Decoupled Learning for Factorial Marked Temporal Point Processes

Weichang Wu
Shanghai Jiao Tong University
blade091@sjtu.edu.cn

Junchi Yan*
Shanghai Jiao Tong University
yanjunchi@sjtu.edu.cn

Xiaokang Yang
Shanghai Jiao Tong University
xkyang@sjtu.edu.cn

Hongyuan Zha
Georgia Institute of Technology
zha@cc.gatech.edu

ABSTRACT

This paper presents a factorial marked temporal point process model and presents efficient learning methods. In conventional (multi-dimensional) marked temporal point process models, an event is often encoded by a single discrete variable (marker). We describe the factorial marked point processes whereby time-stamped event is factored into multiple markers. Accordingly the size of the infectivity matrix modeling the effect between pairwise markers is in exponential order regarding the number of discrete markers.

We propose a decoupled learning method with two learning procedures: i) directly solving the model based on two techniques: Alternating Direction Method of Multipliers and Fast Iterative Shrinkage-Thresholding Algorithm; ii) involving a reformulation that transforms the original problem into a Logistic Regression model for more efficient learning. Moreover, a sparse group regularizer is added to identify the key profile features and event labels. Empirical results on real world datasets demonstrate the efficiency of our decoupled and reformulated method.

CCS CONCEPTS

• **Mathematics of computing** → **Stochastic processes**; • **Information systems** → **Data stream mining**; • **Theory of computation** → *Theory and algorithms for application domains*;

KEYWORDS

Factorial Temporal Point Process, Decoupled Learning, Alternating Direction Method of Multipliers, Fast Iterative Shrinkage-Thresholding Algorithm

*Corresponding author.

KDD '18, August 19–23, 2018, London, United Kingdom

ACM ISBN 978-1-4503-5552-0/18/08…$15.00
https://doi.org/10.1145/3219819.3220035

ACM Reference Format:
Weichang Wu, Junchi Yan*, Xiaokang Yang, and Hongyuan Zha. 2018. Decoupled Learning for Factorial Marked Temporal Point Processes. In *KDD '18: The 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, August 19–23, 2018, London, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3219819.3220035

1 INTRODUCTION AND BACKGROUND

Events are ubiquitous across different domains and applications. In e-commerce, events refer to the transactions associated with users, items. In health informatics, an event sequence can be a series of treatments over time of a patient. In predictive maintenance, events can carry important log data for when the failure occurs and what is the type. In all these examples, effectively modeling and predicting the dynamic behavior is of vital importance for practical usefulness.

Marked temporal point process Point process [2] is a useful tool for modeling the event sequence with arbitrary timestamp for each event. An event in point process can carry extra information called marker. The marker typically refers to event type and lies in the discrete label space i.e. a finite category set $\{1, ..., m\}$[1].

Factorial marked temporal point process For the above mentioned marked point process, the event is represented by a single mark as a single discrete variable. But in many applications, event can carry multiple markers. For instance, a movement to a new job carries both the labels for position and company, which can be treated by two orthogonal markers with different values. Though such cases are ubiquitous, the factorial marked point processes have drawn little attention in literature as existing literatures mostly work with a single marker [13, 22, 25]. Inspired by Factorial Hidden Markov Models [7], we introduce the factorial marked temporal point process, in which the event is represented by

[1] A general concept can be found in [6]: *a marked point pattern is one in which each point of the process carries extra information called a mark, which may be a random variable, several random variables, a geometrical shape, or some other information.* In this paper, we focus on discrete labels for marks. The marked point process is also termed by *multi-dimensional point process* [13], where each dimension refers to a discrete mark value.

multiple markers, and propose a decoupling method to learn the process.

Intensity function and problem statement One core concept for point process is the intensity function $\lambda(t)$, which represents the expected instantaneous rate of events at time t conditioned on the history. One basic intensity function is the constant $\lambda(t) = \lambda_0$ over time, as used in the homogeneous Poisson process. Another popular form is the one used by the Hawkes process [9]: $\lambda(t) = \gamma_0 + \alpha \sum_{t \in \tau} \gamma(t, t_i)$, where τ denotes the event history and $\gamma(t, t_i) \geq 0$ is a marker-vs.-marker infectivity kernel capturing the temporal dependency between events at t and at t_i.

In this paper, we are interested in describing a factorial marked point process for event marker prediction task by using the history event information and the individual level profile of an event taker. We focus on the next-event label estimation, distributed over more than one markers. In particular, our empirical study focuses on individual level next job prediction involving both position and company for LinkedIn users, and duration prediction in current ICU department and transition prediction to next ICU department for patients in MIMIC-II database [8].

2 RELATED WORK AND CONTRIBUTION

Learning for temporal point process Point process is powerful for modeling event sequence with timestamp in continuous time space. Early work dates back to the Hawkes processes [9] which shows appropriateness for self-exciting and mutual-exciting process like earthquake and its aftershock [18, 19]. The learning is fulfilled by maximum likelihood estimation by directly computing the gradient and Hessian matrix w.r.t. the log-likelihood function [20]. Recently more modernized machine learning approaches devise new efficient algorithms for learning the parameters of the specified point process. Nonparametric Expectation-Maximization (EM) algorithm is proposed in [11] for multiscale Hawkes Processes using the majorization-minimization framework, which shows superior efficiency and robustness compared with sampling based estimation methods. [25] extends the technique to handle the multi-dimensional Hawkes process by adding a low-rank and sparsity regularization term in the maximum likelihood estimation (MLE) based loss.

Factorial model Though almost all of these works mentioned above involve the infectivity matrix for model parameters learning, none of them considers the factorial temporal point process case, i.e. an event type is factored into multiple markers, which leads to the explosion of the infectivity matrix size. The idea of factorizing events or states into multiple variables is employed in [7] for Hidden Markov Models (HMM) using variational methods to solve data mining task like capturing statistical structure, but little literature is found about its utility in point process. To our best knowledge, this is the first work of factorial marked point process learning for event marker prediction. Note timestamp prediction can be approximated by predicting a predefined time interval as time duration marker.

Sparse regularization for point process Sparse regularization is a well-established technique in traditional classification and regression models, such as the ℓ_1 regularizer [17], group Lasso [16], sparse group regularizer [21], etc. Recent point process models have also found their applications like the ℓ_1 regularization used in [12] to ensure the sparsity of social infectivity matrix, the nuclear norm in [25] to get a low-rank network structure and the group Lasso in [23] for feature selection. We propose to use the sparse group regularizer, which encourages the nonzero elements focusing on a few columns obeying with the intuition that only a few features and labels play the major role in event dynamics. We find little work in literature on group sparse regularizer for point process learning.

Contributions The main contributions of this paper are:

1) We introduce the concept of factorial marked point process for event marker prediction, and propose a decoupled learning algorithm to simplify the factorial model by decoupling the marker mutual relation in modeling. The method outperforms general marked point process on real-world datasets.

2) We present a multi-label Logistic Regression (LR) perspective and devise reformulation towards a class of point process discriminative learning problems. It eases the learning of these processes by using on-the-shelf LR solver.

Besides these major contributions, we also make additional improvements in proposing a regularized learning objective, which we will include for completeness.

3 PROPOSED MODEL AND OBJECTIVE FUNCTION

3.1 Factorial point process

Factorial point process refers to the processes in which event can be factorized into multiple markers. Except the *job movement prediction* and *ICU department prediction* mentioned in Introduction, many application cases can be described by the factorial point process, while haven't been explored yet. For instance, a *weather forecast* containing *temperature*, *humidity*, *precipitation*, and *wind* can be seen as a factorial point process with 4 markers, with each marker having discrete or continuous values. Obviously these factors affect each other, e.g. the *humidity* today is influenced by the *precipitation* and *temperature* in recent few days. The conventional marked point process could only model one of these factors using a single marker without considering the infectivity between these factors. A factorial point process with multiple markers for the event is essential.

