Online Continuous-Time Tensor Factorization Based on Pairwise Interactive Point Processes

Hongteng Xu[1,2], **Dixin Luo**[2], **Lawrence Carin**[2],
[1] Infinia ML, Inc. [2] Department of Electrical and Computer Engineering, Duke University,
hongteng.xu@infiniaml.com, dixin.luo@duke.edu, lcarin@duke.edu

Abstract

A continuous-time tensor factorization method is developed for event sequences containing K "modalities." Each data element is a point in a $\mathcal{C}_1 \times \mathcal{C}_2 \times \cdots \times \mathcal{C}_K$ tensor, where $\mathcal{C}_k$ is the *discrete* alphabet associated with modality k. Each tensor data element has an associated time of occurence $t \in \mathbb{R}_+$ and a feature vector $\boldsymbol{f} \in \mathbb{R}^D$. We model such data based on pairwise interactive point processes, and the proposed framework connects pairwise tensor factorization with a feature-embedded point process. The model accounts for interactions within each modality, interactions across different modalities, and continuous-time dynamics of the interactions. Model learning is formulated as a convex optimization problem, based on online alternating direction method of multipliers. Compared to existing state-of-the-art methods, our approach captures the latent structure of the tensor and its evolution over time, obtaining superior results on real-world datasets.

1 Introduction

Exploring latent structure in real-world multimodal data ($i.e.$, tensors) is a significant problem for analysis of underlying generative mechanisms and for predicting unobserved instances. Tensor factorization (TF) provides a flexible and effective way to learn such latent structure, decomposing the data into latent factors. Tensor factorization has been widely used in many applications and has achieved encouraging performance, $e.g.$, for recommendation systems [Rendle and Schmidt-Thieme, 2010], signal processing [Ozerov *et al.*, 2011], and computer vision [Shashua and Hazan, 2005].

Such data are often observed sequentially, and may be analyzed as multimodal event sequences in continuous time. For example, user behavior in a social network can be treated as a multimodal event ($i.e.$, message sender × message receiver × message type) with a time stamp, and the collection of such events can be formulated as a tensor whose elements contain the counts of the events up to a certain time. This tensor is time-varying, and the relationships between its modalities can change. Accordingly, its latent factors are likely to evolve over time. Therefore, we seek the development of i) a continuous-time tensor factorization (CTTF) for multimodal event sequences; and ii) a method for updating the model online, capturing the evolution of the latent factors over time.

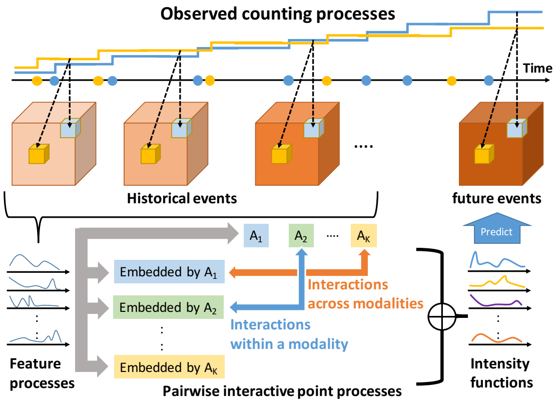


Figure 1: Illustration of our scheme. The feature processes represent the temporal processes of the event-associated features, and the $\{A_k\}$ are embedding matrices of the feature processes.

To address these *desiderata*, we present an online continuous-time tensor factorization method, based on a novel model called pairwise interactive point process. As shown in Fig. 1, the target continuous-time tensor is represented as a set of counting processes, corresponding to multimodal events. The intensity functions of these counting processes are factorized in a pairwise manner, simultaneously representing interactions within each factorization modality and across different modalities, by the inner products of latent factors. Each latent factor is a function of time and embedded features, capturing influences of historical events on current and future ones. Based on the model, we can predict the expected number ($i.e.$, counting processes) of future events.

The model is learned by fitting the expected counting process with the observed one, formulated here as a convex optimization problem, solved effectively by online alternating direction method of multipliers (ADMM). We analyze the convergence and the computational complexity of our method, and compare it with the state-of-the-art methods on two real-world datasets. Experiments show that the proposed method achieves superior performance on predicting future events.

2 Related Work

2.1 Tensor factorization

The Tucker decomposition [Tucker, 1966] factorizes a tensor into a smaller core tensor and a latent factor matrix for each modality; this approach may suffer from high computational complexity. To reduce complexity, the canonical decomposition [Kolda and Bader, 2009] assumes the core tensor is diagonal. A pairwise decomposition method is proposed in [Rendle and Schmidt-Thieme, 2010], which further reduces the computational complexity of tensor factorization, and achieves encouraging results in recommendation systems. The tensor factorization methods in [Rai *et al.*, 2015] leverage side information, to improve the learning results. Bayesian tensor factorization methods [Acharya *et al.*, 2015; Charlin *et al.*, 2015] are also used widely, with scalability improved in [Hu *et al.*, 2015] for count data. However, analysis of the continuous-time evolution of tensor factorization, while an important practical problem, is still relatively understudied.

Existing temporal tensor factorization methods mainly combine tensor factorization with time series-based models [Xiong *et al.*, 2010; Dunlavy *et al.*, 2011; Rafailidis and Nanopoulos, 2014; Acharya *et al.*, 2015; Charlin *et al.*, 2015], which only address data with discrete time steps or predefined decays [Koren, 2010], rather than continuous-time event sequences. Additionally, few of them consider online learning for tensor factorization.

2.2 Point processes

Point processes have been considered in many fields, *e.g.*, for analyzing financial [Bacry *et al.*, 2015] and social network [Blundell *et al.*, 2012] data, and in healthcare [Xu and Zha, 2017; Xu *et al.*, 2017a] and recommendation [Wang *et al.*, 2016] systems. To describe the influences of historical events on current and future ones, a series of complicated models (*e.g.*, the Hawkes process [Hawkes, 1971] and the correcting process [Xu *et al.*, 2015; 2017b]) have been proposed. These early works aimed to capture the dynamics of event sequences associated with a single data modality. Based on these models, many extensions have been proposed, to analyze multimodal (tensor) event sequences, *e.g.*, the multi-task multi-dimensional Hawkes process [Luo *et al.*, 2015]. Recently, point processes parameterized by embedded features have been proposed [Wang *et al.*, 2016], which model the events associated with features. However, extension of these methods to continuous-time tensor factorization is still an open problem.

