac{R(\mathcal{R}_i^{\mathsf{T}}) \Pi_{k=i+1}^j q_\theta(\mathcal{R}_k|\mathcal{R}_{k-1}) q_\theta(\mathsf{T}|\mathcal{R}_j)}{R(\mathcal{R}_j^{\mathsf{T}}) q_\theta(\mathsf{T}|\mathcal{R}_i)} \right)^2, \tag{22}$$

We train the GFlowNet with stochastic gradient

$$\mathbb{E}_{\mathcal{R}_{0:n} \sim q_\theta} [\nabla_\theta \mathcal{L}_{SubTB}(\mathcal{R}_{0:n})] \tag{23}$$

## G. Experimental Details

### G.1. Dataset Details

| | # Target Predicates | # Body Predicates | Events Average Length |
|---|---|---|---|
| Synthetic@5 (w/o sc) | 2 | 3 | 30.19 |
| Synthetic@10 (w/o sc) | 5 | 5 | 30.34 |
| Synthetic@20 (w/o sc) | 7 | 13 | 30.29 |
| Synthetic@40 (w/o sc) | 8 | 32 | 30.82 |
| StackOverflow (w/o sc) | 10 | 22 | 40.00 |
| EPIC-KITCHEN-100 (w/ sc) | 7 | 60 | 36.76 |
| MIMIC3 (w/ sc) | 3 | 62 | 20.01 |

*Table 5.* Event Dataset Statistics

We evaluate our methods on one synthetic dataset and three user behavior datasets. We consider each event type presented in the event history as a unique predicate and emphasize on the model's ability to predict only pertinent target predicates. The overall data statistics is presented in Table 5. We provide details on the preparation and utilization of each below.

**Synthetic Dataset.** This dataset comprises four sets of synthetic event history data generated using the Temporal Logic Point Process (Li et al., 2020). Specifically, we employ pre-defined logical rules along with their weights, as outlined in Eq. (4), to construct the intensity function, and then apply thinning algorithms to generate new events. To evaluate the scalability of the proposed model, we have created four distinct groups of synthetic data, with the number of event types varying from five to forty, and an average sequence length of 30 events.

**StackOverflow** (Leskovec & Krevl, 2014). This dataset encompasses two years of user awards from a question-and-answer website, documenting each user's sequence of badges. There are 22 distinct types of badges in total. However, since each event type is represented solely by a numerical ID, the dataset lacks semantically meaningful information. We focus on a subset of 142 records, each with an average sequence length of 40 event tokens.

**EPIC-KITCHEN-100** (Damen et al., 2021). This dataset originates from a large-scale, first-person (egocentric) vision dataset, featuring multi-faceted, audio-visual, non-scripted recordings in natural settings, specifically the wearers' homes. It captures daily kitchen activities over multiple days. We have utilized the annotated action sequences, focusing only on text, and extracted them to create a temporal event history of cooking verbs. This was achieved by omitting the entities that the human subjects interacted with. The frequencies of each verb, derived from the Epic-100 dataset, are visualized in Fig. 6. In this dataset, we specifically focus on eight verbs: `put-in`, `rinse`, `put-on`, `pour`, `stir`, `peel`, `chop`, and `slice`, as our target predicates. The model is tasked with reasoning about the actions preceding each target verb and learning the underlying structure that culminates in these targets. We concentrated on a subset of 400 event histories, each with an average sequence length of 36.76 events, resulting in 60 distinct event types in total.

