Graphical Models Meet Temporal Point Processes

Debarun Bhattacharjya[1] Abir De[2]
Tian Gao[1] Søren Wengel Mogensen[3]

[1]IBM T. J. Watson Research Center, NY, USA

[2]Indian Institute of Technology, Mumbai, India

[3]Lund University, Sweden

UAI 2022 Tutorial, August 1, 2022

{debarunb,tgao}@us.ibm.com, *abir@cse.iitb.ac.in*,
soren.wengel_mogensen@control.lth.se

Events are everywhere!

Event Datasets

- web logs
- customer transactions
- financial events
- insurance claims
- brain activity neural spikes
- social network messages
- ...

Applications

- preventive maintenance
- health outcome prediction
- scientific discovery
- knowledge discovery
- information diffusion
- recommendation systems
- ...

Motivating analyses

DESCRIPTIVE

What event types ***directly influence*** the occurrence of event type X?
What ***order of events*** makes event type X more (or less) likely to occur?
What is a measure of the ***pairwise causal relationship*** b/w event types Y and X?

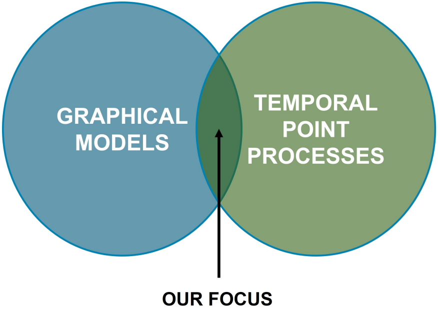
PREDICTIVE

What event type will occur next?
When will the next event happen?
How many events of type X will happen in the next month?

PRESCRIPTIVE

Given history, ***what action*** maximizes expected rewards from future events?
What ***would have happened*** if event type Y had happened (or not) in the past?

Scope



What is covered

- Foundations of graphical models of TPPs
- Learning graphical models of TPPs
- TPPs on network data

What is not covered

- Causal models
- Continuous-time reinforcement learning
- …

Agenda

- Background on Temporal Point Processes
- Graphs and Temporal Point Processes
- Parametric Graphical Event Models
- —— BREAK ——
- Neural Temporal Point Processes
- Temporal Point Processes on Network Data
- —— DISCUSSION ——

Background on TPPs

Overview

In this part of the tutorial, we'll

- introduce event data sets and temporal point processes,
- introduce graphs and local independence.

This is a general and nonparametric approach to graphical modeling of temporal point processes. Later parts will look at parametric models.

Event data

An *event data set* is a collection $D = \{(l_k, t_k)\}_{k=1}^{N}$ where

- t_k is the *event time* of the k^{th} event, $t_{k_0} \leq t_{k_1}$ for $k_0 \leq k_1$,
- l_k is the *label* of the k^{th} event, $l_k \in \mathcal{L} = \{1, \ldots, M\}$.

We will write $\{(L_k, T_k)\}_{k \geq 1}$ for the corresponding random variables.

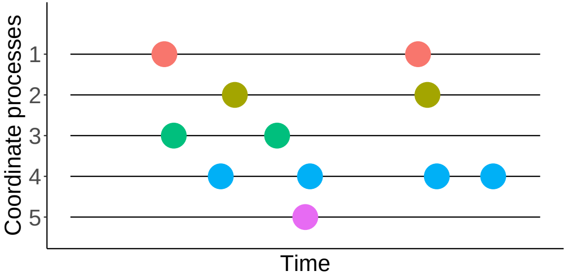
Examples

Illustration on a single time line of data set with three coordinate processes/event types ($M = 3$),

$$D = \{(A, 2), (B, 3), (C, 4), (B, 6), \ldots\}.$$

Examples

Illustration of event data set with five coordinate processes/event types where vertical placement represents coordinate process/event type ($M = 5$),



Temporal point processes (TPPs)

Event data sets can be modelled using *(temporal) point processes*. A (multivariate) point process, $X_t = (X_t^1, \ldots, X_t^M)$, is a stochastic process and

$$X_t^i = \sum_{k \geq 1, L_k = i} \delta(t - T_k).$$

We identify each $i \in \mathcal{L}$ with a *coordinate process*, X_i. One can specify a distribution of the point process using the *conditional intensities*, λ_t^i. These are themselves stochastic processes and for each time point t

$$\lambda_t^i = \lim_{h \downarrow 0} P(\text{an event of type } i \text{ occurs in } (t, t+h] \mid \mathcal{H}_t)$$

where $\mathcal{H}_t$ is a σ-algebra generated by the evolution of the process until time t.

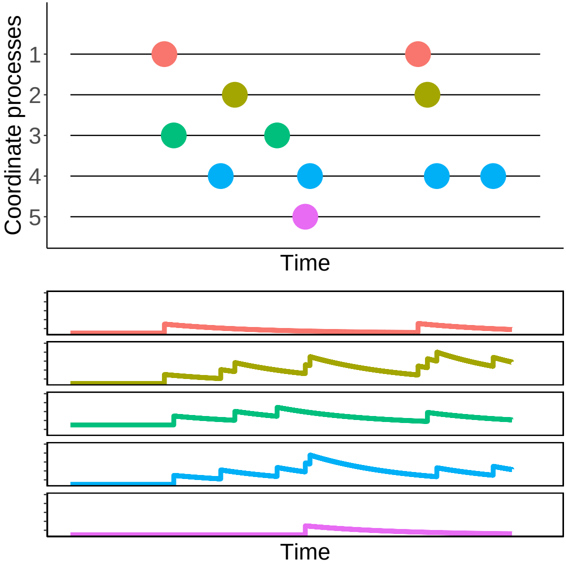
Conditional intensities (Hawkes process)

As an example of how to specify the distribution of a point process using the conditional intensities, we consider the *(linear) Hawkes process*.

$$\lambda_t^j = \mu_j + \sum_{i \in \mathcal{L}} \left(\sum_{\substack{k: T_k < t, \\ L_k = i}} f^{ji}(t - T_k) \right)$$

where μ_i are nonnegative constants and f^{ji} are nonnegative functions.