Learning factorial point process is challenging. Taking *job movement prediction* with two markers *company* c and *position* p as example: to predict the probability of user x's n-th job (c_n, p_n), we need to learn a 4-dimension tensor representing the impact of history *companies* $\{c_i\}_{i=1}^{n-1}$ on c_n and p_n, the impact of history *positions* $\{p_i\}_{i=1}^{n-1}$ on c_n and p_n, respectively. In point process, it means we need to learn

a set of intensity functions including $\lambda(c,c)$, $\lambda(c,p)$, $\lambda(p,c)$ and $\lambda(p,p)$. This simple case considers no infectivity between different sequences, i.e., if we also consider the impact of another user y's job movement on user x's choice of c_n and p_n, we would compute a 6-dimension tensor to measure the complete infectivity, with two extra intensities $\lambda(y,c)$ and $\lambda(y,p)$.

There are ways to simplify factorial point process learning, e.g. we can treat the combination of multiple factors as one marker, and use the conventional marked point process model, but this will lead to explosion of the size of the infectivity matrix. In this paper, we explore a simple decoupling solution that decouple the factorial point process into separate models of different markers respectively. As shown in Fig. 1 for the instance of 2 markers c and p, we decouple the original infectivity matrix into smaller ones by introducing 4 tensor variable $\boldsymbol{a}_{pp}$, $\boldsymbol{a}_{pc}$, $\boldsymbol{a}_{cp}$ and $\boldsymbol{a}_{cc}$. We will present the decoupled model in details in the following section. For the better illustration of the model, we summarize all important notations in Table 1.

3.2 Decoupled learning for factorial point process

More generally, we discuss the situation that event can be factorized into Z markers $(m_1, m_2, \ldots, m_Z)$. Given event sequence u with event marker $m_i = \{(m_{i,1}, m_{i,2}, \ldots, m_{i,Z})\}$ for $m_{i,z} \in \{1, 2, ..., M_z\}$ where $z \in \{1, 2, \ldots, Z\}$, the intensity function of a conventional marked point process for marker m is defined by:

$$\lambda_m^u(t) = f\left(\boldsymbol{\theta}_m^\top \boldsymbol{x}_o^u h(t) + \sum_{i:t_i^u<t} \boldsymbol{a}_{mm_i} g(t, t_i)\right), \qquad (1)$$

where $\boldsymbol{x}_0^u \in \mathbb{R}^M$ is the time-invariant features of sequence taker u extracted from its profile, like *Self-introduction* of LinkedIn users or patients' *diagnose* in MIMIC-II database, and $\boldsymbol{\theta}_m \in \mathbb{R}^M$ is the corresponding coefficients.

For the choice of the three functions f, h, g in Eq. 1, there are many forms in the literature that can be abstracted by the above framework, and popular ones are depicted in Table 2.

For marked point process model, when the marker contains multiple label dimensions, one major bottleneck is that this model involves the infectivity matrix $\boldsymbol{a}$ with size $(\prod_{z=1}^Z M_z) \times (\prod_{z=1}^Z M_z)$ to measure the directional effect between m_i and m_j. More generally, the size of the infectivity matrix $\boldsymbol{a}$ is $O(n^{2Z})$ (assume all dimensions M_z have same number of values by n), which incurs learning difficulty.

To mitigate the challenge for learning the above parameter matrix, especially when $\prod_{z=1}^Z M_z$ is large while the sequences are relatively short, we propose the decoupled factorial point process model to linearly decouple the above intensity function into Z interdependent point processes

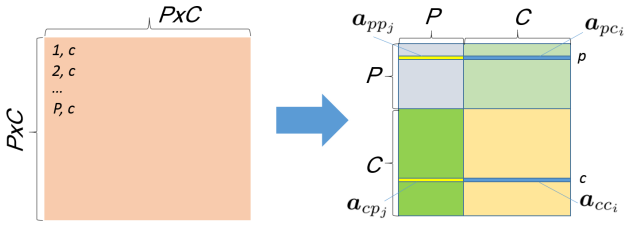


Figure 1: Decoupling infectivity matrix between two markers C and P from size $\mathbb{R}^{(P\times C)\times(P\times C)}$ to $\mathbb{R}^{(P+C)\times(P+C)}$. Note the indicated rows for $\boldsymbol{a}_{pp_j}$, $\boldsymbol{a}_{pc_i}$, $\boldsymbol{a}_{cp_j}$, $\boldsymbol{a}_{cc_i}$.

$\lambda_{m_z}^u(t), z \in \{1, 2, \ldots, Z\}$ for the z-th marker by:

$$f\Bigg(\underbrace{\boldsymbol{\theta}_z^\top \boldsymbol{x}_o^u h(t)}_{\text{effect by profile}} + \underbrace{\sum_{i:t_i^u<t} \boldsymbol{a}_{zz_i} \boldsymbol{b}_{z_i}^u g(t, t_i)}_{\text{effect by former markers } z_i} + \underbrace{\sum_{\substack{y=1 \\ y\neq z}}^{Z} \sum_{j:t_j^u<t} \boldsymbol{a}_{zy_j} \boldsymbol{b}_{y_j}^u g(t, t_j)}_{\text{effect by former markers } y_i}\Bigg) \qquad (2)$$

where $\boldsymbol{b}_{z_i}^u \in \{0,1\}^{M_z}$, $\boldsymbol{b}_{y_j}^u \in \{0,1\}^{M_y}$ is the binary indicators connecting through the influence of one's former marker z_i and markers y_j. Note the row vector $\boldsymbol{a}_{zz_i} \in \mathbb{R}^{1\times M_z}$ is the parameter for intra-influence within marker $\{z_i\}$, and $\boldsymbol{a}_{zy_i} \in \mathbb{R}^{1\times M_y}$ is for inter-influence between z_i and y_i. The above vectors are illustrated in Fig. 1 when $Z = 2$, using notation c as z_1 and p as z_2.

In fact, functions $f(\cdot)$, $g(\cdot)$, $h(\cdot)$ are predefined and some embodiments can be chosen from Table 2. For the time being, we do not specify these functions while focus on solving the learning problem in a general setting. One concrete example is presented in Fig. 2. See the caption for more details.

3.3 Next-event marker prediction with regularizer

Loss function for discriminative learning Based on the defined intensity function, we write out the probability $P(m, t|\mathcal{H}_t^u)$ for event $m = \{m_1, m_2, \ldots, m_Z\}$ happens at time t, conditioned on u's history $\mathcal{H}_t^u$:

$$\begin{aligned} P(m,t|\mathcal{H}_t^u) =& \lambda_m^u(t) \exp\left(-\sum_{m'} \int_{t_I^u}^t \lambda_{m'}^u(s) ds\right) \\ =& \frac{\lambda_m^u(t)}{\sum \lambda_{m'}^u(t)} \times \frac{\sum_{m'} \lambda_{m'}^u(t)}{\exp\left(\sum \int_{t_I^u}^t \lambda_{m'}^u(s) ds\right)} \\ =& P(m|t, \mathcal{H}_t^u) \times P(t|\mathcal{H}_t^u), \end{aligned} \qquad (3)$$

where $\lambda_m^u(t)$ is the event intensity, and $P(t|\mathcal{H}_t^u)$ is the conditional probability that this event happens at time t given history $\mathcal{H}_t^u$. $P(m|t, \mathcal{H}_t^u)$ is the probability that the happened event is m given current time t and history $\mathcal{H}_t^u$.

