Scalable Nonparametric Factorization for High-Order Interaction Events

Zhimeng Pan, Zheng Wang, Shandian Zhe
School of Computing, University of Utah
z.pan@utah.edu, wzhut@cs.utah.edu, zhe@cs.utah.edu

Abstract

Interaction events among multiple entities are ubiquitous in real-world applications. Although these interactions can be naturally represented by tensors and analyzed by tensor decomposition, most existing approaches are limited to multilinear decomposition forms, and cannot estimate complex, nonlinear relationships in data. More importantly, the existing approaches severely underexploit the time stamps information. They either drop/discretize the time stamps or set a local window to ignore the long-term dependency between the events. To address these issues, we propose a Bayesian nonparametric factorization model for high-order interaction events, which can flexibly estimate/embed the static, nonlinear relationships and capture various long-term and short-term excitations effects, encoding these effects and their decaying patterns into the latent factors. Specifically, we use the latent factors to construct a set of mutually excited Hawkes processes, where we place a Gaussian process prior over the background rates to estimate the static, nonlinear relationships of the entities and propose novel triggering kernels to embed the excitation strengths and their time decaying rates among the interactions. For scalable inference, we derive a fully-decomposed model evidence lower bound to dispose of the huge covariance matrix and expensive log summation terms. Then we develop an efficient stochastic optimization algorithm. We show the advantage of our approach in four real-world applications.

Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108. Copyright 2020 by the author(s).

1 Introduction

Interactions of multiple entities are ubiquitous in real world. For example, purchase behaviours can be considered as three-way interactions between *customers*, *commodities* and *stores*. Many applications have presented massive data for interaction events, which record the time stamps when a variety of (high-order) interactions occurred. These events reflect not only complex relationships of the entities, but also rich and long-term temporal influences between their interactions. To analyze these data, we consider extracting a set of latent factors to represent each participating entity. With the factor representations, we can look into the hidden structures of the entities, *e.g.,* communities and anomalies, and extract discriminative features to make predictions, *e.g.,* predicting missing/future interactions.

While high-order interactions can be represented by tensors and we can use various tensor decomposition methods (Tucker, 1966; Harshman, 1970; Chu and Ghahramani, 2009; Kang et al., 2012; Choi and Vishwanathan, 2014) to estimate the factors, these methods mainly use a multilinear decomposition form, hence might be inadequate to capture more complex, nonlinear relationships of the entities. More importantly, these methods severely underexploit the time stamps information. They either ignore the time stamps of the interactions and aggregate the observations into a count tensor (Chi and Kolda, 2012; Hansen et al., 2015; Hu et al., 2015b), or discretize the time stamps into rough steps, such as weeks or months, and overlook the dependencies of the interactions in the same step (Xiong et al., 2010; Schein et al., 2015, 2016). Very recently, Zhe and Du (2018) used Hawkes processes to estimate the fine-grained, local triggering effects among the interactions. However, they used a small window to limit the range of the dependent events so as to compromise with the computational cost, and hence cannot capture the full-spectrum of temporal effects, especially those long-term ones, from the previous interactions. Therefore, they may still miss important temporal patterns.

To address these issues, we propose a nonparametric Bayesian factorization model for high-order interaction events, which not only is flexible enough to estimate the static, nonlinear relationships of entities, but also can capture the full-spectrum, long-term excitation effects among the interactions, encoding these effects and their time-decaying patterns into the latent factors to reveal the structures within the complex temporal dependencies. Specifically, we use the latent factors to construct a set of mutually excited Hawkes processes, where each process samples the occurrences of a particular interaction. We use a Gaussian process (GP) prior to sample the background rate of each interaction as a (nonlinear) function of the latent factors of the participating entities, so as to capture and embed the complex yet static relationships between those entities. We then propose a novel triggering function that uses the kernel (similarity) of the latent factors to model the triggering strengths and their (time) decaying rates for all the previously happened events. In this way, various excitation (causal) patterns are encoded into the latent factors, from which we can discover clustering structures in terms of the excitation effects. For scalable inference, we use the variational sparse GP framework (Hensman et al., 2013) and Jensen's equality on the log rate function, to derive a fully-decomposed log model evidence lower bound. Based on the bound, we develop an efficient stochastic optimization algorithm. Its complexity is only proportional to the mini-batch size, and still estimates all the long-term excitation effects.

For evaluation, we examined our approach on four real-world datasets. Our model often largely improves upon the prediction accuracy of the existing methods that use Poisson processes, discrete time factors, and local time dependency window to incorporate the temporal information. The learning curves show that our inference algorithm converges fast and is immune to overfitting. Finally, we looked into the learned factors by our approach on crime reports data, and found interesting clustering structures, such as neighbouring patrol areas and strongly associated crime types.

2 Notations and Background

Suppose we observe K-way interactions that involve K types of entities (*e.g., customers*, *items* and *stores*). In each type k, the number of entities is d_k. We index each entity of type k by i_k ($1 \leq i_k \leq d_k$). Furthermore, we use a tuple $\mathbf{i} = (i_1, \ldots, i_K)$ to index a particular interaction of the K types' entities. We denote a sequence of N interaction events by $S = [(s_1, \mathbf{i}_1), \ldots, (s_N, \mathbf{i}_N)]$, where $s_1 \leq \ldots \leq s_N$ are the time stamps of the events, and each $\mathbf{i}_n$ is the index of the interaction for event n ($1 \leq n \leq N$). Note that there can be (many) duplicated indices in $[\mathbf{i}_1, \ldots \mathbf{i}_N]$, because one interaction can occur multiple times. From the observed interaction events S, we aim to learn the latent factor representations for all the participant entities. We denote the latent factors for each entity j of type k by an r_k dimensional vector $\mathbf{u}_j^k$. Correspondingly, we can introduce K latent factor matrices, $\mathcal{U} = \{\mathbf{U}^1, \ldots, \mathbf{U}^K\}$, to represent all the entities. Each $\mathbf{U}^k$ is $d_k \times r_k$, and its rows are the latent factors for the entities of type k.

To estimate the latent factors from S, a straightforward approach is to use tensor decomposition algorithms. We can introduce a K-mode tensor $\mathcal{Y} \in \mathbb{R}^{d_1 \times \cdots \times d_K}$, where the k-th mode includes the d_k entities of type k. To convert the events S into a tensor, a commonly used strategy is to drop the time stamps, and count the occurrence of each interaction. We then set each tensor entry $\mathcal{Y}(i_1, \ldots, i_K)$ to the frequency of the corresponding interaction. Given the tensor $\mathcal{Y}$, a classical decomposition approach is Tucker decomposition (Tucker, 1966), which assumes $\mathcal{Y} = \mathcal{W} \times_1 \mathbf{U}^1 \times_2 \ldots \times_K \mathbf{U}^K$, where $\mathcal{W} \in \mathbb{R}^{r_1 \times \cdots \times r_K}$ is a parametric tensor, and $\times_k$ is the mode-k tensor matrix product (Kolda, 2006), which resembles the ordinary matrix-matrix product. When we set all $r_k = r$, and constrain $\mathcal{W}$ to be diagonal, Tucker decomposition becomes CANDECOMP/PARAFAC (CP) decomposition (Harshman, 1970). While numerous tensor decomposition methods have been proposed, *e.g.,* (Chu and Ghahramani, 2009; Kang et al., 2012; Choi and Vishwanathan, 2014), most of them are inherently based on the CP or Tucker decomposition form. To conduct count tensor decomposition, a Poisson process likelihood is usually chosen to model the interaction frequency in each entry $\mathbf{i}$, $p(y_\mathbf{i}) \propto \exp(-\lambda_\mathbf{i} T)\lambda_\mathbf{i}^{y_\mathbf{i}}$ where T is the total time span, and Tucker/CP decomposition is applied to the rates $\{\lambda_\mathbf{i}\}$ or log rates $\{\log(\lambda_\mathbf{i})\}$ (Chi and Kolda, 2012; Hu et al., 2015b). Despite their convenience, both the Tucker/CP forms are mutilinear to the latent factors and hence cannot capture more complicated, nonlinear relationships between the entities.

