Deep Mixture Point Processes: Spatio-temporal Event Prediction with Rich Contextual Information

Maya Okawa[1], Tomoharu Iwata[2], Takeshi Kurashima[1], Yusuke Tanaka[1], Hiroyuki Toda[1] and Naonori Ueda[2]
[1]NTT Service Evolution Labs, NTT Corporation, Kanagawa, Japan
{maya.ookawa.af, takeshi.kurashima.uf, yusuke.tanaka.rh, hiroyuki.toda.xb}@hco.ntt.co.jp
[2]NTT Communication Science Labs, NTT Corporation, Kyoto, Japan
{tomoharu.iwata.gy, naonori.ueda.fr}@hco.ntt.co.jp

ABSTRACT

Predicting when and where events will occur in cities, like taxi pick-ups, crimes, and vehicle collisions, is a challenging and important problem with many applications in fields such as urban planning, transportation optimization and location-based marketing. Though many point processes have been proposed to model events in a continuous spatio-temporal space, none of them allow for the consideration of the rich contextual factors that affect event occurrence, such as weather, social activities, geographical characteristics, and traffic. In this paper, we propose DMPP (Deep Mixture Point Processes), a point process model for predicting spatio-temporal events with the use of rich contextual information; a key advance is its incorporation of the heterogeneous and high-dimensional context available in image and text data. Specifically, we design the intensity of our point process model as a mixture of kernels, where the mixture weights are modeled by a deep neural network. This formulation allows us to automatically learn the complex nonlinear effects of the contextual factors on event occurrence. At the same time, this formulation makes analytical integration over the intensity, which is required for point process estimation, tractable. We use real-world data sets from different domains to demonstrate that DMPP has better predictive performance than existing methods.

ACM Reference Format:
Maya Okawa[1], Tomoharu Iwata[2], Takeshi Kurashima[1], Yusuke Tanaka[1], Hiroyuki Toda[1] and Naonori Ueda[2]. 2019. Deep Mixture Point Processes: Spatio-temporal Event Prediction with Rich Contextual Information. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, Anchorage, Alaska, USA, 11 pages. https://doi.org/10.1145/3292500.3330937

1 INTRODUCTION

In cities, large volumes of event data are being generated by human activities. Such event data includes information about time and geolocation, indicating where and when each event occurred. For instance, taxi pick-up records are represented as a list of events consisting of the pick-up locations and the departure times. Crimes are recorded together with the time and location at which the crime took place. Predicting events is a key component of applications in many fields such as urban planning, transportation optimization and location-based marketing. If taxi dispatch service operators can estimate with high accuracy the future taxi pick-up times and locations, they can allocate taxis to the right places and the right times in advance. Criminal incident prediction will help law enforcement agencies to implement effective police activities that can suppress criminality.

Predicting spatio-temporal events, however, is extremely challenging, because event occurrence is determined by various contextual factors. Such contextual features also include geographical characteristics, e.g., transportation networks [11] and land use [3]; temporal attributes, e.g., day of week and weather conditions [4, 39]; and other features, e.g., social and traffic information [4, 35, 40]. This demands a framework that can capture the complex dynamics of event occurrence given the contextual features present. The conventional approach to this problem is based on regression models [16, 31, 40]. They are intended to model the aggregated number of events within a predefined spatial region and time interval, which is fundamentally different from our task. We focus more on the point process approach to model a sequence of events in continuous time and space, without aggregation, by using explicit information about location and/or time; and predicting the precise time and location at which each event will occur.

Point process is a sophisticated framework for modeling a sequence of events in continuous time and space; it directly estimates an *intensity* function that describes the rate of events occurring at any location and any time. The influence of the contextual features can be modeled by special point process models [6, 12, 14], where the intensity function is described as a function of covariates, i.e., the contextual features. However, this approach has a fundamental limitation. In many practical cases, their assumptions on the functional form of covariates may be too restrictive to capture complex and intricate effects of contextual features; they do not accommodate unstructured data such as images and texts. Most contextual features take the form of unstructured representations. For example, information about geographical characteristics can be obtained from map images. Traffic and social event information can be expressed in the form of natural language expressions.