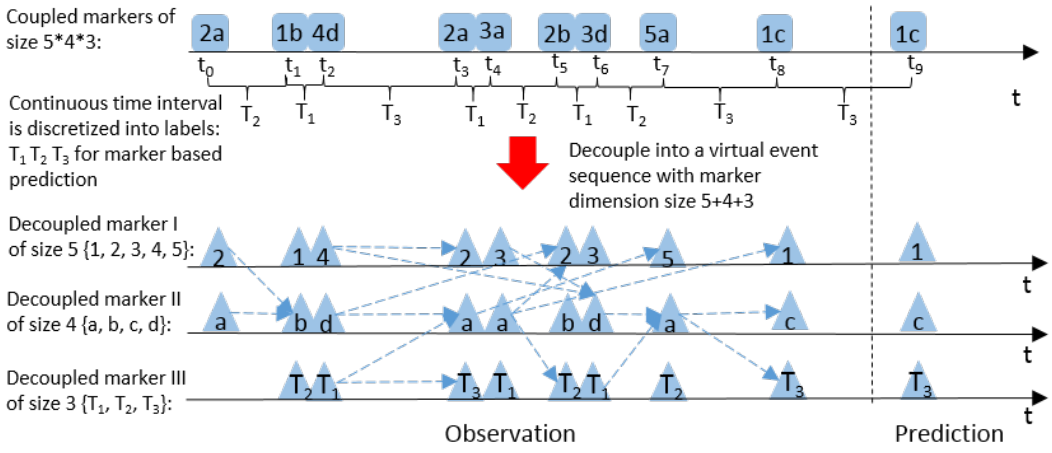


Figure 2: Example of our decoupling perspective on factorial event sequence learning. The raw event sequence is represented with three-marker of label space $5 \times 4 \times 3$. Our decouple model treats the raw sequence as an overlay of three sequences whose marker space is 5, 4, 3 respectively and then the whole marker space's dimension is in linear: $5 + 4 + 3$. The directed dash arrows between events sketch the effect from previous events to future events: not only within one of the three sequences but also across the three sequences. Two particular attentions shall be paid to our model: 1) the method is designated to predict next event's marker but not for its continuous occurrence timestamp (see more details later in the paper). To enable next event time prediction, we discretize the time interval into several levels as illustrated by $\{T_2, T_1, T_3\}$. 2) On the other hand, the accurate timestamp $\{t_1, t_2, \dots\}$ rather than the discretized version $\{T_2, T_1, T_3\}$, is used for learning of the point process model which makes sure our model can capture the fine-grained raw time information.

Based on the above equation, most existing point process learning methods e.g. [5, 11, 26] fall into the generative learning framework aiming to maximize the joint probability of all observed events via a maximum likelihood estimator $\max_{\boldsymbol{\Theta}} \prod_{u,i} P(m_i^u, t_i^u | \mathcal{H}_{t_i^u}^u)$.

However, such an objective function is not tailored to the particular task at hand: instead of taking care of handling the posterior probability of the whole event sequence, we are more interested in predicting the next event and its mark information. To enable a more discriminative learning paradigm to boost the next event prediction accuracy, a recent work [23] suggests to focus on $P(m|t, \mathcal{H}_t^u)$ instead of $P(m, t|\mathcal{H}_t^u)$ as the loss function for learning.

In the decoupled point process model, the dependency between different markers has been measured by inter-influence parameters, i.e., the dependency between process $\lambda_{m_z}^u(t)$ for marker m_z and process $\lambda_{m_y}^u(t)$ for marker m_y has been measured by parameter $\boldsymbol{a}_{zy_i}$ and $\boldsymbol{a}_{yz_i}$ (see Eq. 2) in an independent fashion. In the same spirit, here we simplify the probability $P(m|t, \mathcal{H}_t^u)$ by an independence assumption:

$$\begin{aligned} P(m|t, \mathcal{H}_t^u) =& P(m_1, m_2, \dots, m_Z|t, \mathcal{H}_t^u) \\ =& \prod_{z=1}^{Z} P(m_z|t, \mathcal{H}_t^u) = \prod_{z=1}^{Z} \frac{\lambda_{m_z}^u(t)}{\sum_{m_z'} \lambda_{m_z'}^u(t)}, \end{aligned} \tag{4}$$

where $P(m_z|t, \mathcal{H}_t^u)$ is the normalized intensity function. This simplification leads to the following loss:

$$\begin{aligned} L(\boldsymbol{\Theta}) =& -\sum_{u=1}^{U} \sum_{i=1}^{N^u} \sum_{z=1}^{Z} \sum_{m_z=1}^{M_z} 1\{m_{i,z}^u = m_z\} \log P(m_z|t_{i-1}^u, \mathcal{H}_{t_{i-1}^u}^u) \\ =& -\sum_{u=1}^{U} \sum_{i=1}^{N^u} \log \left(\prod_{z=1}^{Z} \frac{\lambda_{m_{i,z}}^u(t_{i-1}^u)}{\sum_{m_z'} \lambda_{m_z'}^u(t_{i-1}^u)} \right), \end{aligned} \tag{5}$$

where $1\{statement\}$ is an indicator returning 1 if true, otherwise 0. And

$$\boldsymbol{\Theta} = \{\boldsymbol{\Theta_1}; \boldsymbol{\Theta_2}; \dots; \boldsymbol{\Theta_z}; \dots; \boldsymbol{\Theta_Z}\} \in \mathbb{R}^{(\sum_{z=1}^{Z} M_z) \times (M + \sum_{z=1}^{Z} M_z)}$$

is the parameters whereby

$$\boldsymbol{\Theta_z} = [\boldsymbol{\theta_z}^\top, \boldsymbol{a_{zz}}, \boldsymbol{a_{zy}}|_{y=1, y \neq z}^{Z}] \in \mathbb{R}^{M_z \times (M + \sum_{z=1}^{Z} M_z)}.$$

Sparse group regularization Since the model involves many parameters for learning, a natural idea is introducing sparsity to reduce the complexity. Incorporating both group sparsity and overall sparsity, we use a sparse group regularizer [21] as the regularization, a combination of ℓ_1 regularization $\|\boldsymbol{\Theta}\|_1$ and a group lasso $\|\boldsymbol{\Theta}\|_{1,2} = \sum_{j=1}^{M + \sum_{z=1}^{Z} M_z} \|\theta_j\|_2$. The group lasso encourages the nonzero elements concentrated on a few columns in the whole matrix $\boldsymbol{\Theta}$, and the rest part is assumed to be zeros. The ℓ_1 regularization encourages the whole matrix $\boldsymbol{\Theta}$ to be sparse. The behind rationale is that only a few profile features and event marker values will be the main contributor to the point process. This means only

Symbol	Definition
U	The number of event sequences.
u	The u-th sequence, $u \in \{1, \dots, U\}$
N^u	The number of events in u-th sequence.
i	The i-th event in a sequence, $i \in \{1, \dots, N^u\}$.
Z	An event is factorized into Z markers.
z	The z-th marker for an event, $z \in \{1, \dots, Z\}$.
M_z	The z-th marker can take M_z discrete values.
M	The dimension of profile feature.
m_z	The value of the z-th marker, $m_z \in \{1, \dots, M_z\}$.
$m^u_{i,z}$	The value of i-th event's z-th marker in u-th sequence.
$\lambda^u_{m_{i,z}}$	The intensity of the i-th event's z-th marker in u-th sequence.
$\boldsymbol{\theta}_z$	The parameter modeling the influence of profile feature to marker m_z
$\boldsymbol{a}_{zz_i}$	The parameter modeling intra-influence of history marker m_{z_i} to marker m_z.
$\boldsymbol{a}_{zy_j}$	The parameter modeling inter-influence of history marker m_{y_j} to marker m_z

Table 1: Important Notations

Model	$f(x)$	$h(t)$	$g(t,t')$
Modulated Poisson process (**MPP**) [15]	x	1	1
Hawkes process (**HP**) [11]	x	1	$e^{-w(t-t')}$
Self-correcting process (**SCP**) [10]	e^x	t	1
Mutually-correcting process (**MCP**) [23]	e^x	$t - t_I$	$e^{-\frac{(t-t')^2}{\sigma^2}}$

Table 2: Parametric forms of popular point processes.

a few columns will be activated. As a result, the regularized objective is:

$$\min_{\boldsymbol{\Theta}} L(\boldsymbol{\Theta}) + \lambda_1 \|\boldsymbol{\Theta}\|_1 + \lambda_2 \sum_{j=1}^{N} \|\theta_j\|_2, \tag{6}$$

where $N = M + \sum_{z=1}^{Z} M_z$, $\boldsymbol{\Theta} \in \mathbb{R}^{(\sum_{z=1}^{Z} M_z) \times N}$, $\lambda_1 = \alpha\lambda$ and $\lambda_2 = (1-\alpha)\lambda$, λ is the regularization weight, $\alpha \in (0,1)$ controls the balance between overall and group sparsity.