More importantly, dropping the time stamps and performing count tensor decomposition can ignore the temporal dependencies among the events, which are crucial to capture the temporal patterns. A more refined strategy is to discretize the time stamps into a few steps, *e.g.,* weeks or months, augment the count tensor with a time mode (Xiong et al., 2010; Schein et al., 2015, 2016), and then jointly estimate the time factors $\{\mathbf{t}_1, \mathbf{t}_2, \ldots\}$. To allow a smooth transition between the consecutive time factors, a conditional Gaussian prior can be further assigned over each $\mathbf{t}_k$, $p(\mathbf{t}_k|\mathbf{t}_{k-1}) = \mathcal{N}(\mathbf{t}_k|\mathbf{t}_{k-1}, \sigma^2\mathbf{I})$ (Xiong et al., 2010). Nonetheless, the events within the same step are still considered to occur independently (*i.e.,* Poisson process), hence they still overlook their temporal influences on each other and may miss interesting and important temporal patterns.

3 Model

To overcome the problems of the existing approaches, we propose a Bayesian nonparametric factorization model for the high-order interaction events, which can not only capture/embed the static, nonlinear relationships between different entities, but also estimate various short-term/long-term triggering (or causal) effects among the events, encoding these effects and their decaying patterns into the latent factors. Our model is presented as follows.

3.1 Hawkes Processes for Events Modeling

First, while Poisson processes (PPs) are commonly used to model events data due to their convenience and elegance (Schein et al., 2015), they assume independent increments, namely, the events happened in (any) distinct time ranges are independent to each other. Hence, Poisson processes cannot capture the rich and subtle temporal dependencies between the events. To overcome this limit, we use Hawkes processes (HPs) (Hawkes, 1971), a much more flexible point process for events modeling. In general, a Hawkes process samples a sequence of events $\{t_1, \ldots, t_N\}$ according to a rate function of time t,

$$\lambda(t) = \lambda_0 + \sum_{t_n < t} h(t - t_n),$$

where λ_0 is the static, background rate, and $h(\Delta_t)$ is the triggering kernel/function, describing the strength that a previously happened event s_n triggers or causes the occurrence of an event at t. The triggering effects usually decay in time; hence a commonly used triggering function is the exponential decay function,

$$h(\Delta_t) = \alpha \exp(-\frac{\Delta_t}{\tau})$$

where α controls the strength, and τ determines the decaying rate. Given the event sequence $\{t_1, \ldots, t_N\}$, the joint probability of the Hawkes process is

$$p(\{t_1, \ldots t_n\}) = e^{-\int_0^T \lambda(t)} \prod_{j=1}^{N} \lambda(t_j) \tag{1}$$

where T is the total time span over all the events. Therefore, Hawkes processes separate the static (*i.e.,* the background rate) and temporal components (*i.e.,* the triggering function) that drives the occurrence of events, and is flexible enough to capture all kinds of short-term/long-term excitation effects from previously happened events, with their time stamps taken into account. Hence, we will ground our model in the Hawkes process framework.

3.2 Interaction Events Factorization

To factorize the observed high-order interaction events, we use latent factors to construct a set of mutually excited Hawkes processes. For each interaction $\mathbf{i} = (i_1, \ldots, i_K)$, we use a Hawkes process to sample its event sequence. The occurrence of $\mathbf{i}$ can be triggered/caused by not only the same interaction happened before, but also different previous interactions sampled by other Hawkes processes. Specifically, the rate function of each interaction $\mathbf{i}$ is defined as

$$\lambda_{\mathbf{i}}(t) = \lambda_{\mathbf{i}}^0 + \sum_{s_n < t} h_{\mathbf{i}_n \to \mathbf{i}}(t - s_n), \tag{2}$$

where $\lambda_{\mathbf{i}}^0$ is the background rate and $h_{\mathbf{i}_n \to \mathbf{i}}(\Delta_t)$ is the triggering kernel that describes the strength that each previously occurred interaction $\mathbf{i}_n$ encourages/causes interaction $\mathbf{i}$ to occur at time t. Note that $\mathbf{i}_n$ can be the same as or different from $\mathbf{i}$.

We now use the latent factors to model the background rate and the triggering kernel. First, in order to capture the static, nonlinear relationships between the entities that participate in each interaction $\mathbf{i}$, we model the background rate $\lambda_{\mathbf{i}}^0$ as a (nonlinear) function of their latent factors associated with $\mathbf{i}$, $\mathbf{x_i} = [\mathbf{u}_{i_1}^1; \ldots; \mathbf{u}_{i_K}^K]$. To allow a flexible function estimation and to ensure the positiveness of the background rate, we place a GP prior over its logarithm, $g(\mathbf{x_i}) = \log \lambda_0(\mathbf{x_i}) \sim \mathcal{GP}(\cdot|0, k(\mathbf{x_i}, \mathbf{x_i}))$, where $k(\cdot, \cdot)$ is the covariance (kernel) function. Suppose we have m distinct interactions $Q = \{\mathbf{j}_1, \ldots, \mathbf{j}_m\}$ in total, the function values $\mathbf{g} = [g(\mathbf{x}_{\mathbf{j}_1}), \ldots, g(\mathbf{x}_{\mathbf{j}_m})]$ hence follow a multivariate Gaussian distribution,

$$p(\mathbf{g}|\mathcal{U}) = \mathcal{N}(\mathbf{g}|\mathbf{0}, \mathbf{K}_{mm}) \tag{3}$$

where $\mathbf{K}_{mm}$ is the covariance (kernel) matrix on $\mathbf{X}_Q = [\mathbf{x}_{\mathbf{j}_1}, \ldots, \mathbf{x}_{\mathbf{j}_m}]$, and each element $[\mathbf{K}_{mm}]_{r,c} = k(\mathbf{x}_{\mathbf{j}_r}, \mathbf{x}_{\mathbf{j}_c})(1 \leq r, c \leq m)$ is a covariance (kernel) function of the latent factors associated with the corresponding interactions.