In this paper, we propose an event prediction method that effectively incorporates such unstructured data into the point process model. Motivated by the recent success of the deep learning approach, we use it to enhance the point process model. The naive approach is to directly model the intensity by a deep neural network.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
KDD '19, August 4–8, 2019, Anchorage, AK, USA
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330937

Unfortunately, this approach triggers the intractable optimization problem as integral computations are required to determine the likelihood needed for estimation.

We address this through a novel formulation of spatio-temporal point processes. Specifically, we design the intensity as a deep mixture of experts, whose mixture weights are modeled by a deep neural network. This method, called DMPP (Deep Mixture Point Processes), enables us to incorporate unstructured contextual features (e.g., road networks and social/traffic event descriptions) into the predictive model, and to automatically learn their complex effects on event occurrence. Moreover, this formulation yields a tractable optimization problem. Our mixture model-based approach permits the likelihood to be determined from tractable integration. Learning can be done with simple back-propagation.

We conduct experiments on three real-world data sets from multiple urban domains and show that our DMPP consistently outperforms existing methods in event prediction tasks. The experiments also demonstrate that DMPP provides useful insights about why and under which circumstances events occur. By utilizing a recently developed self-attention mechanism [24, 25], DMPP helps us better understand how the contextual features influence event occurrence. Such insights could further aid policy makers in creating more effective strategies. The main contributions of this paper are as follows:

- We propose DMPP, a novel method for spatio-temporal event prediction. It accurately and effectively predicts spatio-temporal events by leveraging the contextual features, such as map images and social/traffic event descriptions, that impact event occurrence.
- We integrate the deep learning approach into the point process framework. Specifically, we extract the intensity by using a deep mixture of experts, whose mixture weights are modeled by a deep neural network. This formulation allows us to utilize the information present in unstructured contextual features, and to automatically discover their complex effects on event occurrence, while at the same time yielding tractable optimization.
- We develop an efficient estimation procedure for training and evaluating DMPP.
- We conduct extensive experiments on real-world data sets from three urban domains. With regard to event occurrence, the proposed method achieves better predictive performance than all existing methods on all data sets (Section 6).

2 RELATED WORK

Point process is a general mathematical framework for modeling a sequence of events; it directly estimates the rate of event occurrence, by using explicit information about location and/or time. Early work mainly focused on the temporal aspect of events. The temporal Hawkes processes [13] are a class of temporal point process models that can capture burst phenomena; in these models, the probability of future events is assumed to be strengthened by past events, with the influence decaying exponentially over time. They have been used for analysing disease transmissions [5], financial transactions [1, 2], terrorist attacks [28], social activities [10, 17], search behaviors [23], and so on. Recent studies have expanded its application to human mobility modeling. Wang *et al.* (2017) proposed Hawkes process variant to identify trip purpose. Du *et al.* (2016) presented a recurrent marked temporal point process (RMTPP) and demonstrated its effectiveness in predicting the timing of taxi pick-ups. Log Gaussian Cox process (LGCP) has been used to effectively model temporal events, such as wildfires [30] and infrastructure failures [9], in which the logarithm of the intensity is assumed to be drawn from a Gaussian process. The spatio-temporal point process is a more general framework, and considers both spatial and temporal domains. The spatio-temporal self-exciting point processes, an extension of temporal Hawkes processes, have been used for modeling seismicity [27], contagious diseases [29], and crime incidents [26], among other applications The spatio-temporal LGCP has been applied to model wildfires [30] and infrastructure failures [9].

All these methods, however, have one fundamental limitation: they ignore contextual features even though they are known to influence event occurrence. Human activities are largely influenced by environmental features, i.e., weather, geographical characteristics and traffic conditions. These features must be considered to accurately predict future events. Their influence has been modeled by a special point process model, called the proportional hazards model [6]; it treats the intensity rate as a function of covariates. One major limitation of this model is that it assumes that the contextual features create only linear effects. Most features have highly non-linear effects on real world event occurrence. The simplest solution is to fit non-linear functions, such as polynomials [14] and splines [12], to covariates. Unfortunately, their assumptions may be too restrictive to capture complex and intricate effects of contextual features. Also, this approach forces us to carefully choose or design the functional form of the covariates so that they accurately capture reality. However, in practice, how the contextual features influence event occurrence is largely unknown.

