Self-Modulating Nonparametric Event-Tensor Factorization

Abstract

Tensor factorization is a fundamental framework to analyze high-order interactions in data. Despite the success of the existing methods, the valuable temporal information are severely underused. The timestamps of the interactions are either ignored or discretized into crude steps. The recent work although formulates event-tensors to keep the timestamps in factorization and can capture mutual excitation effects among the interaction events, it overlooks another important type of temporal influence, *inhibition*. In addition, it uses a local window to exclude all the long-term dependencies. To overcome these limitations, we propose a self-modulating nonparametric Bayesian factorization model. We use the latent factors to construct mutually-governed, general random point processes, which can capture various short-term/long-term, excitation/inhibition effects, so as to encode the complex temporal dependencies into factor representations. In addition, our model couples with a latent Gaussian process to estimate and fuse nonlinear yet static relationships between the entities. For efficient inference, we derive a fully decomposed model evidence lower bound to dispense with the huge kernel matrix and costly summations inside the rate and log rate functions. We then develop an efficient stochastic optimization algorithm. We show the advantage of our method in four real-world applications.

1. Introduction

Interactions between multiple entities (or nodes) are common in real-world applications. For example, consumers' activities can be viewed as interactions between *customers*, *service items* and *providers*. These (high-order) interactions are naturally represented by tensors and analyzed by tensor factorization, which learns a set of latent factors to represent each participant. With the factor representations, we can discover the hidden structures within the entities, *e.g.,* clusters and outliers, and extract useful features to make predictions (in downstream tasks).

While many excellent methods for tensor factorization have been proposed (Tucker, 1966; Harshman, 1970; Chu and Ghahramani, 2009; Kang et al., 2012; Choi and Vishwanathan, 2014), they mainly conduct a multilinear decomposition, and might be inadequate to capture more complex, nonlinear relationships. More importantly, they severely underuse the valuable temporal information along with the data. Most methods either drop the time stamps of the interactions, summarizing the events as a count tensor (Chi and Kolda, 2012; Hansen et al., 2015; Hu et al., 2015b), or discretize the time stamps into crude steps (*e.g.,* weeks or months), ignoring the temporal dependencies in the same step (Xiong et al., 2010; Schein et al., 2015; 2016). Recently, Zhe and Du (2018) formulates event-tensors (where the tensor entries are sequences of interaction events) to preserve the accurate timestamps. They used Hawkes processes to estimate the fine-grained, triggering effects between the interactions. However, their approach overlooks *inhibition*, another ubiquitous effect between the events, and can still miss important temporal patterns. In addition, to compromise on the computational cost, they use a small time window to restrict the range of dependent events, and cannot capture long-term temporal influences of the interactions.

To overcome these limitations, we propose a self-modulating nonparametric Bayesian factorization model for event tensors, which not only can capture static, nonlinear relationships of the entities, but also is flexible enough to capture a variety of short-term and long-term, excitation and inhibition effects among the interaction events, encoding these complex temporal effects into the factor representations. Specifically, we use the latent factors to construct a set of mutually-governed, general random point processes to sample the observed interaction events. We first use a latent Gaussian process (GP) to sample a function of the factor representations to determine the type of temporal effect between each pair of interactions. The strength of the effect is further modelled as a kernel (similarity) function of their factors. In this way, both the type and strength of the effect are absorbed into the factors, from which we can discover the underlying temporal structures. We then couple with another latent GP to sample the base rate as a (nonlinear) function of the factors, in order to estimate and fuse complex yet static relationships between the entities. We use a scaled softplus function to additively integrate all the positive and negative influences from previous interactions to construct the rate function. For efficient inference, we take advantage of the convexity and log concavity of the rate function, and use the sparse variational GP framework (Hensman et al., 2013) and Jensen's inequality to derive a fully decomposed model evidence lower bound (ELBO). Based on the ELBO, we develop an efficient stochastic optimization algorithm. The complexity of our algorithm is only proportional to the size of the mini-batches, while it captures all the long-term dependencies among the interactions.

For evaluation, we examined our method on four real-world datasets. Our model nearly always achieves better predictive performance than the existing methods using Poisson processes, time factors, and Hawkes processes to incorporate temporal information. The training curves show that our inference algorithm converges reasonably fast and is resistant to overfitting. Finally, by looking into the latent factors estimated by our approach, we found interesting/meaningful structures both within the entities and within the events. We also found interesting temporal influence patterns.

2. Background

Tensor Factorization. We denote a K-mode tensor by $\mathcal{Y} \in \mathbb{R}^{d_1 \times \ldots \times d_K}$. The k-th mode includes d_k entities or nodes (*e.g.,* customers). Each entry is indexed by a tuple $\mathbf{i} = (i_1, \ldots, i_K)$ and stands for the interaction of the corresponding K nodes. The entry value is denoted by $y_{\mathbf{i}}$. To decompose $\mathcal{Y}$, we introduce K latent factor matrices $\mathcal{U} = \{\mathbf{U}^1, \ldots, \mathbf{U}^K\}$ to represent all the tensor nodes. Each $\mathbf{U}^k = [\mathbf{u}_1^k; \ldots; \mathbf{u}_{d_k}^k]^\top$, which is $d_k \times r_k$, and each $\mathbf{u}_t^k$ are the r_k latent factors of node t in mode k. We aim to use $\mathcal{U}$ to recover the observed entries in $\mathcal{Y}$. A classical 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 $\mathcal{W} \in \mathbb{R}^{r_1 \times \ldots \times r_K}$ is a parametric tenor, and $\times_k$ is the mode-k tensor matrix product (Kolda, 2006), which resembles the ordinary matrix-matrix product. If we set all $r_k = r$ and $\mathcal{W}$ to be diagonal, Tucker decomposition becomes CANDECOMP/PARAFAC (CP) decomposition (Harshman, 1970). While numerous tensor factorization 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 form. However, since both forms are mutilinear to the latent factors, they are incapable of capturing more complicated, nonlinear relationships in data.

Factorization with Temporal Information. Real-world tensors are often supplemented with detailed temporal information, namely, the timestamps of the observed interactions.

To incorporate these information, traditional methods either drop the timestamps to perform count tensor decomposition (Chi and Kolda, 2012; Hu et al., 2015b), or discretize the timestamps into time steps, *e.g.,* weeks or months, augment the tensor with a time mode (Xiong et al., 2010; Schein et al., 2015; 2016), and jointly estimate the time factors. Both approaches can be viewed as using Poisson processes to model the interaction events, $p(y_{\mathbf{i}}) \propto e^{-\lambda_{\mathbf{i}} T} \lambda_{\mathbf{i}}^{y_{\mathbf{i}}}$, where $y_{\mathbf{i}}$ is the interaction count in entry $\mathbf{i}$ (with/without a time step), and $\lambda_{\mathbf{i}}$ is the event rate. The factorization is performed on $\{\lambda_{\mathbf{i}}\}$ or $\{\log(\lambda_{\mathbf{i}})\}$, typically with Tucker/CP forms. Despite their simplicity and convenience, these methods disregard the rich and vital temporal dependencies between the interactions, due to the independent increment assumption in Poisson processes. To mitigate this issue, Zhe and Du (2018) formulated *event-tensor* to maintain all the accurate timestamps. In an event-tensor, each entry is an event sequence of a particular interaction, rather than a numerical value. Zhe and Du (2018) modelled the observed entries as a set of mutually excited Hawkes processes (Hawkes, 1971). The rate of the events in each entry $\mathbf{i}$ is