Additionally, although methods have been proposed in [Hall and Willett, 2016; Yang *et al.*, 2017] to explore online learning of point processes, these methods only deal with Hawkes processes in a nonparametric manner. The online learning of parametric point processes, with more general formulations, has not been investigated deeply.

3 Proposed Model

3.1 Notations and problem statement

Many real-world data are generated sequentially, and can be formulated as a continuous-time event sequence with multiple data modalities (*i.e.*, data may be mapped to a tensor). A typical example is online shopping records, in which pairs of users and items are listed along with a time stamp. Such data contain two "modalities," corresponding to users and items, and features may be available to describe each transaction.

We describe event sequences with K modalities as a set of triads, denoted $\{(t_n, \boldsymbol{\zeta}_n, \boldsymbol{f}_n)\}_{n=1}^{N}$. The time stamp for event n is denoted $t_n \in [0, T]$; $\boldsymbol{\zeta}_n = [c_n^1, ..., c_n^K] \in \mathcal{C}_1 \times ... \times \mathcal{C}_K$ represents the multimodal event type, where $\mathcal{C}_k$ is the categorical set of the k-th modality and c_n^k is the index of the event corresponding to the k-th modality; and $\boldsymbol{f}_n \in \mathbb{R}^D$ is an event-dependent feature vector.

We represent the event sequence as a continuous-time tensor, *i.e.*, $\boldsymbol{N}(t) = [N_{\boldsymbol{\zeta}}(t)] \in \mathbb{Z}_+^{|\mathcal{C}_1| \times ... \times |\mathcal{C}_K|}$, where $\boldsymbol{\zeta} = [c^1, ..., c^K]$ represents a particular multimodal event type, $|\mathcal{C}_k|$ represents the cardinality of $\mathcal{C}_k$, and $\mathbb{Z}_+$ represents nonnegative integers. Each element $N_{\boldsymbol{\zeta}}(t)$ is a counting process, recording the number of type-$\boldsymbol{\zeta}$ events that have occurred up to time t. For each $\boldsymbol{\zeta}$, the counting process $N_{\boldsymbol{\zeta}}(t)$ can be characterized by the expected instantaneous occurence rate of the event, conditioned on historical observations, called the *intensity function*:

$$\lambda_{\boldsymbol{\zeta}}(t) = \frac{\mathbb{E}[dN_{\boldsymbol{\zeta}}(t)|\mathcal{H}_t]}{dt}, \ \boldsymbol{\zeta} \in \mathcal{C}_1 \times ... \times \mathcal{C}_K, \ t \in [0, T], \quad (1)$$

where $\mathcal{H}_t = \{(t_n, \boldsymbol{\zeta}_n, \boldsymbol{f}_n) | t_n < t\}$ contains historical events before time t. The intensity functions of all types of events formulate a continuous-time *intensity tensor* $\boldsymbol{\Lambda}(t) = [\lambda_{\boldsymbol{\zeta}}(t)] \in \mathbb{R}_+^{|\mathcal{C}_1| \times ... \times |\mathcal{C}_K|}$. In this setup we do not explicitly model feature vectors $\boldsymbol{f}_n$, which are assumed as observed covariates associated with events; however, the intensity function is dependent on prior feature vectors and other characteristics of previous events.

Problem statement: Given observed counting processes $\boldsymbol{N}(t)$, we aim to explore the latent structures of its intensity tensor $\boldsymbol{\Lambda}(t)$, capture the evolution of the latent structures over time, and predict the number of events in the future based on learning results.

3.2 Pairwise interactive point processes

The simplest approach for modeling the intensity tensor $\boldsymbol{\Lambda}(t)$ is to learn each $\lambda_{\boldsymbol{\zeta}}(t)$ independently [Du *et al.*, 2015]. Such a strategy may be questionable because the modalities of real-world data often have interactions, and hence there is likely statistical dependencies between the set of $\{\lambda_{\boldsymbol{\zeta}}(t)\}$. To overcome this problem, many existing factorization methods, especially those applied in recommendation systems, consider the pairwise interactions between different modalities in an inner product manner. In particular, for a tensor $\boldsymbol{\Lambda}$, those methods represent its element $\lambda_{\boldsymbol{\zeta}}$ as the sum of the inner products of different latent factors, *i.e.*, $\lambda_{\boldsymbol{\zeta}} = \sum_{k \neq k'} \boldsymbol{u}_{k,c^k}^\top \boldsymbol{u}_{k',c^{k'}}$.[1] For the well-known collaborative filtering problem for a 2D matrix ($K = 2$) [Koren *et al.*, 2009; Park and Chu, 2009], $\boldsymbol{\zeta} = [c^1, c^2]$, with $c^1 \in \mathcal{C}^1$ and $c^2 \in \mathcal{C}^2$. For any $\lambda_{[c^1,c^2]}$ associated with the $|\mathcal{C}_1| \times |\mathcal{C}_2|$

[1] Pairwise factorization is applicable to both time-dependent and time-invariant tensors, so here we ignore the notation of time.

matrix, the construction $\lambda_{\boldsymbol{\zeta}} = \sum_{k \neq k'} \boldsymbol{u}_{k,c^k}^\top \boldsymbol{u}_{k',c^{k'}}$ yields $\lambda_{[c^1,c^2]} = \boldsymbol{u}_{1,c^1}^\top \boldsymbol{u}_{2,c^2}$, where the set of vectors $\{\boldsymbol{u}_{1,c^1}\}_{c^1 \in \mathcal{C}^1}$ represent the entities along axis one, and $\{\boldsymbol{u}_{2,c^2}\}_{c^2 \in \mathcal{C}^2}$ represent the entities along axis two. For 3D tensors this approach generalizes to pairwise factorization [Rendle and Schmidt-Thieme, 2010].