This paper constructs a novel point process method called DMPP; it extends the spatio-temporal point process with a deep learning model. The pioneering work by [8, 37, 38] is most related to our approach. However, they focus only on the temporal dynamics of event occurrence, and so ignore spatial dynamics. Also, those methods are optimized to predict the timing of the next event. Instead, we are interested in predicting longer event sequences. Moreover, none of these methods accept contextual features.

3 PRELIMINARIES

In this section, before introducing our method, we first provide the necessary theoretical background to the point process.

Point process is a random sequence of event occurrences over a domain. We assume here a sequence of events with known times and locations. Let $\mathbf{x} = (t, s)$ be the event written as the pair of time $t \in \mathbb{T}$ and location $s \in \mathbb{S}$, where $\mathbb{T} \times \mathbb{S}$ is a subset of $\mathbb{R} \times \mathbb{R}^2$. In the following, we denote the number of events falling in subset A of $\mathbb{T} \times \mathbb{S}$ as $N(A)$. The general approach to identifying a point process is to estimate the "intensity" $\lambda(\mathbf{x})$. The intensity $\lambda(\mathbf{x})$ represents the rate of event occurrence in a small region, and is defined as

$$\lambda(\mathbf{x}) = \lambda(t, s) \equiv \lim_{|dt| \to 0, |ds| \to 0} \frac{\mathbb{E}[N(dt \times ds)]}{|dt||ds|}, \tag{1}$$

where dt is a small interval around time t, $|dt|$ is its duration, ds is a small region containing location s, and $|ds|$ is its area. $\mathbb{E}$ indicates an expectation measure. The functional form of intensity is designed to appropriately capture the underlying dynamics of event occurrence.

Given a sequence of events $\mathcal{X} = \{\mathbf{x}_i = (t_i, \mathbf{s}_i)\}_{i=1}^N$, $t_i \in \mathbb{T}$ and $\mathbf{s}_i \in \mathbb{S}$, the likelihood is given by

$$p(\mathcal{X}|\lambda(\mathbf{x})) = \prod_{i=1}^{N} \lambda(\mathbf{x}_i) \cdot \exp\left(- \int_{\mathbb{T}\times\mathbb{S}} \lambda(\mathbf{x}) d\mathbf{x} \right). \quad (2)$$

4 DEEP MIXTURE POINT PROCESSES

This section presents the proposed method referred to as DMPP (Deep Mixture Point Processes). We first introduce the notations and definitions used in this paper. We then provide the model formulation of DMPP followed by parameter learning and prediction. The neural network architecture used in DMPP is detailed in the Appendix.

4.1 Problem Definition

We introduce the notations used in this paper and formally define the problem of event prediction.

Let $\mathcal{X} = \{\mathbf{x}_i = (t_i, \mathbf{s}_i)\}_{i=1}^N$ denote a sequence of events over space and time, where $(t_i, \mathbf{s}_i) \in \mathbb{T} \times \mathbb{S} \in \mathbb{R} \times \mathbb{R}^2$ and N is the total number of events known.

Further, we are also given contextual information associated with the spatio-temporal region $\mathbb{T} \times \mathbb{S}$. Let $\mathcal{D} = A_1, A_2, ..., A_K$ be a set of contextual features, where A_k is the k-th feature, and K is the number of contextual features. Examples of the contextual features include weather, social/traffic event information and geographical characteristics. The social/traffic event information may be a collection of social/traffic event descriptions that include locations and times. In this case, $A_.$ is represented by a set of four-element tuples, each of which has the following format: `<time, latitude, longitude, description>`. Information about the geographical characteristics can be obtained from map images.

Given the contextual features $\mathcal{D}$ up to time $T + \Delta T$, and the event sequence $\mathcal{X}$ up to time T, we aim to learn a predictor that:

- predicts times and locations of events in the future time window $[T, T + \Delta T]$;
- predicts the number of events within any given spatial region and the time period in $[T, T + \Delta T]$,

by leveraging $\mathcal{D}$ and $\mathcal{X}$.

