Probabilistic Attention-to-Influence Neural Models for Event Sequences

Xiao Shou [1] Debarun Bhattacharjya [2] Tian Gao [2] Dharmashankar Subramanian [2] Oktie Hassanzadeh [2]
Kristin P. Bennett [1]

Abstract

Discovering knowledge about which types of events influence others, using datasets of event sequences without time stamps, has several practical applications. While neural sequence models are able to capture complex and potentially long-range historical dependencies, they often lack the interpretability of simpler models for event sequence dynamics. We provide a novel neural framework in such a setting – a probabilistic attention-to-influence neural model – which not only captures complex instance-wise interactions between events but also learns influencers for each event type of interest. Given event sequence data and a prior distribution on type-wise influence, we efficiently learn an approximate posterior for type-wise influence by an attention-to-influence transformation using variational inference. Our method subsequently models the conditional likelihood of sequences by sampling the above posterior to focus attention on influencing event types. We motivate our general framework and show improved performance in experiments compared to existing baselines on synthetic data as well as real-world benchmarks, for tasks involving prediction and influencing set identification.

1. Introduction

Multivariate event sequences are commonplace in a variety of applications. They involve occurrences of different types of events from a known event label set; these events are either observed regularly, i.e. at discrete time units, or occur without meaningful or reliable time stamps associated with each event instance. Given a dataset of such sequences, one may be interested in discovering *which event types* (rather than specific instances) have an influence on the occurrence of other event types of interest. Consider the following illustrative examples across different domains:

- In healthcare, patient-level historical insurance data are commonly aggregated by month (Mavroudeas et al., 2021). Decision makers at the insurance company are keen to learn about which prior event types impact the occurrence of later high-cost events, so as to invest in better preventative programs.
- Event sequences such as execution traces and error logs arise naturally in software systems, and have been studied extensively in data mining and related fields (Mannila et al., 1997; Ammons et al., 2002; Chandola et al., 2009). It can be beneficial to analyze such sequences and identify preceding event types that affect critical events such as failures (to improve system reliability) or web-based attacks (to strengthen computer security).
- There are well-known natural language processing (NLP) approaches for extracting events and event sequences from textual corpora (Chambers & Jurafsky, 2008). It is easier to extract sequences and often impossible to obtain time stamps for events, since they are typically not mentioned in the source corpora. Identifying how certain event types are influenced by others using extracted sequences could be a potential source of information regarding pairwise causal/entailment relations for knowledge graphs (Sap et al., 2019; Heindorf et al., 2020; Hassanzadeh, 2021); this in turn could be beneficial for downstream tasks such as question answering.

In the above examples as well as in other domains, *knowledge discovery* about the influence of some types of events on others can be of great interest to practitioners who engage with event sequence data. This knowledge could drastically improve model interpretability and therefore model trust for important tasks – for instance, when a learned event sequence model is used for prediction – but it could also be of direct interest to analysts, scientists, or experts who wish to gain further insights about sequence dynamics in a particular domain.

There are however challenges around discovering such knowledge from multivariate event sequences: **A**) it is not sufficient to capture how specific past instances of events influence an event type, but to aggregate these effects mean-

[1]Rensselaer Polytechnic Institute, Troy, NY, USA [2]IBM AI Research, Yorktown Heights, NY, USA. Correspondence to: Xiao Shou <shoux@rpi.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

ingfully to study type-wise influences, **B**) one needs to allow for learning potentially long-range historical influences while making minimal modeling assumptions, and **C**) there may be complex underlying historical dependencies including ones that involve non-linear interactions from prior event occurrences.

Transformer-based attention models have achieved huge success for sequence modeling in natural language processing (Devlin et al., 2018; Liu et al., 2019) and time series (Zerveas et al., 2021). Recent theoretical work shows that transformers are universal approximators of sequence-to-sequence functions (Yun et al., 2019; Kratsios et al., 2021), which to some extent explains its wide adoption in machine translation (Vaswani et al., 2017), language modeling (Radford et al., 2018; 2019) and question answering (Yang et al., 2020; Liu et al., 2019). These language models typically only capture instance-wise (or token-wise) effects via attention mechanism; they cannot be directly applied to extract meaningful type-type effects. While type-wise influences have been analyzed through summation (Xiao et al., 2019) or simple averaging (Zhang et al., 2020) during post-processing of an attention model, these approaches do not adequately attend to challenge **A** mentioned previously because the learning of each model is entirely driven by maximizing the respective loglikelihood.

Our setting is related to Granger causality for time series (Granger, 1969), since we are essentially interested in finding a subset of the full event label set with predictive power for event types of interest. Prior work focuses on multivariate time series (Dahlhaus & Eichler, 2003) and multivariate categorical time series (Tank et al., 2021) (which reduces to a Markov chain model for the univariate case). Note that a multivariate event sequence can be viewed as a univariate categorical time series, since exactly one event can happen at any position in the sequence. Summary Markov models (SuMMs) have recently been introduced as a family of models to capture Granger causality in event sequences, where the learned subset is referred to as an *influencing set* (Bhattacharjya et al., 2022). The specific proposed models in the family are however intended for small datasets and unable to adequately handle complex and long-range historical dependencies, thus falling short on challenges **B** and **C**. Note that classical Markov models have a tabular representation whose state-space grows exponentially with the order of the model, and whose data requirements also grow exponentially for estimation. To overcome these challenges, we pursue an attention-based neural version of a summary Markov model, exploiting a neural function approximation based compact representation to capture complex and long-range historical dependencies. We demonstrate its advantages on a prediction task for many real-world event sequences as well as around the identification of influencers.

Contributions. In this paper, we propose a neural framework to address all three challenges. Specifically, our proposed probabilistic attention-to-influence neural model aims to not only capture local dynamics of event sequences but also extract the underlying influencers for event types of interest. Our contributions are as follows: 1) we propose a novel integration of a probabilistic variational inference approach with a deterministic attention mechanism for multivariate event sequences; 2) we design a novel attention-to-influence and influence-to-attention formulation which captures the conversion from instance-wise effects to type-wise influence in the neural attentive framework; 3) we demonstrate that our proposed model exhibits state-of-the-art results on a probabilistic prediction task; 4) we also highlight how the model can discover meaningful influencers for event types of interest, which has far-reaching practical implications around knowledge discovery in many domains.

2. Related Work

Related Neural Event Models. Recurrent neural networks (RNNs) and variants have been the workhorse for many sequence modeling tasks in NLP and multivariate time series prior to transformer-based models. "Times" in these sequences act as the indices or positions of the observed data. In particular, RNNs have been applied to predict the next word given a historical sequence in language models (Mikolov et al., 2010). They are also capable of modeling complex dynamics in irregular time event streams (Du et al., 2016; Mei & Eisner, 2016; Omi et al., 2019; Xiao et al., 2017; Shchur et al., 2019; Gao et al., 2020), where time stamps associated with the events are modeled as a point process. Self-attention mechanism has been introduced in recurrent networks for event sequences (Xiao et al., 2019) involving both time series and event stream data. Contemporary transformer-based models have famously demonstrated superior predictive power in language modeling (Radford et al., 2019) as well as in event streams (Zhang et al., 2020; Zuo et al., 2020; Gu, 2021), due to their ability to capture long-range complex dependencies. However, we note that these neural models are primarily geared towards prediction tasks and are not directly useful for identifying influencing sets.

Related Temporal Models. Related work on modeling multivariate temporal data includes discrete-time graphical models such as (dynamic) Bayesian networks and related time series graphs (Pearl, 2014; Murphy, 2012; Dahlhaus & Eichler, 2003; Eichler et al., 2017), as well as continuous-time graphical event models that represent process independence in a point process (Didelez, 2008; Gunawardana & Meek, 2016; Bhattacharjya et al., 2018). Due to the discrete nature of multivariate event sequences, the classic k^{th}-order Markov chain model can be used to capture the dynamics. The family of summary Markov models (Bhattacharjya et al., 2022) enables a generalized notion of state dependence in variable-order Markov models (Begleiter et al., 2004) and extracts a set of influencers for event types of interest. Another relevant line of work is to use Granger causality to model categorical time series with mixture transition distribution (Tank et al., 2021).

3. Influencing Sets in Event Sequences

In this section, we first introduce basic notation and terminology, and then motivate the feasibility and need for our probabilistic attention-based neural model towards identifying influencing sets of event types of interest from event sequence datasets.

3.1. Event Sequences & History-Dependent Dynamics

An **event sequence dataset** (Mannila et al., 1997) $\mathbf{D}$ involves K sequences indexed by k, each of the form $\mathbf{D_k} = [l_i]_{i=1}^{N_k}$, where event label (or type) l_i at position i in the sequence is from a discrete label set, $l_i \in \mathcal{L}$. Thus, our setting is restricted to marks being categorical. The total number of events and event types are denoted N and M respectively, i.e. $N = \sum_{k=1}^{K} N_k$ and $M = |\mathcal{L}|$. All the events preceding a label at position i in a sequence comprise its **history**, $h_i = \{(j, l_j)\}_{j=1}^{i-1}$. We denote the **restricted history** with respect to label set $\mathbf{Z} \subset \mathcal{L}$, which only includes prior occurrences from $\mathbf{Z}$, as $h_i^{\mathbf{Z}} = \{(j, l_j) : j < i, l_j \in \mathbf{Z}\}$. Subscript i is omitted when referring to a generic position.

The generating dynamics of an event sequence dataset can be modeled using history-dependent probabilities $\theta_{x|h}$, which denotes the probability of observing event label $X \in \mathcal{L}$ at any position in the sequence given history h. Parameterization is therefore through probabilities $\Theta = \{\Theta_X : X \in \mathcal{L}\}$, $\Theta_X = \{\theta_{x|h}\}$ s.t. $\sum_{X \in \mathcal{L}} \theta_{x|h} = 1$, since exactly one event label must occur in any position.