Conditional intensities (Hawkes process)

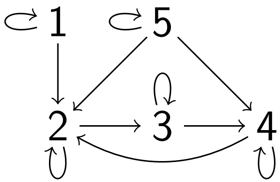


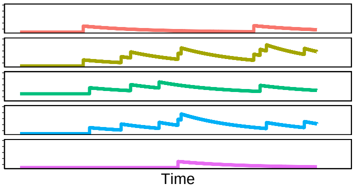
Graphs and TPPs

Graphs

We will use a *directed graph*, $\mathcal{G} = (\mathcal{L}, \mathcal{E})$, to represent sparsity in how coordinate processes influence each other.

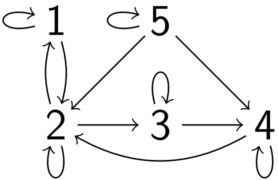
- $\mathcal{L}$ is the node set (same as the label set/index set of the coordinate processes).
- $\mathcal{E}$ is a set of edges, that is, ordered pairs, (i, j), such that $i, j \in \mathcal{L}$.





Graphs

A *walk* is an alternating sequence of adjacent nodes and edges.
A *path* is a walk such that no node is repeated.



- $1 \leftarrow 2 \rightarrow 3 \rightarrow 3$ is a walk
- $1 \leftarrow 2 \rightarrow 3$ is a path

As there may be multiple edges between a pair of nodes, a sequence of nodes does not define unique walk in itself.

Local independence

Definition

Let $A, B, C \subseteq \mathcal{L}$. We say that B is *locally independent* of A given C, and write $A \not\rightarrow_\lambda B \mid C$ if for all $i \in B$, $E(\lambda_t^i \mid \sigma(X_{0:t}^{A \cup C}))$ does not depend on tracks in A.

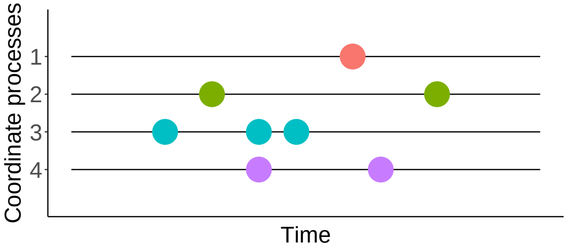
Local independence has been studied by, e.g., Schweder (1970), Aalen (1987), and Didelez (2008). One can also define local independence in other classes of processes, see e.g. (Commenges and Gégout-Petit, 2009; Mogensen, Malinsky, and Hansen, 2018). It is similar to Granger causality in (discrete-time) time series.
Local independence is a *ternary relation*, analogous to conditional independence of random variables. However, local independence is *asymmetric*,

$$A \not\rightarrow_\lambda B \mid C \not\Rightarrow B \not\rightarrow_\lambda A \mid C$$

Local independence

Definition

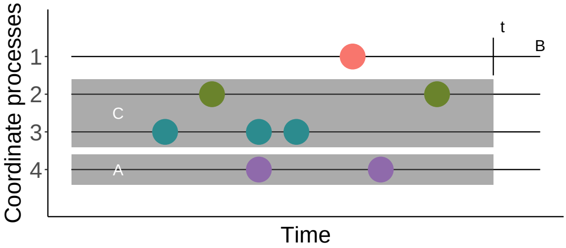
Let $A, B, C \subseteq \mathcal{L}$. We say that B is *locally independent* of A given C, and write $A \not\rightarrow_\lambda B \mid C$ if for all $i \in B$, $E(\lambda_t^i \mid \sigma(X_{0:t}^{A \cup C}))$ does not depend on tracks in A.



Local independence

Definition

Let $A, B, C \subseteq \mathcal{L}$. We say that B is *locally independent* of A given C, and write $A \not\rightarrow_\lambda B \mid C$ if for all $i \in B$, $E(\lambda_t^i \mid \sigma(X_{0:t}^{A \cup C}))$ does not depend on tracks in A.



Local independence graphs

Given a stochastic process, we define its *local independence graph* to be the *directed graph* (DG), $\mathcal{G} = (\mathcal{L}, \mathcal{E})$, such that for $i, j \in \mathcal{L}$

$$i \not\to_{\mathcal{G}} j \Leftrightarrow i \not\to_{\lambda} j \mid \mathcal{L} \setminus \{i\}$$

The implication from left to right is the *pairwise Markov property* ($i \not\to_{\mathcal{G}} j$ denotes that there is no edge from i to j in $\mathcal{G}$).