4.2 Model Formulation

In this work, we construct a novel point process method for spatio-temporal event prediction that can incorporate unstructured contextual features such as map images and social/traffic event descriptions. Our point process intensity must be designed so that it is flexible enough to capture the highly complex effects of contextual features, while at the same time being tractable. Deep learning models have proven to be an extremely useful, especially in automatically extracting the meaningful information contained in the unstructured data including images and text descriptions. Inspired by this, we propose a novel formulation of point process model by integrating it with deep learning approach. The proposed method is referred to as DMPP (Deep Mixture Point Processes). In particular, we model the intensity by a neural network function that accepts contextual features as its input.

Intensity function. We develop a flexible and computationally effective way of using kernel convolution to specify the intensity function. Formally, we design the intensity as a function of contextual features:

$$\lambda(\mathbf{x}|\mathcal{D}) = \int f\Big(\mathbf{u}, \mathbf{Z}(\mathbf{u}; \mathcal{D}); \theta\Big) k(\mathbf{x}, \mathbf{u}) d\mathbf{u}, \quad (3)$$

where $\mathbf{u} = (\tau, \mathbf{r})$ for $\tau \in \mathbb{T}$ and $\mathbf{r} \in \mathbb{S}$, $k(\cdot, \mathbf{u})$ is a kernel function centered at $\mathbf{u}$. $f(\cdot)$ is any deep learning model that returns a nonnegative scalar, and θ denotes a set of the parameters of the deep neural network. $\mathbf{Z}(\mathbf{u}, \mathcal{D}) = \{Z_1(\mathbf{u}; A_1), ..., Z_K(\mathbf{u}; A_K)\}$ is a set of the feature values at the spatio-temporal point $\mathbf{u}$, where Z_k is defined as the operator to extract values of k-th feature at $\mathbf{u}$. As one example, social/traffic event descriptions can be represented by tuples of time, location and event descriptions. In this case, operator Z outputs a list of social/traffic event descriptions scheduled within $[\tau - \Delta\tau, \tau + \Delta\tau]$ and located within a predefined distance, $||\mathbf{r} - \mathbf{r}'|| < \Delta\mathbf{r}$, given $\mathbf{u} = (\tau, \mathbf{r})$. As another example, given a map image representing geographical characteristics, Z returns the feature vectors (e.g., RGB values) of the map image around $\mathbf{r}$. The formulation of Equation (10) is built upon a process convolution approach [15, 20, 21]; but we extend it so that the point process intensity accepts unstructured contextual features, by integrating it with a deep neural network. This extension enables us to integrate unstructured contextual data, and automatically learn their complex effects on event occurrence. Although being flexible and expressive, this intensity is intractable as it involves the integral of the neural network function $f(\cdot)$. Thus, by introducing J representative points $\mathcal{U} = \{\mathbf{u}_j\}_{j=1}^J$ in the spatio-temporal region, we obtain a discrete approximation to Equation (10):

$$\lambda(\mathbf{x}|\mathcal{D}) = \sum_{j=1}^{J} f\Big(\mathbf{u}_j, \mathbf{z}_j; \theta\Big) k(\mathbf{x}, \mathbf{u}_j), \quad (4)$$

where each point $\mathbf{u}_j = (\tau_j, \mathbf{r}_j)$ consists of its time $\tau_j \in \mathbb{T}$ and location $\mathbf{r}_j \in \mathbb{S}$. Here we define $\mathbf{Z}(\mathbf{u}_j; \mathcal{D})$ as $\mathbf{z}_j$, which represents the contextual feature vector associated with the j-th point $\mathbf{u}_j$. Consequently, the intensity is described as a mixture of kernel experts, in which mixture weights are modeled by a deep neural network whose inputs are contextual features. The resulting model yields the automatic learning of their influences as well as making the learning problem tractable (discussed in Section 4.3).