4 LEARNING ALGORITHM

In this section, we first present our tailored algorithm to the presented model. Then we give a new perspective and show how to reformulate it into a Logistic Regression task.

Following the scheme of ADMM, we propose a FISTA [1] based method with Line Search and two soft-shrinkage operators to solve the subproblems of ADMM. The algorithm is summarized in Alg. 2.

4.1 Soft Shrinkage Operator

First we review two soft-shrinkage operators [3] solving the following two basic minimization problem, as will be used later.

- The minimization problem

$$\arg\min_{\boldsymbol{\theta} \in \mathbb{R}^p} \{\|\boldsymbol{\theta}\|_1 + \frac{u}{2}\|\boldsymbol{\theta} - \boldsymbol{r}\|^2\},$$

with $u > 0$, $\boldsymbol{\theta} \in \mathbb{R}^p$, $\boldsymbol{r} \in \mathbb{R}^p$, has a closed-form solution given by the soft-shrinkage operator $\mathbf{shrink_{1,2}}$ defined:

$$\boldsymbol{\theta}^* = \text{shrink}_{1,2}(\boldsymbol{r}, 1/u) \triangleq \text{sign}(\boldsymbol{r}) \cdot \max\{0, |\boldsymbol{r}| - 1/u\},$$

where $\text{sign}(\cdot)$ is the sign function.

- The minimization problem

$$\arg\min_{\boldsymbol{\theta} \in \mathbb{R}^p} \{\|\boldsymbol{\theta}\|_2 + \frac{u}{2}\|\boldsymbol{\theta} - \boldsymbol{r}\|^2\},$$

with $u > 0$, $\boldsymbol{\theta} \in \mathbb{R}^p$, $\boldsymbol{r} \in \mathbb{R}^p$, has a closed-form solution given by the soft-shrinkage operator $\mathbf{shrink_{2,2}}$ defined:

$$\boldsymbol{\theta}^* = \text{shrink}_{2,2}(\boldsymbol{r}, 1/u) \triangleq \frac{\boldsymbol{r}}{\|\boldsymbol{r}\|_2} \cdot \max\{0, \|\boldsymbol{r}\|_2 - 1/u\}.$$

4.2 ADMM iteration scheme

To solve the minimization problem defined in Eq. 6 using ADMM solver, we add two auxiliary variables β, γ. The augmented Lagrangian function for Eq. 6 is $L^s_u(\boldsymbol{\Theta}, \boldsymbol{\beta}, \boldsymbol{\gamma}) =$

$$L(\boldsymbol{\Theta}) + \lambda_1\|\boldsymbol{\Theta}\|_1 + \lambda_2 \sum_{j=1}^{N} \|\gamma_j\|_2 - \boldsymbol{\beta}^T(\boldsymbol{\Theta} - \boldsymbol{\gamma}) + \frac{u}{2}\|\boldsymbol{\Theta} - \boldsymbol{\gamma}\|_2^2, \tag{7}$$

where $\boldsymbol{\beta} = (\beta_1, \dots, \beta_N)$, $\boldsymbol{\gamma} = (\gamma_1, \dots, \gamma_N)$, $\boldsymbol{\Theta} = (\theta_1, \dots, \theta_N)$, and $\beta_j, \gamma_j, \theta_j \in \mathbb{R}^{C+P}$.

The iterative scheme is given by:

$$\begin{aligned} \boldsymbol{\Theta}^{k+1} &= \arg\min_{\boldsymbol{\Theta}} L^s_u(\boldsymbol{\Theta}, \boldsymbol{\beta}^k, \boldsymbol{\gamma}^k) \qquad (8)\\ &= \arg\min_{\boldsymbol{\Theta}} \{L(\boldsymbol{\Theta}) - \boldsymbol{\beta}^T(\boldsymbol{\Theta} - \boldsymbol{\gamma}) + \frac{u}{2}\|\boldsymbol{\Theta} - \boldsymbol{\gamma}\|_2^2 + \lambda_1\|\boldsymbol{\Theta}\|_1\} \\ \boldsymbol{\gamma}^{k+1} &= \arg\min_{\boldsymbol{\gamma}} L^s_u(\boldsymbol{\Theta}^{k+1}, \boldsymbol{\beta}^k, \boldsymbol{\gamma}) \qquad (9)\\ &= \arg\min_{\boldsymbol{\gamma}} \{\lambda_2 \sum_{j=1}^{N} \|\gamma_j\|_2 - \boldsymbol{\beta}^T(\boldsymbol{\Theta} - \boldsymbol{\gamma}) + \frac{u}{2}\|\boldsymbol{\Theta} - \boldsymbol{\gamma}\|_2^2\} \\ \boldsymbol{\beta}^{k+1} &= \boldsymbol{\beta}^k - u(\boldsymbol{\Theta}^{k+1} - \boldsymbol{\gamma}^{k+1}) \qquad (10) \end{aligned}$$

Therefore the optimization of Function 7 has been divided into two sub-problems defined as Eq. 8 and Eq. 9. While for Eq. 9, the update of $\boldsymbol{\gamma}$, it has a closed-form solution given by operator $\mathbf{shrink_{2,2}}$ as follows

$$\begin{aligned} \gamma_j^{k+1} &= \arg\min_{\gamma_j} \{\lambda_2\|\gamma_j\|_2 + \frac{u}{2}\|\gamma_j - (\boldsymbol{\Theta}^{k+1} - \boldsymbol{\beta}^k/u)_j\|_2^2\} \\ &= \mathbf{shrink}_{2,2}((\boldsymbol{\Theta}^{k+1} - \boldsymbol{\beta}^k/u)_j, \lambda_2/u) \qquad (11)\\ &= \nu_j^k - P_{\mathcal{D}_j}(\nu_j^k), \end{aligned}$$

where $\nu_j^k = (\boldsymbol{\Theta}^{k+1} - \boldsymbol{\beta}^k/u)_j$, $\mathcal{D}_j$ denotes the ball in p_j-dimension centered at 0 with radius λ_2/u [3].

4.3 FISTA with line search

To solve Eq. 8 we define $g(\boldsymbol{\Theta}) = L(\boldsymbol{\Theta}) + \frac{u}{2}\|\boldsymbol{\Theta} - \boldsymbol{\gamma}^k - \boldsymbol{\beta}^k/u\|_2^2$, then $\boldsymbol{\Theta}^{k+1}$ can be obtained by solving $\boldsymbol{\Theta} = \arg\min_{\boldsymbol{\Theta}} \{g(\boldsymbol{\Theta}) + \lambda_1\|\boldsymbol{\Theta}\|_1\}$ through a FISTA method [1] with line search to compute the step size. The Algorithm is summarized in Alg.1

4.4 Reformulating to Logistic Regression task

Based on the intensity function Eq. 2, the loss function Eq. 5, we show how to reformulate the learning of the decoupled point process as a multi-class Logistic Regression task. One

Algorithm 1: FISTA($\boldsymbol{\Theta}^k$)