Next, in order to estimate a variety of the excitation effects between the interactions and their decaying patterns, we define the triggering function as

$$h_{\mathbf{i}_n \to \mathbf{i}}(\Delta_t) = k_1(\mathbf{x}_{\mathbf{i}_n}, \mathbf{x_i}) \cdot \exp(-\frac{\Delta_t}{k_2(\mathbf{x}_{\mathbf{i}_n}, \mathbf{x_i})}) \tag{4}$$

where $k_1(\cdot, \cdot)$ and $k_2(\cdot, \cdot)$ are two kernel functions that measure the similarity between $\mathbf{x}_{\mathbf{i}_n}$ and $\mathbf{x_i}$. In our model, the excitation strength between two types of interactions $\mathbf{i}_n$ and $\mathbf{i}$, as well as how fast it decays along with time are determined by the similarity/closeness between the associated entities. The closeness is measured by the kernel functions (k_1 and k_2) of the latent factor representations. The closer the entities that participate in two interactions, the stronger and longer the excitation effect. In this way, the excitation effects and their time decaying patterns are encoded into the latent factors, which enables us to uncover hidden structures underlying these temporal influences — *e.g.,* clusters/communities of entities that more strongly and persistently trigger each other to interact with other groups of entities, *e.g.,* "ordering the same types of food" and "buying similar styles of bags/clothes".

We assign a standard Gaussian prior over each latent factor. Given the observed interaction events $S = [(s_1, \mathbf{i}_1), \ldots, (s_N, \mathbf{i}_N)]$, the joint probability of our model, according to (2), (3) and (4), is given by

$$p(S,\mathbf{g},\mathcal{U}) = \prod_k \prod_{i_k} \mathcal{N}(\mathbf{u}_{i_k}^k|0,\mathbf{I})\mathcal{N}(\mathbf{g}|\mathbf{0},\mathbf{K}_{mm}) \cdot \prod_{\mathbf{i}\in Q} \exp\big\{-\int_0^T \lambda_{\mathbf{i}}(t)\mathrm{d}t\big\} \prod_{n=1}^N \lambda_{\mathbf{i}_n}(s_n) \quad (5)$$

where each $\lambda_{\mathbf{i}}^0 = \exp\big(g(\mathbf{x}_{\mathbf{i}})\big)$.

4 Algorithm

We now present the model estimation algorithm. The exact inference for our model is computationally infeasible for large data due to the GP likelihood in (3), which requires us to compute an $m \times m$ kernel matrix $\mathbf{K}_{mm}$ and its inverse. When the number of distinct interactions (*i.e.,* m) is large, the computation is infeasible. Moreover, the calculation of each $\lambda_{\mathbf{i}_n}(s_n)$ (see (5)) needs the triggering kernel from all the previously happened events $\{s_1,\ldots,s_{n-1}\}$ (see (2)), and hence is very expensive when the number of observed events (*i.e.,* N) is large. To scale up to both large m and N, we use sparse variational GP framework (Titsias, 2009; Hensman et al., 2013) and Jensen's inequality to derive a fully-decomposable model evidence lower bound (ELBO), based on which we develop an efficient stochastic optimization algorithm.

4.1 Fully-Decomposable Evidence Lower Bound

Specifically, to use sparse GP, we first introduce a small set of pseudo inputs $\mathbf{Z} = [\mathbf{z}_1,\ldots,\mathbf{z}_l]$ where $l \ll m$. We denote the values of the latent function g on $\mathbf{Z}$ by $\mathbf{b} = [g(\mathbf{z}_1),\ldots,g(\mathbf{z}_l)]$. Since $g(\cdot)$ is sampled from GP, we have its projection on $\mathbf{Z}$ and $\mathbf{X}_Q$ (the latent factors associated with all the distinct interactions, see (3)), namely $\mathbf{b}$ and $\mathbf{g}$, jointly follow a multi-variate Gaussian distribution where the covariance matrix is a kernel matrix on both $\mathbf{Z}$ and $\mathbf{X}_Q$. We can further decompose their joint probability by

$$p(\mathbf{b},\mathbf{g}) = p(\mathbf{b})p(\mathbf{g}|\mathbf{b}) \quad (6)$$

where $p(\mathbf{b}) = \mathcal{N}(\mathbf{b}|\mathbf{0},\mathbf{K}_{ll})$ and $p(\mathbf{g}|\mathbf{b})$ is a conditional Gaussian distribution, $p(\mathbf{g}|\mathbf{b}) = \mathcal{N}(\mathbf{g}|\mathbf{K}_{ml}\mathbf{K}_{ll}^{-1}\mathbf{b},\mathbf{K}_{mm} - \mathbf{K}_{ml}\mathbf{K}_{ll}^{-1}\mathbf{K}_{lm})$. Here $\mathbf{K}_{ll}$ is the covariance (kernel) matrix on $\mathbf{Z}$, $\mathbf{K}_{ml}$ is the cross covariance matrix between $\mathbf{X}$ and $\mathbf{Z}$ where each element $[\mathbf{K}_{ml}]_{rc} = k(\mathbf{x}_{\mathbf{j}_r},\mathbf{z}_c)$, and $\mathbf{K}_{lm} = \mathbf{K}_{ml}^\top$. We now can augment the joint probability of our model with the pseudo outputs $\mathbf{b}$,

$$p(S,\mathbf{g},\mathbf{b},\mathcal{U}) = p(\mathbf{b})p(\mathbf{g}|\mathbf{b})p(S,\mathcal{U}|\mathbf{g}) \quad (7)$$

where $p(S,\mathcal{U}|\mathbf{g}) = \prod_k \prod_{i_k} \mathcal{N}(\mathbf{u}_{i_k}^k|0,\mathbf{I}) \prod_{\mathbf{i}\in Q} \exp\big\{-\int_0^T \lambda_{\mathbf{i}}(t)\mathrm{d}t\big\} \prod_{n=1}^N \lambda_{\mathbf{i}_n}(s_n)$. Note that when we marginalize out $\mathbf{b}$, we recover the original probability (5). Based on (7), we now construct a variational evidence lower bound to avoid computing the full covariance matrix $\mathbf{K}_{mm}$ and its inverse which are prohibitively expensive for large m. To this end, we introduce a variational posterior for $\mathbf{b}$ and $\mathbf{g}$,

$$q(\mathbf{b},\mathbf{g}) = q(\mathbf{b})p(\mathbf{g}|\mathbf{b}) \quad (8)$$

where $q(\mathbf{b}) = \mathcal{N}(\mathbf{b}|\boldsymbol{\mu},\boldsymbol{\Sigma})$ is a Gaussian distribution. To ensure the positive definiteness of $\boldsymbol{\Sigma}$ and to ease computation, we further parameterize $\boldsymbol{\Sigma}$ by its Cholesky decomposition, $\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}^\top$ where $\mathbf{L}$ is a lower triangular matrix. Then we can derive the variational lower bound from

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{b},\mathbf{g})}\big[\log \frac{p(S,\mathbf{g},\mathbf{b},\mathcal{U})}{q(\mathbf{b},\mathbf{g})}\big].$$