**MIMIC-3** (Johnson et al., 2016). This dataset comprises electronic health records of patients admitted to the intensive care unit (ICU). We specifically focus on patients diagnosed with sepsis, extracting medications, lab tests, outputs, and diagnoses to form text-based temporal event histories. The frequencies of the various event types related to sepsis are illustrated in 7. In this dataset, we concentrate on three key event types: `survival`, `urine_output_low`, and `normal_blood_pressure`. Our analysis is based on a subset of 477 event histories, each with an average sequence length of 20 event tokens, resulting in a total of 62 unique event types.
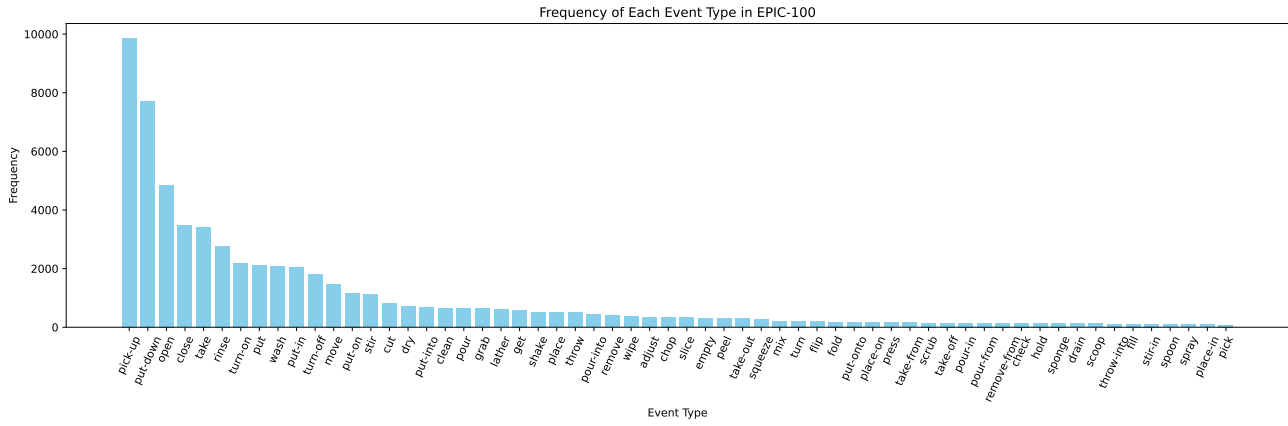
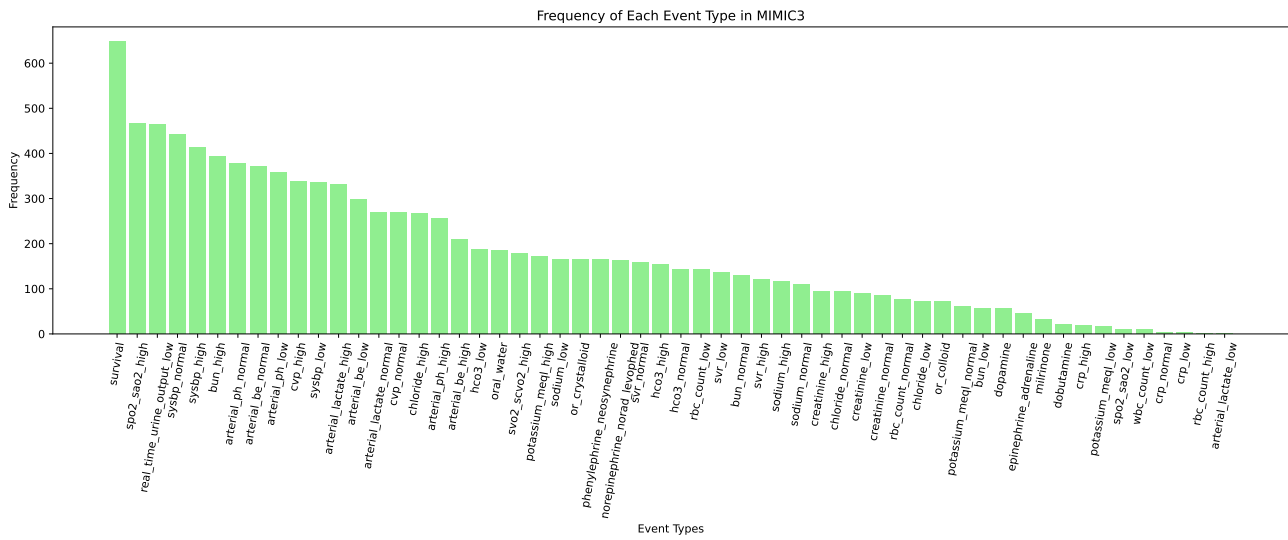*Figure 6.* Distribution of Semantic Event types in **EPIC-100**



*Figure 7.* Distribution of Semantic Event types in **MIMIC-3**

### G.2. Praising LLM Outputs

We detail two distinct methodologies for parsing outputs from large language models (LLMs) and querying corresponding probabilities.