In the above model, $\lambda_{\boldsymbol{\zeta}}(t)$ only depends on the cross-modality inner products between the elements of $\boldsymbol{\zeta}$. A natural extension is to also include intra-modality terms, as

$$\lambda_{\boldsymbol{\zeta}}(t) = \eta_{\boldsymbol{\zeta}}(t) + \underbrace{\sum\nolimits_{k=2}^{K} \sum\nolimits_{k'=1}^{k-1} \boldsymbol{u}_{k,c^k}^\top(t) \boldsymbol{u}_{k',c^{k'}}(t)}_{\text{Interactions across different modalities}}, \quad (2)$$

$$\eta_{\boldsymbol{\zeta}}(t) = \sum\nolimits_{k=1}^{K} \eta_{c^k}(t) = \underbrace{\sum\nolimits_{k=1}^{K} \boldsymbol{u}_{k,c^k}^\top(t) \boldsymbol{u}_{k,c^k}(t)}_{\text{Self-interactions within each modality}}, \quad (3)$$

where the L-dimensional vector $\boldsymbol{u}_{k,c^k}(t)$ with $c^k \in \mathcal{C}_k$ is the time-dependent latent factor corresponding to the index c^k of the k-th modality. For $k = 1, ..., K$, $\boldsymbol{U}^k(t) = [\boldsymbol{u}_{k,1}(t), ..., \boldsymbol{u}_{k,|\mathcal{C}_k|}(t)] \in \mathbb{R}^{L \times |\mathcal{C}_k|}$ is the latent factor matrix corresponding to the k-th modality.

The work in [Wang *et al.*, 2016] also considered a base intensity $\eta_{\boldsymbol{\zeta}}(t)$, like in (2), but in that prior work the base intensity was time-independent. Similar to the work in [Koren, 2010], we represent the base intensity as the sum of the basis corresponding to different modalities (the $\eta_{c^k}(t)$ in (3)). Moreover, to avoid additional parameters, we reuse the latent factors to parameterize the basis ($\eta_{c^k}(t) = \boldsymbol{u}_{k,c^k}^\top(t) \boldsymbol{u}_{k,c^k}(t)$). Such a strategy implies that the base intensity reflects the self-interactions within each modality.

In summary, (2) means that event type $\boldsymbol{\zeta} = [c^1, ..., c^K]$ has an expected occurrence rate at time t controlled simultaneously by the interactions within each modality and the interactions across different modalities. Combining (2) with (3), we rewrite (2) as

$$\lambda_{\boldsymbol{\zeta}}(t) = \sum\nolimits_{k=1}^{K} \sum\nolimits_{k'=1}^{k} \boldsymbol{u}_{k,c^k}^\top(t) \boldsymbol{u}_{k',c^{k'}}(t). \quad (4)$$

Like the work in [Du *et al.*, 2015; Xu *et al.*, 2015; Wang *et al.*, 2016], we represent each latent factor $\boldsymbol{u}_{k,c^k}(t)$ as a time-dependent embedding of observed features:

$$\boldsymbol{u}_{k,c^k}(t) = \boldsymbol{A}_k \Big[\sum\nolimits_{c_n^k = c^k, t_n < t} \overset{1}{} \boldsymbol{f}_n \kappa(t-t_n)\Big] = \boldsymbol{A}_k \hat{\boldsymbol{f}}_{c^k}(t), \quad (5)$$

where $\boldsymbol{A}_k \in \mathbb{R}^{L \times (D+1)}$ is the k-th embedding matrix, the $\hat{\boldsymbol{f}}_{c^k}(t)$ is the weighted sum of the historical feature vectors up to time t, and $\kappa(t) \geq 0$ is a predefined kernel function deciding the weights. We can find that the latent factor $\boldsymbol{u}_{k,c^k}$ is the sum of two components: i) a time-invariant component corresponding to the first column of $\boldsymbol{A}_k$, and ii) a time-varying component corresponding to the multiplication between the remaining D columns of $\boldsymbol{A}_k$, denoted as $\boldsymbol{A}_k^{2:D+1}$ and the accumulated features, $i.e.$ $\boldsymbol{A}_k^{2:D+1}(\sum_{c_n^k = c^k, t_n < t} \boldsymbol{f}_n \kappa(t - t_n))$. To ensure that the intensity function is physically-meaningful, here we require that the elements of all embedding matrices $\boldsymbol{A}_k$'s and features $\boldsymbol{f}_n$'s are nonnegative, such that the intensity function is nonnegative as well.

4 Online Continuous-Time Tensor Factorization

4.1 Reformulation of the problem

Given observed counting process $\boldsymbol{N}(t)$, we aim to learn the intensity tensor $\boldsymbol{\Lambda}(t)$ and the parameters $\{\boldsymbol{A}_k\}_{k=1}^K$ in the model defined by (4)-(5). Unlike maximum likelihood estimation (MLE) proposed in most existing works [Lewis and Mohler, 2011; Luo *et al.*, 2015; Wang *et al.*, 2016], we learn our pairwise interactive point process model as a regularized linear predictor, for efficient online learning. Specifically, instead of maximizing the likelihood of the observed event sequence, we aim to fit the observed counting process $\boldsymbol{N}(t)$ with their expectation $\mathbb{E}[\boldsymbol{N}(t)]$, which is estimated by the integration of the intensity tensor, $i.e.$, $\int_0^T \boldsymbol{\Lambda}(s)ds$. Given the event sequence $\{(t_n, \boldsymbol{\zeta}_n, \boldsymbol{f}_n)\}_{n=1}^N$, we present the following squared loss function as the objective function:

$$\mathcal{R}(\{\boldsymbol{A}_k\}_{k=1}^K) = \sum_{n=1}^{N} \Big| \frac{1}{t_n} \Big(N_{\boldsymbol{\zeta}_n}(t_n) - \int_0^{t_n} \lambda_{\boldsymbol{\zeta}_n}(s)ds \Big) \Big|^2. \quad (6)$$

We use the factor $\frac{1}{t_n}$ to rescale the weight of the n-th estimation error, because the variance of the averaged intensity, $i.e.$, $\mathbb{E}[\frac{N(t) - \mathbb{E}[N(t)]}{t}]^2$ is generally bounded.