From (7) and (8), we can see that the conditional Gaussian term $p(\mathbf{g}|\mathbf{b})$ is cancelled in the fraction inside of the logarithm. This is the key step to reduce the cost, because the conditional covariance is a giant $m \times m$ matrix that includes $\mathbf{K}_{mm}$. We can arrange the terms and obtain

$$\mathcal{L} = \log\big(p(\mathcal{U})\big) - \mathrm{KL}\big(q(\mathbf{b})\|p(\mathbf{b})\big) - T\sum_{\mathbf{i}\in\mathbf{Q}} \mathbb{E}_q\big[e^{g(\mathbf{x}_{\mathbf{i}})}\big] - \sum_{\mathbf{i}\in\mathbf{Q}}\sum_{n=1}^N \int_{s_n}^T h_{\mathbf{i}_n\to\mathbf{i}}(t-s_n)\mathrm{d}t + \sum_{n=1}^N \mathbb{E}_q\big[\log\big(\lambda_{\mathbf{i}_n}(s_n)\big)\big], \quad (9)$$

where $p(\mathcal{U}) = \prod_k \prod_{i_k} \mathcal{N}(\mathbf{u}_{i_k}^k|0,\mathbf{I})$ and KL is the Kullback-Leibler divergence. Note that to calculate each expectation term, we do not need to use the full variational posterior $q(\mathbf{b},\mathbf{g})$, because it only involves one particular interaction and all the other elements in g are marginalized out. Take $\mathbb{E}_q\big[e^{g(\mathbf{x}_{\mathbf{i}})}\big]$ for an example. We only need to use $q\big(\mathbf{b},g(\mathbf{x}_{\mathbf{i}})\big) = q(\mathbf{b})p\big(g(\mathbf{x}_{\mathbf{i}})|\mathbf{b}\big)$ to calculate the expectation. Here $p\big(g(\mathbf{x}_{\mathbf{i}})|\mathbf{b}\big) = \mathcal{N}\big(g(\mathbf{x}_{\mathbf{i}})|\gamma_{\mathbf{i}}(\mathbf{b}),\sigma_{\mathbf{i}}^2\big)$ is a scalar conditional Gaussian distribution, where $\boldsymbol{\gamma}_{\mathbf{i}}(\mathbf{b}) = \mathbf{k}_{\mathbf{i}l}\mathbf{K}_{ll}^{-1}\mathbf{b}$, and $\sigma_{\mathbf{i}}^2 = k_{\mathbf{i}\mathbf{i}} - \mathbf{k}_{\mathbf{i}l}\mathbf{K}_{ll}^{-1}\mathbf{k}_{l\mathbf{i}}$, $\mathbf{k}_{\mathbf{i}l} = [k(\mathbf{x}_{\mathbf{i}},\mathbf{z}_1),\ldots,k(\mathbf{x}_{\mathbf{i}},\mathbf{z}_l)]$ and $\mathbf{k}_{l\mathbf{i}} = \mathbf{k}_{\mathbf{i}l}^\top$. Hence the computation is cheap.

However, it is still expensive to calculate each log rate function $\log\big(\lambda_{\mathbf{i}_n}(s_n)\big)$ in (9). According to (2), this is a log summation term, and we have

$$\log\big(\lambda_{\mathbf{i}_n}(s_n)\big) = \log\big(\lambda_{\mathbf{i}_n}^0 + \sum_{j=1}^N \delta(s_j < s_n) h_{\mathbf{i}_j\to\mathbf{i}_n}(\Delta_{nj})\big)$$

where $\delta(\cdot)$ is an indicator function and $\Delta_{nj} = s_n - s_j$. Hence, the complexity is proportional to the number of observed events N. When N is large, it is very costly. To address this issue, we can partition the events into mini-batches of size M, $\mathcal{B} = \{B_1,\ldots,B_{N/M}\}$. Then we have $\lambda_{\mathbf{i}_n}(s_n) = \lambda_{\mathbf{i}_n}^0 + \frac{M}{N}\sum_{k=1}^{N/M} \frac{N}{M}\sum_{j\in B_k} \delta(s_j < s_n) h_{\mathbf{i}_j\to\mathbf{i}_n}(\Delta_{nj})$. We can view the rate as an expectation, $\lambda_{\mathbf{i}_n}(s_n) = \mathbb{E}_{p(k)}[X_k^n]$, where $p(k) = \frac{M}{N}$, k can take values from $\{1,\ldots,N/M\}$, and

$$X_k^n = \lambda_{\mathbf{i}_n}^0 + \frac{N}{M}\sum_{j\in B_k} \delta(s_j < s_n) h_{\mathbf{i}_j\to\mathbf{i}_n}(\Delta_{nj}). \quad (10)$$

Therefore, we can use Jensen's inequality to obtain

$$\log\big(\lambda_{\mathbf{i}_n}(s_n)\big) = \log(\mathbb{E}_{p(k)}[X_k^n]) \geq \mathbb{E}_{p(k)}[\log(X_k^n)].$$

We substitute this lower bound for each log rate term $\log\big(\lambda_{\mathbf{i}_n}(s_n)\big)$in (9), and obtain a fully-decomposable evidence lower bound,

$$\mathcal{L}^f = \log\big(p(\mathcal{U})\big) - \mathrm{KL}\big(q(\mathbf{b})\|p(\mathbf{b})\big) - T\sum_{\mathbf{i}\in Q}\mathbb{E}_q\big[e^{g(\mathbf{x}_\mathbf{i})}\big]$$
$$- \sum_{\mathbf{i}\in\mathbf{Q}}\sum_{n=1}^{N}\int_{s_n}^{T} h_{\mathbf{i}_n\to\mathbf{i}}(t-s_n)\mathrm{d}t + \sum_{n=1}^{N}\mathbb{E}_{p(k)}\mathbb{E}_q[\log(X_k^n)]. \tag{11}$$

In this way, we move most of the summation in each $\log\big(\lambda_{\mathbf{i}_n}(s_n)\big)$ to the outside of log, leaving a tiny amount of summation (over the mini-batch) inside, *i.e.,* X_k^n. Based on the fully-decomposed new bound, we can develop efficient stochastic optimization for model estimation.

4.2 Stochastic Optimization

We aim to maximize the evidence lower bound $\mathcal{L}^f$ in (11) to estimate the variational posterior q, the latent factors and the other parameters. Despite the decomposed form of $\mathcal{L}^f$, it is expensive to compute because of the summation and double summation terms. In order to develop an efficient optimization algorithm, we further partition the distinct interactions Q into mini-batches of size D, $\mathcal{Q} = \{Q_1, \ldots, Q_{m/D}\}$. We partition the events into mini-batches of size F. Note that we can re-use the previous partition $\mathcal{B}$ or choose a new one. We leave the flexibility here, and denote the event batches by $\mathcal{C} = \{C_1, \ldots, C_{N/F}\}$. Now we arrange $\mathcal{L}^f$ as

$$\mathcal{L}^f = -\mathrm{KL}\big(q(\mathbf{b})\|p(\mathbf{b})\big) - T\sum_j \frac{D}{m}\sum_{\mathbf{i}\in Q_j}\frac{m}{D}\mathbb{E}_q\big[e^{g(\mathbf{x}_\mathbf{i})}\big]$$
$$-\sum_j\sum_t \frac{D}{m}\frac{F}{N}\sum_{\mathbf{i}\in Q_j, n\in C_t}\frac{m}{D}\frac{N}{F}\phi(\mathbf{i},\mathbf{i}_n,s_n) + \log\big(p(\mathcal{U})\big)$$
$$+\sum_t\sum_k \frac{F}{N}\frac{M}{N}\sum_{n\in C_t, k\in B_k}\frac{N}{F}\mathbb{E}_q\log(X_k^n), \tag{12}$$