$$\lambda_{\mathbf{i}}(t) = \lambda_{\mathbf{i}}^0 + \sum\nolimits_{s_n \in A(t)} k(\mathbf{x}_{\mathbf{i}_n}, \mathbf{x}_{\mathbf{i}}) h_0(t - s_n) \quad (1)$$

where $\lambda_{\mathbf{i}}^0$ is the background rate, $A(t)$ is a local time window that specifies the range of dependent events happened before t (*e.g.,* 50 past events nearest to t), $\mathbf{i}_n$ is the entry which the previous interaction at time s_n belongs to, $\mathbf{x}_{\mathbf{i}_n}$ and $\mathbf{x}_{\mathbf{i}}$ are the latent factors associated with entry $\mathbf{i}_n$ and $\mathbf{i}$ respectively, $k(\cdot, \cdot)$ is a kernel function that measures their similarity, and $h_0(\cdot)$ is a base triggering kernel that measures how the triggering effect decays along with time. From the rate function (1), we can see that the model can capture the (local) excitation effects of the previously happened interactions on the current one, and the triggering strength are (partly) encoded into the latent factors — the closer the corresponding factors of the two entries, the stronger the strength.

3. Model

Although Zhe and Du (2018)'s model can capture fine-grained, mutual triggering effects among the interactions, it ignores another important class of temporal influence — *inhibition*. The inhibition effect between the events are ubiquitous in real world. For example, a customer who has recently purchased a Surface laptop is unlikely to buy an MacBook; people who voted for one president candidate are unlikely to support another candidate in a short term. In practice, among the interaction events can be mixed excitation and inhibition effects, resulting in complex temporal dependencies. In addition, the model uses a local time window $A(t)$ to specify a small range of the dependent events for each interaction (see (1)). Although this can save much computational cost for the rate function and its logarithm in model estimation (especially for a large number of events), it excludes all the long-term influences of the interaction events on each other, and hence can miss many interesting and valuable temporal patterns. To overcome these problems, we propose a self-modulating nonparametric Bayesian event-tensor factorization model, presented as follows.

3.1. Notations for Event-Tensor

First, let us supplement a few notations. In the event-tensor, for each observed entry $\mathbf{i}$, we denote its event sequence by $y_{\mathbf{i}} = [s^1_{\mathbf{i}}, \ldots, s^{n_{\mathbf{i}}}_{\mathbf{i}}]$, *i.e.,* the time stamps the interaction $\mathbf{i}$ occurred, and $n_{\mathbf{i}}$ is the number of occurrences. Note that each entry represents a particular type of interaction. We can merge the event sequences of all the observed entries into a single sequence, $S = [(s_1, \mathbf{i}_1) \ldots, (s_N, \mathbf{i}_N)]$, where $s_1 \leq \ldots \leq s_N$ are all the time stamps, each $\mathbf{i}_n$ indexes the entry that event s_n belongs to, *i.e.,* the particular interaction occurred at s_n.

3.2. Self-Modulating Nonparametric Factorization

We now consider to use the latent factors $\mathcal{U}$ to construct a general random point process to accommodate both the triggering and inhibition effects among the interaction events. One basic assumption in (Zhe and Du, 2018) (see (1)) is that the closer (or more similar) the factor representations of two interactions, the stronger their mutual excitation effects. This is true in many applications, for example, "the event that *user* A purchased *commodity* B may excite A's friend C to purchase B as well". Obviously, the factors of A and C are expected to be close because they are in the same community (*i.e.,* friends) and so are the factor representations for the interactions (A, B) and (C, B). However, in many other cases, closer factor representations may on the contrary lead to stronger inhibition effects. For example, the event that user A has purchased Surface laptop B can strongly suppress A to buy MacBook C (aforementioned); the event that athlete A has won the champion of Game B deprives of the possibility that his competitor C wins B. Therefore, to model the strength of the temporal influence of a previously occurred interaction $\mathbf{j}$ on the current one $\mathbf{i}$, we still use a kernel function of their factor representations, $k(\mathbf{x}_{\mathbf{j}}, \mathbf{x}_{\mathbf{i}})$, where $\mathbf{x}_{\mathbf{i}} = [\mathbf{u}^1_{i_1}; \ldots; \mathbf{u}^K_{i_K}]$ and $\mathbf{x}_{\mathbf{j}} = [\mathbf{u}^1_{j_1}; \ldots; \mathbf{u}^K_{j_K}]$. However, to detect the type of the influence, we consider to learn a discriminative function of the factor representations, $g(\mathbf{x}_{\mathbf{j}}, \mathbf{x}_{\mathbf{i}})$, where $g(\mathbf{x}_{\mathbf{j}}, \mathbf{x}_{\mathbf{i}}) > 0$ indicate that $\mathbf{j}$ will trigger the occurrence of $\mathbf{i}$ and otherwise inhibit. To flexibly estimate $g(\cdot)$, we place a Gaussian process (GP) prior (Rasmussen and Williams, 2006) — a nonparametric function prior that accommodates various complex functions. Hence, the latent function values $\mathbf{g}$ for every pair of observed entries will follow a multivariate Gaussian distribution,

$$p(\mathbf{g}|\mathcal{U}) = \mathcal{N}\big(\mathbf{g}|\mathbf{0}, \kappa_g(\mathbf{X}_g, \mathbf{X}_g)\big), \tag{2}$$

where each row of the input matrix $\mathbf{X}_g$ corresponds to a pair of entries, and are the concatenation of the associated factors, $\kappa_g(\cdot, \cdot)$ is the covariance (kennel) function.

Now, we define a raw rate function for each entry $\mathbf{i}$ that integrates both the triggering and suppressing effects from the previous interactions,

$$\tilde{\lambda}_{\mathbf{i}}(t) = \lambda^0_{\mathbf{i}} + \sum_{s_n<t} \tanh\big(g(\mathbf{x}_{\mathbf{i}_n}, \mathbf{x}_{\mathbf{i}})\big) k(\mathbf{x}_{\mathbf{i}_n}, \mathbf{x}_{\mathbf{i}}) h_0(t - s_n) \tag{3}$$

where $\lambda^0_{\mathbf{i}}$ is the background rate and $h_0(\cdot)$ is a base kernel that describes the how the strength of the influence decays with time. In our experiments, we chose the commonly used exponential decay kernel, $h_0(\Delta) = \exp(-\frac{\Delta}{\tau})$. Note that we use $\tanh(\cdot)$ to squeeze the values of $g(\cdot)$ into $[-1, 1]$ without changing the sign. The reason is that we use $g(\cdot)$ to just determine the influence types (excitation or inhibition); for clear interpretability, we do not want to confound it with the influence strength (which are modelled by the other components, *i.e.,* $k(\cdot, \cdot)$ and $h_0(\cdot)$).