Minimizing (6) directly is non-convex because it involves the terms like $\boldsymbol{A}_k^\top \boldsymbol{A}_{k'}$, $k \neq k'$. Fortunately, we can reformulate the problem as a convex optimization problem by rewriting the intensity function as

$$\begin{aligned} \lambda_{\boldsymbol{\zeta}}(t) &= \sum\nolimits_{k=1}^{K} \sum\nolimits_{k'=1}^{k} \hat{\boldsymbol{f}}_{c^k}^\top(t) \boldsymbol{A}_k^\top \boldsymbol{A}_{k'} \hat{\boldsymbol{f}}_{c^{k'}}(t) \\ &= \sum\nolimits_{k=1}^{K} \sum\nolimits_{k'=1}^{k} \hat{\boldsymbol{f}}_{c^k}^\top(t) \boldsymbol{X}_{kk'} \hat{\boldsymbol{f}}_{c^{k'}}(t) \\ &= \sum\nolimits_{k=1}^{K} \sum\nolimits_{k'=1}^{k} \mathrm{tr}(\boldsymbol{F}_{c^k c^{k'}}^\top(t) \boldsymbol{X}_{kk'}) = \tilde{\boldsymbol{f}}_{\boldsymbol{\zeta}}^\top(t) \boldsymbol{x}. \end{aligned} \quad (7)$$

Here, $\boldsymbol{X}_{kk'} = \boldsymbol{A}_k^\top \boldsymbol{A}_{k'}$, $\boldsymbol{F}_{c^k c^{k'}}(t) = \hat{\boldsymbol{f}}_{c^k}(t) \hat{\boldsymbol{f}}_{c^k}^\top(t)$ is the outer product of the accumulated feature and its transpose, and tr$(\cdot)$ calculates the trace of matrix. The parameter, denoted as $\boldsymbol{x} \in \mathbb{R}^{D'}$, $D' = \frac{K(K+1)(D+1)^2}{2}$, is a vectorization of all matrices $\boldsymbol{X}_{kk'}$'s, $i.e.$, $\boldsymbol{x} = \mathrm{vec}(\{\boldsymbol{X}_{kk'}\}_{k \geq k'})$. Similarly, $\tilde{\boldsymbol{f}}_{\boldsymbol{\zeta}}(t) = \mathrm{vec}(\{\boldsymbol{F}_{c^k c^{k'}}\}_{k \geq k'})$ is the vectorized representation of the $\boldsymbol{F}_{c^k c^{k'}}(t)$'s.

Based on (7), we rewrite the loss function in (6) as

$$\mathcal{R}(\boldsymbol{x}) = \|\boldsymbol{N}_0^{t_N} - \boldsymbol{F}_0^{t_N} \boldsymbol{x}\|_2^2, \quad (8)$$

where the vector $\boldsymbol{N}_0^{t_N} = [\frac{1}{t_n} N_{\boldsymbol{\zeta}_n}(t_n)] \in \mathbb{R}^N$, and the matrix $\boldsymbol{F}_0^{t_N} = [\frac{1}{t_n} \int_0^{t_n} \tilde{\boldsymbol{f}}_{\boldsymbol{\zeta}_n}^\top(s)ds] \in \mathbb{R}^{N \times D'}$. Moreover, the parameter vector $\boldsymbol{x}$ has structure. In particular, we can construct a symmetric rank-L matrix $\boldsymbol{X}$ from the parameter vector $\boldsymbol{x}$ as

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_{11}, \cdots, \boldsymbol{X}_{1K} \\ \vdots \quad \ddots \quad \vdots \\ \boldsymbol{X}_{K1}, \cdots, \boldsymbol{X}_{KK} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_1^\top \\ \vdots \\ \boldsymbol{A}_K^\top \end{bmatrix} [\boldsymbol{A}_1, ..., \boldsymbol{A}_K], \quad (9)$$

where $\boldsymbol{x}$ vectorizes the lower triangular submatrices of $\boldsymbol{X}$. For convenience, we denote the operator that constructs $\boldsymbol{X}$

from $\boldsymbol{x}$ as $\boldsymbol{X} = \text{map}(\boldsymbol{x})$. Considering the nonnegative and low-rank properties of $\boldsymbol{X}$, the final optimization problem is

$$\min_{\boldsymbol{x}\geq\boldsymbol{0}} \mathcal{R}(\boldsymbol{x}) + \alpha\|\text{map}(\boldsymbol{x})\|_*, \tag{10}$$

where $\|\cdot\|_*$ calculates the nuclear norm of the matrix. This regularizer imposes a low-rank constraint on $\boldsymbol{X}$ and its significance is controlled by α. By solving (10), we learn the parameters of our point process model as a regularized linear predictor.

4.2 Online learning strategy

To track the evolution of the model with new data, we solve the optimization problem in (10) in the framework of online learning. In particular, we can develop an online learning algorithm based on the stochastic alternating direction method of multipliers (ADMM) [Ouyang *et al.*, 2013; Zhong and Kwok, 2014]. In the initial phase, we first estimate the parameters from the observations till time T, denoted as $\boldsymbol{x}_T$. When new events occur in $[T, T_{new}]$, we can calculate the new counting processes and the new embedded features quickly, denoted as $\boldsymbol{N}_T^{T_{new}}$ and $\boldsymbol{F}_T^{T_{new}}$. We can then update the parameters by solving

$$\begin{aligned}&\min_{\boldsymbol{x}\geq\boldsymbol{0},\boldsymbol{y},\boldsymbol{z}} \|\boldsymbol{N}_T^{T_{new}} - \boldsymbol{F}_T^{T_{new}}\boldsymbol{x}\|_2^2 + \rho\|\boldsymbol{x}-\boldsymbol{y}\|_2^2\\ &+\alpha\|\text{map}(\boldsymbol{y})\|_* + 2\rho(\boldsymbol{z}^\top(\boldsymbol{x}-\boldsymbol{y})) + \beta\|\boldsymbol{x}-\boldsymbol{x}_T\|_2^2,\end{aligned} \tag{11}$$

where $\boldsymbol{y}$ is an auxiliary variable and $\boldsymbol{z}$ is a dual variable. The first term corresponds to the squared loss function for the new data, and the subsequent three terms are argumented Lagrangian terms of ordinary ADMM. Additionally, we add $\|\boldsymbol{x}-\boldsymbol{x}_T\|_2^2$ to ensure that the new estimation should not be too far from the previous one. The weight of this term is controlled by a parameter β. We can minimize (11) by alternating optimization. In the i-th step, we obtain $\boldsymbol{x}_i$ by updating $\boldsymbol{x}_{i-1}$ based on new observations, and the optimization problem is

$$\begin{aligned}\boldsymbol{x}_i =&\arg\min_{\boldsymbol{x}\geq\boldsymbol{0}} \|\boldsymbol{N}_{T_{i-1}}^{T_i} - \boldsymbol{F}_{T_{i-1}}^{T_i}\boldsymbol{x}\|_2^2\\ &+\rho\|\boldsymbol{x}-\boldsymbol{y}_{i-1}+\boldsymbol{z}_{i-1}\|_2^2 + \beta\|\boldsymbol{x}-\boldsymbol{x}_{i-1}\|_2^2.\end{aligned} \tag{12}$$