3.2. Neural Summary Markov Models

We wish to model local generating dynamics for a set of event labels where not all the event labels in history are relevant. We consider a sequence **summary function** $s(\cdot)$ for label set $\mathbf{Z}$ that maps any restricted history $h^{\mathbf{Z}}$ at any sequence position to some **summary state**. Unlike Bhattacharjya et al. (2022) who only consider models where the summary states $s_{\mathbf{Z}}$ belong to some discrete range $\Sigma_{\mathbf{Z}}$, we allow for more complex historical summaries. In particular, we consider a neural network such that any restricted history can be summarized by state $s_{\mathbf{Z}} = s_\phi(h^{\mathbf{Z}})$, where ϕ are network parameters for the summary function and $s_{\mathbf{Z}} \in \mathbb{R}^d$ for some vector dimension d. Any history h is **consistent** with state $s_{\mathbf{Z}}$ if the summary function applied to h restricted to $\mathbf{Z}$ results in $s_{\mathbf{Z}}$, i.e. $s(h^{\mathbf{Z}}) = s_{\mathbf{Z}}$.

Example 3.1. *Figure 1 shows an example event sequence with 5 event labels, $\mathcal{L} = \{A, B, C, D, E\}$. Consider a label of interest C. An ordinal summary function summarizes restricted history through a vector specifying the order in which the restricted labels occur in recent history, after disregarding repeated occurrences. In this example, with a look-back of 3 positions, this function maps prior history restricted to $\{A, B\}$ to $\emptyset$, $[B]$, $[B, A]$, $[A]$ and $[A]$ at the 5 occurrences of C respectively. Here $\emptyset$ denotes absence of both A and B type events. In contrast, an autoregressive summary function (Tank et al., 2021) summarizes the restricted history by specifying occurrences of labels in each look-back position. In this example, this maps to $[\emptyset, \emptyset, \emptyset]$, $[\emptyset, \emptyset, B]$, $[B, B, A]$, $[A, \emptyset, A]$, $[A, \emptyset, \emptyset]$. Note that event sequences in applications such as healthcare may require more complex summary functions where some combination of older events that reflect some chronic condition together with more recent events may represent realistic dynamics.*

A neural summary function summarizes any restricted history into a vector. The generative local dynamics are modeled using a decoder that maps from any summary state to a multinomial probability distribution. Formally, dynamics for a subset of the event labels $\mathbf{X}$ is modeled using **probability function** $\theta_x^{\mathbf{Z}} = \theta_\psi(s_{\mathbf{Z}})$ such that $0 \leq \theta_x^{\mathbf{Z}} \leq 1, \forall X \in \mathbf{X}$. Furthermore, when $\mathbf{X} \subset \mathcal{L}$, there is an additional probability for when none of the labels in $\mathbf{X}$ occur, denoted $\theta_{\mathcal{L}\setminus\mathbf{X}}^{\mathbf{Z}}$, where $0 \leq \theta_{\mathcal{L}\setminus\mathbf{X}}^{\mathbf{Z}} \leq 1$ and $\theta_{\mathcal{L}\setminus\mathbf{X}}^{\mathbf{Z}} = 1 - \sum_x \theta_x^{\mathbf{Z}}$. If on the other hand $\mathbf{X} = \mathcal{L}$, then $\sum_x \theta_x^{\mathbf{Z}} = 1$. Here ψ are network parameters for $\theta(\cdot)$. The probability of event label X occurring given any history h, with restriction to label set $\mathbf{Z}$, can be obtained through a composition of the summary and probability functions: $\theta_{x|h} = \theta_\psi(s_\phi(h^{\mathbf{Z}}))$. The notion of an influencing set can now be formalized as follows:

Definition 3.2. A label set $\mathbf{U}$ is an **influencing set** for event labels $\mathbf{X}$ under summary function $s_\phi(\cdot)$ if for all summary states $s_{\mathbf{U}}$, $\sum_{X \in \mathbf{X}} \theta_{x|h} = \sum_{X \in \mathbf{X}} \theta_{x|h'}$ for all h, h' consistent with $s_{\mathbf{U}}$. $\mathbf{U}$ is **minimal** if the condition cannot be satisfied after removing any label in $\mathbf{U}$.

We generalize a previous result (Bhattacharjya et al., 2022) and show the existence of a minimal influencing set:

Theorem 3.3. *There is a unique minimal influencing set for any label set $\mathbf{X} \subseteq \mathcal{L}$ corresponding to any summary function $s_\phi(\cdot)$ and for any event sequence dynamics probability function $\theta_\psi(\cdot)$.*

All proofs can be found in section D of the Appendix. We now propose a generative family of models for local dynamics in event sequences where only historical occurrences of a subset of event types play a role.

Definition 3.4. A **neural summary Markov model** for event label set $\mathbf{X} \subseteq \mathcal{L}$ includes:

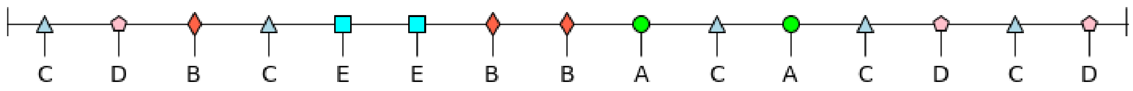

Figure 1. An illustrative event sequence over labels $\{A, B, C, D, E\}$ where $X = C$ is of interest. An example of event labels from healthcare is as follows: A: Medication refill, B: Cardiac event, C: Emergency visit, D: Hypertension diagnosis, E: Anxiety diagnosis.

- A set of labels $\mathbf{U}$ that is an influencing set for labels $\mathbf{X}$.
- A summary function $s_\phi(\cdot)$ which maps any restricted history $h^{\mathbf{U}}$ to $s_{\mathbf{U}} \in \mathbb{R}^d$.
- A multinomial probability function $\theta_\psi(\cdot)$ which takes any summary state $s_{\mathbf{U}}$ as input.

Since a unique minimal influencing set exists for local dynamics for any subset of labels, it may be possible to learn a model to discover this minimal set from an event sequence dataset. An important application is around learning a model for a single label of interest X at a time; here one is interested in identifying event types that influence whether X will happen or not, in any sequence position.

We remind readers that an event sequence is a univariate categorical time series, therefore exactly one event can happen at any position. This effectively couples all the labels together as probabilities must sum to one, which can often result in a situation where the full label set $\mathcal{L}$ is the influencing set for each label individually. The following example is illustrative in that regard.

Example 3.5. *Figure 1 shows an event sequence whose dynamics generation is topology-based, i.e. driven by some underlying graph. Specifically, this sequence is generated from Graph-1 in Figure 3(a), where the un-normalized probability of a label occurrence depends only on the number of historical occurrences of its parents in the topology. Due to the nature of data generation and the coupling of probabilities over labels, the influencing set for each label when considered individually is $\mathcal{L}$, even though there are two subgraphs in the underlying graph that are disconnected ($\{A, B, C\}$ and $\{D, E\}$). This is a toy-problem version of many real-world applications where event sequences are driven by underlying physical networks in systems.*

The above example demonstrates that it could be of practical importance to identify the *extent* to which a label influences another, even when all labels are effectively influencers. In the next section, we describe a specific neural summary Markov model that suitably tackles this challenge, and later show through an experiment how the learned influencers can help recover the underlying topology in Section 5.2.

4. Probabilistic Attention-to-Influence

We propose a probabilistic attention-to-influence neural model that helps identify influencers for event types of interest. For simplicity of exposition, we describe the model when there is a single event type of interest X, but the concepts extend to local dynamics over label set $\mathbf{X} \subseteq \mathcal{L}$. The model consists of two parts: an attention encoder and an attention-to-influence decoder as shown in Figure 2. The former leverages the attention mechanism from transformer architecture to model pairwise interactions of event types with type X and the latter learns a dynamical model given an influencing set. Our goal is to provide a probabilistic estimate of influence given event sequences $\mathbf{D}$ as in Figure 1. Let $\mathcal{V}$ be a binary random vector which encodes how various event(s) in $\mathbf{U}$ jointly influence X. We aim to find a good approximation to the posterior $p(\mathcal{V}|\mathbf{D})$ given a prior on $p(\mathcal{V})$. With the aid of variational inference (Zhang et al., 2018; Zhang & Yan, 2021), this posterior distribution given sequences $\mathbf{D}$ can be approximated by a global mean field variational distribution $q_\omega(\mathcal{V}|\mathbf{D})$ (parametrized by attention network ω). It is worth highlighting that the true posterior may contain coupling effects of influencers on X (i.e. the presence of Y and absence of Z with respect to the restrict history of $\{Y, Z\}$ on X); $q_\omega(\mathcal{V}|\mathbf{D})$ contains only decoupled effects. Naturally, the evidence lower bound (ELBO) (Hoffman et al., 2013) can be expressed as:

$$\mathbb{L}(\omega, \psi; \mathbf{D}) = \mathbb{E}_{q_\omega}[\log \frac{p(\mathcal{V})}{q_\omega(\mathcal{V}|\mathbf{D})}] + \mathbb{E}_{q_\omega}[\log \theta_\psi(\mathbf{D}|\mathcal{V})] \quad (1)$$

The first term in Equation 1 is negative KL divergence between $q_\omega(\mathcal{V}|\mathbf{D})$ and $p(\mathcal{V})$. Assuming $p(\mathcal{V}) = \prod_Y p(\mathcal{V}_{XY} = 1)$ for any $Y \in \mathcal{L}$, where $\mathcal{V}_{XY}$ is an indicator for whether Y is an influencer of X, we obtain a component-wise analytical form:

$$\mathbb{E}_{q_\omega}[\log \frac{p(\mathcal{V})}{q_\omega(\mathcal{V}|\mathbf{D})}] = \sum_Y \mathbb{E}_{q_\omega(\mathcal{V}_{XY}|\mathbf{D})}[\log \frac{p(\mathcal{V}_{XY})}{q_\omega(\mathcal{V}_{XY}|\mathbf{D})}] \quad (2)$$