Configuration of representative points. The set of representative points is structured as follows. We first introduce M discrete points placed uniformly along time axis within $[0, T + \Delta T]$ to define time points $\mathcal{T}$: $0 = \tau'_1 < ... < \tau'_M = T + \Delta T$. Similarly, we set L discrete points within the spatial region to define space points $\mathcal{S}$: $\mathbf{r}'_1, ..., \mathbf{r}'_L$, where $\mathbf{r}'_l \in \mathbb{S}$. The set of representative points is defined by the Cartesian product of the space and time points: $\mathcal{U} = \{(\tau, \mathbf{r}) \mid \tau \in \mathcal{T} \wedge \mathbf{r} \in \mathcal{S}\}$. Therefore $J = ML$. There are some options in locating the representative points, either fixing them or optimizing them in terms of spatial coordinates. In this paper, we choose the former and fix them on a regular grid, as simplifies the computation. Note that the number of representative points, J, determines the trade-off between approximation accuracy and computation complexity. Larger J improves approximation, while

reducing computational cost. A sensitivity analysis of the impact of J is given in the experimental section.

Kernel function. We can make various assumptions as to the kernel function $k(\mathbf{x}, \mathbf{u}_j)$. For example, we can use a Gaussian kernel:

$$k(\mathbf{x}, \mathbf{u}_j) = \exp\Big(- (\mathbf{x} - \mathbf{u}_j)^\top \Sigma^{-1} (\mathbf{x} - \mathbf{u}_j)\Big), \tag{5}$$

where Σ is a 3×3 covariance matrix (bandwidth) of the kernel. Other kernel functions, such as Matern, sigmoid, periodic (trigonometric), and compactly supported kernels [36] are viable alternatives.

Neural network model. The neural network model $f(\cdot)$ can be designed to suit the input data. We consider the general case wherein image features (e.g., map images) and text features (e.g., social/traffic event descriptions) are available. We propose an attention network architecture that fully exploits visual and textual information. The proposed architecture consists of three components: *an image attention network* to extract image features, *a text attention network* to encode text features, and *a multimodal fusion module*. We construct *the image attention network* by combining a CNN with the spatial attention model proposed by [25]. We design *the text attention network* on a CNN designed for sentences [18] and an attention mechanism [24]. The structured texts are split into words and then processed by the text attention network. Lastly, the extracted features from images and texts are fused into a single representation via *the multimodal fusion module*, and input to the intensity function. We detail each component of the neural network in Appendix A.

4.3 Parameter Learning

Given a list of observed events up to time T (total of N events) $\mathcal{X}$, the logarithm of the likelihood function is written as

$$\begin{aligned}\log p(\mathcal{X}|\lambda(\mathbf{x})) &= \sum_{i=1}^{N} \log \lambda(\mathbf{x}_i|\mathcal{D}) - \int_{\mathbb{T}\times\mathbb{S}} \lambda(\mathbf{x}|\mathcal{D}) d\mathbf{x} \\ &= \sum_{i=1}^{N} \log \sum_{j=1}^{J} f(\mathbf{u}_j, \mathbf{z}_j; \theta) k(\mathbf{x}_i, \mathbf{u}_j) \\ &\quad - \sum_{j=1}^{J} f(\mathbf{u}_j, \mathbf{z}_j; \theta) \int_{\mathbb{T}\times\mathbb{S}} k(\mathbf{x}, \mathbf{u}_j) d\mathbf{x}, \end{aligned} \tag{6}$$

where $\mathbb{T}\times\mathbb{S}$ is the domain of the observation. Notably, the above log-likelihood can be solved tractably with integratable kernel functions. Our mixture model-based approach with representative points allows the neural network model $f(\cdot)$, which cannot be integrated analytically in general, to be moved outside the integral. This permits us to use the simple back-propagation algorithm. For many well-known kernel functions, such as Gaussian, polynomial, the integral of the second term is written as closed-form solutions or approximations. In the case of the Gaussian kernel, it is described by an error function. During the training phase, we adopt mini-batch optimization. Over the set of indices selected in a mini-batch $\mathcal{I}$, by normalizing the first term in Equation (6), the objective function can be written as

$$\begin{aligned}\log p(\mathcal{X}|\lambda(\mathbf{x})) &= \frac{N}{|\mathcal{I}|} \sum_{i\in\mathcal{I}} \log \sum_{j=1}^{J} f(\mathbf{u}_j, \mathbf{z}_j; \theta) k(\mathbf{x}_i, \mathbf{u}_j) \\ &\quad - \sum_{j=1}^{J} f(\mathbf{u}_j, \mathbf{z}_j; \theta) \int_{\mathbb{T}\times\mathbb{S}} k(\mathbf{x}, \mathbf{u}_j) d\mathbf{x}, \end{aligned} \tag{7}$$

where $|\mathcal{I}|$ denotes the mini-batch size. We apply back-propagation to find all the model parameters, $\Theta = \{\Sigma, \theta\}$, that maximize the above log-likelihood, by taking the derivative of Equation (7) w.r.t. kernel parameter Σ and neural network parameters θ.