This problem can be solved efficiently by projected gradient descent. Given $\boldsymbol{x}_i$, we update $\boldsymbol{y}_i$ by solving a low-rank approximation problem, and soft-thresholding is applied.

$$\begin{aligned}\boldsymbol{y}_i =&\arg\min_{\boldsymbol{y}} \rho\|\boldsymbol{y}-\boldsymbol{x}_i-\boldsymbol{z}_{i-1}\|_F^2 + \alpha\|\text{map}(\boldsymbol{y})\|_*\\ =&\text{vec}(S_{\frac{\alpha}{\rho}}(\text{map}(\boldsymbol{x}_i+\boldsymbol{z}_{i-1}))),\end{aligned} \tag{13}$$

where the operator $S_\delta(\cdot)$ shrinks the singular values of the matrix with a threshold δ, and the operators map$(\cdot)$ and vec$(\cdot)$ are defined as previously. Finally, we obtain the new dual variable $\boldsymbol{z}_i$ by $\boldsymbol{z}_i = \boldsymbol{z}_{i-1} + (\boldsymbol{x}_i - \boldsymbol{y}_i)$. The scheme of our algorithm is shown in Algorithm 1, where the operator $(\cdot)_+$ keeps all nonnegative values from changing and sets all negative values to zeros.

It should be noted that for each time T_i we can construct the low-rank symmetric matrix $\boldsymbol{X}_i$ from the estimated parameters, *i.e.*, $\boldsymbol{X}_i = \text{map}(\boldsymbol{x}_i)$. When the embedding matrices $\{\boldsymbol{A}_k\}_{k=1}^K$ in the interval $[T_{i-1}, T_i]$ are required, according to (9) we can apply the symmetric nonnegative matrix factorization [Kuang *et al.*, 2012] to $\boldsymbol{X}_i$ and obtain them accordingly.

Algorithm 1 Online CFFT

```
Input: An initial sequence in $[0, T_0]$, parameters $\alpha, \beta, \rho, \tau$.
Output: A set of $\boldsymbol{x}_i$'s for different time intervals.
Initialize $\boldsymbol{x}_0 = ((((\boldsymbol{F}_0^{T_0})^\top \boldsymbol{F}_0^{T_0})^{-1}(\boldsymbol{F}_0^{T_0})^\top \boldsymbol{N}_0^{T_0})_+$,
$\boldsymbol{y}_0 = \boldsymbol{x}_0$, $\boldsymbol{z}_0 = \boldsymbol{0}$.
for $i = 1, 2, 3, ...$ do
    Calculate $\boldsymbol{N}_{T_{i-1}}^{T_i}$ and $\boldsymbol{F}_{T_{i-1}}^{T_i}$. Set $\boldsymbol{x}_i = \boldsymbol{x}_{i-1}$.
    $\boldsymbol{\Phi} = (\boldsymbol{F}_{T_{i-1}}^{T_i})^\top \boldsymbol{F}_{T_{i-1}}^{T_i} + (\rho+\beta)\boldsymbol{I}$.
    $\boldsymbol{b} = (\boldsymbol{F}_{T_{i-1}}^{T_i})^\top \boldsymbol{N}_{T_{i-1}}^{T_i} + \rho(\boldsymbol{y}_{i-1} - \boldsymbol{z}_{i-1}) + \beta\boldsymbol{x}_{i-1}$.
    Projected gradient descent (Inner iterations):
    while not converge do
        Update $\boldsymbol{x}$ by $\boldsymbol{x}_i = (\boldsymbol{x}_i - 2\tau(\boldsymbol{\Phi}\boldsymbol{x}_i - \boldsymbol{b}))_+$.
    Update $\boldsymbol{y}$ by (13).
    Update $\boldsymbol{z}$ by $\boldsymbol{z}_i = \boldsymbol{z}_{i-1} + (\boldsymbol{x}_i - \boldsymbol{y}_i)$.
```

4.3 Prediction of future events

After learning the pairwise interactive point process model, we can predict future events. In particular, given the model trained till time T, we can simulate a set of events happening in the following time interval $[T + dT]$ by Ogata's thinning method [Ogata, 1981]. The expected number of the events with a specific type in this target interval is estimated by averaging the increments of the counting processes in different simulation trials.

5 Further Analysis

5.1 Convexity and convergence

Instead of learning the matrices $\{\boldsymbol{A}_k\}_{k=1}^K$ directly, our method reformulates the parameters as a vector $\boldsymbol{x}$ and learns by convex optimization. In the case that the observed event sequences are generated by a stationary point process, the parameters converge to a global optimal solution of (10) with the increase of data. Via the analysis in [Ouyang *et al.*, 2013], the convergence rate of $\boldsymbol{x}$ is guaranteed to be $\mathcal{O}(\frac{\log t}{t})$, where t is the number of iterations. Furthermore, even if the target point process is not temporal stationary, our online learning method can still ensure that we can obtain a global optimal solution in each step (*i.e.*, for the data in specific time intervals). In such a situation, the sequence of optimums $\{\boldsymbol{x}_i\}$ reflects the evolution of the target point process. The performance of our method on convergence is shown in Fig. 2(a) for further verification.