where $\phi(\mathbf{i},\mathbf{i}_n,s_n) = \int_{s_n}^{T} h_{\mathbf{i}_n\to\mathbf{i}}(t-s_n)\mathrm{d}t$ and X_k^n is defined in (10). Then the bound can be viewed as the expectation of a stochastic objective,

$$\mathcal{L}^f = \mathbb{E}_{p(k),p(\alpha),p(v)}[\tilde{\mathcal{L}}^f_{k,\alpha,v}], \tag{13}$$

where $p(v) = \frac{D}{m}$, $v \in \{1,\ldots,m/D\}$, $p(\alpha) = \frac{F}{N}$, $\alpha \in \{1,\ldots,N/F\}$, and

$$\tilde{\mathcal{L}}^f_{k,\alpha,v} = \log\big(p(\mathcal{U})\big) - \mathrm{KL}\big(q(\mathbf{b})\|p(\mathbf{b})\big)$$
$$-T\sum_{\mathbf{i}\in Q_v}\frac{m}{D}\mathbb{E}_q\big[e^{g(\mathbf{x}_\mathbf{i})}\big] - \sum_{\mathbf{i}\in Q_v, n\in C_\alpha}\frac{m}{D}\frac{N}{F}\phi(\mathbf{i},\mathbf{i}_n,s_n)$$
$$+\sum_{n\in C_\alpha, k\in B_k}\frac{N}{F}\mathbb{E}_q\log(X_k^n). \tag{14}$$

Now, we can develop a stochastic optimization algorithm based on (13). Each time, we first sample a mini-batch Q_v, B_k and C_α, and then compute the gradient of the stochastic bound $\tilde{\mathcal{L}}^f_{k,\alpha,v}$ in (14) as an unbiased stochastic gradient of $\mathcal{L}^f$. Note that $\mathbb{E}_q\big[e^{g(\mathbf{x}_\mathbf{i})}\big]$ is analytical and so is the gradient. However, the expectation term $\mathbb{E}_q[\log(X_k^n)]$ is intractable to compute, because the exponential of the latent function value, $\lambda^0_{\mathbf{i}_n} = e^{g(\mathbf{x}_{\mathbf{i}_n})}$, is inside the log (see (10)). To address this problem, we use the reparameterization trick (Kingma and Welling, 2013). We first generate a parameterized sample $\tilde{\mathbf{b}} = \boldsymbol{\mu} + \mathbf{L}\boldsymbol{\eta}$ where $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$. Then, we generate a parameterized sample for $g(\mathbf{x}_{\mathbf{i}_n})$, $\tilde{g}_{\mathbf{i}_n} = \boldsymbol{\gamma}_\mathbf{i}(\tilde{\mathbf{b}}) + \sigma_\mathbf{i}\epsilon$ where $\epsilon \sim \mathcal{N}(0,1)$. We substitute $\tilde{g}_{\mathbf{i}_n}$ for $g(\mathbf{x}_{\mathbf{i}_n})$ in each X_k^n, and hence can obtain an unbiased stochastic estimate of $\tilde{\mathcal{L}}^f_{k,\alpha,v}$. We then calculate the gradient of that estimate, which will be an unbiased stochastic gradient of $\tilde{\mathcal{L}}^f_{k,\alpha,v}$ and in turn an unbiased stochastic gradient of $\mathcal{L}^f$. Then we can use any stochastic optimization algorithm to maximize $\mathcal{L}^f$, so as to jointly update the variational posteriors $q(\mathbf{b})$, latent factors $\mathcal{U}$, kernel parameters and the other parameters. The computation of the stochastic gradient (see (14)) restricts the double summation to be across the mini-batches only, and hence can largely reduce the cost.

4.3 Algorithm Complexity

The time complexity of our algorithm is $\mathcal{O}(D_0F_0 + F_0M_0 + (D_0+F_0)l^3)$ where D_0, F_0 and M_0 are the mini-batch sizes of $\mathcal{Q}$, $\mathcal{C}$ and $\mathcal{B}$ respectively. Therefore, the computational cost is proportional to the mini-batch sizes. The space complexity is $\mathcal{O}(l^2 + \sum_{k=1}^{K} d_k r_k)$ which is to store the covariance matrix of the pseudo outputs $\mathbf{b}$ and latent factors $\mathcal{U}$.

5 Related Work

High-order interactions are naturally represented by multidimensional arrays, or tensors, and analyzed by tensor decomposition. Canonical tensor decomposition approaches include CP (Harshman, 1970) and Tucker (Tucker, 1966) decompositions, based on which many other models or algorithms have been developed (Shashua and Hazan, 2005; Chu and Ghahramani, 2009; Sutskever et al., 2009; Acar et al., 2011; Hoff, 2011; Kang et al., 2012; Yang and Dunson, 2013; Rai et al., 2014; Choi and Vishwanathan, 2014; Hu et al., 2015a; Rai et al., 2015), just to name a few. Recently, a few nonparametric Bayesian decomposition methods (Xu et al., 2012; Zhe et al., 2015, 2016) were proposed to estimate nonlinear relationships in data, and have been shown improved prediction accuracy upon the the multilinear methods. When dealing with temporal information, traditional methods either simply decompose the counts (Chi and Kolda, 2012; Hansen et al., 2015; Hu et al., 2015b), or discretize the time stamps into a few time steps, and perform the count decomposition across the time steps (Xiong et al., 2010; Schein et al., 2015, 2016). Although very useful, these methods overlook rich and subtle temporal dependencies among the interactions, and may miss important temporal patterns. Recently, Zhe and Du (2018) used the Hawkes processes to capture the local triggering effects between the interactions.They set a time window, and assume only the preceding events in the window have the excitation effects. To ensure their inference algorithm to be scalable, they have to set the window size to be small enough, say, a few hundreds. By contrast, our model does not impose a window, and estimates the excitations from all the preceding events, short-term and long-term. In addition, Zhe and Du (2018) assumes the triggering kernel has a constant decaying rate across all the interactions. Our approach assumes heterogeneous decaying rates and hence is more flexible. We model the decaying rate as another kernel of the latent factors, and hence can encode more temporal effects into the latent factor representations. Our approach can be considered a natural generalization of Zhe and Du (2018)'s model.

Hawkes processes (HPs) are a popular family of point process models to study the mutual triggering relationships for all kinds of events. There have been many works that use HPs to uncover temporal relationships in different applications, such as (Blundell et al., 2012; Tan et al., 2016; Linderman and Adams, 2014; Du et al., 2015; He et al., 2015; Wang et al., 2017). Meanwhile, many works have also been proposed for general learning of HPs, such as nonparametric triggering kernel estimation (Zhou et al., 2013), learning the Granger causality (Xu et al., 2016), learning short doubly-censored event sequences (Xu et al., 2017) and online estimation (Yang et al., 2017). Recently, Mei and Eisner (2017) proposed neural Hawkes processes, which can model all kinds of complex temporal dependency among the event through a continuous LSTM. Different from these excellent works, our work places a factorization model in a Hawkes process framework, so as to capture and embed abundant, short-term and long-term excitation effects from high-order interaction events.