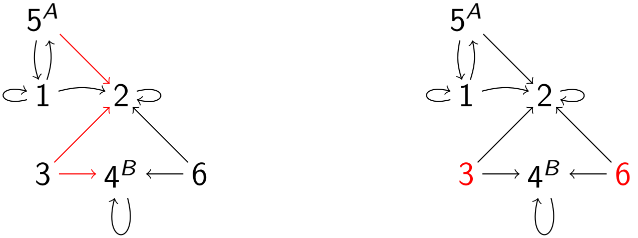
Intuitively, the edge $i \to_{\mathcal{G}} j$ is omitted if what happens at time t in process j does not depend (directly) on the past of process i.

δ-separation

δ-separation is a graphical separation criterion, analogous to d-separation in DAGs. δ-separation from A to B given C for disjoint sets $A, B, C \subseteq \mathcal{L}$ occurs when a certain kind of walk is absent in the graph. The most important difference from d-separation is the fact that only walks with a *head* at j can be connecting from i to j given some set C. We will just give some examples.

δ-separation

δ-separation is a graphical separation criterion, analogous to d-separation in DAGs. δ-separation from A to B given C for disjoint sets $A, B, C \subseteq \mathcal{L}$ occurs when a certain kind of walk is absent in the graph.



Left: a walk (in red) which is δ-connecting from 5 to 4 given $C = \{2\}$, and not δ-connecting given $C = \{3\}$.
Right: $A = \{4\}$ is δ-separated from $B = \{5\}$ by any C such that $\{3, 6\} \subseteq C$.

Markov properties

Under some regularity conditions, the *global Markov property* holds (Didelez, 2008; Mogensen, Malinsky, and Hansen, 2018).
If B is δ-separated from A given C in the graph $\mathcal{D}$, then we write $A \perp_\delta B \mid C \ [\mathcal{D}]$. δ-separation is not symmetric.

Theorem (The global Markov property)

Let X be a TPP and let $\mathcal{D}$ be its local independence graph. Let $A, B, C \subseteq V$. Then

$$A \perp_\delta B \mid C \ [\mathcal{D}] \Rightarrow A \not\to_\lambda B \mid C.$$

This gives a connection between TPPs and their local independence graphs.

More general graphs and structure learning

- *Directed mixed graphs* (include bidirected edges $\leftrightarrow$ as well as directed edges) and μ-*separation* allow graphical marginalization to model partially observed systems (Mogensen and Hansen, 2020).
    - Analogous to MAGs and ADMGs with *m*-separation in DAG-based models.
- One can learn (marginalized) local independence graphs based on tests of local independence (Meek, 2014; Mogensen, Malinsky, and Hansen, 2018; Christgau, Petersen, and Hansen, 2022).
    - Analogous to structure learning in DAG-models based on tests of conditional independence.

Parametric Graphical Event Models

Overview

(Dynamic) Graphical Models

- Discrete-time
  - Dynamic Bayes nets
  - Time series graphs
- Continuous-time
  - Continuous-time Bayes nets
  - Local independence graphs/graphical event models

Parametric (Multivariate) TPPs

The literature makes various assumptions about history dependence. Examples:

- Poisson networks (Rajaram, Graepel, and Herbrich, 2005)
- Piecewise-constant intensity models (Gunawardana, Meek, and Xu, 2011)
- Multivariate Hawkes processes (Zhou, Zha, and Song, 2013)
- Proximal GEMs (Bhattacharjya, Subramanian, and Gao, 2018).
- Ordinal GEMs (Bhattacharjya, Gao, and Subramanian, 2020).

PGEM: Preliminaries

- Event dataset $\mathbf{D} = \{(l_i, t_i)\}, i = 1, \cdots, N; l_i \in \mathcal{L}, |\mathcal{L}| = M$, where t_i are assumed temporally ordered b/w 0 and T.
- Inter-event times b/w events labels Z and X are denoted $\hat{t}_{zx}$ for $Z \neq X$; $\hat{t}_{xx}$ for $Z = X$ includes period at the end.

Example

- $M = 3$ labels; $N = 7$ events
- $\{\hat{t}_{ac}\} = \{2, 8\}$; $\{\hat{t}_{bc}\} = \{1, 7\}$; $\{\hat{t}_{bb}\} = \{3, 7, 7\}$

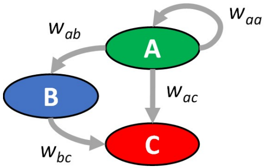
PGEM: Formulation

Definition

A proximal graphical event model includes:

- A graph $\mathcal{G}$ with a node for each label X in $\mathcal{L}$.
- A window for every edge: $\mathcal{W} = \{w_x : \forall X\} = \{w_{zx} : \forall Z \in \mathbf{U}\}$.
- An intensity parameter for every node X and instantiation $\mathbf{u}$ of its parents' occurrences, $\Lambda = \{\lambda_{x|\mathbf{u}}^{w_x} : \forall X \in \mathcal{L}\}$.

Assumption: A label's intensity depends on whether its parents have occurred at least once in their respective recent (i.e. proximal) histories.