5.2 Sample complexity

Equation (8) measures the empirical risk between the normalized counting processes and their expectations estimated by a linear predictor. According to Theorem 1 in [Shamir, 2015], we have the following proposition:

Proposition 5.1. *Suppose N events are observed for a point process with K event types, i.e., $\boldsymbol{\zeta} \in \mathcal{C}_1 \times ... \times \mathcal{C}_K$. If its counting process satisfies $\frac{1}{t}N_{\boldsymbol{\zeta}}(t) < Y$ for all t and $\boldsymbol{\zeta}$, and its intensity function can be decomposed as in (4), which contains K latent processes with $D + 1$ dimensions, then we can* *construct a linear predictor with a D'-dimensional parameter vector $\boldsymbol{x}$, where $D' = \frac{K(K+1)(D+1)^2}{2}$ and $\|\boldsymbol{x}\|_2 \leq B$, and the bound on the excess risk $\mathbb{E}[\mathcal{R}(\hat{\boldsymbol{x}}) - \mathcal{R}(\boldsymbol{x}^*)]$ satisfies*

$$\begin{aligned}&\mathbb{E}[\mathcal{R}(\hat{\boldsymbol{x}}) - \mathcal{R}(\boldsymbol{x}^*)]\\ &\leq \mathcal{O}\left(\min\left\{Y^2, \frac{B^2 + D'Y^2\log(1+\frac{N}{D'})}{N}, \frac{BY}{\sqrt{N}}\right\}\right),\end{aligned} \tag{14}$$

where $\hat{\boldsymbol{x}}$ is the optimum learned from observed events and $\boldsymbol{x}^*$ is the ground truth of the model.

According to (14) we can find that i) when data is insufficient ($i.e.$, $N < \min\{D', \frac{B^2}{Y^2}\}$), the learning results suffers from over-fitting and the excess risk and $\frac{1}{t^2}N_{\boldsymbol{\zeta}}^2(t)$ have the same order of magnitude; ii) when considering many modalities ($i.e.$, large K) or high-dimensional features ($i.e.$, large D), the bound of excess risk is likely to be $\mathcal{O}(\frac{BY}{\sqrt{N}})$ even if we have a large amount of samples. Fortunately, in many practical applications the number of modalities is limited ($K = 2$ or 3 commonly) and the low-rank assumption of latent processes is applied, so that the bound $\mathcal{O}(\frac{B^2+D'Y^2\log(1+\frac{N}{D'})}{N})$ is achievable when sufficient events are observed.

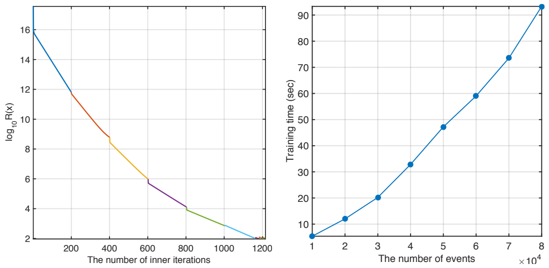
5.3 Computational complexity

The bottleneck of our method is calculating the matrix $\boldsymbol{F}_{T_{i-1}}^{T_i}$ in each step. Each row of $\boldsymbol{F}_{T_{i-1}}^{T_i}$ involves the integration of intensity function, which considers all events happening before a specific time t. The computation becomes intractable with the increase of historical events. This problem can be solved when defining the kernel function $\kappa(t)$ in (5) as a monotonically-decreasing function, $e.g.$, exponential function. In particular, we set $\kappa(t) = w\exp(-wt)$, $t \geq 0$. As a result, when we calculate the rows of $\boldsymbol{F}_{T_{i-1}}^{T_i}$, the historical events whose timestamps $t_{history} \ll T_{i-1}$ can be ignored because $\kappa(T_{i-1} - t_{history}) \approx 0$. Suppose that there are N events in $[T_{i-1}, T_i]$ and for each event we consider at most M historical events to calculate the corresponding row of $\boldsymbol{F}_{T_{i-1}}^{T_i}$, then the computation of $\boldsymbol{F}_{T_{i-1}}^{T_i}$ involves $\mathcal{O}(MN)$ operations. As a result, suppose that the dimension of the parameter $\boldsymbol{x}$ is D', then the per-iteration complexity of our method is $\mathcal{O}(D'^2MN)$. Note that although $D' = \mathcal{O}(K^2D^2)$, it is generally ignorable compared with MN because the number of modalities K and the dimension of feature D are always limited in practice. A typical runtime curve in the case with $K = 2$ and $D = 20$ is shown in Fig. 2(b), verifying the above analysis. The proposed method is implemented in MATLAB.

6 Experimental Results

6.1 Competitors and evaluations

We evaluate our method (**CTTF**) on two real-world datasets, and we segment each into several subsets. For each subset, we use all the events up to a predefined timestamp as the training data, and the remaining events as the testing data. The hyper-parameters, including $\{\rho, \alpha, \beta\}$ in our method and the w in kernel $\kappa(t)$ are selected using 10-fold cross validation with grid search. Using the prediction method mentioned in Sec. 4.3, we predict the number of events happening in the testing time interval and report the Mean Absolute Error (MAE) between the predicted and true number. The mean and the standard deviation of the MAEs obtained in different subsets are shown as errorbars in Fig. 3(a) and 3(b).



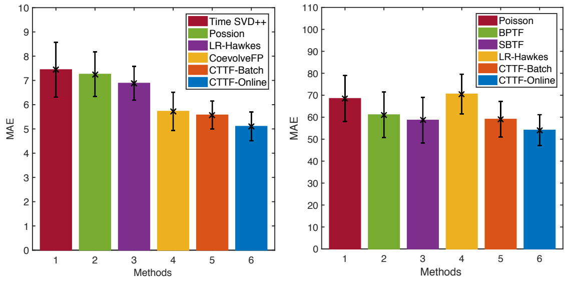
(a) Convergence (b) Runtime

Figure 2: Illustrations of the convergence and the complexity of our method. The testing sequence is from the IPTV dataset. The dataset contains two modalities (7, 100 users and 436 drama programs) and each event is associated with a 20-dimensional feature vector describing the attributes of the corresponding drama. In (a), the logarithm of loss function obtained in 10 steps is shown. In each step, we import a batch of events to update $\boldsymbol{x}$ by our method. The x-axis represents the total number of projected gradient descent for all the steps and the colors of curves correspond to different steps.