Input: $\boldsymbol{\Theta}^k$ from last iteration
Initialize $\boldsymbol{\Theta}^{(k,0)} = \boldsymbol{v}^{(0)} = \boldsymbol{\Theta}^k$, threshold $\epsilon = 0.01$, $i = 1$,$\tau_i = \frac{2}{i+1}$,$t_0 = \hat{t} > 0$, $\eta = 0.8$;
while $\frac{\|\boldsymbol{\Theta}^{(k,i)} - \boldsymbol{\Theta}^{(k,i-1)}\|_2}{\|\boldsymbol{\Theta}^{(k,i)}\|_2^2} \leq \epsilon$ **do**
  $\boldsymbol{y} = (1-\tau_k)\boldsymbol{\Theta}^{(k,i-1)} + \tau_k \boldsymbol{v}^{(i-1)}$, $t = t_{i-1}$;
  $\boldsymbol{\Theta}^{(k,i)} = \mathbf{shrink}_{1,2}(\boldsymbol{y} - \frac{1}{t}\nabla g(\boldsymbol{y}), \lambda_1/t)$;
  while $g(\boldsymbol{\Theta}^{(k,i)}) >$ $g(\boldsymbol{y}) + \nabla g(\boldsymbol{y})^T(\boldsymbol{\Theta}^{(k,i)} - \boldsymbol{y}) + \frac{1}{2t}\|\boldsymbol{\Theta}^{(k,i)} - \boldsymbol{y}\|_2^2$ **do**
    $t = \eta \cdot t$;
    $\boldsymbol{\Theta}^{(k,i)} = \mathbf{shrink}_{1,2}(\boldsymbol{y} - \frac{1}{t}\nabla g(\boldsymbol{y}), \lambda_1/t)$;
  $\boldsymbol{v}^{(i)} = \boldsymbol{\Theta}^{(k,i-1)} + \frac{1}{\tau_k}(\boldsymbol{\Theta}^{(k,i)} - \boldsymbol{\Theta}^{(k,i-1)})$, $i = i+1$;
$\boldsymbol{\Theta}^{k+1} = \boldsymbol{\Theta}^{(k,i)}$, **return** $\boldsymbol{\Theta}^{k+1}$;

Algorithm 2: Decoupled Learning of Factorial Point Process

Input: two associated marked point process $\{(c_i, p_i, t_i)\}$, $\lambda_1 > 0$,$\lambda_2 > 0$, threshold $\epsilon = 0.01$
Initialize $(\boldsymbol{\Theta}, \boldsymbol{\beta}, \boldsymbol{\gamma}) = (\mathbf{0}, \mathbf{0}, \mathbf{0})$;
while $\frac{\|\boldsymbol{\Theta}^k - \boldsymbol{\Theta}^{k-1}\|_2}{\|\boldsymbol{\Theta}^k\|_2} \leq \epsilon$ **do**
  Update $\boldsymbol{\Theta}^{k+1}$ via $\boldsymbol{\Theta}^{k+1} = \mathbf{FISTA}(\boldsymbol{\Theta}^k)$;
  Compute $\boldsymbol{\gamma}^{k+1}$ via Eq. 11; Update $\boldsymbol{\beta}^{k+1}$ via $\boldsymbol{\beta}^{k+1} = \boldsymbol{\beta}^k - u(\boldsymbol{\Theta}^{k+1} - \boldsymbol{\gamma}^{k+1})$;
  $k = k+1$;
Output: $\boldsymbol{\Theta}^k$

obvious merit of this reformulation is the reuse of on-the-shelf LR solvers e.g. http://www.yelab.net/software/SLEP/ [14] with little parameter tuning. In contrast, the algorithm presented in Alg.2 involves more parameters and is more computationally costly as shown in Fig. 4 and Table 3.

For event taker u at time t, by separating the event taker u's feature

$$\boldsymbol{f}_{\boldsymbol{t}}^{\boldsymbol{u}} = [x_0^u h(t), \sum_{i:t_i^u<t} b_{z_i}^u g(t,t_i), \sum_{y=1,y\neq z}^{Z} \sum_{j:t_j^u} b_{y_j}^u g(t,t_j)]$$

and $\boldsymbol{f}_{\boldsymbol{t}}^{\boldsymbol{u}} \in \mathbb{R}^{M+\sum_{z=1}^{Z} M_z}$ from the parameters $\boldsymbol{\Theta}_z$, the conditional intensity function in Eq. 2 can be written as:

$$\lambda_{m_z}^u(t) = \exp(\boldsymbol{\Theta}_z \boldsymbol{f}_{\boldsymbol{t}}^{\boldsymbol{u}}) \tag{12}$$

Therefore the probability $P(p|t, \mathcal{H}_t^u)$ can be written as:

$$P(m_z|t, \mathcal{H}_t^u) = \frac{\exp(\boldsymbol{\Theta}_z \boldsymbol{f}_{\boldsymbol{t}}^{\boldsymbol{u}})}{\sum_{z'} \exp(\boldsymbol{\Theta}_{z'} \boldsymbol{f}_{\boldsymbol{t}}^{\boldsymbol{u}})}. \tag{13}$$

This is exactly the same probability function of sample $\boldsymbol{f}_{\boldsymbol{t}}^{\boldsymbol{u}}$ belonging to class m_z for a Softmax classifier of M_z classes, and $\boldsymbol{\Theta}_z$ is the parameter.

Hence the log-loss function in Eq. 5 becomes:

$$L(\boldsymbol{\Theta}) = -\sum_{u=1}^{U}\sum_{i=1}^{N^u} \log\left(\prod_{z=1}^{Z} \frac{\exp(\boldsymbol{\Theta}_z \boldsymbol{f}_{\boldsymbol{t_i}}^{\boldsymbol{u}})}{\sum_{z'} \exp(\boldsymbol{\Theta}_{z'} \boldsymbol{f}_{\boldsymbol{t_i}}^{\boldsymbol{u}})}\right),$$

which is the sum of Z Softmax classifiers' loss functions.

So far we have reformulated the decoupled learning of the factorial marked point process to the learning of Z Softmax classifiers. For the z-th classifier, it takes $\boldsymbol{f}_{\boldsymbol{t}}^{\boldsymbol{u}}$ from the sample and classify it to one of M_z markers $\hat{m}_z$. In the following experiments, we will show that the reformulated learning method in fact optimizes the same loss function as Alg.2.

4.5 Event marker prediction

After learning parameters $\boldsymbol{\Theta} = \{\boldsymbol{\Theta}_z|_{z=1}^Z\}$, we can predict the next event markers $m = \{\hat{m}_z|_{z=1}^Z\}$ at t, given history $\mathcal{H}_t^u$ by computing $P(m_z|t, \mathcal{H}_t^u)$ (see Eq. 13). The predictions $\hat{c}$ and $\hat{m}_z$ are given by $\hat{m}_z = \arg\max_{\hat{m}_z \in M_z} P(m_z|t, \mathcal{H}_t^u)$. It is important to note that though our model technically only issues discrete output as it is inherently a classification model, while in practice the future events' timestamp can be predicted by an approximated discrete duration as done in our experiments. In this regard, we treat the future timestamp as a marker.

5 EMPIRICAL STUDY AND DISCUSSION

5.1 Dataset and protocol

To verify the potential of the proposed model, we apply it to a *LinkedIn-Career* dataset crawled and de-identified from LinkedIn to predict user's next company $\hat{c}$, next position $\hat{p}$ and duration $\hat{t}$ of current job; an ICU dataset extracted from public medical database MIMIC-II [8] to predict patient's transition to the next ICU department $\hat{c}$ and duration of stay $\hat{p}$ in current department. Experiments are conducted under Ubuntu 64bit 16.04LTS, with i7-5557U 3.10GHz×4 CPU and 8G RAM. For the convenience of replicating the experiments, the crawled *de-identified LinkedIn-Career* dataset and the code is available on Github[2].