6 Experiment

6.1 Predictive Performance

Datasets. We first examined the predictive performance of our approach. To this end, we used the following four real-world datasets. (1) *UFO*[1], the reports of UFO sightings over the last century. We extracted the list of two-way interactions *(UFO shape, city)*. The unique numbers of UFO shapes and cities are 28 and 13,713, respectively. We have 70,418 observed interactions from which 45,045 are distinct from each other. (2) *Article*[2], a 12-month log (03/2016 - 02/2017) extracted from CI&T's Internal Communication platform (DeskDrop), including the user operation history on the shared articles, such as LIKE and FOLLOW. We extracted the list of three-way interaction *(user, operation, article id)*. The number of unique users, operations and articles are 1895, 5 and $2,987$, respectively. The number of the observed events is $72,312$ in which there are $50,938$ distinct interactions. (3) *SLC-Crime*[3], police case reports from Salt Lake City, Utah, in the year of 2014. We extracted the events of two-way interactions *(beat, crime type)* where each beat represents a police patrol area. The occurrence of one interaction means one particular type of crime was reported in the specific patrol area (*i.e.,* beat). We have 24 different beats and 26 crime types. The number of observed events is 58,680 and there are 597 unique interactions. (4) *Chicago-Crime*[4], police case reports from Chicago, 2018. Similar to *SLC-Crime*, we extracted the events of *(beat, crime type)*. In Chicago, we have 276 different beats and 32 crime types. The total number of occurred interactions is 262,698 in which 5,757 are unique.

Competing approaches. We compared our method with the following classical and/or state-of-the-art tensor factorization approaches that incorporate the temporal information. (1) CP-PTF, the homogeneous Poisson process (PP) tensor decomposition that uses CP to decompose the event rate of each entry. We applied CP to the logarithm of the rates to ensure positive predictions, without the need for extra constraints on the latent factors. (2) CPT-PTF, which is similar to (Schein et al., 2015). We discretized the time points into steps and augmented the tensor with a time mode. We then introduced time factors for each step. Inside each time step, we still used PPs to decompose the logarithm of event rates. Following (Liang Xiong, 2010), we assigned a conditional Gaussian prior to capture the dependency between the time factors. (3) CP-NPTF, non-homogeneous Poisson process tensor factorization where the event rate of each entry $\mathbf{i}$ is defined as $\lambda_{\mathbf{i}}(t) = t \cdot \exp\big(\mathrm{CP}(\mathbf{i})\big)$. Here $\mathrm{CP}(\mathbf{i})$ is the CP decomposition of the entry $\mathbf{i}$. (4) GP-PTF, the PP tensor factorization using GPs to model the logarithm of event rate as a (nonlinear) function of the latent factors. This is the same formulation of our approach in modeling the background rate. (5) GP-NPTF, non-homogeneous PP tensor factorization using a GP to replace the CP decomposition in the rate defined for CP-NPTF. (6) HP-Local (Zhe and Du, 2018), Hawkes process event-tensor decomposition that uses a local window to define the rate function so to capture the local excitation effects between neighbouring interaction events.

Experimental settings. We varied the number of latent factors from $\{2, 5, 8, 12\}$. For all the methods that involve GPs, we used the same variational sparse GP framework as in our approach, and set the number of pseudo inputs to 128; we used SE-ARD kernel. We implemented our

[1]`https://www.kaggle.com/NUFORC/ufo-sightings/data`

[2]`www.kaggle.com/gspmoreira/articles-sharing-reading-from-cit-deskdrop/data`

[3]`https://opendata.utah.gov/browse?category=Public%20Safety`

[4]`https://data.cityofchicago.org/Public-Safety/Crimes-2018/3i3m-jwuy`

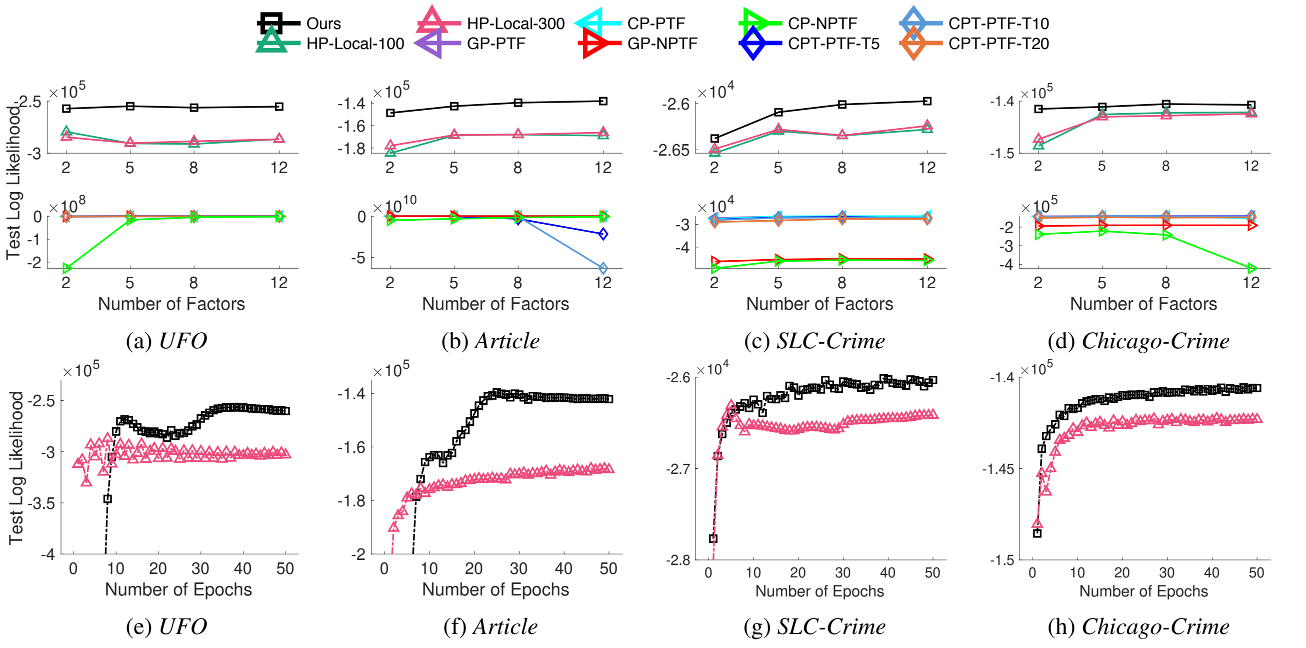


Figure 1: Test log likelihood on each dataset with different numbers of latent factors (top row), and the learning curve when the number of latent factors is set to 8 (second row). CPT-PTF-T{5,10,20} denote running CPT-PTF with 5, 10 and 20 time steps. HP-Local-{100, 300} means running HP-Local with window size 100 and 300.

approach with TensorFlow (Abadi et al., 2016), and used ADAM (Kingma and Ba, 2014) for stochastic optimization. We set the mini-batch size to 64. We chose the learning rate from $\{5 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}, 5 \times 10^{-3}, 10^{-2}\}$. We ran each method for 50 epochs, which has already shown convergence. For HP-Local, we used the original MATLAB implementation[5] for *UFO*, *Article* and *SLC-Crime*. However, it was too slow on *Chicago-Crime*, which includes a much larger number of events; so we re-implemented HP-Local with TensorFlow and used ADAM for stochastic optimization. We chose the same settings, *e.g.,* the mini-batch size and range of learning rates, as in our method. All the other approaches were implemented with MATLAB. For training, we used the first 40K, 50K, 40K and 200K events from *UFO*, *Article*, *SLC-Crime* and *Chicago-Crime*, respectively. We used the remaining 30.4K, 22.3K, 18.7K and 62.7K events for test. For CPT-PTF, the number of time steps was varied from {5, 10, 20}. For HP-Local, we examined the window size from {100, 300}. Note that we used very long test event sequences to examine our method's performance in capturing long-term temporal effects. We calculated the test log-likelihood of each method, and report the results in Figure 1.