4.4 Prediction

Here we present a procedure for future event prediction.

We denote representative points within the test period $\mathbb{T}^* = (T, T + \Delta T]$ as $\mathcal{U}^* = \{(\tau, \mathbf{r}) \mid T < \tau \leq T + \Delta T\} \subset \mathcal{U}$. Given the learned parameters of the neural network, $\hat{\theta}$, we first calculate $f(\mathbf{u}_j, \mathbf{z}_j; \hat{\theta})$ for each representative point. Using the set of estimated functions $\{f(\mathbf{u}_j, \mathbf{z}_j; \hat{\theta})\}_{\mathbf{u}_j \in \mathcal{U}^*}$ and the estimated kernel parameter $\hat{\Sigma}$, we derive intensity $\hat{\lambda}(\mathbf{x})$ for the test period based on Equation (4).

Given the sequence of events observed in the test period $\mathbb{T}^*$, $\mathcal{D} = \{\mathbf{x}_{N+1}, ..., \mathbf{x}_{N+n}\}$, analogous to Equation (6), the log-likelihood for the test data is calculated as

$$\begin{aligned}\mathcal{L}^* = \log p(\mathcal{D}|\hat{\lambda}(\mathbf{x})) &= \sum_{i=N+1}^{N+n} \log \sum_{\mathbf{u}_j \in \mathcal{U}^*} f\Big(\mathbf{u}_j, \mathbf{z}_j; \hat{\theta}\Big) k(\mathbf{x}_i, \mathbf{u}_j) \\ &\quad - \sum_{\mathbf{u}_j \in \mathcal{U}^*} f\Big(\mathbf{u}_j, \mathbf{z}_j; \hat{\theta}\Big) \int_{\mathbb{T}^*\times\mathbb{S}} k(\mathbf{x}, \mathbf{u}_j) d\mathbf{x}. \end{aligned} \tag{8}$$

The point process model can be used to predict the expected number of events. The number of events is derived by integrating the estimated intensity over specific time period $P \subset \mathbb{T}^*$ and region of interest $Q \subset \mathbb{S}$ such that

$$N(P \times Q) = \int_{P\times Q} \hat{\lambda}(\mathbf{x}) d\mathbf{x} = \sum_{\mathbf{u}_j \in \mathcal{U}^*} f\Big(\mathbf{u}_j, \mathbf{z}_j; \hat{\theta}\Big) \int_{P\times Q} k(\mathbf{x}, \mathbf{u}_j) d\mathbf{x}, \tag{9}$$

where $N(A)$ is the number of events that fall into subset A. As discussed in Section 4.3, the above integral has a tractable solution.

5 EXPERIMENTS

In this section, we use real-world data sets from different domains to evaluate the predictive performance of our model.

5.1 Data Sets

5.1.1 Event data. We used three event data sets from different domains collected in New York City and Chicago from Jan 1, 2016 to April 1, 2016 (the observation period is 13 weeks). The details are as follows.

NYC Collision Data. New York City vehicle collision (NYC Collision) data set contains ~ 32 thousand motor vehicle collisions. Every collision is recorded in the form of time and location (latitude and longitude coordinates).

Chicago Crime Data. Chicago crime data set is a collection of reported incidents of crime that occurred in Chicago; it contains ~ 13 thousand records, each of which shows time, and latitude and longitude of where the crime happened.

NYC Taxi Data. New York City taxi pick-up (NYC Taxi) data set consists of ~ 30 million pick-up records in New York City collected by the NYC Taxi and Limousine Commission (TLC). Each record contains pick-up time, latitude and longitude coordinate. To reduce data size, we randomly selected 100 thousand events for our experiment.