Next, to obtain a positive rate function so as to build a valid point process, we use a scaled soft-plus function $\gamma_s(\cdot)$ to transform the raw rate $\tilde{\lambda}_{\mathbf{i}}(t)$,

$$\lambda_{\mathbf{i}}(t) = \gamma_s\big(\tilde{\lambda}_{\mathbf{i}}(t)\big) = s \log\big(1 + \exp(\frac{\tilde{\lambda}_{\mathbf{i}}(t)}{s})\big) \tag{4}$$

where $s > 0$. It is trivial to show that when $s \to \infty$, $\lambda_{\mathbf{i}}(t) \to \max\big(\tilde{\lambda}_{\mathbf{i}}(t), 0\big)$. Therefore, the scaled soft-plus can considerably maintain the additive structure in our raw rate definition in (3). While other transformation operators are also possible, *e.g.,* $\exp(\cdot)$, we found empirically that the scaled softplus exhibits superior and excellent performance.

Finally, to estimate the complex yet static relationships between the entities and fuse the relationships into the latent factors, we model the background rate $\lambda^0_{\mathbf{i}}$ in each entry $\mathbf{i}$ as a nonlinear function of the associated factors, $f(\mathbf{x}_{\mathbf{i}})$. To this end, we place another GP prior over $f(\cdot)$. Then the background rate values $\mathbf{f}$ for all the observed entries are sampled from a multivariate Gaussian distribution,

$$p(\mathbf{f}|\mathcal{U}) = \mathcal{N}\big(\mathbf{f}|\mathbf{0}, \kappa_f(\mathbf{X}_f, \mathbf{X}_f)\big), \tag{5}$$

where each row of $\mathbf{X}_f$ are the concatenated factors associated with one entry, and $\kappa_f(\cdot, \cdot)$ is the covariance (or kernel) function. Note that we do not need to constrain $f(\cdot) > 0$, because via the soft-plus transformation (4), we will always obtain a non-negative event rate.

We place a standard Gaussian prior over all the latent factors $\mathcal{U}$. Given the observed interaction events $\mathcal{S}$ (from all the entries), the joint probability of our model is given by

$$\begin{aligned} p(\mathcal{S}, \mathbf{g}, \mathbf{f}, \mathcal{U}) = &\prod_k \prod_{i_k} \mathcal{N}(\mathbf{u}^k_{i_k}|\mathbf{0}, \mathbf{I}) \\ &\cdot \mathcal{N}\big(\mathbf{g}|\mathbf{0}, \kappa_g(\mathbf{X}_g, \mathbf{X}_g)\big)\mathcal{N}\big(\mathbf{f}|\mathbf{0}, \kappa_f(\mathbf{X}_f, \mathbf{X}_f)\big) \\ &\cdot \prod_{\mathbf{i}} \exp\big(-\int_0^T \lambda_{\mathbf{i}}(t)\mathrm{d}t\big) \prod_{n=1}^N \lambda_{\mathbf{i}_n}(s_n) \end{aligned} \tag{6}$$

where T is the total time span across all the events. Note that the last row is the likelihood of our proposed mutually governed, general random point process on the observed entries (Daley and Vere-Jones, 2007).

4. Algorithm

The estimation of our model is challenging. First, the exact inference of our model is infeasible for large data because the GP likelihoods (2) and (5) requires us to compute $M^2 \times M^2$ and $M \times M$ covariance (kernel) matrices respectively and their inverse, where M is the number of observed entries (*i.e.,* distinct interactions). When M is large, the computation is prohibitively costly. Second, the calculation of each rate $\lambda_{\mathbf{i}_n}(s_n)$ in the joint probability (6) needs to go through all the previously happened interactions $\{s_1, \ldots, s_{n-1}\}$ (see (3)), and therefore is expensive for a large number of events N. Third, due to the softplus transformation in (4), the integral over each rate function in (6) does not have a closed form and is intractable to compute.

To address these challenges, we take advantage of the variational sparse GP framework (Hensman et al., 2013) and the properties of our rate function to derive a fully decomposed model evidence lower bound (ELBO). Based on the ELBO, we develop a stochastic, mini-batch optimization algorithm that are efficient to both large M and N. Our algorithm is presented as follows.

4.1. Fully Decomposed Model Evidence Lower Bound

First, to use the variational sparse GP, we introduce pseudo inputs $\mathbf{Z}_g = [\mathbf{z}_1^g, \ldots, \mathbf{z}_{m_g}^g]^\top$ and $\mathbf{Z}_f = [\mathbf{z}_1^f, \ldots, \mathbf{z}_{m_f}^f]^\top$ for the two latent functions $g(\cdot)$ and $f(\cdot)$, respectively, where $m_g \ll M^2$ and $m_f \ll M$. We denote the function values at these pseudo inputs by $\mathbf{b}_g = [g(\mathbf{z}_1^g), \ldots, g(\mathbf{z}_{m_g}^g)]^\top$ and $\mathbf{b}_f = [f(\mathbf{z}_1^f), \ldots, f(\mathbf{z}_{m_f}^f)]$, which we refer to as the pseudo outputs. Then we can augment our model by jointly sampling $\{\mathbf{f}, \mathbf{b}_f\}$ and $\{\mathbf{g}, \mathbf{b}_g\}$. Due to the GP priors of $g(\cdot)$ and $f(\cdot)$, both $\{\mathbf{g}, \mathbf{b}_g\}$ and $\{\mathbf{f}, \mathbf{b}_f\}$ follow a multivariate Gaussian distribution, and the covariance (kernel) matrices are computed on $\{\mathbf{X}_f, \mathbf{Z}_f\}$ and $\{\mathbf{X}_g, \mathbf{Z}_g\}$, respectively. We can further decompose the joint prior by

$$p(\mathbf{g}, \mathbf{b}_g) = p(\mathbf{b}_g) p(\mathbf{g}|\mathbf{b}_g) \tag{7}$$

where $p(\mathbf{b}_g) = \mathcal{N}\big(\mathbf{b}_g|\mathbf{0}, \kappa_g(\mathbf{Z}_g, \mathbf{Z}_g)\big)$, $p(\mathbf{g}|\mathbf{b}_g) = \mathcal{N}(\mathbf{g}|\mathbf{m}_{g|b}, \boldsymbol{\Sigma}_{g|b})$ is a conditional Gaussian distribution, $\mathbf{m}_{g|b} = \kappa_g(\mathbf{X}_g, \mathbf{Z}_g)\kappa_g(\mathbf{Z}_g, \mathbf{Z}_g)^{-1}\mathbf{b}_g$ and $\boldsymbol{\Sigma}_{g|b} = \kappa_g(\mathbf{X}_g, \mathbf{X}_g) - \kappa_g(\mathbf{X}_g, \mathbf{Z}_g)\kappa_g(\mathbf{Z}_g, \mathbf{Z}_g)^{-1}\kappa_g(\mathbf{Z}_g, \mathbf{X}_g)$. Similarly, we can decompose

$$\begin{aligned} p(\mathbf{f}, \mathbf{b}_f) &= p(\mathbf{b}_f) p(\mathbf{f}|\mathbf{b}_f) \\ &= \mathcal{N}\big(\mathbf{b}_f|\mathbf{0}, \kappa_f(\mathbf{Z}_f, \mathbf{Z}_f)\big)\mathcal{N}(\mathbf{f}|\mathbf{m}_{f|b}, \boldsymbol{\Sigma}_{f|b}) \end{aligned} \tag{8}$$