The second term is the conditional likelihood of observing the sequences for a given influencing set which can be approximated by Monte Carlo sampling in closed-form with sample $\mathcal{V}_j$:

$$\mathbb{E}_{q_\omega} log \theta_\psi(\mathbf{D}|\mathcal{V}) \approx \frac{1}{J} \sum_{j=1}^{J} \sum_{k=1}^{K} \sum_{i=1}^{N_k} log \theta_\psi(l_i|\mathcal{V}_j) \quad (3)$$

The neural summary function $s_\phi(\cdot)$ in our framework is implicitly embedded as the transformer network which takes a restricted history and outputs a history representation. It

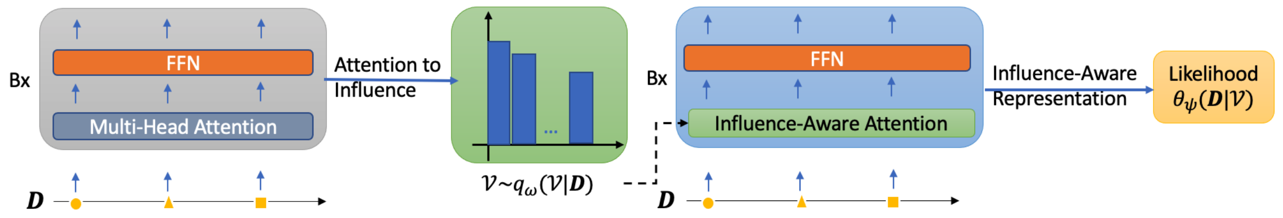
Bx
FFN
Multi-Head Attention
Attention to Influence
$\mathcal{V} \sim q_\omega(\mathcal{V}|\boldsymbol{D})$
Bx
FFN
Influence-Aware Attention
Influence-Aware Representation
Likelihood $\theta_\psi(\boldsymbol{D}|\mathcal{V})$
$\boldsymbol{D}$
$\boldsymbol{D}$

Figure 2. Neural architecture of the proposed probabilistic attention-to-influence neural model.

gets absorbed into dynamics probability function $\theta_\psi(\cdot)$ in our attention decoder network ψ. We describe computations in further detail as follows.

4.1. Encoder

The encoder is designed to infer influences between different event types given sequences $\mathbf{D}$. We apply positional embedding (Vaswani et al., 2017) to the sequences and combine the embeddings with one-hot encoded vectors for event types to form the embedded input for the attention module. The B blocks of multi-head self-attention are applied to obtain a high level representation of the sequences. Attention output $\mathbf{A}_{enc}^{(i)}$ of the i^{th} block is computed as: $\mathbf{A}_{enc}^{(i)} = \text{softmax}(\frac{\mathbf{Q}^{(i)}(\mathbf{K}^{(i)})^T}{\sqrt{M_k}})\mathbf{V}^{(i)} = \mathbf{S}_{enc}^{(i)}\mathbf{V}^{(i)}$ where $\mathbf{Q}^{(i)}$, $\mathbf{K}^{(i)}$, $\mathbf{V}^{(i)}$ are query, key and value matrices in the i^{th} block; they are linear transformations of the embedded input multiplied by trainable weights $\mathbf{W}_Q^{(i)}$, $\mathbf{W}_K^{(i)}$, and $\mathbf{W}_V^{(i)}$ respectively. M_k is the dimension of the key matrix. We extract the averaged attention score matrix from the B^{th} block, namely $\mathbf{S}_{enc}^{(B)}$. Each entry (i, j) of $\mathbf{S}_{enc}^{(B)} \in \mathbb{R}_+^{N_k \times N_k}$ signifies the past influence of event instance l_j on event instance l_i through dot-product attention mechanism. Our approach in fact naturally embeds the following assumption: any event instance l_i in a sequence $\mathbf{D}_k$ only attends to past instance l_j where $l_j \in \mathcal{L}$. l_j and l_i can be of the same type. This assumption can be justified by the temporal ordering of event sequences. In practice, it is easily achieved by imposing a strictly upper triangular attention mask on each sequence. The same assumption applies to the decoder. For a more general attention score matrix, we modify with exponential decay:

$$\widetilde{\mathbf{S}}_{enc}^{(B)} = \mathbf{S}_{enc}^{(B)} \odot \exp^{-\gamma \Delta T}, \tag{4}$$

where ΔT denotes position distances for events in each sequence $\mathbf{D_k}$ and γ is a hyper-parameter which controls the decay of influence of event l_j on l_i and $\odot$ is entry-wise multiplication. The justification of such decay is the more recent occurrences of an influencing event should impact an event of interest more than older occurrences.

Attention-to-Influence. We summarize how an event type influences another based on our proposed Attention-to-Influence (A2I) formulation to obtain an unscaled influence matrix:

$$\text{A2I}(\widetilde{\mathbf{S}}_{enc}^{(B)}) = \mathbf{P}^\top \widetilde{\mathbf{S}}_{enc}^{(B)} \mathbf{P}, \tag{5}$$

where $\mathbf{P} \in \{0, 1\}^{N_k \times M}$ is a binary indicator matrix which specifies the type of events occurring in each $\mathbf{D_k}$ for the N_k event instances [1]. In particular, each row of $\mathbf{P}$ is an M-dimensional one-hot vector. We emphasize the above equation captures the transformation from instance-instance interaction to type-type interaction efficiently. We take the row with respect to event of interest X, i.e. $A2I_X(\widetilde{\mathbf{S}}_{enc}^{(B)}) \in \mathbb{R}^M$ from Equation 5, which signifies influences of other event types on X. We apply a linear transformation and impose a sigmoidal activation to ensure a probability distribution:

$$q_\omega(\mathcal{V}|\mathbf{D}) := \sigma(\mathbf{W}_q\, \text{A2I}_X(\widetilde{\mathbf{S}}_{enc}^{(B)}) + \mathbf{b}_q) \tag{6}$$

where $\mathbf{W}_q$ and $\mathbf{b}_q$ are trainable weights and biases. Many mappings from real values to binary probability distributions ($f : A2I_X \rightarrow [0, 1]^M$) are possible. In our case, we use the sigmoidal function due to its effectiveness in neural networks. Sampling from $q_\omega(\mathcal{V}|\mathbf{D})$ is straightforward, however since the distribution is discrete, Gumbel-Softmax (Maddison et al., 2017; Jang et al., 2016) is used to provide differentiable samples for back-propagation in our neural network.

4.2. Decoder

The goal of the attention decoder is to model the dynamics by leveraging the underlying influencing set. In particular, we incorporate a sampled influencing set into the learning of the dynamics. We make the following simplifying assumption which describes the relation between attention and influence:

Assumption 4.1. For any pair of events (l_i, l_j) where $i < j$ in a sequence $\mathbf{D_k}$, event instance l_j attends to event instance l_i if and only if the associated event type of l_i is in the influencing set of type for l_j.

The above assumption is aligned with the Granger causal related notion in various event models (Didelez, 2008; Bhattacharjya et al., 2018; Yu et al., 2020): only parent processes

[1]In practice, we add an extra dimension 0 for padded events.

can affect a child process, and other processes are considered to be conditionally locally independent from the child process. This enables us to directly modify the attention score through a given influencing set for an event type of interest.

Influence-to-Attention. We propose the learning of influence-aware attention in the following. For each event instance in sequence $\mathbf{D_k}$, we check the event type it influences given a sampled influencing set $\mathcal{V}_j$ for X, fill entries related to events of non-interest $\mathcal{L} \setminus X$ with 1's to form $\widetilde{\mathcal{V}_j} \in \{0,1\}^{M \times M}$ and construct influence indicator $\mathbf{I} \in \{0,1\}^{N_k \times N_k}$:

$$\mathbf{I} = \mathbf{P}\widetilde{\mathcal{V}_j}\mathbf{P}^\top \tag{7}$$

where $\mathbf{P}$ is the binary indicator matrix same as in Equation 5. We combine an attention score matrix in the decoder $\mathbf{S}^{(i)}_{dec} = \text{softmax}(\frac{\mathbf{Q}^{(i)}(\mathbf{K}^{(i)})^T}{\sqrt{M_k}})$ with influence indicator $\mathbf{I}$ to derive the influence-aware attention score:

$$\tilde{\mathbf{S}}^{(i)}_{dec} = \mathbf{S}^{(i)}_{dec} \odot \mathbf{I} \tag{8}$$

For an h-head influence-aware attention module in the i^{th} block, we concatenate the above representation from each head and multiply it by a trainable weight matrix $\mathbf{W}^{(i)}_O$ to form a new representation: $\tilde{\mathbf{A}}^{(i)}_{dec} = [\tilde{\mathbf{S}}^{(i)}_{dec,1}, \tilde{\mathbf{S}}^{(i)}_{dec,2}, ..., \tilde{\mathbf{S}}^{(i)}_{dec,h}]\mathbf{W}^{(i)}_O$. In the B-block multi-head self-attention, the i^{th} block outputs a high level representation $\tilde{\mathbf{H}}^{(i)}$ from a feed forward network with trainable weights and biases $\mathbf{W}^{(i)}_{FC}$'s and $\mathbf{b}^{(i)}_{FC}$'s:

$$\tilde{\mathbf{H}}^{(i)} = \text{ReLU}(\tilde{\mathbf{A}}^{(i)}_{dec}\mathbf{W}^{(i)}_{FC,1} + \mathbf{b}^{(i)}_{FC,1})\mathbf{W}^{(i)}_{FC,2} + \mathbf{b}^{(i)}_{FC,2} \tag{9}$$