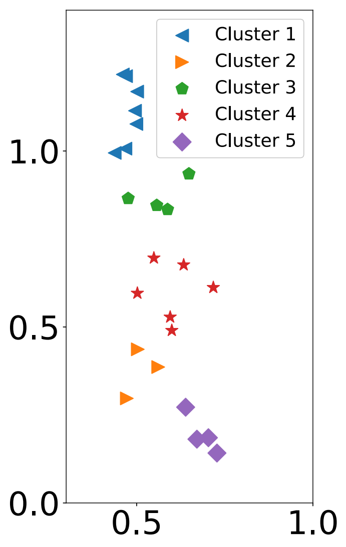
Results. As we can see from Figure 1a-d (top row), our approach consistently outperforms all the competing methods, in many cases by a large margin (*e.g.,* see the results on *UFO*, *Article* and *SLC-Crime*). Hence, it demonstrates the advantages of our factorization approach in terms of prediction accuracy. Note that the second best method is always HP-Local (see the top sub-figures in Figure 1a-d), showing that capturing the temporal dependencies among the interactions is critical to achieve superior predictive performance. The test log-likelihoods of many baselines are close and so their curves overlap. However, in general, CP based approaches are comparable to or worse than GP-based methods (see CP-NPTF, represented by the green curve, for an example in Figure 1a-d). Therefore, it confirms the advantage of the nonlinear decomposition, and also implies the presence of the nonlinear relationships between the entities in the data.

We also examined the learning behaviour of our approach, as shown in Figure 1 e-h. As we can see, our algorithm converges fast. On each dataset, the test log-likelihood achieved the maximum value around 35 epochs and kept stable until the total 50 epochs were finished (although in some cases with mild perturbations). Overall, there are no obvious over-fitting phenomena. The reason might be that our training objective is a lower bound of the model evidence (so is HP-Local), which can effectively prevent our model from over-fitting the training data.

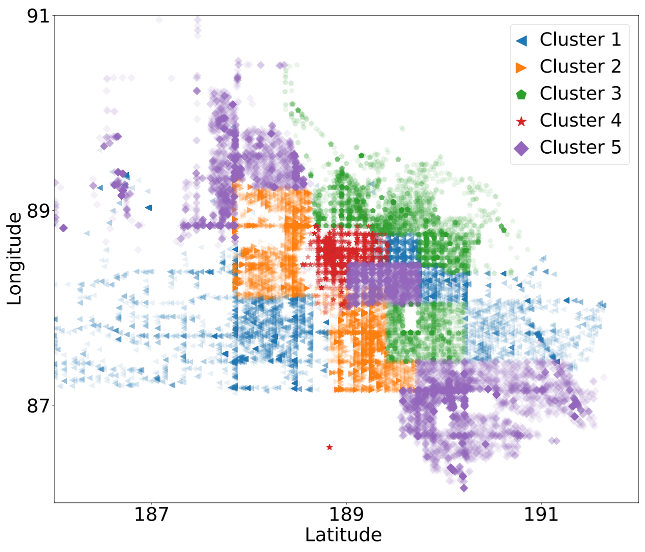
6.2 Structure Investigation

Next, we looked into the hidden structures that can be discovered by our approach. To this end, we set the number of latent factors to 2, and run our algorithm on *SLC-Crime* and *Chicago-Crime*. We first ran the k-means algorithm

[5] https://github.com/yishuaidu/Stochastic-Nonparametric-Event-Tensor-Decomposition



(a) Clusters of the latent factors of the beats.



(b) Location of the crime events.

Figure 2: The structures found in the latent factors of the beats in *SLC-Crime* (a), and the actual locations of the crime events and the beats to which the locations belong to (b).

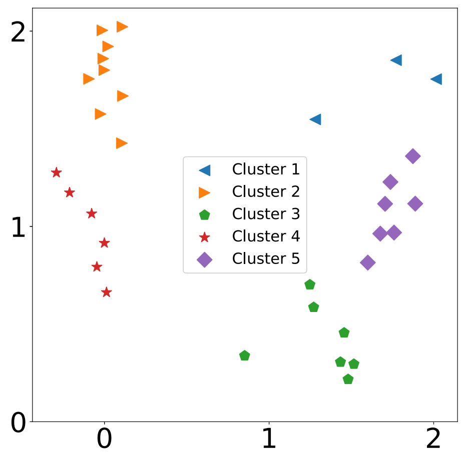


Figure 3: Clusters of the latent factors of the crime types in *Chicago-Crime* data.

on the learned factors of different beats (*i.e.,* police patrol areas) from *SLC-Crime*, and found 5 clusters. Note that we used BIC to select the optimal cluster number. As shown in Figure 2 a, these latent factors have clear structures. Interestingly, these clusters are adjacent successively rather than stay away from each other, implying that their actual locations might be neighbouring as well. To verify this and also examine the distribution of the clusters of the beats in real world, we drew the locations (*i.e.,* latitudes and longitudes) of all the crime events in the data. The locations that belong to the same beat were drawn with the same color, which is consistent with the whole cluster the beat belongs to. As we can see from Figure 2 b, the beats in the same cluster are often neighbouring each other, and further expand to a larger region. This is reasonable, because the crimes can trigger each other in neighbouring areas, for example, gangsters often fight in same or nearby blocks for revenge or competing particular turfs. From Figure 2 b, we can also see that different clusters of beats, forming larger regions, are adjacent successively — this is consistent with the cluster locations in terms of the factor representations.

Next, we ran the k-means algorithm on the latent factors of the 32 crime types from *Chicago-Crime*. Again, we used BIC to identify the optimal cluster number, and found 5 clusters. From Figure 3, we can see the structures are very clear, implying an interesting relationships of different types of crimes. For example, STALKING and SEX OFFENSE are in cluster 1. The two crimes are strongly associated — many victims were sexually offended after being stalked, and in California, people who stalk will be registered as a sex offender[6]. For another example, ARSON and ROBBERY are in the same cluster as well — the two crimes can be associated as well, after robbery, the criminals might commit arson to distract the police so as to find more time to escape.

7 Conclusion

We have presented a Bayesian nonparametric model to factorize high-order interaction events. Our model can capture and embed nonlinear relationships between different entities, as well as various long-term/short-term excitation effects among the events and their decaying patterns. In addition, our model estimation algorithm is efficient and scalable to a large number of events and interaction types.

[6]`https://victimsofcrime.org/our-programs/past-programs/stalking-resource-center/stalking-laws/criminal-stalking-laws-by-state/california`

Acknowledgement

This work was supported by the NSF IIS-1910983.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A system for large-scale machine learning. In 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pages 265–283.