where $\mathbf{m}_{f|b}$ and $\boldsymbol{\Sigma}_{f|b}$ are the conditional mean and covariance matrix given $\mathbf{b}_f$ respectively, similar to $\mathbf{m}_{g|b}$ and $\boldsymbol{\Sigma}_{g|b}$. The joint probability of the augmented model is then

$$\begin{aligned} &p(\mathcal{S}, \mathbf{g}, \mathbf{b}_g, \mathbf{f}, \mathbf{b}_f, \mathcal{U}) \\ &= p(\mathbf{b}_g) p(\mathbf{g}|\mathbf{b}_g) p(\mathbf{b}_f) p(\mathbf{f}|\mathbf{b}_f) p(\mathcal{U}, \mathcal{S}|\mathbf{f}, \mathbf{g}) \end{aligned} \tag{9}$$

where $p(\mathcal{U}, \mathcal{S}|\mathbf{f}, \mathbf{g}) = \prod_k \prod_{i_k} \mathcal{N}(\mathbf{u}_{i_k}^k|\mathbf{0}, \mathbf{I}) \prod_{\mathbf{i}} \exp\big(-\int_0^T \lambda_{\mathbf{i}}(t)\mathrm{d}t\big) \prod_{n=1}^N \lambda_{\mathbf{i}_n}(s_n)$. Note that if we marginalize out the pseudo outputs $\mathbf{b}_g$ and $\mathbf{b}_f$, we will recover the original model (6). Based on (9), we now construct a variational model evidence lower bound (ELBO) to avoid calculating the full covariance matrices $\kappa_g(\mathbf{X}_g, \mathbf{X}_g)$ and $\kappa_f(\mathbf{X}_f, \mathbf{X}_f)$, which is infeasible for large M. To do so, we introduce a variational posterior for $\{\mathbf{g}, \mathbf{b}_g, \mathbf{f}, \mathbf{b}_f\}$,

$$q(\mathbf{g}, \mathbf{b}_g, \mathbf{f}, \mathbf{b}_f) = q(\mathbf{b}_g) p(\mathbf{g}|\mathbf{b}_g) q(\mathbf{b}_f) p(\mathbf{f}|\mathbf{b}_f), \tag{10}$$

where $q(\mathbf{b}_g) = \mathcal{N}(\mathbf{b}_g|\boldsymbol{\mu}_g, \mathbf{S}_g)$ and $q(\mathbf{b}_f) = \mathcal{N}(\mathbf{b}_f|\boldsymbol{\mu}_f, \mathbf{S}_f)$. We further parameterize $\mathbf{S}_g$ and $\mathbf{S}_f$ by their Cholesky decompositions, $\mathbf{S}_g = \mathbf{L}_g\mathbf{L}_g^\top$ and $\mathbf{S}_f = \mathbf{L}_f\mathbf{L}_f^\top$, to ensure their positive definiteness. We then derive the EBLO from

$$\begin{aligned} \mathcal{L} &= \mathbb{E}_{q(\mathbf{g}, \mathbf{b}_g, \mathbf{f}, \mathbf{b}_f)} \log \frac{p(\mathcal{S}, \mathbf{g}, \mathbf{b}_g, \mathbf{f}, \mathbf{b}_f, \mathcal{U})}{q(\mathbf{g}, \mathbf{b}_g, \mathbf{f}, \mathbf{b}_f)} \\ &= \mathbb{E}_q \log \frac{p(\mathbf{b}_g)\cancel{p(\mathbf{g}|\mathbf{b}_g)}p(\mathbf{b}_f)\cancel{p(\mathbf{f}|\mathbf{b}_f)}p(\mathcal{U}, \mathcal{S}|\mathbf{f}, \mathbf{g})}{q(\mathbf{b}_g)\cancel{p(\mathbf{g}|\mathbf{b}_g)}q(\mathbf{b}_f)\cancel{p(\mathbf{f}|\mathbf{b}_f)}}. \end{aligned}$$

Now we can see that the full conditional Gaussian distributions $p(\mathbf{g}|\mathbf{b}_g)$ and $p(\mathbf{f}|\mathbf{b}_f)$ are both cancelled. We only need to compute the covariance matrices for $p(\mathbf{b}_g)$ and $p(\mathbf{b}_f)$, which are $m_g \times m_g$ and $m_f \times m_f$, respectively. Hence, the computational cost is greatly reduced. Rearranging the terms, we have

$$\begin{aligned} \mathcal{L} &= \log(p(\mathcal{U})) - \mathrm{KL}\big(q(\mathbf{b}_g)\|p(\mathbf{b}_g)\big) - \mathrm{KL}\big(q(\mathbf{b}_f)\|p(\mathbf{b}_f)\big) \\ &- \sum_{\mathbf{i}} \mathbb{E}_q\Big[\int_0^T \lambda_{\mathbf{i}}(t)\mathrm{d}t\Big] + \sum_{n=1}^N \mathbb{E}_q\big[\log\big(\lambda_{\mathbf{i}_n}(s_n)\big)\big], \end{aligned} \tag{11}$$

where $p(\mathcal{U}) = \prod_k \prod_{i_k} \mathcal{N}(\mathbf{u}_{i_k}^k|\mathbf{0}, \mathbf{I})$ and $\mathrm{KL}(\cdot\|\cdot)$ is the Kullbalk Leibler divergence. To handle the intractable integral in (11), we rewrite it as an expectation, $\int_0^T \lambda_{\mathbf{i}}(t)\mathrm{d}t = \mathbb{E}_{p(t)}[T\lambda_{\mathbf{i}}(t)]$ where $p(t) = \mathrm{Uniform}(0, T)$. Then we can sample t to obtain an unbiased estimate of the integral and conduct stochastic optimization (which we will discuss later). The ELBO now is

$$\begin{aligned} \mathcal{L} &= \log(p(\mathcal{U})) - \mathrm{KL}\big(q(\mathbf{b}_g)\|p(\mathbf{b}_g)\big) - \mathrm{KL}\big(q(\mathbf{b}_f)\|p(\mathbf{b}_f)\big) \\ &- \sum_{\mathbf{i}} \mathbb{E}_q\mathbb{E}_{p(t)}[T\lambda_{\mathbf{i}}(t)] + \sum_{n=1}^N \mathbb{E}_q\big[\log\big(\lambda_{\mathbf{i}_n}(s_n)\big)\big]. \end{aligned} \tag{12}$$

However, the computation of each rate $\lambda_{\mathbf{i}}(t)$ and log rate $\log\big(\lambda_{\mathbf{i}_n}(s_n)\big)$ is still quite expensive. According to (3) and (4), they couple a summation of the temporal influences (excitation or inhibition) from all the previously happened events in the (scaled) softplus and log-softplus function, and the time complexity is (on average) $\mathcal{O}(N)$. Since we need to compute N log rates, the total complexity is $\mathcal{O}(N^2)$. Therefore, it will be very costly for large N. To address this issue, we observe the following fact.