Each $\tilde{\mathbf{H}}^{(i)}$ is then sequentially fed into $i+1^{th}$ block until the B^{th} block is reached. It is worth emphasizing that the high level representation $\tilde{\mathbf{H}}^{(B)}(i)$ not only summarizes the history prior to event l_i, but also the impact of the influencing set on label of interest over time. The history dependent log likelihood term in Equation 3 can thus be succinctly expressed as $\theta_\psi(l_{i+1}|\mathcal{V}_j) = \theta_\psi(l_{i+1}|\tilde{\mathbf{H}}^{(B)}(i))$. [2] The generated labels can be modeled via a multinomial distribution:

$$\theta_\psi(l_{i+1} = m|\tilde{\mathbf{H}}^{(B)}(i)) = \frac{\exp(\mathbf{W}_{m,:}\tilde{\mathbf{H}}^{(B)}(i) + \mathbf{b}_m)}{\sum_{m=1}^{M}\exp(\mathbf{W}_{m,:}\tilde{\mathbf{H}}^{(B)}(i) + \mathbf{b}_m)} \tag{10}$$

where $\mathbf{W}_{m,:}$ is the m^{th} row of the corresponding trainable weight matrix and $\mathbf{b}_m$ is m^{th} entry of the corresponding bias term. The Influence-to-Attention mechanism concisely integrates pairwise influence in the learning of event dynamics by repeatedly modifying the attention score matrix in B-blocks of multi-head self-attention module.

[2] $\theta_\psi(l_1|\mathcal{V}_j)$ is obtained by a padded event.

The following theorem shows that our model is more general for learning type-wise effects than existing instance-based attention models (Xiao et al., 2019; Zhang et al., 2020), which are a special case of our method. Furthermore, we provide some evidence to support the claim that our approach does not introduce any additional complexity as stated in Theorem 4.3, and we provide proof sketches for further reference in the Appendix.

Theorem 4.2. *Post-processing by accumulating instance-wise attention scores to type-wise effects as in (Xiao et al., 2019; Zhang et al., 2020) is equivalent to considering a full and deterministic influencing set,* $\mathbf{U} = \mathcal{L}$*, or equivalently,* $q_\omega(\mathcal{V} = \mathbb{1}|\mathbf{D}) = 1$ *in our model.*

Theorem 4.3. *Our model achieves the same order of computational complexity, runtime compute and memory requirements compared to other transformer based models (Vaswani et al., 2017) if the cardinality of event types* M *is much smaller than the embedding dimension* d*,* $M \ll d$ *.*

Algorithm 1 Topology-based event sequence generator (with Python pseudo code)

```
Input: Given label set: $\mathcal{L}$, underlying topology/graph that
       guides sequence generation: $\mathcal{G}$, number of events per
       sequence: seq_len, multiplicative factor: mult_factor
Output: A generated event sequence $D$
hist_info_dict = {}
  $D$ = []
  for $i \leftarrow 1$ to seq_len do
      un-normalized_prob_vec = []
      for lab in $\mathcal{L}$ do
          un-normalized_prob = 1
          counts = 0
          for par of lab in $\mathcal{G}$ do
              if par in hist_info_dict then
                  counts += 1
          end
          if counts $\neq 0$ then
              un-normalized_prob = mult_factor * counts
              un-normalized_prob_vec.append(un-normalized_prob)
      end
      prob_vec = $\frac{un-normalized\_prob\_vec}{sum(un-normalized\_prob\_vec)}$
      this_lab = np.random.choice($\mathcal{L}$, p=prob_vec)
      $D$.append(this_lab)
      if this_lab in hist_info_dict then
          hist_info_dict[this_lab] += 1
      else
          hist_info_dict[this_lab] = 1
      end
  end
Return: $D$
```

5. Experiments

We implement the multi-head attention module in the encoder and influence-aware attention in the decoder in PyTorch, following standard procedure to train the model. Details around implementation and training can be found in section A of the Appendix. We test the performance of our model on benchmark real-world datasets on tasks for prediction as well as identification of influencing sets such as from data simulated by a topology-based event sequence generator.

5.1. Prediction

In this set of experiments, we are interested in the local dynamics for each label of interest X in an event sequence. Our task is to predict whether X will happen next, based on prior history.

Models, Baselines & Metric. We test 2 variants of our probabilistic-attention-to-influence neural models, namely uniform-2 and uniform-τ. The former means we sample 2 samples $\mathcal{V}_j$ ($J = 2$ in Equation 3) from the posterior $q_\omega(\mathcal{V}|\mathbf{D})$; the latter specifies that instead of sampling, we manually place pre-specified thresholds τ's on the posterior. The τ values are selected through a separate dev dataset. We compare our models against existing ones: parametric summary Markov models (i.e. BSuMM and OSuMM) (Bhattacharjya et al., 2022), k^{th} order Markov chains (MC) for varying k, logistic regression (LR) with a varying look-back of k positions, and an LSTM model (Hochreiter & Schmidhuber, 1997). We train baseline models (MC, SuMMs, LR and LSTM) according to the exact procedure described in Appendix B.3 in (Bhattacharjya et al., 2022). Additionally, we compare 2 state-of-the-art neural models: a modification of the transformer Hawkes process (THP) (Zuo et al., 2020) and transformer for next token (TNT) prediction in NLP (Devlin et al., 2018). Without time stamps, THP in our setting utilizes constant timesteps [1,2,3,4,...] for its temporal encoding, and TNT leverages transformer structure and maximizes the log-likelihood of event sequences $\mathbf{D}$: log $\theta_\psi(\mathbf{D}) = \sum_{k=1}^{K}\sum_{i=1}^{N_k} log\theta_\psi(l_i)$ where θ_ψ is the predictive multinomial distribution in Equation 13. More details around THP and TNT are discussed in section B of the Appendix. Since our task involves binary prediction, i.e. whether the next event is the event of interest X or not given prior history, we evaluate based on negative log loss (identical to *log likelihood*), a suitable metric for predicting low probability events (Bishop, 2006) since all models generate probabilistic predictions.

Datasets. We consider 5 real event datasets in different domains curated previously (Bhattacharjya et al., 2022). Each dataset is randomly split into 70%-15%-15% train, dev, and test set. Models are trained with train set and hyperparameters for models are selected using the dev set. Once training completes, evaluation is conducted on the held-out test set. We briefly describe each of them here, and give summary statistics in Table 1.

Table 1. Dataset summary: # of event labels (M), # of sequences (K) and # of events (N).

Dataset	M	K	N
Beige Books	15	349	3271
Diabetes	13	65	20210
LinkedIn	10	1000	2932
Stack Overflow	20	1000	71254
Timelines	337	529	18549

Diabetes (Frank & Asuncion, 2010) contains daily events around meal intake, exercise activity, insulin dosage and changes in blood glucose measurements for 67 diabetic patients. **Stack Overflow** (Grant & Betts, 2013) is a question-answering website where users get engaged in the online community; our dataset contains events for engagement of 1000 users (chosen from (Du et al., 2016)). **LinkedIn** (Xu et al., 2017) contains user-level employment-related events for 1000 LinkedIn users. **Beige Books** contains sequences of topics extracted from documents published by the United States Federal Reserve Board about economic conditions in the US. **Timelines** contains sequences of event-related Wikidata (Vrandecic & Krötzsch, 2014) concepts from Wikipedia articles.

Results. Table 2 shows negative log loss averaged over labels of interest for models on a held-out test set. Results with additional baseline settings are in section B of the Appendix. For all datasets except Timelines, all labels are considered to be of interest. The labels of interest in Timelines are chosen as in prior work (Bhattacharjya et al., 2022). The proposed models are generally high performing, achieving best or second best performance on most datasets, although TNT is a close competitor. The most significant improvement of our models over non-neural baselines is on

Table 2. Negative log loss, averaged over labels of interest on the test sets. 3^{rd} order Markov chains (MC) and logistic regression (LR) with a look-back of 3 are shown. Best results are in bold; second best are in italics.

Model	Beige Books	Diabetes	LinkedIn	Stack Overflow	Timelines
3-MC	-37.66	-473.96	-119.55	-1435.12	-1343.52
3-LR	-36.85	-506.05	*-92.23*	-1277.84	-160.84
BSuMM	-36.11	-497.90	-114.52	-1242.59	-141.42
OSuMM	-38.07	**-432.89**	-115.63	-1246.64	-142.18
LSTM	-63.65	-595.57	-135.92	-1246.45	-135.98
TNT	**-20.38**	-453.53	-96.79	**-120.41**	-136.67
THP	-38.41	*-433.69*	-133.53	-1277.01	*-131.53*
Uniform-τ	-21.54	-448.95	**-91.21**	*-121.12*	**-125.29**
Uniform-2	*-21.06*	-447.77	-95.96	-134.70	-135.30

Table 3. Results on topology recovery on Graph-1. Best results are highlighted in bold. Samples of 100 and 900 sequences are generated for model comparison.

	BSuMM		OSuMM		THP		TNT		Uniform-τ		Uniform-2	
Event	100	900	100	900	100	900	100	900	100	900	100	900
B	0.62(0.18)	0.50(0.02)	0.67(0.26)	0.63(0.07)	0.33(0.07)	0.32(0.09)	0.32(0.06)	0.32(0.10)	0.62(0.07)	0.65(0.05)	**1.00(0.00)**	**1.00(0.00)**
C	0.56(0.19)	0.66(0.04)	0.51(0.14)	0.65(0.2)	0.56(0.12)	0.51(0.17)	0.57(0.00)	0.52(0.16)	0.57(0.00)	0.57(0.00)	**1.00(0.00)**	**1.00(0.00)**
D	0.66(0.07)	0.41(0.03)	0.69(0.12)	0.50(0.00)	0.50(0.26)	0.48(0.25)	0.83(0.27)	0.83(0.32)	**1.00(0.00)**	**1.00(0.00)**	**1.00(0.00)**	**1.00(0.00)**
E	0.65(0.06)	0.41(0.03)	0.67(0.12)	0.50(0.00)	0.47(0.25)	0.50(0.24)	0.75(0.31)	0.84(0.27)	0.47(0.04)	0.44(0.05)	**0.80(0.16)**	**1.00(0.00)**

Table 4. Results on Graph-2. Best results are highlighted in bold.