1. For LLMs that are locally accessible, such as OPT series, our approach aligns with that of (Hu et al., 2023a). Here, we directly query the probability of subsequent tokens given a target sentence, avoiding the need for parsing. To illustrate, consider an 'action space' defined by $\{A, B, C, D\}$. To get the probability of action $A$, we tokenize it into k tokens $[w_1^A, w_2^A, ..., w_k^A]$. Then $p(A)$ is computed as:

$$p(A) = p(w_k^A|w_{k-1}^A, ..., w_1^A)p(w_{k-1}^A|w_{k-2}^A, ..., w_1^A)...p(w_1^A).$$

2. For LLMs that are not locally accessible, like GPT, we employ a different technique. Regular Expressions are used to isolate target sentences. Then, we leverage the 'logprobs' parameter in the OpenAI Chat Completions API to ascertain the probabilities of target tokens. For example, a common pattern in our analysis is '#Event NAME#', which allows us to capture the output event by extracting the NAME component. In cases where the NAME fails to be parsed, we adopt the approach from logic-LM (Pan et al., 2023), making a random guess across all possible event types.
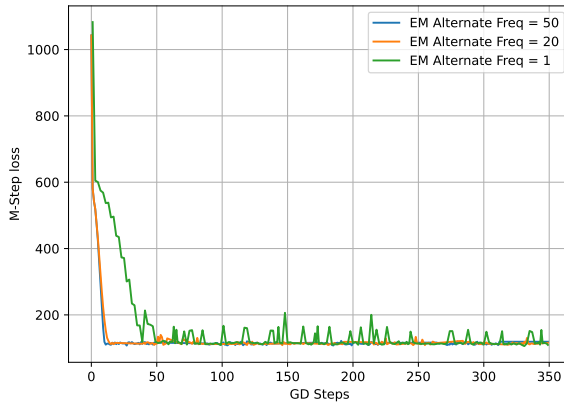
### G.3. Additional Experiments



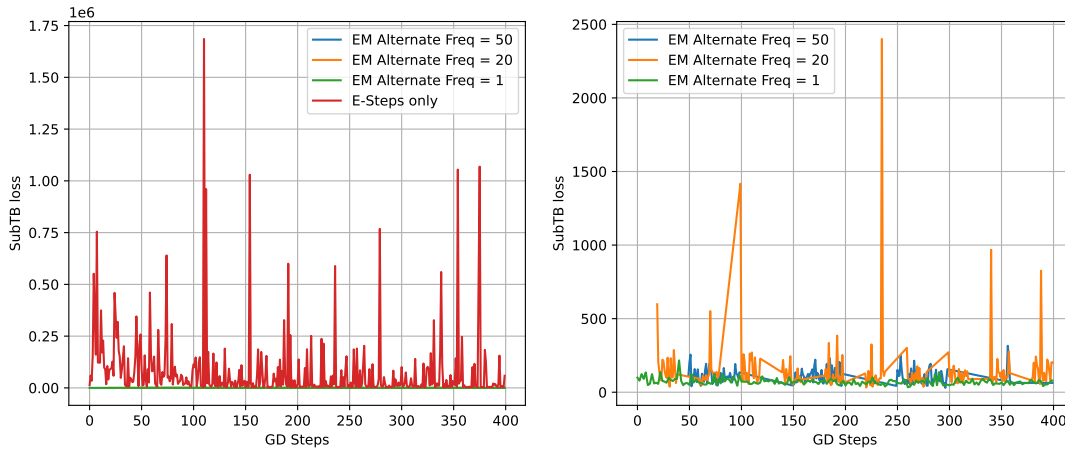*Figure 8.* M-Steps converges rate under different EM-loops alternating frequencies



*Figure 9.* E-Steps converge rate under different EM-loops alternating frequencies. **Left**: Four different alternating frequencies including E-steps only. **Right**: Excluding E-steps only from Left.