Lemma 4.1. *The scaled soft-plus function* $\gamma_s(x) = s\log\big(1+\exp(x/s)\big)$ *($s > 0$) is convex and* $\log\big(\gamma_s(x)\big)$ *is concave.*

The proof is given in the supplementary material. Based on this property, we can use Jensen's inequality to further derive an ELBO that fully decomposes these expensive summations. Specifically, we first rewrite the raw rate function (3) as

$$\tilde{\lambda}_{\mathbf{i}}(t) = \lambda^0_{\mathbf{i}} + \sum_{n=1}^{N} \delta(s_n < t) h_{\mathbf{i}_n \to \mathbf{i}}(\mathbf{x}_{\mathbf{i}_n}, \mathbf{x}_{\mathbf{i}}, t - s_n)$$

where $\delta(\cdot)$ is the indicator function and $h_{\mathbf{i}_n \to \mathbf{i}}(\mathbf{x}_{\mathbf{i}_n}, \mathbf{x}_{\mathbf{i}}, t - s_n) = \tanh\big(g(\mathbf{x}_{\mathbf{i}_n}, \mathbf{x}_{\mathbf{i}})\big) k(\mathbf{x}_{\mathbf{i}_n}, \mathbf{x}_{\mathbf{i}}) h_0(t - s_n)$. We then partition the observed events into mini-batches of size Q: $\{\mathcal{B}_1, \ldots, \mathcal{B}_{N/Q}\}$, and rearrange the summation as $\tilde{\lambda}_{\mathbf{i}}(t) = \lambda^0_{\mathbf{i}} + \frac{Q}{N}\sum_{k=1}^{N/Q} \frac{N}{Q} \sum_{n \in \mathcal{B}_k} \delta(s_n < t) h_{\mathbf{i}_n \to \mathbf{i}}(\mathbf{x}_{\mathbf{i}_n}, \mathbf{x}_{\mathbf{i}}, t - s_n)$. Thereby, we can view the raw rate as an expectation, $\tilde{\lambda}_{\mathbf{i}}(t) = \mathbb{E}_{p(k)}[X^{\mathbf{i}}_k]$ where $p(k) = Q/N$, $k \in \{1, \ldots, N/Q\}$, and

$$X^{\mathbf{i}}_k = \lambda^0_{\mathbf{i}} + \frac{N}{Q} \sum_{n \in \mathcal{B}_k} \delta(s_n < t) h_{\mathbf{i}_n \to \mathbf{i}}(\mathbf{x}_{\mathbf{i}_n}, \mathbf{x}_{\mathbf{i}}, t - s_n).$$

Since the rate $\lambda_{\mathbf{i}}(t) = \gamma_s\big(\tilde{\lambda}_{\mathbf{i}}(t)\big)$ and $\gamma_s(\cdot)$ is convex, we can apply Jensen's inequality to obtain $\lambda_{\mathbf{i}}(t) = \gamma_s(\mathbb{E}_{p(k)}[X^{\mathbf{i}}_k]) \leq \mathbb{E}_{p(k)}[\gamma_s(X^{\mathbf{i}}_k)]$ and so

$$-\mathbb{E}_q \mathbb{E}_{p(t)}[\lambda_{\mathbf{i}}(t)] \geq -\mathbb{E}_q \mathbb{E}_{p(t)} \mathbb{E}_{p(k)}[\gamma_s(X^{\mathbf{i}}_k)]. \tag{13}$$

Similarly, the raw rate inside each log rate $\lambda_{\mathbf{i}_n}(s_n)$ can also be viewed as an expectation, $\tilde{\lambda}_{\mathbf{i}}(s_n) = \mathbb{E}_{p(k)}[Y^n_k]$, where

$$Y^n_k = \lambda^0_{\mathbf{i}_n} + \frac{N}{Q} \sum_{j \in \mathcal{B}_k} \delta(s_j < s_n) h_{\mathbf{i}_j \to \mathbf{i}_n}(\mathbf{x}_{\mathbf{i}_j}, \mathbf{x}_{\mathbf{i}_n}, s_n - s_j).$$

Since $\log(\gamma_s(\cdot))$ is concave, we can apply Jensen's inequality to obtain

$$\begin{aligned} \log\big(\lambda_{\mathbf{i}_n}(s_n)\big) &= \log(\gamma_s(\mathbb{E}_{p(k)}[Y^n_k])) \\ &\geq \mathbb{E}_{p(k)}[\log(\gamma_s(Y^n_k))]. \end{aligned} \tag{14}$$

Finally, we substitute the lower bounds in (13) and (14) for each expected rate and log rate in (12), respectively. We then obtain a fully decomposed ELBO,

$$\begin{aligned} \mathcal{L}^+ &= \log(p(\mathcal{U})) - \mathrm{KL}\big(q(\mathbf{b}_g)\|p(\mathbf{b}_g)\big) - \mathrm{KL}\big(q(\mathbf{b}_f)\|p(\mathbf{b}_f)\big) \\ &- \sum_{\mathbf{i}} \mathbb{E}_q \mathbb{E}_{p(t)} \mathbb{E}_{p(k)}[T\gamma_s(X^{\mathbf{i}}_k)] + \sum_{n=1}^{N} \mathbb{E}_q \mathbb{E}_{p(k)}[\log(\gamma_s(Y^n_k))]. \end{aligned} \tag{15}$$

In this way, we move out most of the summation in each softplus and log softplus function, leaving only a very light summation across the mini-batch, *i.e.,* $X^{\mathbf{i}}_k$ and Y^n_k. The ELBO is additive on the observed entries, events and mini-batch set (via $p(k)$). Thereby, we can develop a stochastic optimization algorithm for efficient model estimation.

4.2. Stochastic Optimization

We now maximize the ELBO $\mathcal{L}^+$ in (15) to estimate the variational posterior q, the latent factors $\mathcal{U}$ and the other parameters. This ELBO is not analytical because the expectation terms are for the softplus or log softplus functions and do not have closed forms . Hence, we resort to stochastic optimization. In order to develop an efficient algorithm, we further partition all the observed entries (*i.e.,* distinct interactions) into mini-batches of size C: $\{\mathcal{C}_1, \ldots, \mathcal{C}_{M/C}\}$, and all the observed events into mini-batches of D: $\{\mathcal{D}_1, \ldots, \mathcal{D}_{N/D}\}$. Note that we can also reuse the previous partition of the events, $\{\mathcal{B}_1, \ldots, \mathcal{B}_{N/Q}\}$. Next, we rearrange

$$\begin{aligned} \mathcal{L}^+ &= \log(p(\mathcal{U})) - \mathrm{KL}\big(q(\mathbf{b}_g)\|p(\mathbf{b}_g)\big) - \mathrm{KL}\big(q(\mathbf{b}_f)\|p(\mathbf{b}_f)\big) \\ &- \sum_j \frac{C}{M} \sum_{\mathbf{i} \in \mathcal{C}_j} \frac{M}{C} \mathbb{E}_q \mathbb{E}_{p(t)} \mathbb{E}_{p(k)}[T\gamma_s(X^{\mathbf{i}}_k)] \\ &+ \sum_l \frac{D}{N} \sum_{n \in \mathcal{D}_l} \frac{N}{D} \mathbb{E}_q \mathbb{E}_{p(k)}[\log(\gamma_s(Y^n_k))]. \end{aligned} \tag{16}$$