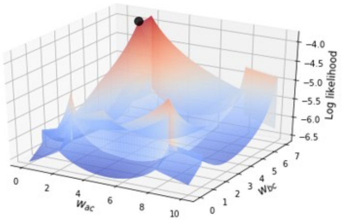


PGEM: Score-based Learning (1 of 3)

Given graph $\mathcal{G}$ and windows $\mathcal{W}$:

- $LL = \sum_X \sum_{\mathbf{u}} \left(-D(\mathbf{u})\lambda_{x|\mathbf{u}} + N(x,\mathbf{u})\log\left(\lambda_{x|\mathbf{u}}\right)\right)$, where:
    - $N(x,\mathbf{u})$: # of times X occurs and condition $\mathbf{u}$ is true
    - $D(\mathbf{u})$: duration over the horizon where condition $\mathbf{u}$ is true
- $BIC = LL - \log(T)\sum_X 2^{|\mathbf{U}|}$ (Score decomposes!)
- Max. likelihood estimates: $\hat{\lambda}_{x|\mathbf{u}} = \frac{N(x,\mathbf{u})}{D(\mathbf{u})}$

Given $\mathcal{G}$, finding the optimal $\mathcal{W}$ is a combinatorial problem!



PGEM: Score-based Learning (2 of 3)

Theorem

For a node X with single parent Z, the log likelihood maximizing window w_{zx} either belongs to or is a left limit of a window in the candidate set $\mathcal{W}^* = \{\hat{t}_{zx} \cup max\{\hat{t}_{xx}\}\}$.

- Intuition: the optimal is at (or limit to) points where the counts $N(x, \mathbf{u})$ change.

Theorem

Using BIC as score, if $2^{\mathbf{U}} > \frac{N(x)(1-\log N(x))}{\log T}$ for parents $\mathbf{U}$ of X then no proper superset of $\mathbf{U}$ can be X's optimal parents.

- Could help with efficient parent set, similar to Bayes nets (Campos and Ji, 2011).

PGEM: Score-based Learning (3 of 3)

Learning Problem: Given event dataset **D**, learn PGEM $\{\mathcal{G}, \mathcal{W}, \Lambda\}$.

Outer loop performs graph search

- Score-based forward backward search for parents **U** for every event label X

Inner loop computes:

- "Optimal" windows using one of two heuristics
- LL, score and MLE estimates for intensity rates

Heuristics:

- **FBS-IW** (independent windows)
- **FBS-CW** (conditional windows)

PGEM: Constraint-based Learning

Recent work (Bhattacharjya et al., 2022) considers testing for process independence, analogous to methods that test for conditional independence in Bayes nets (Spirtes, Glymour, and Scheines, 2000).

Algorithm 1 PC Algorithm for Parent Discovery in GEMs

Inputs: Event label $X \in \mathcal{L}$, event dataset D (over $\mathcal{L}$), threshold parameter for tester α
Outputs: Parents $\mathbf{U}$ for X

$\mathbf{U} = \mathcal{L}$
for all Y in $\mathcal{L}$ **do**
  flag = False, $n = 0$, $\mathbf{Z}^* = \mathbf{U} \setminus Y$
  while $n \leq |\mathbf{Z}^*|$ and flag = False **do**
    for all $\mathbf{Z}$ that are subsets of size n in $\mathbf{Z}^*$ **do**
      Obtain score from a process independence test, checking if $Y \not\rightarrow X|\mathbf{Z}$
      if score $\leq \tau = f(\alpha)$ (indicating process independence) **then**
        flag = True, $\mathbf{U} = \mathbf{U} \setminus Y$, break from loop
    $n = n + 1$

This assumes we have access to a **process independence tester!**

OGEM: Preliminaries

Definitions

- A masking function $\phi(\cdot)$ takes a sequence of events and returns a sub-sequence where a label is never repeated.
- An order instantiation for labels $\mathbf{Z}$ is a permutation of any subset, obtained at time t by applying $\phi(\cdot)$ to events from $\mathbf{Z}$ occurring within the interval $[\max\{t-w, 0\}, t]$.

Example

The figure below shows order instantiations at occurrences of C with respect to labels $\{A, B\}$ using window $w = 5$.

Tabular OGEM

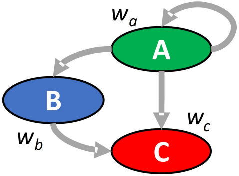
Definition

A tabular graphical event model with $\phi(\cdot)$ includes:

- A graph $\mathcal{G}$ with a node for each label X in $\mathcal{L}$.
- A window for every node: $\mathcal{W} = \{w_x : \forall X \in \mathcal{L}\}$.
- An intensity parameter for every node X and order instantiation $\mathbf{o}$ of its parents' occurrences, $\Lambda = \{\lambda_{x|\mathbf{o}}^{w_x} : \forall X\}$.

Limitations:

- $\#$ of order instantiations are super-exponential in $|\mathbf{U}|$.
- Complex models are hard to learn.
- Not all order instantiations will be observed in the data.



OGEM: Tree Representation

Basic idea: Make some order instantiations share parameters!

Definition

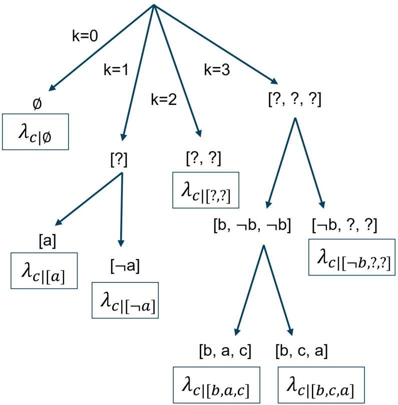
An order representation $\mathbf{r}$ of length $k < |\mathbf{U}|$ for a set of labels $\mathbf{Z}$ is a sequence of slots that are either filled with a label in $\mathbf{Z}$ or restricted by a subset of $\mathbf{Z}$. $\mathbf{r}$ is feasible if consistent with at least one $\mathbf{o}$.