Dataset The *LinkedIn-Career* Dataset contains 5,006 users crawled from *information technology* (IT) industry on LinkedIn[3], including their *Self-introduction*, *Technical skills* and *Working Experience* after de-identification preprocess. We collect samples in IT industry because: i) The staff turnover rate is high, which makes it easier to collect suitable samples; ii) The IT industry is most familiar to the authors, and our domain knowledge can help better curate the raw data. We extract profile features from users' *Self-introduction* and *Technical skills*, and get users' history company and position $\{(c_i, p_i)\}$ from *Working Experience*. After we exclude samples with zero job movement, we have a so-called *LinkedIn-Career* benchmark, involving 2,403 users, 57 IT companies, 10 kinds of positions and 4 kinds of durations. The dataset is to some extent representative for IT industry. For companies, we have large corporations like *Google*, *Facebook*, *Microsoft* and medium-sized enterprise like *Adobe*, *Hulu*, *VMWare*. For positions we have technical positions like *engineer*, *senior engineer*, *tech lead*, and management positions like *manager*, *director*, *CEO*. For durations we discretize

[2]https://github.com/blade091shenwei/factorial-marked-point-process
[3]https://www.linkedin.com/

the duration of stay in a position or company as *temporary*(within 1 year), *short-term*(1-2 years), *medium-term*(2-3 years) and *long-term*(more than 3 years). The goal is to predict user's next company $\hat{c}$ from $C = 57$ companies, next position $\hat{p}$ from $P = 10$ positions and duration of stay in current company and position $\hat{t}$ from $T = 4$ durations.

The *ICU* dataset contains 30,685 patients from MIMIC-II database, including patients' *diagnose*, *treatment record*, *transition* between different ICU departments and *duration* of stay in the departments. The goal is to predict patient's next ICU department $\hat{c}$ from $C = 8$ departments including Coronary care unit (CC), Anesthesia care unit (ACU), Fetal ICU (FICU), Cardiac surgery recovery unit (CSRU), Medical ICU (MICU), Trauma Surgical ICU (TSICU), Neonatal ICU (NICU), and General Ward (GW), and predict patient's duration of stay $\hat{t}$ from $T = 3$ kinds of duration including *temporary*(within 1 day), *short-term*(1-5 days), and *long-term* (more than 5 days). The profile features are extracted from patients' *diagnose* (ICD9 code of patients' disease) and *treatment record* (nursing, medication, treatment).

Many peer methods are evaluated as follows:

Intensity function choices Our framework is tested by four point process embodiments: i) Mutually-corrected Processes (**MCP**), ii) Hawkes Process (**HP**), iii) Self-correcting Process (**SCP**) and iv) Modulated Poisson Process (**MPP**). Their characters are briefly compared in Table 2. Note in our experiments, all these models are learned via the reformulated LR algorithm as described in Alg. 2.

Comparison to classic Logistic Regression We test a non-point process approach i.e. the plain LR. For the point process based LR solver, its input is

$$\boldsymbol{f}_{\boldsymbol{t}}^{\boldsymbol{u}} = [x_0^u h(t), \sum_{i:t_i^u<t} b_{z_i}^u g(t,t_i), \sum_{y=1,y\neq z}^{Z} \sum_{j:t_j^u} b_{y_j}^u g(t,t_j)],$$

while the plain LR involves the raw feature as

$$\boldsymbol{f}^{\boldsymbol{u}} = [x_0^u, b_{z_I}^u, \sum_{y=1,y\neq z}^{Z} b_{y_I}^u],$$

which includes user profile feature x_0^u, binary indicator $b_{z_I}^u$ and $\sum_{y=1,y\neq z}^{Z} b_{y_I}^u$ representing one's current state without considering the history states.

Comparison to RNN and RMTPP We also experiment on RNN by treating the prediction task as a sequence classification problem. A dynamic RNN that can compute over sequences with variable length is implemented. Moreover, to explore the effect of discretizing the time interval when making duration prediction, we also experiment on RMTPP (Recurrent Marked Temporal Point Process) proposed by [4]. Instead of predicting a discrete label for duration, it gives a continuous prediction result.

Prediction performance metrics We use prediction accuracy AC to evaluate the performance of the model with four variants AC_c, AC_p, AC_t, AC_{cpt} to denote prediction accuracy for state c (# correct ones out of total predictions), state p, state t and joint c, p, t respectively.

Dataset	Method	AC_c	AC_p	AC_t	AC_{cpt}	Time	Iter. #
Career	Alg.2	32.81	60.67	52.41	10.74	123.8m	147.7
	LR	33.58	60.13	53.96	10.96	46.8s	11.2
ICU	Alg.2	76.63	—	55.74	45.64	764.2m	121.5
	LR	76.98	—	55.63	45.55	55.1s	9.7

Table 3: Comparison of the raw ADMM solver (Alg.2) and the reformulated LR solver: prediction accuracy by percentage for AC_c, AC_p, AC_t, joint prediction accuracy AC_{cpt}, time cost and average iteration count by random initialization for 10 trials. Time and iteration number is the average result.

To evaluate the performance of our discrete duration prediction compared with RMTPP, both MSE (Mean Squared Error) and AC are computed. To compute prediction MSE, the predicted discrete duration is substituted by the intermediate time point of the discrete intervals, e.g., 0.5 years for *temporary* stay, 1.5 years for *short-term* stay and 4 years for *long-term* stay. To compute prediction AC, the predicted continuous duration of RMTPP is discretized using the same criterion by the proposed model.

For *LinkedIn-Career* data, we further compute a precision curve for the top-K position, company and duration predictions as shown in Fig. 3. These metrics are widely used for recommender systems. In fact, as our model is for predicting the next company $\hat{c}$, next position $\hat{p}$ and duration $\hat{t}$ given career history $\mathcal{H}_t^u$, it can be used for recommending companies and posts at the predicted time period $\hat{t}$.

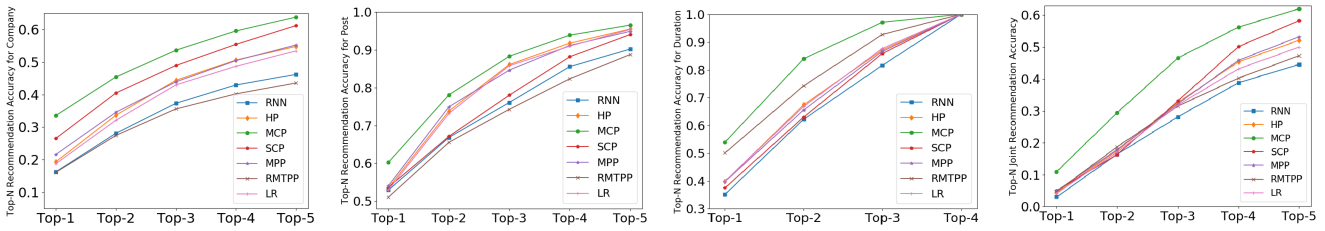
All the experimental results are obtained by commonly used 10-fold cross validation, like [24].

5.2 Results and discussion

We are particularly interested in analyzing the following main bullets via empirical studies and quantitative results.

i) LR solver vs. ADMM solver To make a fair comparison, the LR solver and ADMM solver i.e. Alg.2 are both initialized by a uniform distribution sampling from $(-1, 1)$, and the running time and iteration count are the average of 10-fold cross validation. Table 3 compares LR solver and ADMM solver regarding with accuracy and time cost on the Dataset *LinkedIn-Career* and *ICU*. One can find the prediction accuracy is similar while the ADMM solver is more costive as we find it converges more slowly as shown in Fig. 4. Also, as shown in Alg.2, it involves more hyper-parameters to tune and they have been tuned to their best performance. For comparison between the reformulated LR via the point process framework, and the raw LR using only user profile i.e. LR_{np}, we find the former outperforms in most cases in Table 4 and 5.