Then we can view the ELBO as an expectation of a stochastic ELBO,

$$\mathcal{L}^+ = \mathbb{E}_{p(k), p(j), p(l)}[\tilde{\mathcal{L}}^+_{k,j,l}] \tag{17}$$

where $p(k) = \frac{Q}{N}$, $k \in \{1, \ldots, \frac{N}{Q}\}$, $p(j) = \frac{C}{M}$, $j \in \{1, \ldots, \frac{M}{C}\}$, $p(l) = \frac{D}{N}$, $l \in \{1, \ldots, \frac{N}{D}\}$, and

$$\begin{aligned} \tilde{\mathcal{L}}^+_{k,j,l} &= \log(p(\mathcal{U})) - \mathrm{KL}\big(q(\mathbf{b}_g)\|p(\mathbf{b}_g)\big) - \mathrm{KL}\big(q(\mathbf{b}_f)\|p(\mathbf{b}_f)\big) \\ &- \mathbb{E}_q \mathbb{E}_{p(t)}[T\gamma_s(X^{\mathbf{i}}_k)] + \mathbb{E}_q[\log(\gamma_s(Y^n_k))]. \end{aligned}$$

Now with (17), we can develop an efficient stochastic optimization algorithm. Each time, we first draw a mini-batch $\mathcal{B}_k$, $\mathcal{C}_j$ and $\mathcal{D}_l$ from $p(k)$, $p(j)$ and $p(l)$ respectively, and then seek to compute $\nabla \tilde{\mathcal{L}}^+_{k,j,l}$ as an unbiased estimate of $\nabla \mathcal{L}^+$. However, the expectation term in $\tilde{\mathcal{L}}^+_{k,j,l}$ is still intractable. To address this issue, we use the reparameterization trick (Kingma and Welling, 2013) to generate a parameterized sample for $\mathbf{b}_f$ and each background rate $\lambda^0_{\mathbf{i}} = f(\mathbf{x}_{\mathbf{i}})$ in $X^{\mathbf{i}}_k$ and Y^n_k. Since $q(\mathbf{b}_f)$ is Gaussian and $p(f(\mathbf{x}_{\mathbf{i}})|\mathbf{b}_f)$ is conditional Gaussian, it is straightforward to obtain the sample $\tilde{\mathbf{b}}_f = \boldsymbol{\mu}_f + \mathbf{L}_f \boldsymbol{\eta}$ and $\tilde{\lambda}^0_{\mathbf{i}} = \mu^0_{\mathbf{i}} + \sigma^0_{\mathbf{i}}\epsilon$, where $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\epsilon \sim \mathcal{N}(0, 1)$, $\mu^0_{\mathbf{i}} = \kappa_f(\mathbf{x}_{\mathbf{i}}, \mathbf{Z}_f)\kappa_f(\mathbf{Z}_f, \mathbf{Z}_f)^{-1}\tilde{\mathbf{b}}_f$ and $(\sigma^0_{\mathbf{i}})^2 = \kappa_f(\mathbf{x}_{\mathbf{i}}, \mathbf{x}_{\mathbf{i}}) - \kappa_f(\mathbf{x}_{\mathbf{i}}, \mathbf{Z}_f)\kappa_f(\mathbf{Z}_f, \mathbf{Z}_f)^{-1}\kappa_f(\mathbf{Z}_f, \mathbf{x}_{\mathbf{i}})$. Similarly, we can generate parameterized samples for $\mathbf{b}_g$ and each $g(\mathbf{x}_{\mathbf{i}_n}, \mathbf{x}_{\mathbf{i}})$ in $X^{\mathbf{i}}_k$ and Y^n_k. We then generate a sample for t from $p(t)$.

Now, we substitute all the parameterized samples for the corresponding random variables in $X_k^{\mathbf{i}}$ and Y_k^n, and obtain an unbiased stochastic estimate of $\tilde{\mathcal{L}}_{k,j,l}^+$. We then compute its gradient to obtain the an unbiased estimate of $\nabla\tilde{\mathcal{L}}_{k,j,l}^+$, which in turn is an unbiased estimate of $\nabla\mathcal{L}^+$. We now can apply any stochastic optimization algorithm to maximize $\mathcal{L}^+$ with the stochastic gradient. Note that the computation of the stochastic gradient is only inside the sampled mini-batches. Therefore, the cost is greatly reduced.

4.3. Algorithm Complexity

The time complexity of our inference is $\mathcal{O}\big(Q(C+D)m_g^3 + (C+D)m_f^3\big)$ where Q, C and D are the mini-batch sizes of the three partitions. Therefore, the computational cost is proportional to the mini-batch sizes, rather than determined by the total number of entries M and events N. The space complexity is $\mathcal{O}(m_g^2 + m_f^2 + \sum_{k=1}^K d_k r_k)$, which is to store the prior and posterior matrices for the pseudo outputs $\mathbf{b}_g$ and $\mathbf{b}_f$ and the latent factors $\mathcal{U}$.

5. Related Work

Classical tensor factorization methods include CP (Harshman, 1970) and Tucker (Tucker, 1966) decompositions, which are multilinear. Many other approaches have been proposed based on them (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), to name a few. Recently, several Bayesian nonparametric factorization models (Xu et al., 2012; Zhe et al., 2015; 2016) were proposed to estimate the nonlinear relationships in data. To handle temporal information, current methods either factorize the counts instead (Chi and Kolda, 2012; Hansen et al., 2015; Hu et al., 2015b), or discretize the time stamps into crude steps and then factorize the counts in each step (Xiong et al., 2010; Schein et al., 2015; 2016). Despite their success, these methods ignore the rich temporal dependencies between the interactions, and can miss important temporal patterns. Recently, Zhe and Du (2018) used the Hawkes processes to capture various, fine-grained triggering effects among the interactions. The method outperforms the previous approaches in terms of prediction, and discovers interesting clusters with temporal meanings. However, this method ignores another important temporal effect, inhibition, hence can still fail to capture complex, mixed temporal dependency patterns. In addition, the method sets a local time window to enable efficient computation, but meanwhile misses the valuable, long-range influences of the events on each other. To overcome these problems, we use latent factors to construct more flexible random point processes to detect and disentangle a variety of mixed excitation and inhibition effects, encoding them into the latent factors. In addition, we develop an efficient inference algorithm that is able to capture all kinds of short-term and long-term dependencies.