Acar, E., Dunlavy, D. M., Kolda, T. G., and Morup, M. (2011). Scalable tensor factorizations for incomplete data. Chemometrics and Intelligent Laboratory Systems, 106(1):41–56.

Blundell, C., Beck, J., and Heller, K. A. (2012). Modelling reciprocating relationships with hawkes processes. In Advances in Neural Information Processing Systems, pages 2600–2608.

Chi, E. C. and Kolda, T. G. (2012). On tensors, sparsity, and nonnegative factorizations. SIAM Journal on Matrix Analysis and Applications, 33(4):1272–1299.

Choi, J. H. and Vishwanathan, S. (2014). Dfacto: Distributed factorization of tensors. In Advances in Neural Information Processing Systems, pages 1296–1304.

Chu, W. and Ghahramani, Z. (2009). Probabilistic models for incomplete multi-dimensional arrays. AISTATS.

Du, N., Farajtabar, M., Ahmed, A., Smola, A. J., and Song, L. (2015). Dirichlet-hawkes processes with applications to clustering continuous-time document streams. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 219–228. ACM.

Hansen, S., Plantenga, T., and Kolda, T. G. (2015). Newton-based optimization for Kullback-Leibler nonnegative tensor factorizations. Optimization Methods and Software, 30(5):1002–1029.

Harshman, R. A. (1970). Foundations of the PARAFAC procedure: Model and conditions for an"explanatory"multi-mode factor analysis. UCLA Working Papers in Phonetics, 16:1–84.

Hawkes, A. G. (1971). Spectra of some self-exciting and mutually exciting point processes. Biometrika, 58(1):83–90.

He, X., Rekatsinas, T., Foulds, J., Getoor, L., and Liu, Y. (2015). Hawkestopic: A joint model for network inference and topic modeling from text-based cascades. In International conference on machine learning, pages 871–880.

Hensman, J., Fusi, N., and Lawrence, N. D. (2013). Gaussian processes for big data. In Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, pages 282–290. AUAI Press.

Hoff, P. (2011). Hierarchical multilinear models for multiway data. Computational Statistics & Data Analysis, 55:530–543.

Hu, C., Rai, P., and Carin, L. (2015a). Zero-truncated poisson tensor factorization for massive binary tensors. In UAI.

Hu, C., Rai, P., Chen, C., Harding, M., and Carin, L. (2015b). Scalable bayesian non-negative tensor factorization for massive count data. In Proceedings, Part II, of the European Conference on Machine Learning and Knowledge Discovery in Databases - Volume 9285, ECML PKDD 2015, pages 53–70, New York, NY, USA. Springer-Verlag New York, Inc.

Kang, U., Papalexakis, E., Harpale, A., and Faloutsos, C. (2012). Gigatensor: scaling tensor analysis up by 100 times-algorithms and discoveries. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 316–324. ACM.

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

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.

Kolda, T. G. (2006). Multilinear operators for higher-order decompositions, volume 2. United States. Department of Energy.

Liang Xiong, Xi Chen, T.-K. H. J. S. J. G. C. (2010). Temporal collaborative filtering with bayesian probabilistic tensor factorization. In Proceedings of SDM.

Linderman, S. and Adams, R. (2014). Discovering latent network structure in point process data. In International Conference on Machine Learning, pages 1413–1421.

Mei, H. and Eisner, J. M. (2017). The neural hawkes process: A neurally self-modulating multivariate point process. In Advances in Neural Information Processing Systems, pages 6754–6764.

Rai, P., Hu, C., Harding, M., and Carin, L. (2015). Scalable probabilistic tensor factorization for binary and count data. In IJCAI.

Rai, P., Wang, Y., Guo, S., Chen, G., Dunson, D., and Carin, L. (2014). Scalable Bayesian low-rank decomposition of incomplete multiway tensors. In Proceedings of the 31th International Conference on Machine Learning (ICML).

Schein, A., Paisley, J., Blei, D. M., and Wallach, H. (2015). Bayesian poisson tensor factorization for inferring multilateral relations from sparse dyadic event counts. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 1045–1054. ACM.

Schein, A., Zhou, M., Blei, D. M., and Wallach, H. (2016). Bayesian poisson tucker decomposition for learning the structure of international relations. In Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, ICML'16, pages 2810–2819. JMLR.org.

Shashua, A. and Hazan, T. (2005). Non-negative tensor factorization with applications to statistics and computer vision. In Proceedings of the 22th International Conference on Machine Learning (ICML), pages 792–799.

Sutskever, I., Tenenbaum, J. B., and Salakhutdinov, R. R. (2009). Modelling relational data using bayesian clustered tensor factorization. In Advances in neural information processing systems, pages 1821–1828.

Tan, X., Naqvi, S. A., Qi, A. Y., Heller, K. A., and Rao, V. (2016). Content-based modeling of reciprocal relationships using hawkes and gaussian processes. In UAI.

Titsias, M. K. (2009). Variational learning of inducing variables in sparse gaussian processes. In International Conference on Artificial Intelligence and Statistics, pages 567–574.

Tucker, L. (1966). Some mathematical notes on three-mode factor analysis. Psychometrika, 31:279–311.

Wang, Y., Ye, X., Zha, H., and Song, L. (2017). Predicting user activity level in point processes with mass transport equation. In Advances in Neural Information Processing Systems, pages 1644–1654.

Xiong, L., Chen, X., Huang, T.-K., Schneider, J., and Carbonell, J. G. (2010). Temporal collaborative filtering with bayesian probabilistic tensor factorization. In Proceedings of the 2010 SIAM International Conference on Data Mining, pages 211–222. SIAM.

Xu, H., Farajtabar, M., and Zha, H. (2016). Learning granger causality for hawkes processes. In International Conference on Machine Learning, pages 1717–1726.

Xu, H., Luo, D., and Zha, H. (2017). Learning hawkes processes from short doubly-censored event sequences. In nternational Conference on Machine Learning.

Xu, Z., Yan, F., and Qi, Y. (2012). Infinite Tucker decomposition: Nonparametric Bayesian models for multiway data analysis. In Proceedings of the 29th International Conference on Machine Learning (ICML).

Yang, Y. and Dunson, D. (2013). Bayesian conditional tensor factorizations for high-dimensional classification. Journal of the Royal Statistical Society B, revision submitted.

Yang, Y., Etesami, J., He, N., and Kiyavash, N. (2017). Online learning for multivariate hawkes processes. In Advances in Neural Information Processing Systems, pages 4937–4946.

Zhe, S. and Du, Y. (2018). Stochastic nonparametric event-tensor decomposition. In Advances in Neural Information Processing Systems, pages 6856–6866.

Zhe, S., Xu, Z., Chu, X., Qi, Y., and Park, Y. (2015). Scalable nonparametric multiway data analysis. In Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, pages 1125–1134.

Zhe, S., Zhang, K., Wang, P., Lee, K.-c., Xu, Z., Qi, Y., and Ghahramani, Z. (2016). Distributed flexible nonlinear tensor factorization. In Advances in Neural Information Processing Systems, pages 928–936.

Zhou, K., Zha, H., and Song, L. (2013). Learning triggering kernels for multi-dimensional hawkes processes. In International Conference on Machine Learning, pages 1301–1309.