We consider the following methods as baselines, for comparison. In the case of 2D tensor (matrix) data, we implement four methods: classical matrix factorization **TimeSVD++** [Koren, 2010], dynamic **Poisson** factorization [Charlin *et al.*, 2015], low-rank regularized Hawkes process (**LR-Hawkes**) [Du *et al.*, 2015], and the co-evolutionary feature process (**CoevolveFP**) [Wang *et al.*, 2016]. In the case of 3D tensor data, besides the Poisson and the CoevolveFP, we compare with Bayesian probabilistic tensor factorization (**BPTF**) [Xiong *et al.*, 2010], and the scalable Bayesian nonnegative tensor factorization method (**SBTF**) [Hu *et al.*, 2015].

Similar to our work, the LR-Hawkes and the CoevolveFP are point process-based methods, which estimate the number of future events by the simulation method in Sec. 4.3. The TimeSVD++ can also describe event sequences in continuous time. It estimates the number of future events directly by learned latent factors and a predefined decay function of time. For the other methods, which cannot factorize tensors in the continuous-time domain, we first reformulate original data as time series: the event sequence is discretized into several bins and the numbers of the events with different types in each bin are recorded. The training count tensor is fitted by these methods and the testing count tensor is estimated by the learned latent factors.

Additionally, in both these two cases, we implement our CTTF method by batch optimization and online optimization, respectively (**CTTF-Batch** and **CTTF-Online**). The CTTF-Batch assumes that the whole training sequence obeys to a stationary point process, and we learn a single pairwise interactive point process model. The CTTF-Online relaxes the stationary assumption and updates the model online by importing small batches of training events sequentially. The testing sequence is then estimated by the model in last updating.



(a) Comparisons on IPTV (b) Comparisons on GDELT

Figure 3: Experimental results and comparisons.

6.2 IPTV dataset

The IPTV dataset [Wang *et al.*, 2016; Luo *et al.*, 2014] contains the watching history of $7,100$ users, of 436 drama programs in 11 months, with $2,392,010$ events, and $1,420$ binary movie features, including $1,073$ actors, 312 directors, 22 genres, 8 countries and 5 years. There are two modalities corresponding to "user" and "TV program", and each counting process $N(t)$ counts how many viewing behaviors of a specific user-program pair happen till time t. We reduce the dimension of these features to 20 $(= D)$ by nonnegative sparse principal component analysis [Zass and Shashua, 2007]. We can treat the counting processes as a continuous-time 2D tensor, whose elements record the numbers of modality pairs ($i.e.$, user ID and drama ID). We segment the dataset into 47 subsets, each of which corresponds to user watching records over a contiguous span of 192 hours (8 days). For each method, the data in the first 168 hours is used to train the model and the remaining data is used to test.

Figure 3(a) visualizes the comparisons for various methods on the average and the standard deviation of MAEs. We find that the our CTTF-Online method obtains smaller MAE with smaller standard deviation, outperforming the comparison methods. In particular, the TimeSVD++ considers the influence of historical observations on current interactions among different modalities but it does not take advantages of features of events in its framework. The dynamic Poisson method only considers the transition probability between adjacent events, which cannot capture more complicated triggering patterns. The LR-Hawkes method does not consider the interactions across different modalities, considering the event sequences corresponding to different multimodal event types independently. The CoevolveFP method does not consider the interaction with each modality. The LR-Hawkes and CoevolveFP methods are inferior to the proposed CTTF because they only model one-side interactions. Moreover, the CTTF-Online is slightly better than the CTTF-Batch. In practice, the target model often may be locally-stationary rather than globally-stationary, and the testing data is likely to have more similar dynamics to its adjacent training data. The CTTF-Online updates the model online and captures its evolution over time.

6.3 Political science dataset

We next consider a dataset from the Global Database of Events, Location, and Tone (GDELT) [Leetaru and Schrodt, 2013]. It records the dyadic interactions between countries in the form of Country A did something to Country B at a certain time. Here, we consider 88 countries, 16 types of actions, and a time period of $2,395$ days (from years 2007 to 2013), containing $9,825,248$ events. In particular, each event is associated with two attributes called "QuadClass" and "Coldstein", respectively. The "QuadClass" indicates that the action in the event belongs to one of the following four classes: Verbal Cooperation, Material Cooperation, Verbal Conflict, and Material Conflict. The "Coldstein" is in the range of $[-10, 10]$. It represents positive (negative) events by positive (negative) values, and its amplitude indicates the infectivity of the event. Therefore, we represent these two attributes as a 25-dimensional binary feature vector for each event. There are three modalities corresponding to "political action type", "the country applying action" and "the country suffering action", and each $N(t)$ records the number of "country-country-action type" triples up to time t. The counting processes of the interactions among the countries is formulated as a continuous-time 3D tensor. In this experiment, we segment the GDELT dataset into 39 subsets, each of which corresponds to the political events in 9 weeks (63 days). For each method, the data in the first 8 weeks is used to train the model and the remaining data is used to test.

Figure 3(b) visualizes the comparisons for the different methods, on the average and the standard deviation of MAEs. Similar to the experiments on the IPTV dataset, our CTTF-Online method is superior to the alternative approaches. In this experiment, the TimeSVD++ and the CoevolveFP are designed for the 2D case, which are unsuitable for 3D tensor data. The LR-Hawkes method treats the sequences with specific multimodal event types independently, and suffers from over-fitting, yielding an MAE that is larger than the methods. The BPTF is originally designed for the 2D case as well, but it can be easily extended to the 3D case. Similar to the Poisson method, it only considers the transitions between adjacent events, and thus cannot capture high-order relationships. The most competitive baseline of our method in this experiment is the SBFT method, which achieves comparable performance to the CTTF-Batch method. However, it is still inferior to the CTTF-Online method because it ignores the evolution of the model over time.

7 Conclusions

We propose a pairwise interactive point process model to achieve online continuous-time tensor factorization for multimodal event sequences. The learning task of the proposed model can be reformulated as a convex optimization problem, which can be solved by an online ADMM. We analyze the convergence and the computational complexity of our method, and demonstrate its feasibility and superiority on two real-world datasets. In the future, we plan to introduce the online learning strategy into the framework of maximum likelihood estimation, with the goal of achieving a better convergence rate. Additionally, the neural network-based embedding methods will be considered for the latent factors.

Acknowledgment. The work presented here was supported in part by DARPA, DOE, NIH, NSF and ONR.

References

[Acharya *et al.*, 2015] Ayan Acharya, Joydeep Ghosh, and Mingyuan Zhou. Nonparametric Bayesian factor analysis for dynamic count matrices. *arXiv preprint arXiv:1512.08996*, 2015.