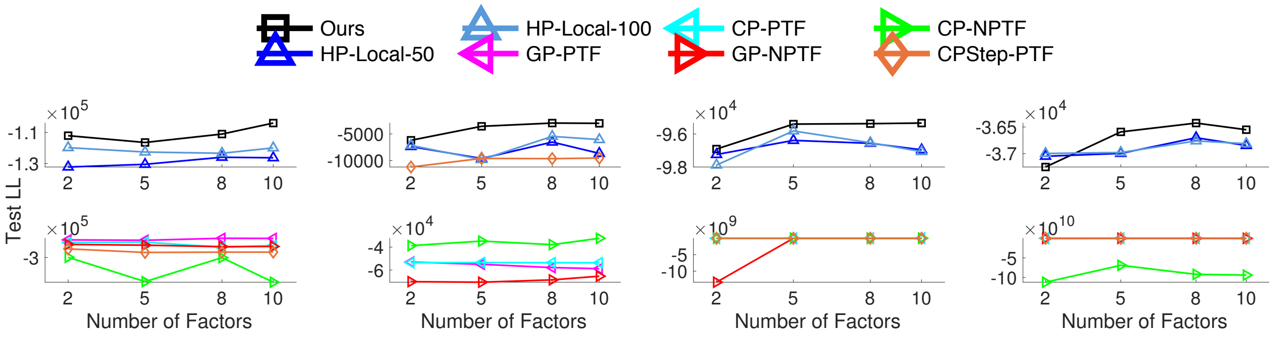
Hawkes process (HP) is a popular class of random point processes to study mutual excitations within various events. Many models use HPs to discover temporal relationships, *e.g.,* (Blundell et al., 2012; Tan et al., 2016; Linderman and Adams, 2014; Du et al., 2015; He et al., 2015; Wang et al., 2017). Several pieces of work were also proposed to improve the learning of HPs, such as nonparametric triggering kernel estimation (Zhou et al., 2013), Granger causality (Xu et al., 2016), 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 uses a continuous LSTM to model complex temporal dependencies among the events (*e.g.,* triggering, suppression and their nonlinear interactions). They also tested to use the softplus function to integrate different temporal effects. Distinct from their excellent work, our method (1) uses the latent factors (rather than free parameters) to construct the point process to encode the temporal effects into the factor representations, and (2) investigates the properties of the rate function to fulfill efficient inference, especially for large numbers of events and event types.

6. Experiment

6.1. Predictive Performance

Datasets. We first examined the predictive performance of our approach on the following four real-world datasets. (1) *Taobao* (`https://tianchi.aliyun.com/dataset/dataDetail?dataId=53`), online shopping behaviours between 07/01/2015 and 11/30/2015 in the largest retail platforms of China. We extracted a 5-mode tensor, of size $980 \times 274 \times 631 \times 58 \times 2$, which represents the interactions of *(user, seller, item, category, action)*. The action can be either "buy" or "click". The number of observed events are $69,833$ where there are $16,609$ distinct interactions (*i.e.,* entries). (2) *UFO* (Zhe and Du, 2018), the UFO sighting reports in 20th century. The dataset is a two-mode event-tensor *(UFO shape, city)*, of size $28 \times 13,713$, including $45,045$ observed entries and $70,418$ sighting events. (3) *Crash* (`https://www.kaggle.com/usdot/nhtsa-traffic-fatalities`), the report of fatal traffic crashes in US, year 2015. We extracted a 4-mode tensor *(state, county, city, landuse-type)*, of size $51 \times 288 \times 2,098 \times 5$. Each entry consists of a sequence of crash events. There are $8,691$ entries and $32,052$ events in total. (4) *LastFM* (`https://grouplens.org/datasets/hetrec-2011/`), a music tagging dataset. From the most active users, popular artists and tags, we extracted a three-mode event-tensor, *(user, artist, tag)*, of size is $191 \times 200 \times 184$. The tensor has $13,399$ observed entries and the same number of events. That means each interaction occurred only once.

Competing Methods. We compared with the following

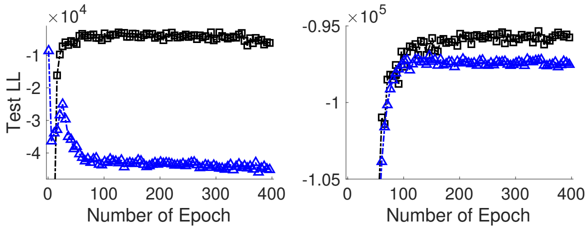


(a) *Taobao* (b) *UFO* (c) *Crash* (d) *LastFM*

Figure 1. Test log likelihood on each dataset with different number of latent factors. HP-Local-$\{50, 100\}$ means running HP-Local with window size 50 and 100. The sub-figures on the top include the results only for our method and HP-Local, except that in (b), CPStep-PTF is also included. All the remaining methods are shown in the bottom sub-figures and their performance are much worse. Note that the test log likelihoods of several baselines are very close on *Crash* and *LastFM* and so their curves overlap.

popular and/or state-of-the-art tensor factorization methods incorporating the temporal information. (1) CP-PTF, the homogeneous Poisson process (PP) tensor factorization, which uses CP to factorize the event rate of each entry. To obtain non-negative rates, we conducted the factorization in the log domain. (2) CPStep-PTF, which is similar to (Schein et al., 2015). It discretizes the timestamps into 5 steps and adds a time mode to the tensor. Time factors are introduced to represent each step and jointly estimated with the other factors. CP is applied to factorize the event rates of each entry in the log domain. (3) GP-PTF, which uses GPs to estimate the log rate of each entry as a nonlinear function of the associated latent factors. This is the same as our approach in modeling the base rate. (4) CP-NPTF, non-homogeneous Poisson process tensor factorization where the event rate is modelled as $\lambda_{\mathbf{i}}(t) = t \cdot \exp\left(\mathrm{CP}(\mathbf{i})\right)$ for each $\mathbf{i}$. Here $\mathrm{CP}(\mathbf{i})$ is CP factorization of entry $\mathbf{i}$. (5) GP-NPTF, non-homogeneous PP tensor factorization that replaces CP with GP in the rate modeling for CP-NPTF. (6) HP-Local (Zhe and Du, 2018), Hawkes process event-tensor decomposition using a local time window to define the rate function and to estimate the local triggering effects among the neighbouring events.