Example

Consider $k = 2$ size orders for $\mathbf{Z} = \{A, B, C\}$.

- Ex #1: $r = [A, \neg A]$ encodes $[A, B]$ and $[A, C]$.
- Ex #2: $r = [?, ?]$ encodes all 6 permutations of pairs in $\{A, B, C\}$.

Illustrative parameter tree for event label C, parents: $\{A, B, C\}$



OGEM: Learning

Learning Problem: Given event dataset **D** and masking function $\phi(\cdot)$, learn $\{\mathcal{G}, \Lambda\}$ for OGEM.

- OGEM-tree: Learn OGEM tree with $\mathcal{W}$ given.
- OGEM-tree-W: As above, but also learn windows $\mathcal{W}$.

Outer loop performs graph search

- Score-based forward backward search for parents **U** for every event label X

Inner loop learns the tree representation, given parents

- Loop over all possible k from 0 to |**U**|
- Learn the optimal subtree for each k by splitting at each node

procedure OPTSUBTREE(event label X, parents $\mathbf{U}$, window w_X, masking function $\phi(\cdot)$, dataset $\mathcal{D}$, subtree length k)
  Initialize representation list $\mathcal{R}$, tree $\mathcal{T}_k$ and model information for representations $\mathcal{I}$ as empty
  Set root of subtree as $r = [?, ?, \cdots]$ (k times)
  Add r to list $\mathcal{R}$ and tree $\mathcal{T}_k$
  Compute all model information (summary stats, lambdas, LL and score) for the root; store in $\mathcal{I}$
  while $\mathcal{R}$ not empty **do**
    Choose any representation r in $\mathcal{R}$ and determine all feasible splits by filling a single slot
    for both children r_C in each feasible split of r **do**
      if $r_C \in \mathcal{I}$ **then**
        Retrieve model information from $\mathcal{I}$
      else
        Compute all model information; store in $\mathcal{I}$
    Consider feasible split with maximum total score
    if feasible split and score improvement from this split over parent > 0 **then**
      Make parent r an internal node of tree $\mathcal{T}_k$
      Add both r_C from this split to list $\mathcal{R}$
    else
      Remove parent r from list $\mathcal{R}$; make it a leaf node
  return Optimal sub-tree $\mathcal{T}_k$ for this k; Model info. $\mathcal{I}$

Empirical Investigation

Task: To compare models around fitting event datasets.

Datasets:

- ICEWS [politics]
- IPTV [TV viewership]
- LinkedIn [employment]
- Mimic [healthcare]
- Stack Overflow [online engagement]

Dataset	N (# events)	M (# labels)	MHP	PGEM	OGEM-tab	OGEM-tree	OGEM-tree-W
ICEWS							
Argentina	3252	104	-1419	-1386	-1369	**-1366**	-1393
Brazil	4249	114	-2169	-2000	-2057	-2050	**-1993**
Colombia	841	79	-528	-534	**-518**	**-518**	-537
Mexico	1905	97	**-760**	-797	-771	-769	-766
IPTV	332980	16	**-64168**	-77009	-75114	-72696	-74491
LinkedIn	2932	10	-1593	-1462	-1478	-1418	**-1406**
Mimic	2419	75	-567	-500	-474	**-429**	-454
Stack Overflow	71254	22	-52543	-48323	-49344	-49192	**-48232**

Table 1: Log likelihood on the test sets.

Methodology:

- Metric: Log likelihood; (70/15/15)% split for train/dev/test sets
- Baselines: multivariate Hawkes process, PGEM, tabular OGEM

Results: OGEM-tree models fit data reasonably compared to baselines.

Neural TPPs

Deep Learning

Advances state-of-the-art performances in numerous tasks and applications.

- e.g., computer vision, NLP, robotics, healthcare, chemistry, astrophysics ...

Advantages:

- Universal function approximator
- Scaling to billions of parameters, with modern computation tools such as GPUs

Parametric GEMs vs. Neural Point Processes

Parametric GEMs

Makes various assumptions about historical dependence + irregular time dynamics:

- Hawkes
- Proximal
- Basis functions

Neural Point Processes

Less restrictive assumptions:

- Base dynamic model: RNN, Transformers
- Irregular dynamics:
  - Hawkes
  - Sampling
  - Integral

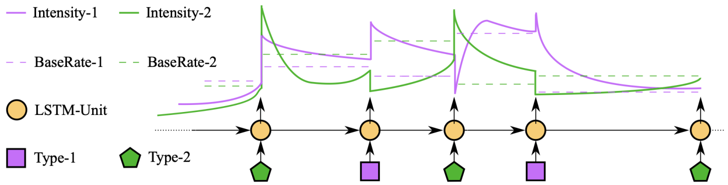
Into to Neural Point Process

Key Ideas:

- Dynamic: to use a RNN/LSTM cell to automatically learn the historical state h, with $\lambda_k(t) = f_k(\mathbf{W}_k^T \mathbf{h}(t))$.

$$\mathcal{L}(D) = \sum_{i}^{N} \log \lambda_{k_i}(t_i) - \int_{t=0}^{T} \lambda(t)dt$$

- Evolution: to let the hidden state continuous evolve (exponentially) at some rate λ_k toward a steady state value



Du et al. (2016) and Mei and Eisner (2016)

Neural Point Process with (some) Graph

How could we extract a graphical representation of (causal) relationships between different events?