### G.4. Training Details

**Implementation Details.** All models are implemented using the PyTorch framework. All the experiments were conducted on a server with 512G RAM, two 64 logical cores CPUS (AMD Ryzen Threadripper PRO 5995WX 64-Cores), and four NVIDIA RTX A6000 GPUs with 50G memory.

**Hyperparameters Selection.** We present the selected hyperparameters on synthetic datasets and three real-world datasets in Table 6 and Table 7 respectively.

**Fine-tuning Quantized Large Language Model.**

In our experiment, we implement QLoRA (Dettmers et al., 2023) to fine-tune the Language Model, effectively reducing memory requirements during LLM finetuning without compromising on performance, as compared to the conventional 16-bit model finetuning process. Specifically, QLoRA employs 4-bit quantization to condense a pre-existing language model. This model's parameters are then set as unchangeable, and a limited set of modifiable parameters are incorporated via Low-Rank Adapters. During the finetuning phase, QLoRA directs gradient updates through these unmodifiable 4-bit quantized pre-trained language model parameters to the Low-Rank Adapters. Only the LoRA layers are adjusted during the training process.

**Prompt Details.** We present the prompts utilized for reasoning, denoted as $P(\mathcal{R}|X, Y)$, and inference, represented by $P(Y|X, \mathcal{R})$, in Tables 8 and 9, respectively. We iteratively grow the logic tree by applying the structure learning prompt to the successive node within the current tree. Importantly, our prompts are crafted using a simple, predefined template. In this template, `events_history` represents an in-text version of the observed event sequence $X$, and `target_event` corresponds to the event id/name associated with $Y$. We also provide three inference examples in Table 9.

*Table 6.* Decriptions and values of hyperparameters used for models trained on the four synthetic datasets.

| HYPERPARAMETERS | VALUE USED | | | |
|---|---|---|---|---|
| | SYNTHETIC@5 | SYNTHETIC@10 | SYNTHETIC@20 | SYNTHETIC@40 |
| EPOCHS | 10 | 10 | 10 | 10 |
| ALTERNATE EVERY | 1 | 1 | 1 | 1 |
| BATCH SIZE | 8 | 8 | 8 | 8 |
| LLM LR | 5e-4 | 5e-4 | 5e-4 | 5e-4 |
| LLM SIZE (E-STEP) | opt-1.3b | opt-1.3b | opt-1.3b | opt-1.3b |
| LLM SIZE (M-STEP) | zephyr-3b | zephyr-3b | zephyr-3b | zephyr-3b |
| LOGIC MODEL UPDATE STEPS | 1 | 1 | 1 | 1 |
| LOGIC MODEL LR | 0.001 | 0.001 | 0.001 | 0.001 |
| LOGIC TREE DEPTH | 3 | 3 | 3 | 3 |
| LOGIC TREE WIDTH | 5 | 8 | 5 | 3 |
| TOP K | 2 | 2 | 2 | 2 |
| WRAMUP LEARNING RATE | True | True | True | True |
| LoRA RANK | 512 | 512 | 512 | 512 |
| LoRA SCALING FACTOR | 512 | 512 | 512 | 512 |
| LoRA DROPOUT | 0. | 0. | 0. | 0. |

*Table 7.* Decriptions and values of hyperparameters used for models trained on the three real-world datasets.