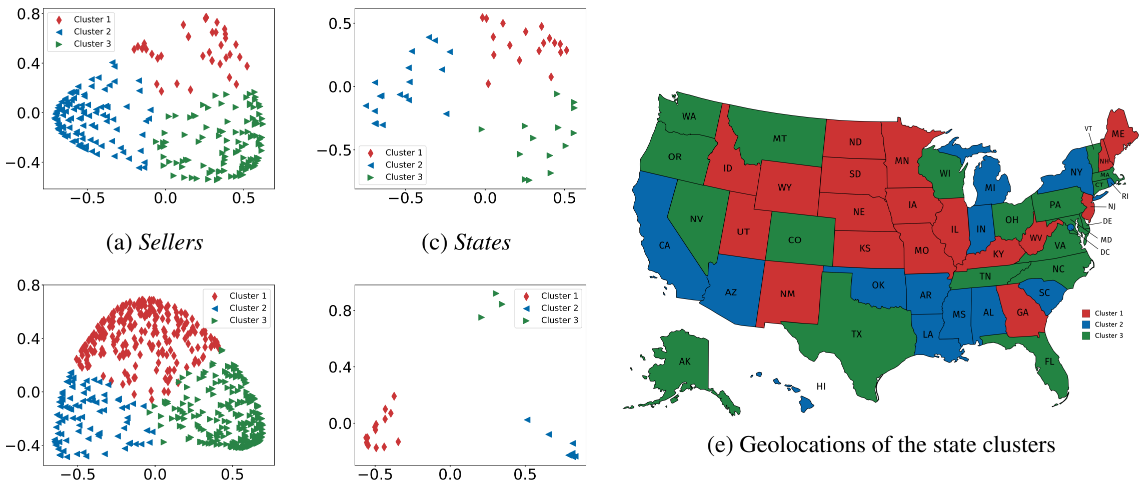
Experimental settings. For all the competing methods that use GPs, we applied the same sparse GP framework as in our method for scalable inference: we set the number of pseudo inputs to 100 and used SE-ARD kernel for which we initialized all the parameters to 1. For a fair comparison, all the methods were initialized with the same set of factors that were independently sampled from $\mathrm{Uniform}(0, 1)$. We implemented our approach and HP-Local with PyTorch and used ADAM (Kingma and Ba, 2014) for stochastic optimization. For both methods, the mini-batch size was set to 100. The learning rate was chosen from $\{5\times10^{-4}, 10^{-3}, 3\times10^{-3}, 5\times10^{-3}, 10^{-2}\}$. We ran each method for 400 epochs, which are sufficient for convergence. All the other methods were implemented with MATLAB. For training, we used the first 40K, 40K, 20K and 10K events from *Taobao*, *UFO*, *Crash* and *LastFM*, respectively. The remaining 29.8K, 30.4K, 12K and 3.3K events were used for test. For HP-Local, we varied the window size from $\{50, 100\}$. Note that the event sequences for test are very long, through which we can examine the performance of our method in capturing long-term temporal effects. We varied the number of the latent from $\{2, 5, 8, 10\}$. We calculated the test log-likelihood of each method, and report the results in Figure 1.



(a) *UFO* (b) *Crash*

Figure 2. The test log likelihood of our method and HP-Local-50 along with the number of training epochs.

Results. As we can see from Fig. 1, our approach consistently outperforms all the competing method except that on *LastFM*, our method is a bit worse than HP-Local when the number of latent factors is 2. The results demonstrate the advantage of our method in terms of predictive performance. Note that in general, when we increased the window size, HP-Local obtained better or similar prediction accuracy (*e.g.,* Fig. 1a); hence it shows that estimating long-range temporal dependencies can help improve the predictive performance. The other competing methods, *e.g.,* CP-PTF, GP-PTF, CP-NPTF and GP-NPTF, are far worse than our model and HP-Local in almost all the cases. For better illustration, we show their test log likelihoods in separate sub-figures at the bottom. Note that these methods are based on homogeneous or non-homogeneous Poisson processes and disregard the temporal dependencies among the interaction events. Therefore, their results further demonstrate the benefit of our model being capable of capturing various,



(a) *Sellers*

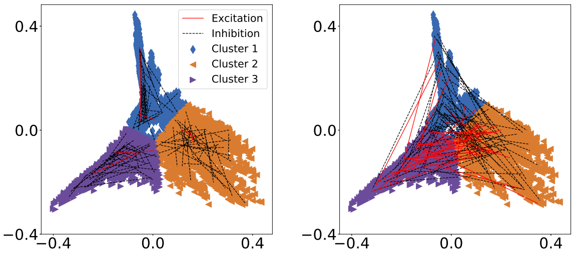
(c) *States*

(e) Geolocations of the state clusters

(b) *Items*

(d) *UFO shapes*

Figure 3. Structures reflected from the latent factors learned by our model on *Taobao* (a, b), *Crash* (c, e) and *UFO* (d). Colors indicate cluster memberships.



(a) *Within clusters*

(b) *Across clusters*

Figure 4. Structures of the events and their temporal influences in *LastFM*.

complex temporal dependencies among the events.

We then investigated the learning behaviour of our model, as shown in Fig. 2. Our learning algorithm converges reasonably fast (around 100 epochs on both *UFO* and *Crash*), and then keeps stable after the convergence. Hence, there is not an apparent overfitting issue and we do not need to use extra tricks such as early stopping. The reason might be that our training objective (15) is a lower bound of the model evidence and can effectively resist overfitting (like many other variational inference algorithms).

6.2. Structure Discovery

Next, we examined if our method can discover hidden structures in the data. To this end, we set the number of latent factors to 10 and ran our algorithm on *Crash*, *UFO* and *Taobao*. Then we applied kernel PCA (Schölkopf et al., 1998) (with RBF kernel) to project the latent factors onto the x-y plane. We then ran k-means to find the potential clusters in the (projected) factor representations of the entities. We used the elbow method (Ketchen and Shook, 1996) to select the number of clusters. As we can see from Fig. 3, these factors reflected clear grouping structures. Note that *Taobao* data have been anonymized so we cannot investigate the meaning of those clusters. But they can be potentially useful for tasks like recommendation (Tran et al., 2018) and click-through-rate prediction (Pan et al., 2019). Then for *Crash* data, we show the actual geolocations of the clustered states in Figure 3e. We can see that states grouped together are often neighbouring with each other. The clusters can be roughly seen as the middle (red), the south (blue) and the east/west coast (green). This is reasonable — the states close by may have similar patterns of the traffic crash events and their chain effects (inhibition/excitation) due to similar driving customs, roads layout and weather patterns.

We also looked into the triggering and inhibition effects among the interaction evens. To this end, we ran our algorithm on *LastFM*. We represented each music tagging event by concatenating the latent factors of participants, *i.e.,* the user, artist and tag. We then used kernel PCA to project the event representations onto the x-y plane, and ran k-means to obtain the clusters. We then randomly sample pairs of events within and across clusters, and show their triggering/inhibition relationships with red/black colors. As shown in Fig. 4a and b, the factor representations of the events present a clear cluster structure. Interestingly, we can see that events within the same cluster mainly inhibit each other (see Fig. 4a), while across different clusters often excite each other (see Fig. 4b). This might because users tend to tag artists with distinct/orthogonal styles, rather than keep tagging the same types of artists with very close tags/labels.

7. Conclusion

We have presented a self-modulating nonparametric event-tensor factorization model. Our model can capture various triggering and inhibition effects among the interaction events, in both short and long ranges. Our inference is also efficient for large numbers of observed entries and events.

References

Acar, E., Dunlavy, D. M., Kolda, T. G., and Mørup, 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.

Daley, D. J. and Vere-Jones, D. (2007). An introduction to the theory of point processes: volume II: general theory and structure. Springer Science & Business Media.

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.

Ketchen, D. J. and Shook, C. L. (1996). The application of cluster analysis in strategic management research: an analysis and critique. Strategic management journal, 17(6):441–458.

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.

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.

Pan, F., Li, S., Ao, X., Tang, P., and He, Q. (2019). Warm up cold-start advertisements: Improving ctr predictions via learning to learn id embeddings. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 695–704.

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).

Rasmussen, C. E. and Williams, C. K. I. (2006). Gaussian Processes for Machine Learning. MIT Press.

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.

Schölkopf, B., Smola, A., and Müller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. Neural computation, 10(5):1299–1319.

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.

Tran, T., Lee, K., Liao, Y., and Lee, D. (2018). Regularizing matrix factorization with user and item embeddings for recommendation. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pages 687–696.

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.