Comparing running time in Table 3, the LR solver has better scalability than ADMM solver. This is because the ADMM solver is a general algorithm for convex optimization with sparse group regularization, while the LR solver works by a special design for the objectives that can be reformulated to Logistic Regression loss. Many algorithmic optimizations



(a) Company (b) Position (c) Duration (d) Company & Position & Duration

Figure 3: Top-5 prediction accuracy on the collected *LinkedIn-Career* dataset out of 57 companies, 10 positions and 4 durations.

		company pred. accuracy $AC_c(\%)$			position pred. accuracy $AC_p(\%)$			duration pred. accuracy $AC_t(\%)$			joint pred. accuracy $AC_{cpt}(\%)$		
sequence	intensity	decoupled	coupled	uni-com	decoupled	coupled	uni-pos	decoupled	coupled	uni-dur	decoupled	coupled	uni-cpt
short	HP	15.37	14.23	14.50	56.99	56.99	56.99	38.97	36.61	36.61	4.59	4.05	4.26
	SCP	17.99	10.01	12.61	58.98	49.75	55.38	43.56	41.81	38.22	4.96	2.69	2.11
	MPP	15.58	13.58	13.96	56.99	56.99	56.99	40.16	36.61	36.61	4.71	3.72	4.21
	MCP	**18.54**	11.04	13.14	**59.42**	51.49	56.90	**46.49**	46.46	41.32	**5.41**	2.52	3.26
	LR_{np}	—	15.35	—	—	56.99	—	—	38.31	—	—	4.46	—
	RNN	—	13.18	—	—	56.98	—	—	36.94	—	—	3.77	—
	RMTPP	—	12.02	—	—	56.03	—	—	44.95	—	—	4.15	—
long	HP	24.45	20.57	25.61	50.80	49.25	49.51	34.52	33.61	34.33	4.36	3.53	4.32
	SCP	35.41	20.49	26.46	51.00	40.54	46.96	33.27	28.99	28.64	7.47	5.46	5.84
	MPP	28.71	21.44	29.57	51.64	49.02	52.90	36.04	33.20	33.76	6.37	4.17	6.49
	MCP	**50.23**	29.98	44.12	**60.45**	50.16	55.65	**47.00**	40.33	37.78	**14.61**	7.21	10.05
	LR_{np}	—	24.59	—	—	49.51	—	—	35.47	—	—	4.14	—
	RNN	—	19.92	—	—	49.24	—	—	33.80	—	—	3.26	—
	RMTPP	—	19.88	—	—	49.17	—	—	44.07	—	—	4.58	—
all	HP	19.33	16.23	19.15	53.50	52.95	52.95	39.85	37.72	37.80	4.39	3.20	4.05
	SCP	26.52	15.54	20.77	53.68	46.88	51.90	37.56	32.30	29.45	5.06	3.32	3.70
	MPP	21.59	16.49	21.73	54.10	52.77	54.49	39.81	35.17	35.72	4.96	3.50	4.81
	MCP	**33.58**	20.30	27.80	**60.13**	52.48	57.56	**53.96**	48.34	45.16	**10.96**	5.44	6.96
	LR_{np}	—	18.74	—	—	52.98	—	—	40.01	—	—	4.21	—
	RNN	—	16.24	—	—	52.93	—	—	35.21	—	—	3.02	—
	RMTPP	—	16.10	—	—	51.07	—	—	50.16	—	—	4.70	—

Table 4: Accuracy comparison for different intensity functions on *LinkedIn-Career* (HP, SCP, MPP, MCP, see Table 2). Numbers in bold denote the best or second-best accuracy on the specified metric and dataset. Learning for all point process based models is via the reformulated LR solver as discussed in the main paper. Long sequence denotes those with more than 2 job transitions. For the non-point process based (classic) LR_{np}, we present its performance for each prediction target.

		duration prediction accuracy $AC_t(\%)$			transition prediction accuracy $AC_c(\%)$			joint prediction accuracy $AC_{ct}(\%)$		
sequence	intensity	decoupled	coupled	uni-duration	decoupled	coupled	uni-transition	decoupled	coupled	uni-dt
all	HP	52.48	51.64	52.91	74.61	73.31	73.77	42.85	41.62	42.40
	SCP	50.14	49.77	49.05	74.22	74.01	70.75	41.04	40.77	39.48
	MPP	53.27	51.88	52.02	74.74	73.42	72.05	43.64	42.37	43.28
	MCP	**55.63**	54.62	50.14	**76.98**	76.58	74.02	**45.55**	45.32	44.89
	LR_{np}	—	39.88	—	—	69.61	—	—	31.13	—
	RNN	—	47.01	—	—	70.54	—	—	36.44	—
	RMTPP	—	54.28	—	—	67.49	—	—	41.93	—

Table 5: Accuracy comparison for different intensity function models on ICU dataset from MIMIC-II.

for Logistic Regression can be used in LR solver, like Efficient Projection in SLEP [14].

ii) Decoupled learning vs. RNN As shown in Table 4 and Table 5, the decoupled marked point process model outperforms RNN. This is because the next-event prediction task for relatively short sequences like dataset *LinkedIn-Career* and *ICU* is not a typical sequence classification task. We need to make prediction on every step of the sequence, rather than make prediction at the end of the whole sequence. That means for the end-to-end RNN sequence classification

model, it needs to deal with sequences with considerably variable length, including a large number of sequences of length 1.

We also compare the accuracy of RMTPP that makes continuous duration prediction, with decoupled model and general RNN in Table 4 and Table 5. Though the duration prediction accuracy of RMTPP is improved compared to general RNN, the decoupled model still have better performance.

To further verify the effect of discretizing the time interval when making duration prediction, the MSE of decoupled-MCP and RMTPP is also compared in Table 6, which shows that the discretization of time interval is to some extent rational.

iii) Infectivity matrix decoupling vs. coupling Table 4 and Table 5 also compare the performance of our decoupled model (see Eq. 2) against the raw coupled model (see Eq. 1), and the simplified model (single-dimension) when only marker c, p or t is considered. This boils down to the single-dimension case and the method is termed by uni-com, uni-pos and uni-dur for dataset *LinkedIn-Career*, and uni-duration and uni-transition for *ICU* respectively. While for uni-cpt, it involves no new model while uses the output of uni-c, uni-p and uni-t to combine them together as the joint prediction. It shows that the decoupled model consistently achieves the best performance, which perhaps is attributed to the reduction of model complexity given relatively limited training data.

Comparing the accuracies in Table 4 for dataset *LinkedIn-Career* and Table 5 for dataset *ICU*, we can see that the improvement in accuracy of decoupled model compared to coupled model or single-dimension model, is more remarkable for *LinkedIn-Career* than that for *ICU*. The reason is that for *LinkedIn-Career*, the coupled state space is decoupled from $C \times P \times T = 2280$ to $C + P + T = 71$, and for *ICU* it is decoupled from $C \times T = 24$ to $C + T = 11$. The *LinkedIn-Career* dataset has a larger coupled state space than *ICU*. So when decoupled to smaller state spaces, the improvement for *LinkedIn-Career* is more notable than that for *ICU*.

iv) Choice of intensity function There are many popular intensity forms and some are listed in Table 2. According to Table 4 and Table 5, the mutually-correcting process (MCP) consistently shows superior performance against other intensity function embodiments. This verifies two simple assumptions that i) the intensity tends to decrease for the moment the event happens, i.e., the desire for new job can be suppressed when a new job is fulfilled for job prediction, and patients' demand for transition to next ICU department decrease after they move into a new department for ICU department transition prediction; ii) the probability of future events is influenced by the history events according to Table 2, i.e., one's transition possibility to new job is related to his/her history career experience, and patient's future ICU department transition procedure is related to his/her history treatment.

v) Influence of sequence length To further explore the performance behavior, we experiment on short-sequences and long-sequences respectively on *LinkedIn-Career*[4]. Results in Table 4 show that the decoupled MCP algorithm has more advantage in long-sequence prediction, suggesting that the decoupled MCP can make better use of history information.