- Difficulty: neural network + time-dependent function

Existing literature focus on two approaches:

1. Attention mechanism
2. A dedicated set of parameters (e.g., a binary gating matrix)

Neural TPPs with (some) Graph

Approach 1:
Attention mechanism is used to compute graphical relationships.

Approach 2:
A dedicated set of parameters is used to model the graph in neural TPPs.

Attention: A Brief Review

Original Attention

Given a current state h_i and several past states h_j

1. Alignment: compute $e_{ij} = f(h_i, h_j)$
2. Weight: $\alpha_{i,j} = softmax(e) = \frac{\exp e_{ij}}{\sum_j \exp e_{ij}}$
3. Context: $c_i = \sum_j \alpha_{ij} h_j$

Bahdanau, Cho, and Bengio (2014)

General Attention

3 components: query q, key k, and values v

1. Alignment: compute $e_{q,k_j} = f(q, k_j)$
2. Weight: $\alpha_{q,k_j} = softmax(\frac{e}{\sqrt{|k|}})$
3. Attention: $\text{Att}(q, k, v) = \sum_j \alpha_{q,k_j} v_j$

Vaswani et al. (2017)

Attention: A Brief Review

Original Attention

Given a current state h_i and several past states h_j

1. Alignment: compute $e_{ij} = f(h_i, h_j)$
2. Weight: $\alpha_{i,j} = softmax(e) = \frac{\exp e_{ij}}{\sum_j \exp e_{ij}}$
3. Context: $c_i = \sum_j \alpha_{ij} h_j$

Bahdanau, Cho, and Bengio (2014)

General Attention

3 components: query q, key k, and value v

1. Alignment: compute $e_{q,k_j} = f(q, k_j)$
2. Weight: $\alpha_{q,k_j} = softmax(\frac{e}{\sqrt{|k|}})$
3. Attention: $\text{Att}(q, k, v) = \sum_j \alpha_{q,k_j} v_j$

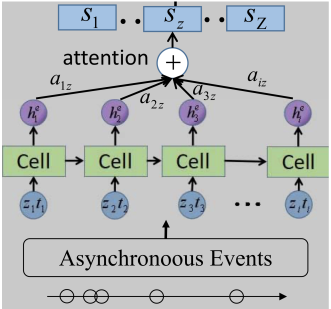
Vaswani et al. (2017)

1/3: Recurrent Point Process Network

Three parts: RNN + Dynamic Decaying + Attention

$$\alpha_{z_i,z} = \text{softmax}(f(h_i^e, u_z)) \Rightarrow G_{k,k'} = \text{mean}(\alpha_{z_i,z})$$

$$s_z(t) = \sum_i \alpha_{z_i,z} h_i^e \exp(-w(t - t_i)), \quad \lambda_z(t) = f(s_z(t))$$



Xiao et al. (2017) and Xiao et al. (2019)

2/3: Multi-Channel Neural GEM

Beyond exponential functions:

- Piece-wise constant function
- Time lags with delayed excitation or inhibition
- Varying time scales among events

Key ideas of MCN-GEM:

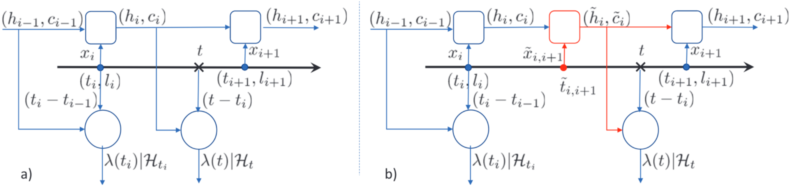
- Nonparametric: utilize time intervals between event arrivals to sample negative evidence
- Spatio-temporal attention

Gao et al. (2020)

2/3: Multi-Channel Neural GEM

RNN + sampling negative evidence (non-event occurrences)

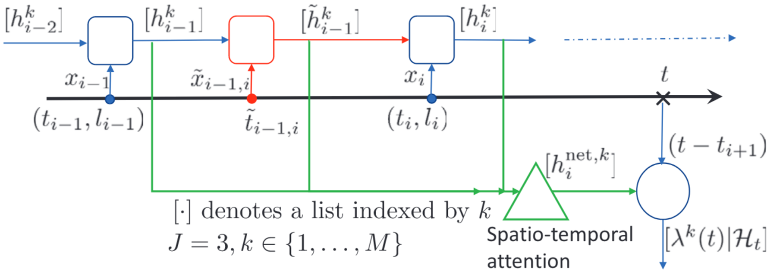
$$\mathcal{L}(D) = \sum_{i}^{N} \log \lambda_{k_i}(t_i) - \sum_{i}^{N+1} \Delta t_i \sum_{k} \lambda_{k_i}(t_i)$$



Gao et al. (2020)

2/3: Multi-Channel Neural GEM

$+$ spatio-temporal attention $\alpha \in R^{J \times K} \Rightarrow G_{k,k'} = \frac{\sum_i^T \sum_j^J \alpha_{ijk'}^k}{|T||J|}$



Gao et al. (2020)

General Attention

Limitations of RNN

- Difficulty to capture the long-term and/or non-sequential dependencies.
- In-efficiency in training and hard to parallel.