	BSuMM		OSuMM		THP		TNT		Uniform-τ		Uniform-2	
Event	100	900	100	900	100	900	100	900	100	900	100	900
B	0.60(0.45)	0.51(0.04)	0.60(0.48)	0.66(0.02)	0.33 (0.05)	0.32(0.07)	0.33(0.03)	0.30(0.11)	**1.00(0.00)**	**1.00(0.00)**	**1.00(0.00)**	**1.00(0.00)**
C	0.32(0.33)	0.76(0.10)	0.35(0.33)	0.56(0.18)	0.56(0.07)	0.57(0.04)	0.57(0.00)	0.57(0.02)	**1.00(0.00)**	**1.00(0.00)**	**1.00(0.00)**	**1.00(0.00)**
D	0.57(0.45)	0.50(0.03)	0.56(0.48)	0.67(0.05)	0.33 (0.03)	0.31(0.08)	0.33(0.00)	0.31(0.10)	**1.00(0.00)**	**1.00(0.00)**	**1.00(0.00)**	**1.00(0.00)**
E	0.33(0.33)	0.78(0.07)	0.31(0.33)	0.57(0.19)	0.53(0.12)	0.56(0.07)	0.57(0.06)	0.57(0.04)	**1.00(0.00)**	**1.00(0.00)**	**1.00(0.00)**	**1.00(0.00)**

Table 5. Comparison of influencing sets on Stack Overflow and LinkedIn.

Dataset	Event	BSuMM	OSuMM	Uniform-τ
Stack Overflow	4	{1, 2, 4, 6, 7, 9}	{2, 4, 5, 6}	[12, 11, 16, 14, 17, 5, 6, 20, 1, 4]
	19	{4, 10, 19}	{10, 19}	[13, 9, 19, 8, 15, 16, 4, 3, 20, 18]
LinkedIn	1	{1, 2, 3, 7, 9}	{1, 3, 7}	[6, 7, 4, 3, 10, 9, 8, 5, 2, 1]
	3	{1, 2, 3, 4, 5}	{1, 3, 4}	[6, 7, 4, 9, 3, 10, 8, 5, 2, 1]

Table 6. Influencing concepts for two Wikidata concepts using Event Registry news snippets.

Event of Interest	Influencing Events (in Order of Decreasing Posterior)
protest	[murder, arson, explosion, school shooting, aviation accident, disease outbreak, economic crisis coup d'état, civil disorder, regime change, earthquake, mass shooting, risk factor fraud, suicide, armed conflict, statutory law, disaster, impeachment, protest, conflict]
economic crisis	[earthquake, procession, crime, bomb attack, attack, theft hazard, disaster, work accident, riot, regime change, infectious disease, disease outbreak explosion, massacre, scandal, industrial disaster, protest accident]

Stack Overflow, which is around 10-fold. The success of transformer models is due to their flexibility in capturing complex dynamics and historical influence in a data-driven approach. The inferior performance of THP (particularly on Stack Overflow) indicates that simply adapting a continuous time point process model to our setting may not capture local dynamics correctly. Our attention-to-influence approach goes well beyond other neural models such as LSTMs to exploit essential characteristics of BSuMM and OSuMM while using the predictive power of transformer models.

5.2. Influencing Set Identification

While many tools exist for next token prediction, our model surpasses mere prediction capabilities. Instead, we focus on a more crucial task for knowledge discovery: identifying influencing sets for event labels. In doing so, we showcase the unique advantages of our model.

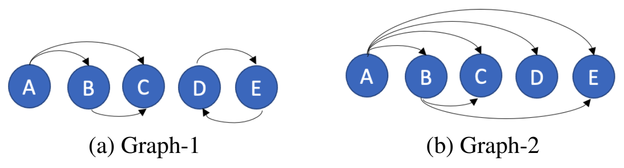

Figure 3. 2 Underlying topological graphs for synthetic event sequence generation. (a) is disconnected and (b) is connected.

Synthetic Experiments. To conduct synthetic experiments, we generate 100 simulations of temporal datasets based on 2 representative topologies, namely Graph-1 (a disconnected graph) and Graph-2 (a connected graph) shown in Figure 3. This is achieved using Algorithm 1. For each simulated dataset, we generate two sets of event sequences: one with $K = 100$ sequences and another with $K = 900$ sequences. Each sequence consists of 20 events.

The dynamics of the generated event sequences are determined by their underlying graphs. The un-normalized probability of a label occurring in a sequence depends solely on the historical occurrences of its parents in the topology. Specifically, we examine the counts of a label's parents in the history and multiply those counts by a parameter 'mult_factor' (set to 10 in our experiments) to obtain the unnormalized probability. Events are then generated according to a normalized probability distribution, ensuring that the sum of probabilities for each label's occurrence at any position is equal to one. Algorithm 1 provides high-level Python pseudo codes for this generation process.

For experimentation and evaluation, the generated dataset is randomly split into train, dev, and test sets, with proportions of 70%, 15%, and 15% respectively. We repeat this entire process 100 times for both Graph-1 and Graph-2. We report the mean F1 score and its standard deviation while comparing the most influential events for a specific event of interest with the underlying parents in the topology. This analysis is performed on the test data. We individually compare the recovered influential events with the underlying parents for each event of interest X, where $X \in \{B, C, D, E\}$[3].

Results. Our model Uniform-2 successfully identifies the most influencing event type(s) for all labels on Graph-1 and Graph-2 in Table 3 and 4, a statistically significantly improved result compared to all baselines. Note the weaker variant Uniform-τ still achieves improved overall performance. The influencing events identified by neural models TNT and THP are averaged attention weights via post-processing (see (Zhang et al., 2020) for more details). Even though TNT is powerful for prediction, it is much inferior for identifying most influencing event types - it only detects the mutual dependence of D and E with reasonable accuracy in Graph-1 while failing others. This empirically suggests post-processing is not suitable for influencing event identification even with state-of-the-art sequence models. Bhattacharjya et al. (2022) show BSuMM and OSuMM are powerful for learning influencing sets; however they fail to distinguish the *extent* among influencing events in our setting. The fact that our neural SuMM recovers the underlying topology well indicates that it is successfully giving a higher posterior for parents in the topology. This is useful in practical applications where it is beneficial to identify the magnitude of influence.

Stack Overflow & LinkedIn. We select exemplar events of interest on Stack Overflow and LinkedIn and their influencing sets discovered by BSuMM, OSuMM, and Uniform-τ [4] in Table 5. The top 10 influencers identified by our model are displayed in descending order of posterior probability. The non neural models select a sparse set of influencers for each label, while the neural model has a larger influencing set. The ability of the proposed model to capture more complex historical dependence on larger influencing sets leads to an order of magnitude improvement in the predictive performance on Stack Overflow, as shown in binary prediction. As the dataset contains longer sequences, it may be challenging for BSuMM and OSuMM with a shorter look-back of 3 positions to capture influences from older event occurrences.

Event Registry. We show an example of influencing set discovery by our model Uniform-τ on a dataset derived from a corpus of news article snippets from Event Registry (Leban et al., 2014). The corpus consists of 40 months of snippets where each is turned into a sequence of concepts from Wikidata (Vrandecic & Krötzsch, 2014) using keyword matching. The listed influencing events (in order of decreasing posterior) for "protest" and "economic crisis" in Table 6 show interesting patterns of how socio-political events have affected one another in recent years. Interestingly, murder is the most influencing event for protest (potentially for justice) in the US, perhaps a remarkable reflection of our contemporary society.

6. Conclusion

We have proposed a novel probabilistic attention-to-influence neural approach for modeling event sequences and learning influencing sets for a given event label set of interest. As far as we are aware, ours is the first probabilistic formulation for attention mechanism that has been applied for tasks of interest to event sequences without time stamps, such as label prediction and influencing set discovery. Our approach is more broadly applicable and could potentially be extended to multivariate temporal point process data. Note that pairwise influence could be incorrectly interpreted as a causal relation in applications, thus we urge appropriate usage of our proposed model to avoid drawing potentially harmful conclusions. Future research directions could explore adapting to memory-efficient transformers (Beltagy et al., 2020; Kitaev et al., 2019) and applying model variants to other temporal event data.

Acknowledgments

This work was supported by the Rensselaer-IBM AI Research Collaboration (http://airc.rpi.edu), part of the IBM AI Horizons Network (http://ibm.biz/AIHorizons). We express our gratitude to the anonymous reviewers and meta-reviewer for their valuable feedback, which greatly contributed to improving the quality of our work.

[3]F1 scores are not computed for label A since it does not have any parents.

[4]Similar results are obtained from Uniform-2.

References

Ammons, G., Bodík, R., and Larus, J. R. Mining specifications. *ACM Sigplan Notices*, 37(1):4–16, 2002.

Begleiter, R., El-Yaniv, R., and Yona, G. On prediction using variable order Markov models. *Journal of Artificial Intelligence Research*, 22:385–421, 2004.

Beltagy, I., Peters, M. E., and Cohan, A. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Bhattacharjya, D., Subramanian, D., and Gao, T. Proximal graphical event models. *Advances in Neural Information Processing Systems*, 31, 2018.