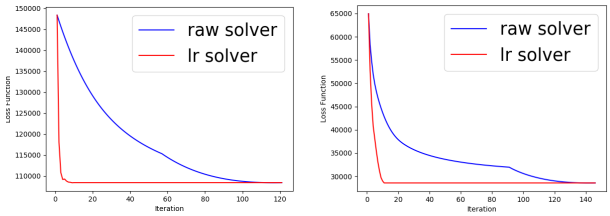


Figure 4: Convergence curve of reformulated LR solver & ADMM solver (Alg.2) by similar random initialization. Left: *ICU* dataset from MIMIC-II; Right: *LinkedIn-Career*.

dataset	LinkedIn		ICU	
model	RMTPP	de-MCP	RMTPP	de-MCP
MSE	9.625	2.934	14.602	4.272

Table 6: MSE comparison of future event duration prediction for RMTPP and decoupled MCP on LinkedIn (in year) and ICU (in day) dataset. Note RMTPP model predicts continuous timestamp value for future events.

dataset	marker	w/o sparse	group lasso	sparse group
Career	company	29.16	31.47	**33.58**
	position	56.99	58.16	**60.13**
	duration	50.56	52.33	**53.96**
	joint (3)	9.53	10.04	**10.96**
ICU	duration	52.68	55.13	**55.63**
	transition	73.45	76.09	**76.98**
	joint (2)	42.20	45.49	**45.55**

Table 7: Accuracy(%) by different regularizers. Sparse group (Eq. 6) combines ℓ_1 regularizer and group lasso.

vi) Influence of sparsity To verify the effect of sparse group regularization, we compare the accuracy of the decoupled MCP model with different regularization settings, including **without sparse regularization**, with **group lasso** for group sparsity and with **sparse group** regularization (a combination of ℓ_1 regularization and group lasso – see Eq. 6) for both overall sparsity and group sparsity. As shown in Table 7, the sparse group regularizer outperforms.

We also explore feature selection functionality by investigating the magnitudes of elements in matrix $\mathbf{\Theta}$. The element Θ_{ij} measures the influence of profile feature or marker j to label i. Small (large) values indicate the corresponding

[4]ICU data is not included in the length test because 96% of the patients in ICU have no more than three transitions.

features or markers have little (high) influence to label. For example, in *LinkedIn-Career* dataset, the numerical values in coefficient column vector corresponding to marker *engineer* are all nonzero, showing that having a working experience as an *engineer* is important in IT industry. For marker *director*, most of the elements in the corresponding coefficient column vector is zero except for the rows of positions CEO and $founder$, suggesting an ascending career path in general.

6 CONCLUSION

We study the problem of factorial point process learning for which the event can carry multiple markers whereby the relevant concept can be found in Factorial Hidden Markov Models [7]. Two learning algorithms are presented: the first is directly based on the raw regularized discriminative prediction objective function which employs ADMM and FISTA techniques for optimization; the second is a simple LR solver which is based on a key reformulation of the raw objective function. Experimental results on two real-world datasets collaborate the effectiveness of our approach.

ACKNOWLEDGEMENT

This work is partially supported by NSFC (61602176), NSFC-Zhejiang Joint Fund for the Integration of Industrialization and Informatization (U1609220), The National Key Research and Development Program of China (2016YFB1001003), Partnership Collaboration Awards by The University of Sydney and Shanghai Jiao Tong University (WF610561702).

REFERENCES

[1] Amir Beck and Marc Teboulle. 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences* 2, 1 (2009), 183–202.
[2] D. Daley and D. Vere-Jones. 2007. *An introduction to the theory of point processes: volume II: general theory and structure*. Springer Science & Business Media.
[3] David L Donoho. 1995. De-noising by soft-thresholding. *IEEE Transactions on Information Theory* 41, 3 (1995), 613–627.
[4] Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. 2016. Recurrent marked temporal point processes: Embedding event history to vector. In *SIGKDD*. ACM.
[5] Nan Du, Yichen Wang, Niao He, Jimeng Sun, and Le Song. 2015. Time-sensitive recommendation from recurrent user activities. In *NIPS*.
[6] Alan E Gelfand, Peter Diggle, Peter Guttorp, and Montserrat Fuentes. 2010. *Handbook of spatial statistics*. CRC press.
[7] Zoubin Ghahramani and Michael I Jordan. 1996. Factorial hidden Markov models. In *Advances in Neural Information Processing Systems*. 472–478.
[8] Ary L. Goldberger, Luis A.N. Amaral, Leon Glass, Jeffrey M. Hausdorff, Plamen Ch. Ivanov, Roger G. Mark, Joseph E. Mietus, George B. Moody, Chung-Kang Peng, and H. Eugene Stanley. 2000. PhysioBank, PhysioToolkit, and PhysioNet: Component of a New Research Resource for Complex Physiologic Signals. *Circulation* 101, 23 (2000), e215–e220.
[9] Alan G Hawkes. 1971. Spectra of some self-exciting and mutually exciting point processes. *Biometrika* 58, 1 (1971), 83–90.
[10] Valerie Isham and Mark Westcott. 1979. A self-correcting point process. *Stochastic Processes and Their Applications* 8, 3 (1979), 335–347.
[11] Erik Lewis and George Mohler. 2011. A nonparametric EM algorithm for multiscale Hawkes processes. *Journal of Nonparametric Statistics* 1, 1 (2011), 1–20.
[12] Liangda Li and Hongyuan Zha. 2014. Learning Parametric Models for Social Infectivity in Multi-Dimensional Hawkes Processes. In *AAAI*. 101–107.
[13] Thomas Josef Liniger. 2009. *Multivariate hawkes processes*. Ph.D. Dissertation. ETH Zurich.
[14] Jun Liu, Shuiwang Ji, and Jieping Ye. 2009. *SLEP: Sparse Learning with Efficient Projections*. Arizona State University. http://www.public.asu.edu/~jye02/Software/SLEP
[15] Chris Lloyd, Tom Gunter, Michael Osborne, and Stephen Roberts. 2015. Variational inference for Gaussian process modulated Poisson processes. In *International Conference on Machine Learning*. 1814–1822.
[16] Lukas Meier, Sara Van De Geer, and Peter Bühlmann. 2008. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 70, 1 (2008), 53–71.
[17] Andrew Y Ng. 2004. Feature selection, L 1 vs. L 2 regularization, and rotational invariance. In *ICML*.
[18] Yosihiko Ogata. 1988. Statistical models for earthquake occurrences and residual analysis for point processes. *Journal of the American Statistical association* 83, 401 (1988), 9–27.
[19] Yosihiko Ogata. 1998. Space-time point-process models for earthquake occurrences. *Annals of the Institute of Statistical Mathematics* 50, 2 (1998), 379–402.
[20] Tohru Ozaki. 1979. Maximum likelihood estimation of Hawkes' self-exciting point processes. *Annals of the Institute of Statistical Mathematics* 31, 1 (1979), 145–155.
[21] Noah Simon, Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2013. A sparse-group lasso. *Journal of Computational and Graphical Statistics* 22, 2 (2013), 231–245.
[22] Shuai Xiao, Junchi Yan, Xiaokang Yang, Hongyuan Zha, and Stephen M Chu. 2017. Modeling the Intensity Function of Point Process Via Recurrent Neural Networks.. In *AAAI*. 1597–1603.
[23] Hongteng Xu, Weichang Wu, Shamim Nemati, and Hongyuan Zha. 2017. Patient flow prediction via discriminative learning of mutually-correcting processes. *IEEE Transactions on Knowledge and Data Engineering* 29, 1 (2017), 157–171.
[24] Junchi Yan, Yu Wang, Ke Zhou, Jin Huang, Chunhua Tian, Hongyuan Zha, and Weishan Dong. 2013. Towards Effective Prioritizing Water Pipe Replacement and Rehabilitation.. In *IJCAI*. 2931–2937.
[25] Ke Zhou, Hongyuan Zha, and Le Song. 2013. Learning social infectivity in sparse low-rank networks using multi-dimensional hawkes processes. In *AISTATS*.
[26] Ke Zhou, Hongyuan Zha, and Le Song. 2013. Learning triggering kernels for multi-dimensional hawkes processes. In *ICML*.