[Bacry *et al.*, 2015] Emmanuel Bacry, Iacopo Mastromatteo, and Jean-François Muzy. Hawkes processes in finance. *Market Microstructure and Liquidity*, 1(01):1550005, 2015.

[Blundell *et al.*, 2012] Charles Blundell, Jeff Beck, and Katherine A Heller. Modelling reciprocating relationships with Hawkes processes. In *NIPS*, 2012.

[Charlin *et al.*, 2015] Laurent Charlin, Rajesh Ranganath, James McInerney, and David M Blei. Dynamic Poisson factorization. In *RecSys*, 2015.

[Du *et al.*, 2015] Nan Du, Yichen Wang, Niao He, Jimeng Sun, and Le Song. Time-sensitive recommendation from recurrent user activities. In *NIPS*, 2015.

[Dunlavy *et al.*, 2011] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. Temporal link prediction using matrix and tensor factorizations. *Transactions on Knowledge Discovery from Data*, 5(2):10, 2011.

[Hall and Willett, 2016] Eric C Hall and Rebecca M Willett. Tracking dynamic point processes on networks. *Transactions on Information Theory*, 62(7):4327–4346, 2016.

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

[Hu *et al.*, 2015] Changwei Hu, Piyush Rai, Changyou Chen, Matthew Harding, and Lawrence Carin. Scalable Bayesian nonnegative tensor factorization for massive count data. In *ECML*, 2015.

[Kolda and Bader, 2009] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[Koren *et al.*, 2009] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8), 2009.

[Koren, 2010] Yehuda Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[Kuang *et al.*, 2012] Da Kuang, Chris Ding, and Haesun Park. Symmetric nonnegative matrix factorization for graph clustering. In *SDM*, 2012.

[Leetaru and Schrodt, 2013] Kalev Leetaru and Philip A Schrodt. GDELT: Global data on events, location, and tone. In *ISA Annual Convention*, 2013.

[Lewis and Mohler, 2011] Erik Lewis and George Mohler. A nonparametric EM algorithm for multiscale Hawkes processes. *Journal of Nonparametric Statistics*, 1(1):1–20, 2011.

[Luo *et al.*, 2014] Dixin Luo, Hongteng Xu, Hongyuan Zha, Jun Du, Rong Xie, Xiaokang Yang, and Wenjun Zhang. You are what you watch and when you watch: Inferring household structures from IPTV viewing data. *Transactions on Broadcasting*, 60(1):61–72, 2014.

[Luo *et al.*, 2015] Dixin Luo, Hongteng Xu, Yi Zhen, Xia Ning, Hongyuan Zha, Xiaokang Yang, and Wenjun Zhang. Multitask multi-dimensional Hawkes processes for modeling event sequences. In *IJCAI*, 2015.

[Ogata, 1981] Yosihiko Ogata. On Lewis' simulation method for point processes. *Transactions on Information Theory*, 27(1):23–31, 1981.

[Ouyang *et al.*, 2013] Hua Ouyang, Niao He, Long Tran, and Alexander Gray. Stochastic alternating direction method of multipliers. In *ICML*, 2013.

[Ozerov *et al.*, 2011] Alexey Ozerov, Cédric Févotte, Raphaël Blouet, and Jean-Louis Durrieu. Multichannel nonnegative tensor factorization with structured constraints for user-guided audio source separation. In *ICASSP*, 2011.

[Park and Chu, 2009] Seung-Taek Park and Wei Chu. Pairwise preference regression for cold-start recommendation. In *RecSys*, 2009.

[Rafailidis and Nanopoulos, 2014] Dimitrios Rafailidis and Alexandros Nanopoulos. Modeling the dynamics of user preferences in coupled tensor factorization. In *RecSys*, 2014.

[Rai *et al.*, 2015] Piyush Rai, Yingjian Wang, and Lawrence Carin. Leveraging features and networks for probabilistic tensor decomposition. In *AAAI*, 2015.

[Rendle and Schmidt-Thieme, 2010] Steffen Rendle and Lars Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *WSDM*, 2010.

[Shamir, 2015] Ohad Shamir. The sample complexity of learning linear predictors with the squared loss. *Journal of Machine Learning Research*, 16:3475–3486, 2015.

[Shashua and Hazan, 2005] Amnon Shashua and Tamir Hazan. Non-negative tensor factorization with applications to statistics and computer vision. In *ICML*, 2005.

[Tucker, 1966] Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[Wang *et al.*, 2016] Yichen Wang, Nan Du, Rakshit Trivedi, and Le Song. Coevolutionary latent feature processes for continuous-time user-item interactions. In *NIPS*, 2016.

[Xiong *et al.*, 2010] Liang Xiong, Xi Chen, Tzu-Kuo Huang, Jeff Schneider, and Jaime G Carbonell. Temporal collaborative filtering with Bayesian probabilistic tensor factorization. In *SDM*, 2010.

[Xu and Zha, 2017] Hongteng Xu and Hongyuan Zha. A Dirichlet mixture model of Hawkes processes for event sequence clustering. In *NIPS*, 2017.

[Xu *et al.*, 2015] Hongteng Xu, Yi Zhen, and Hongyuan Zha. Trailer generation via a point process-based visual attractiveness model. In *IJCAI*, 2015.

[Xu *et al.*, 2017a] Hongteng Xu, Dixin Luo, and Hongyuan Zha. Learning Hawkes processes from short doubly-censored event sequences. In *ICML*, 2017.

[Xu *et al.*, 2017b] Hongteng Xu, Weichang Wu, Shamim Nemati, and Hongyuan Zha. Patient flow prediction via discriminative learning of mutually-correcting processes. *Transactions on Knowledge and Data Engineering*, 29(1):157–171, 2017.

[Yang *et al.*, 2017] Yingxiang Yang, Jalal Etesami, Niao He, and Negar Kiyavash. Online learning for multivariate Hawkes processes. In *NIPS*, 2017.

[Zass and Shashua, 2007] Ron Zass and Amnon Shashua. Nonnegative sparse PCA. In *NIPS*, 2007.

[Zhong and Kwok, 2014] Wenliang Zhong and James Kwok. Fast stochastic alternating direction method of multipliers. In *ICML*, 2014.