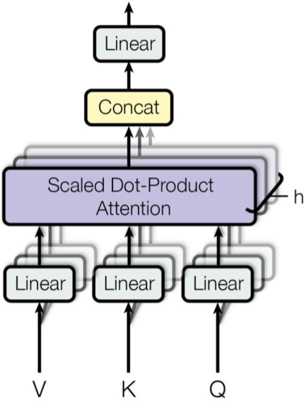
Multi-headed Attention

- $\mathrm{Att}(q, k, v) = \mathrm{softmax}(\frac{qk^T}{\sqrt{|k|}})v$
- concatenation + combination of multiple different attentions

Transformer-based TPPs (Zhang et al., 2020a; Zuo et al., 2020; Gu, 2021)

Multi-Headed Attention: A Review

Attention is all you need...

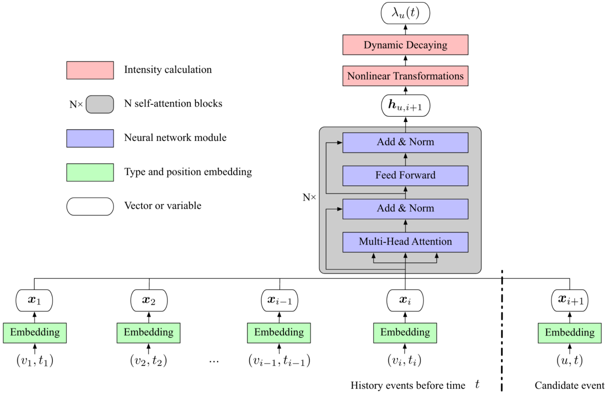


Vaswani et al. (2017)

3/3: Self-Attentive Hawkes Models

Attention is all you need (to extract graphs)...

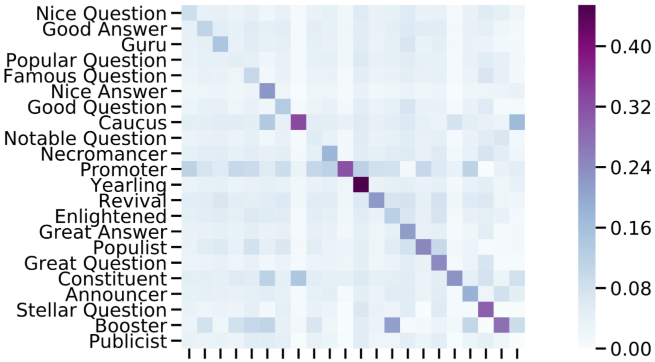
- Attentions + Dynamic Decaying



Zhang et al. (2020a)

3/3: Self-Attentive Hawkes Models

SAHP: similar with a single attention but now with multiple.



Zhang et al. (2020a)

Neural TPPs with Explicit Graph Modeling

Approach 1:
Attention mechanism is used to compute graphical relationships.

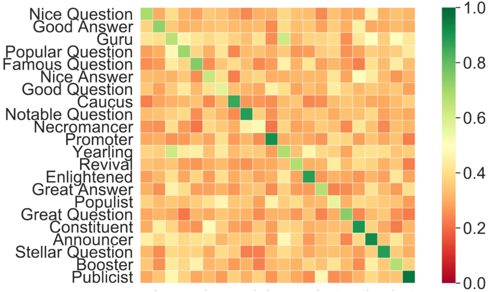
Approach 2:
A dedicated set of parameters is used to model the graph in neural TPPs.

1/2: Learning Neural Point Processes with Latent Graphs

Modify SAHP attention score with explicit graphs

$$\alpha(h_i, h_j) = \mathcal{G}_{(k,k')} \exp(h_i^T h_j)$$

where $G_{k,k'} \sim \text{Ber}(k, k')$



Zhang, Lipani, and Yilmaz (2021)

2/2: Causality from Attributions on Sequence of Events

Explicit Graph Model with Attribution

Definition

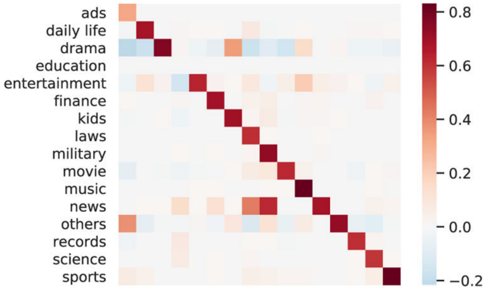
Attribution $A_j(f_k, x_i, \underline{x}_i)$ is defined as the event contribution of the j-th event to the target prediction $f_k(x_i)$ relative to the baseline $f_k(\underline{x}_i)$.

- Base dynamic model: GRU
- Irregular time: semi-parametric weighted Gaussian basis functions
- f: cumulative intensity function

Zhang et al. (2020b)

2/2: Causality from Attributions on Sequence of Events

$$A_{k,k'} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{i} I(k_j^s = k') A_j(f_k, x_i, \underline{x}_i)}{\sum_j^n I(k_j^s = k')}$$



Zhang et al. (2020b)

Summary

- Model and representation
  - Dynamics: RNNs and Transformers
  - Irregular Dynamics: parametric and non-parametric
- Graph representation
  - Attention, multi-headed attention, and graph representation
  - Explicit graph learning

Open Problems

Graphical Models

Growing body of literature to combine deep learning and graphical models.