Bhattacharjya, D., Sihag, S., Hassanzadeh, O., and Bialik, L. Summary Markov models for event sequences. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 4836–4842, 2022.

Bishop, C. M. *Pattern Recognition and Machine Learning*. Springer, 2006.

Chambers, N. and Jurafsky, D. Unsupervised learning of narrative event chains. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, volume 94305, pp. 789–797, 2008.

Chandola, V., Banerjee, A., and Kumar, V. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3): 1–58, 2009.

Dahlhaus, R. and Eichler, M. Causality and graphical models in time series analysis. *Oxford Statistical Science Series*, pp. 115–137, 2003.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Didelez, V. Graphical models for marked point processes based on local independence. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1): 245–264, 2008.

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

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

Feder, A., Keith, K. A., Manzoor, E., Pryzant, R., Sridhar, D., Wood-Doughty, Z., Eisenstein, J., Grimmer, J., Reichart, R., Roberts, M. E., et al. Causal inference in natural language processing: Estimation, prediction, interpretation and beyond. *Transactions of the Association for Computational Linguistics*, 10:1138–1158, 2022.

Frank, A. and Asuncion, A. UCI machine learning repository, 2010. URL http://archive.ics.uci.edu/ml.

Gao, T., Subramanian, D., Shanmugam, K., Bhattacharjya, D., and Mattei, N. A multi-channel neural graphical event model with negative evidence. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3946–3953, 2020.

Granger, C. W. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37: 424–438, 1969.

Grant, S. and Betts, B. Encouraging user behaviour with achievements: An empirical study. In *Proceedings of the IEEE Working Conference on Mining Software Repositories (MSR)*, pp. 65–68, 2013.

Gu, Y. Attentive neural point processes for event forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7592–7600, 2021.

Gunawardana, A. and Meek, C. Universal models of multivariate temporal point processes. In *Proceedings of Artificial Intelligence and Statistics*, pp. 556–563. PMLR, 2016.

Hassanzadeh, O. Building a knowledge graph of events and consequences using Wikidata. In *Proceedings of the 2nd Wikidata Workshop (Wikidata 2021) co-located with the 20th International Semantic Web Conference (ISWC 2021)*, 2021.

Heindorf, S., Scholten, Y., Wachsmuth, H., Ngonga Ngomo, A.-C., and Potthast, M. CauseNet: Towards a causality graph extracted from the Web. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, 2020.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. Stochastic variational inference. *Journal of Machine Learning Research*, 2013.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kitaev, N., Kaiser, L., and Levskaya, A. Reformer: The efficient transformer. In *International Conference on Learning Representations*, 2019.

Kratsios, A., Zamanlooy, B., Liu, T., and Dokmanić, I. Universal approximation under constraints is possible with transformers. *arXiv preprint arXiv:2110.03303*, 2021.

Leban, G., Fortuna, B., Brank, J., and Grobelnik, M. Event Registry: Learning about world events from news. In *Proceedings of the International Conference on World Wide Web*, 2014. doi: 10.1145/2567948.2577024. URL http://doi.acm.org/10.1145/2567948.2577024.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Maddison, C. J., Mnih, A., and Teh, Y. W. The concrete distribution: A continuous relaxation of discrete random variables. In *Proceedings of the International Conference on Learning Representations*, 2017.

Mannila, H., Toivonen, H., and Inkeri Verkamo, A. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1:259–289, 1997.

Mavroudeas, G., Neehal, N., Shou, X., Magdon-Ismail, M., Kuruzovich, J. N., and Bennett, K. P. Predictive modeling for complex care management. In *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 1443–1448. IEEE, 2021.

Mei, H. and Eisner, J. The neural Hawkes process: A neurally self-modulating multivariate point process. *arXiv preprint arXiv:1612.09328*, 2016.

Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., and Khudanpur, S. Recurrent neural network based language model. In *Interspeech*, volume 2, pp. 1045–1048. Makuhari, 2010.

Murphy, K. P. *Machine Learning: A Probabilistic Perspective*. MIT press, 2012.

Omi, T., Ueda, N., and Aihara, K. Fully neural network based model for general temporal point processes. *arXiv preprint arXiv:1905.09690*, 2019.

Pearl, J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 2014.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding by generative pre-training. 2018.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Sap, M., Le Bras, R., Allaway, E., Bhagavatula, C., Lourie, N., Rashkin, H., Roof, B., Smith, N. A., and Choi, Y. ATOMIC: An atlas of machine commonsense for if-then reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 3027–3035, 2019.

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

Tank, A., Li, X., Fox, E. B., and Shojaie, A. The convex mixture distribution: Granger causality for categorical time series. *SIAM Journal on Mathematics of Data Science*, 3 (1):83–112, 2021.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Vrandecic, D. and Krötzsch, M. Wikidata: A free collaborative knowledgebase. *Communications of ACM*, 57(10): 78–85, 2014.

Xiao, S., Yan, J., Yang, X., Zha, H., and Chu, S. M. Modeling the intensity function of point process via recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 1597–1603, 2017.

Xiao, S., Yan, J., Farajtabar, M., Song, L., Yang, X., and Zha, H. Learning time series associated event sequences with recurrent point process networks. *IEEE Transactions on Neural Networks and Learning Systems*, 30(10):3124–3136, 2019.

Xu, H., Luo, D., and Zha, H. Learning Hawkes processes from short doubly-censored event sequences. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 3831–3840, 2017.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Quoc, L. XLNet: Generalized autoregressive pretraining for language understanding. arxiv 2019; 1906.08237, 2020.

Yu, X., Shanmugam, K., Bhattacharjya, D., Gao, T., Subramanian, D., and Xue, L. Hawkesian graphical event models. In *International Conference on Probabilistic Graphical Models*, pp. 569–580. PMLR, 2020.

Yun, C., Bhojanapalli, S., Rawat, A. S., Reddi, S. J., and Kumar, S. Are transformers universal approximators of sequence-to-sequence functions? *arXiv preprint arXiv:1912.10077*, 2019.

Zerveas, G., Jayaraman, S., Patel, D., Bhamidipaty, A., and Eickhoff, C. A transformer-based framework for multivariate time series representation learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 2114–2124, 2021.

Zhang, C., Bütepage, J., Kjellström, H., and Mandt, S. Advances in variational inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):2008–2026, 2018.

Zhang, Q., Lipani, A., Kirnap, O., and Yilmaz, E. Self-attentive Hawkes process. In *Proceedings of the International Conference on Machine Learning*, pp. 11183–11193. PMLR, 2020.

Zhang, Y. and Yan, J. Neural relation inference for multi-dimensional temporal point processes via message passing graph. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 3406–3412, 2021.

Zuo, S., Jiang, H., Li, Z., Zhao, T., and Zha, H. Transformer Hawkes process. In *International Conference on Machine Learning*, pp. 11692–11702. PMLR, 2020.

Appendix for Probabilistic Attention-to-Influence Neural Models for Event Sequences

A. Probabilistic Attention-to-Influence Neural Model

A.1. Training Procedure

Training of the probabilistic attention-to-influence neural model is performed jointly on the encoder and decoder by mini-batch stochastic gradient descent given K training sequences $\{S_k\}_{i=1}^{K}$ and J dev sequences $\{S_j\}_{i=1}^{J}$ namely (S_{tr} and S_{dev}). Adam (Kingma & Ba, 2014) optimizer is used to train our model. We modify S_{dev} to $\widetilde{S_{dev}}$ so that only 2 types of events are observed for binary prediction: event of interest X and event of non-interest $\bar{X}$. We experiment with 4 variants: the prefix uniform and sparse represent the difference of prior setting, i.e. $p(\mathcal{V}_{XY}) = 0.5$ for uniform and $p(\mathcal{V}_{XY}) = 0.2$ for sparse, and the postfix means 2 samples $\mathcal{V}_j$ from the posterior $q_\phi(\mathcal{V}|\mathbf{D})$. Empirically we find 2 samples are efficient compared to other numbers while also achieving good performance. The two variants, uniform-τ and sparse-τ on the other hand, specify that instead of sampling, we manually place pre-specified thresholds τ's on the posterior. The procedure for Uniform-2 is fully described by Algorithm 1 for event X. For multiple events, we iterate through the list and train each model. In the case of Uniform-τ, $q_\omega(V|S') > \tau$ is used to assign $\mathcal{V}_f$ instead of sampling.

Algorithm 2 Training Pseudocode For Probabilistic Attention-to-Influence Neural Model Uniform-F

```
Input: Given Training data $S_{tr} = \{S_k\}_{k=1}^{K}$, Dev data $S_{dev} = \{S_j\}_{j=1}^{J}$, Sample size $F$, prior $p(\mathcal{V})$, mini-batch size $b$,
       Training epochs $N$, event of interest $X$, decay-rate $\gamma$
Output: Optimal parameters $\psi^*$, $\omega^*$
for $epoch \leftarrow 1$ to $N$ do
    for $iteration \leftarrow 1$ to $\lceil \frac{K}{b} \rceil$ do
        sample a batch of sequences $S'$ from $S_{tr}$
        compute $q_\omega(\mathcal{V}|S')$ via Equation 6
        sample $\mathcal{V}_1, ..., \mathcal{V}_F \sim q_\omega(\mathcal{V}|\mathcal{S}')$
        compute $\log\theta_\psi(S'|\mathcal{V}_f)$ via Equation 3
        compute ELBO $\mathbb{L}$ via Equation 1
        back-propagate with gradient $\nabla_{\theta,\phi}\mathbb{L}$
        update parameters of network $\psi, \omega$
    end
    evaluate $\frac{1}{F}\sum_{f=1}^{F} \log\theta_\psi(\widetilde{S_{dev}}|\mathcal{V}_f)$, stop training if not improving in 5 epochs
end
Return: Optimal parameters $\psi^*$, $\omega^*$
```

A.2. Hyperparameters

The multi-head attention modules in the encoder and decoder share the same weights and biases. The experiment setting for hyperparameters in Uniform-2 and Sparse-2 for binary prediction is given in Table 7. The τ values for Uniform-τ are $\{0.4,0.5,0.6\}$ and for Sparse-τ they are $\{0.1,0.2,0.3\}$. The best threshold for each model is chosen based on dev set. All our experiments are performed on a private server (https://idea.rpi.edu/IDEA_Cluster_Access) with TITAN RTX GPU.