| HYPERPARAMETERS | VALUE USED | | |
| --- | --- | --- | --- |
| | EPIC-100 | STACKOVERFLOW | MIMIC-3 |
| EPOCHS | 20 | 20 | 20 |
| ALTERNATE EVERY | 1 | 1 | 1 |
| BATCH SIZE | 2 | 2 | 2 |
| LLM LR | 5e-4 | 5e-4 | 5e-4 |
| LLM SIZE (E-STEP) | opt-1.3b | opt-1.3b | opt-1.3b |
| LLM SIZE (M-STEP) | mistral-7b | zephyr-3b | zephyr-3b |
| LOGIC MODEL UPDATE STEPS | 1 | 1 | 1 |
| LOGIC MODEL LR | 0.001 | 0.001 | 0.001 |
| TREE DEPTH | 3 | 3 | 3 |
| TREE WIDTH | 4 | 3 | 2 |
| TOP K | 2 | 3 | 2 |
| WRAMUP LEARNING RATE | True | True | True |
| LoRA RANK | 512 | 512 | 512 |
| LoRA SCALING FACTOR | 512 | 512 | 512 |
| LoRA DROPOUT | 0. | 0. | 0. |

| | **Bayesion Structure Learning** $P(\mathcal{R}|X,Y)$ |
| --- | --- |
| **Template** | I want you to do the reasoning over social events. Given event list: {total_events} <br><br> We have the observations: {events_history} <br><br> If the activation time of one event happens before Event {target_event}, it means that event could have caused Event {target_event} to be activated. <br> If the activation time of one event do not happens before Event {target_event}, it means that event cannot cause the other event to be activated. <br> Using this logic and based on the previous observation, You need to reason all possible events from above that can cause Event {target_event} to be activated. <br> Start your answer from the most confident one and stop if you cannot find any other events. <br> Answer: Event |

*Table 8.* Prompts used for structure learning

*Table 9.* Prompts Used for Next Event Inference. `rationales` store the text representation of the logic tree by going over all the paths. The reasoning path is highlighted in red color.

| | **Direct Inference** $P(Y|X)$ | **Reasoning based Inference** $P(Y|X, \mathcal{R})$ |
|---|---|---|
| **Template** | I want you to perform inference over social events. {examples} Now you have event: {total_events} We have the observations: {events_history} then, the most likely event (chosen from event list : {possible_events}) to happen after {time} is Event: | I want you to perform inference over social events. {examples} Now you have event: {total_events} and rules: {rationales} We have the observations: {events_history} then, the most likely event (chosen from event list : {possible_events}) to happen after {time} is Event: |
| **Example 1** | Given Events 0, 1 We have the observations: 1. Event 0 is activated at time 0.4<br><br>then, the most likely event (choose from event list: 0, 1) to happen after 0.4 is Event 1 | Given Events 0, 1 and rules: 1. Event 1 ← (Event 0) and (Time of Event 1 after Time of Event 0)<br><br>We have the observations: 1. Event 0 is activated at time 0.4<br><br>then, the most likely event (choose from event list: 0, 1) to happen after 0.4 is Event 1 |
| **Example 2** | Given Events 0, 1, 2<br><br>We have the observations: 1. Event 1 is activated at time 0.2<br><br>then, the most likely event (chosen from event list : 0, 1, 2) to happen after 0.2 is Event 0 | Given Events 0, 1, 2 and rules:<br><br>1. Event 0 ← (Event 1) and (Time of Event 0 after Time of Event 1), 2. Event 0 ← (Event 2) and (Time of Event 0 after Time of Event 2)<br><br>We have the observations: 1. Event 1 is activated at time 0.2<br><br>then, the most likely event (chosen from event list : 0, 1, 2) to happen after 0.2 is Event 0 |
| **Example 3** | Given Events 0, 1, 2<br><br>We have the following observation: 1. Event 0 is activated at time 0.2, 0.3, 0.5 2. Event 1 is activated at time 0.5, 0.6 3. Event 2 is activated at time 0.1, 0.4<br><br>then, the most likely event (chosen from event list : 0, 1, 2) to happen after 0.8 is Event 2 | Given Events 0, 1, 2 and rules:<br><br>1. Event 2 ← (Event 1) and (Event 0) and (Time of Event 2 after Time of Event 1) and (Time of Event 1 after Event 0)<br><br>We have the following observation: 1. Event 0 is activated at time 0.2, 0.3, 0.5 2. Event 1 is activated at time 0.5, 0.6 3. Event 2 is activated at time 0.1, 0.4<br><br>then, the most likely event (chosen from event list : 0, 1, 2) to happen after 0.8 is Event 2 |