- Memory: the cost of storing the representation
- Statistical efficiency: the number of training data
- Runtime: the cost of inference
- Runtime: the cost of sampling

Goodfellow, Bengio, and Courville (2016)

Graphical Event Models

Equivalent and new problems in neural GEM:

- Representation learning of unstructured events
- Structure learning: real validation datasets with graphs
- Inference: generative TPPs
- Latent events

References I

- Aalen, Odd O. (1987). "Dynamic modelling and causality". In: *Scandinavian Actuarial Journal* 1987.3-4, pp. 177–190.
- Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473*.
- Bhattacharjya, D., T. Gao, and D. Subramanian (2020). "Order-dependent event models for agent interactions". In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1977–1983.
- Bhattacharjya, D. et al. (2022). "Process Independence Testing in Proximal Graphical Event Models". In: *Proceedings of the 1st Conference on Causal Learning and Reasoning (CLeaR)*.
- Bhattacharjya, Debarun, Dharmashankar Subramanian, and Tian Gao (2018). "Proximal graphical event models". In: *Advances in Neural Information Processing Systems*. Vol. 31.

References II

- Campos, C. P. de and Q. Ji (2011). "Efficient structure learning of Bayesian networks using constraints". In: *Journal of Machine Learning Research* 12.Mar, pp. 663–689.
- Christgau, Alexander Mangulad, Lasse Petersen, and Niels Richard Hansen (2022). "Nonparametric Conditional Local Independence Testing". In: *arXiv preprint arXiv:2203.13559*.
- Commenges, Daniel and Anne Gégout-Petit (2009). "A General Dynamical Statistical Model with Causal Interpretation". In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 71.3, pp. 719–736.
- Didelez, Vanessa (2008). "Graphical models for marked point processes based on local independence". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 70.1, pp. 245–264.

References III

Du, Nan et al. (2016). "Recurrent Marked Temporal Point Processes: Embedding Event History to Vector". In: *Proc. of the 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pp. 1555–1564.

Gao, Tian et al. (2020). "A Multi-Channel Neural Graphical Event Model with Negative Evidence". In: *Proc. of AAAI Conference on Artificial Intelligence*. Vol. 34. 04, pp. 3946–3953.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press.

Gu, Yulong (2021). "Attentive Neural Point Processes for Event Forecasting". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 9, pp. 7592–7600.

References IV

- Gunawardana, Asela, Christopher Meek, and Puyang Xu (2011). "A Model for Temporal Dependencies in Event Streams". In: *Advances in Neural Information Processing Systems 24 (NIPS).*
- Meek, Christopher (2014). "Toward Learning Graphical and Causal Process Models.". In: *CI at UAI*, pp. 43–48.
- Mei, Hongyuan and Jason Eisner (2016). "The Neural Hawkes Process: A Neurally Self-modulating Multivariate Point Process". In: *arXiv preprint arXiv:1612.09328.*
- Mogensen, Søren Wengel and Niels Richard Hansen (2020). "Markov equivalence of marginalized local independence graphs". In: *The Annals of Statistics* 48.1, pp. 539–559.

References V

- Mogensen, Søren Wengel, Daniel Malinsky, and Niels Richard Hansen (2018). "Causal Learning for Partially Observed Stochastic Dynamical Systems". In: *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence (UAI)*.
- Rajaram, S., T. Graepel, and R. Herbrich (2005). "Poisson-networks: A model for structured point processes". In: *Proceedings of International Workshop on Artificial Intelligence and Statistics*, pp. 277–284.
- Schweder, Tore (1970). "Composable Markov Processes". In: *Journal of Applied Probability* 7.2, pp. 400–410.
- Spirtes, Peter, Clark Glymour, and Richard Scheines (2000). *Causation, Prediction, and Search*. MIT Press.
- Vaswani, Ashish et al. (2017). "Attention is all you need". In: *Advances in neural information processing systems* 30.

References VI

Xiao, Shuai et al. (2017). "Modeling the Intensity Function of Point Process Via Recurrent Neural Networks". In: *Proc. of Conf. on Artificial Intelligence (AAAI)*, pp. 1597–1603.

Xiao, Shuai et al. (2019). "Learning time series associated event sequences with recurrent point process networks". In: *IEEE transactions on neural networks and learning systems* 30.10, pp. 3124–3136.

Zhang, Qiang, Aldo Lipani, and Emine Yilmaz (2021). "Learning neural point processes with latent graphs". In: *Proceedings of the Web Conference 2021*, pp. 1495–1505.

Zhang, Qiang et al. (2020a). "Self-attentive Hawkes Process". In: *International Conference on Machine Learning*. PMLR, pp. 11183–11193.

References VII

Zhang, Wei et al. (2020b). "Cause: Learning granger causality from event sequences using attribution methods". In: *International Conference on Machine Learning*. PMLR, pp. 11235–11245.

Zhou, K., H. Zha, and L. Song (2013). "Learning triggering kernels for multi-dimensional Hawkes processes". In: *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 1301–1309.

Zuo, Simiao et al. (2020). "Transformer Hawkes Process". In: *International Conference on Machine Learning*. PMLR, pp. 11692–11702.