Table 7. Hyperparameters for Uniform-2 and Sparse-2 for binary prediction. n_layers: the number of blocks for multi-headed self-attention module in both encoder and decoder; d_model: the dimension of the value vector after attention has been applied; n_head: the number attention heads; d_inner: the dimension of the hidden layer of the feed forward neural network; d_v: the dimension of the value vector; d_k : the dimension of the key vector; num_samples: the number of samples sampled from the approximate posterior; decay_rate: rate of attention score decay over time.

Parameter Value	Beige Books	Diabetes	LinkedIn	Stack Overflow	Timelines
batch_size	16	16	32	16	16
n_head	4	8	8	4	4
n_layers	4	4	4	4	4
d_model	512	128	256	512	512
d_inner	256	64	128	256	256
d_v	256	64	128	256	256
d_k	256	64	128	256	256
dropout	0.1	0.1	0.1	0.1	0.1
epoch	300	300	300	300	300
num_samples	2	2	2	2	2
learning_rate	0.0001	0.0002	0.00005	0.0001	0.00005
decay_rate	0	0	0	0	0

B. Baselines

B.1. Transformer Hawkes Process (THP)

THP in this study is a straight adaption from (Zuo et al., 2020) [1]. The embedding layer includes a time embedding and event-type embedding. While in our setting where no timestamps are given, we supply with regularly spaced times: [1,2,3,4,...,n], i.e. $t_j = j$. Time embedding is achieved through:

$$[z(t_j)]_i = \begin{cases} \cos(t_j/10000^{\frac{i-1}{d}}) & \text{if } i \text{ is odd} \\ \sin(t_j/10000^{\frac{i}{d}}) & \text{if } i \text{ is even} \end{cases} \tag{11}$$

where t_j is a timestamp and d is the dimension of encoding. Time embedding and one-hot encoded types are combined to form the embedded input $\mathbf{X}$. For sequence $S = \{t_i, y_i\}_{i=1}^{L}$, time embedding $\mathbf{z_i}$ for each instance is specified in Eq. 11 and for the entire sequence with length L, the embedding is $\mathbf{Z} \in R^{M\times L}$. Type embedding are through the product of a trainable embedding matrix $\mathbf{U} \in \mathbb{R}^{M\times K}$ and one hot encoded vectors $\mathbf{y_i}$'s for all type instances, i.e. $\mathbf{X} = (\mathbf{UY} + \mathbf{Z})^T$ where $\mathbf{Y} = [\mathbf{y_1}, \mathbf{y_2}, ..., \mathbf{y_L}]$. $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$ are query, key and value matrix; they are linear transformations of $\mathbf{X}$, i.e. $\mathbf{Q} = \mathbf{XW}^Q$, $\mathbf{K} = \mathbf{XW}^K$, $\mathbf{V} = \mathbf{XW}^V$ where $\mathbf{W}^Q$, $\mathbf{W}^K$, $\mathbf{W}^V$ are trainable weights. We optimize the same objective as implemented by the authors. A few recommended sets of hyperparameters are used for training and the best set is selected according to results on dev. The set of hyperparameters corresponding to the reported results is in Table 8.

Table 8. Hyperparameters for THP.

Parameter Value	Beige Books	Diabetes	LinkedIn	Stack Overflow	Timelines
batch_size	4	4	4	4	4
n_head	4	4	8	8	8
n_layers	4	4	4	1	2
d_model	64	64	64	64	64
d_inner	32	32	32	32	32
d_v	32	32	32	32	32
d_k	32	32	32	32	32
dropout	0.1	0.1	0.1	0.1	0.1
epoch	100	100	100	100	100
learning_rate	0.001	0.001	0.0002	0.0005	0.0005

[1]https://github.com/SimiaoZuo/Transformer-Hawkes-Process

B.2. Transformer for Next Token (TNT)

We implement with Pytorch a B-block attention-based transformer network to model the dynamics and seek to maximize the log-likelihood of event sequences $\mathbf{D}$ in Equation 12:

$$log\theta_{\psi}(\mathbf{D}) = \sum_{k=1}^{K}\sum_{i=1}^{N_k} log\theta_{\psi}(l_i) \tag{12}$$

We use an un-modified history representation $\mathbf{H}^{(B)}$ from the B^{th} block and the generated labels can be modeled via a multinomial distribution:

$$\theta_{\psi}(l_{i+1} = m|\mathbf{H}^{(B)}(i)) = \frac{\exp(\mathbf{W}_{m,:}\mathbf{H}^{(B)}(i) + \mathbf{b}_m)}{\sum_{m=1}^{M}\exp(\mathbf{W}_{m,:}\mathbf{H}^{(B)}(i) + \mathbf{b}_m)} \tag{13}$$

where $\mathbf{W}_{m,:}$ is the m^{th} row of the corresponding trainable weight matrix and $\mathbf{b}_m$ is m^{th} entry of the corresponding bias term. We perform prediction experiments and hyperparameters are selected from the best performing model from dev set. The hyperparameters of TNT model are shown in Table 9.

Table 9. Hyperparameters for TNT.

Parameter Value	Beige Books	Diabetes	LinkedIn	Stack Overflow	Timelines
batch_size	16	16	16	16	16
n_head	4	4	4	4	4
n_layers	4	4	4	4	4
d_model	512	256	256	512	256
d_inner	256	128	128	256	128
d_v	256	128	128	256	128
d_k	256	128	128	256	128
dropout	0.1	0.1	0.1	0.1	0.1
epoch	300	300	300	300	300
learning_rate	0.0001	0.0001	0.0001	0.0001	0.0001

B.3. Results for Prediction

We show more comprehensive prediction results in Table 10. We highlight that our model consistently improves over all baselines on most event datasets.

B.4. Event Registry

Event Registry is a dataset derived from a corpus of news article snippets from Event Registry (Leban et al., 2014). The corpus consists of 40 months of snippets where each is turned into a sequence of concepts from Wikidata (Vrandecic & Krötzsch, 2014) using keyword matching. The concepts are 50 event-related classes (event types). We only keep sequences of length 3 or more, which results in 15,821 sequences. We further split into train and dev set for our probablistic attention-to-influence neural model to learn. In particular we used Uniform-τ as our learner. We use the following set of (hyper)parameters batch_size = 16, n_head = 4, n_layers =2, d_model = 128, d_inner = 64, d_v = 64 , d_k = 64, dropout = 0.1, epoch = 300, learning_rate = 0.0001 and decay_rate = 0. Threshold τ = {0.4,0.5,0.6} are used and the best value is selected based on highest log-likelihood on dev set.

C. Results on Influencing Set Discovery

The BSuMM and OSuMM parametric models are learned using Algorithm 1 in (Bhattacharjya et al., 2022). The 2 models are used to identify the influencing events for each label of interest $\{B, C, D, E\}$ from the synthetic datasets. The learned influencing sets are compared to the ground truth parents in the underlying topology. We use the following for experiments: prior (Dirichlet) parameter $\alpha = 0.1$, penalty weight $\gamma = 1$, look-back κ is a hyper-parameter that is chosen using train/dev from $\{1, 5, 10\}$ positions.

Model/Dataset	Beige Books	Diabetes	LinkedIn	Stack Overflow	Timelines
0-MC	-118.46	-543.05	-133.41	-1367.31	-176.31
1-MC	-60.91	-513.01	-110.58	-1278.96	-154.47
2-MC	-40.37	-488.42	-112.55	-1283.66	-611.78
3-MC	-37.66	-473.96	-119.55	-1435.12	-1343.52
4-MC	-41.42	-535.66	-122.57	-1895.76	-1656.31
3-LR	-36.85	-506.05	-92.23	-1277.84	-160.84
5-LR	-36.15	-497.92	-93.37	-1263.54	-184.05
10-LR	-36.35	-497.94	-93.37	-1253.04	-200.55
BSuMM	-36.11	-497.90	-114.52	-1242.59	-141.42
OSuMM	-38.07	**-432.89**	-115.63	-1246.64	-142.18
LSTM	-63.65	-595.57	-135.92	-1246.45	-135.98
THP	-38.41	-433.69	-133.53	-1277.01	-131.53
TNT	**-20.38**	-453.53	-96.79	-120.41	-136.67
Uniform-τ	-21.54	-448.95	**-91.21**	-121.12	**-125.29**
Uniform-2	-21.06	-447.77	-95.96	-134.70	-135.30
Sparse$-\tau$	21.34	-448.18	-91.88	**-108.11**	-126.68
Sparse-2	-20.89	-464.19	-96.63	-121.83	-128.87

Table 10. Negative log loss (log likelihood), averaged over labels of interest, for various models computed on the test sets. k^{rd} order Markov chains (MC) range from $k = 0$ to 4, and logistic regression (LR) with a look-back of $k = 3$, $k = 5$ and $k = 10$ are shown. Best results are highlighted in Bold.

We identify influencing events with TNT models by postprocessing. Specifically we learn a TNT model from training set and obtain the attention score for instance-wise interaction from the last block in the attention module. Type-wise effects are obtained through averaging the attention scores over all attention heads and all sequences. An optimal threshold is obtained from type-wise effects on dev set and then is applied on test set to obtain an F1 score. We use the following set of (hyper)parameters: batch_size = 32, n_head = 4, n_layers =4, d_model = 128, d_inner = 64, d_v = 64 , d_k = 64, dropout = 0.1, epoch = 100 and learning_rate = 0.0001.

The proposed probabilistic attention-to-influence neural model estimates the approximate posterior $q_\omega(\mathcal{V}|\mathbf{D})$. An optimal threshold is obtained through dev set and then is applied on test set. The training and (hyper)parameter selection for the model can be found in Table 11. Uniform-τ uses Default setting for Graph-2 and the setting for event E in Graph-1 for all events in Graph-1. The results of the 3 models on Graph-2 are shown in Tables 4 and ours show improvements over baselines.

Table 11. Hyperparameters in Uniform-2 for influencing set recovery. Default Hyperparameters are used for inference on Graph-1 for event B,C and all events on Graph-2.

Parameter Value	Default	event D in Graph-1	event E in Graph-1
batch_size	32	32	32
n_head	4	4	4
n_layers	2	3	4
d_model	128	128	128
d_inner	64	64	64
d_v	64	64	64
d_k	64	64	64
dropout	0.1	0.1	0.1
epoch	100	100	100
num_samples	2	2	2
learning_rate	0.0001	0.0001	0.0001
decay_rate	5	1	1

D. Proofs

D.1. Proof of Theorem 3

We follow (Bhattacharjya et al., 2022) in setting up a contradiction, while considering dynamics for event label set $\mathbf{X}$ using summary function $s_\phi(\cdot)$. Specifically, we consider two distinct influencing sets $\mathbf{U}$ and $\mathbf{U}'$, and show that they cannot be both distinct and minimal. We use $\tilde{\Theta}_\mathbf{X}$ to denote the sum of probabilities over label set $\mathbf{X} \subseteq \mathcal{L}$, i.e. $\tilde{\Theta}_\mathbf{X} = \{\tilde{\theta}_{x|h}\}$ where $\tilde{\theta}_{x|h} = \sum_{X \in \mathbf{X}} \theta_{x|h}$.

Since $\mathbf{U}$ and $\mathbf{U}'$ are influencing sets, for all summary states $s_\mathbf{U}, s_{\mathbf{U}'} \in \mathbb{R}^d$, $\tilde{\theta}_{x|h} = \tilde{\theta}_{x|h'}$ for all h, h' consistent with $s_\mathbf{U}$ and $s_{\mathbf{U}'}$. Taking the intersection set $\mathbf{U}^* = \mathbf{U} \cap \mathbf{U}'$, since the sets are distinct, $\mathbf{U}^* \subset \mathbf{U}$ and $\mathbf{U}^* \subset \mathbf{U}'$. Note that applying the summary function to the smaller set $\mathbf{U}^*$ (which could potentially be the empty set $\emptyset$) instead of $\mathbf{U}$ or $\mathbf{U}'$ essentially reduces elements in the input historical sequence, due to the history restriction. Thus there are at least as many histories h consistent with a mapping $s_{\mathbf{U}^*} = s_\phi(h^{\mathbf{U}^*})$ rather than say $s_\mathbf{U} = s_\phi(h^\mathbf{U})$. Furthermore, if there are two histories h, h' consistent with some $s_\mathbf{U}$, they are consistent with $s_{\mathbf{U}^*}$, therefore $\tilde{\theta}_{x|h} = \tilde{\theta}_{x|h'}$ for all h, h' consistent with this $s_{\mathbf{U}^*}$. Since this is true for all summary states and $\mathbf{U}^* \subset \mathbf{U}$, $\mathbf{U}$ cannot be minimal and this sets up a contradiction.

D.2. Proof of Theorem 4.2

Without loss of generality, consider an attention-based transformer model for event sequences, i.e. TNT (see section B.2). Let the ψ_T^* denote the set of parameters that maximize the log-likelihood function 12. Let S be the typewise effects learned from the post-processing procedure, which converts instance-wise attention scores to by (summarization (Xiao et al., 2019) and then) averaging attention scores of B^{th} block over all sequences (Zhang et al., 2020) from the learned TNT model with ψ_T^*.

Our model can learn S with the following steps. The approximated conditional log-likelihood in Equation 3 in main text becomes deterministic, if we use a deterministic posterior $q_\omega(\mathcal{V})$ i.e. $q_\omega(\mathcal{V} = \mathbb{1}|\mathbf{D}) = 1$ (with $\mathbf{U} = \mathcal{L}$), $\mathbb{E}_{q_\omega} log\theta_\psi(\mathbf{D}|\mathcal{V}) = \sum_{k=1}^{K} \sum_{i=1}^{N_k} log\theta_\psi(l_i)$. The deterministic posterior $q_\omega(\mathcal{V} = \mathbb{1}|\mathbf{D}) = 1$ also implies we do not modify any attention scores by our construction of influence-to-attention, i.e. influence indicator $\mathbf{I} = \mathbb{1}$ in Equation 8 in main text. In addition, any prior $p(\mathcal{V})$ the $KL(q_\omega(\mathcal{V})|p(\mathcal{V}))$ term in Equation 1 in main text will be constant. Hence ψ_T^* also maximizes ELBO $\mathcal{L}$ given $q_\omega(\mathcal{V} = \mathbb{1}|\mathbf{D}) = 1$. For example, the same averaging scores S can be achieved by the following: let $W_q = 0$, b_q be a vector whose entries are very large and let the rest of parameters ω in encoder be arbitrary constants (by freezing during training), which ensures $q_\omega(\mathcal{V} = \mathbb{1}|\mathbf{D}) = 1$. We train our model and the learned optimal parameters ψ^* in the decoder can be the same as ψ_T^*, which gives the same post-processing averages S.

D.3. Proof Sketch of Theorem 4.3

The major difference between our model and other transformer based baselines is usage of the attention-to-influence and influence-to-attention mechanism. The additional complexity is from the computation of equations (4), (5), (6), (7) and (8). Let $n = max_k(N_k)$, the maximum length of all the sequences and m be total number of labels. The computational complexities for (4), (5), (6), (7) and (8) respectively are $\mathcal{O}(n^2)$, $\mathcal{O}(mn^2 + m^2n)$, $\mathcal{O}(m)$, $\mathcal{O}(mn^2 + m^2n)$ and $\mathcal{O}(n^2)$. Hence (5) and (7) dominate over other terms. Consider one layer of attention block. Linearly transforming the rows of an embedding X to compute the query Q, key K, and value V matrices, each of which has shape (n, d) where d is the dimension of embedding, amounting to a computational complexity of $\mathcal{O}(nd^2)$. Computing a typical attention score matrix and post-multiplying a value matrix (Equation (1) in (Vaswani et al., 2017)) results in a computational complexity of $\mathcal{O}(n^2d)$. Hence a typical layer involves $\mathcal{O}(nd^2 + n^2d)$. m is typical smaller than d or n, as in a typical recurrent multivariate event sequence as shown in Tables 6, 10, 11 in the Appendix. In addition, our encoder and decoder share weights. Thus our model roughly has the same computational complexity as other transformer baselines.

The runtime compute from our model is comparable to baseline transformer models. This is due to the fact that the additional number of FLOPS introduced in our model are dominated by equation (7), since we only need to compute (5) from the last layer and need to compute for (7) for each layer. Consider a typical dot-product attention (i.e., Equation (1) in (Vaswani et al., 2017)). Such a computation requires $4dn^2 - nd \approx 4dn^2$ FLOPS where d is the hidden dimension; and computing equation (7) requires $2m^2n - mn + 2mn^2 - n^2 \approx 2m^2n + 2mn^2$. A typical dataset in our case involves m being on the order of 10, d 100 and n 100, which results in computation of attention values being much more expensive. This should be true for both forward pass and back-propagation.

The memory requirements for our model are comparable to any transformer model since a few additional quantities are of much smaller order of magnitude compared to the transformer model. For example, in practice, we only need to store a few additional terms: $exp^{-\gamma\Delta T}$ (n $\times$ n) from equation (4) and P (n $\times$ m) matrix from (5), $\mathcal{V}_j$ (m $\times$ m) and $\tilde{\mathcal{V}}_j$ (m $\times$ m), and other associated quantities in (4)-(8) can be derived. In practice, for $exp^{-\gamma\Delta T}$ we only need to store one row ($n \times 1$), because such decay pattern is universal to all event instances. Consider a typical n being an order of magnitude larger than m. Then memory requirement for storing the above items will be dominated by attention matrix ($n \times n$) which is standard in the transformer models.

E. Brief Discussion on Assumption and Limitation

Our assumptions are clearly described in Sections 4.1 and 4.2, i.e. temporal order of event sequences and relation between attention and influence. We discuss limitations in Section 6, that our model should not be interpreted to draw causal conclusion. In this paper we have already mitigated some of the risk by stressing use of the term ”influencers” instead of ”causes” and in practice, we would add even more verbiage and reasonably detailed instructions and caveats to avoid misuse. To further mitigate the risk of harmful causal conclusions, one should carefully conduct controlled experiment to verify and validate the causal claims, if possible. A future research direction is to establish a solid framework for causal inference (definitions of treatment, covariates, outcomes and counterfactual distributions, and etc.) in temporal event sequences similar to the work proposed in natural language sequences (Feder et al., 2022). Such work could be beneficial for drawing causal conclusions for temporal event sequences, under some assumptions. To process and model long sequences more efficiently, our model relies on memory-efficient transformers. Alternatively, to optimize efficiency, we can employ a sparse (self) attention module. This approach can maintain comparable performance on prediction tasks while also preserving the performance on influencing set identification. The monotonic nature of the sigmoidal transformation in Equation (6) ensures the retention of the relative order of influencing events, contributing to the preservation of performance in this aspect.