For the 13-week observation period, we selected the last seven days as the test set, the last seven days before the test period as the validation set, and used the remaining data as the training set. Thus, $T = 120960$min and $\Delta T = 10080$min.

5.1.2 Urban contextual data. We used the following urban data as the contextual features.

Map Image. As the image features, we used *the map image* of the cities acquired from OpenStreetMap (OSM) database [1]. For each representative point $\mathbf{u}_j = (\tau_j, \mathbf{r}_j)$, we extracted the image around $\mathbf{r}_j$ (i.e., about 300m× 500m square grid space) and used its RGB vector as the input of the *image attention network*.

Social/Traffic Event Description. We collected *traffic events* (e.g., major street construction works and street events) and *social events* (e.g., sports events, musical concerts and festivals) in New York City as held by the 511NY website[2] during Jan 2016 through April 2016. The 13 week period contained a total of 8,968 descriptions. Each social/traffic event record contains a description of the event, as well as its start time t_s, end time t_e and location (latitude and longitude coordinates). For each representative point $\mathbf{u}_j = (\tau_j, \mathbf{r}_j)$, we extracted social/traffic event descriptions satisfying $t_s < \tau_j < t_e$ and located within a predefined distance, Δr from $\mathbf{r}_j$. If the point contains more than one sentence, we selected the spatially closest one. If the point contains no sentences, we used dummy variables. We used their descriptions, a sequence of 1-of-K coded word vectors, as the input of the text network. In this paper, we set Δr to the walking distance of 620m, following [4].

5.2 Experimental Setup

Hyper-parameters of each model are tuned by grid-search on the validation set. For DMPP, we used the Adam algorithm [19] as the optimizer, with $\beta_1 = 0.01$, $\beta_2 = 0.9$, and learning rate of 0.01. For *the multimodal fusion module* of DMPP, we tuned the hyper-parameters as follows: layer size n_l in {1,2,3,4}; number of units per layer n_u in {16, 32, 64}. The mini-batch size $|\bar{I}|$ is selected from the set {8, 16, 32}. Following the prior settings in [8], we also applied L2 regularization with λ=0.001 in both models. The number of representative points are tuned on the validation set in terms of the number of time points, M, and the number of space points, L. The tested combinations were $M = \{24, 28, 168\}$ and $L = \{4, 8, 10, 12\}$. We used three kinds of kernel functions: uniform, Gaussian, compactly supported Gaussian (the definitions are provided in Appendix B). Also, we explored various map styles: OSM default (the original map of Figure 4a), Watercolor (the original map of Figure 4b), Greyscale. The best settings for DMPP are given in the corresponding section. The default settings for each component of the neural network are described in Appendix A.

5.3 Evaluation Metrics

We evaluated the predictive performance using two metrics: **LogLike** (predictive log-likelihood) and **MAPE** (Mean Absolute Percentage Error). For the first metric, given the learned model, we calculated log-likelihood on the test data (**LogLike**) for each event as $\mathcal{L}^*/N_t$, where $\mathcal{L}^*$ is the test log-likelihood defined by Equation (7) and N_t is the number of test events. **MAPE** is used to evaluate the performance of event number prediction; it is defined as the absolute difference between the predicted number of events and the actual number: $\text{MAPE} = \sum_{r=1}^{N_r} \sum_{t=1}^{N_b} |n_{r,t} - \hat{n}_{r,t}|/n_{r,t}$, where $n_{r,t}$ is the number of events observed in the r-th grid cell and t-th time interval, and $\hat{n}_{r,t}$ is the corresponding prediction. N_r is the number of grid cells and N_b is the number of time bins for which predictions are made. In our experiment, we partitioned the region of interest using a 10×10 uniform grid, and divided the test period (seven days) into 14 time bins with a fixed uniform interval of 12 hours. Therefore $N_r = 100$ and $N_b = 14$. For DMPP, we predicted the number of events for each pair of spatial grid cell and future time bin, using Equation (6).

5.4 Comparison Methods

We compared the proposed model and its variants with three existing methods.

- HP (Homogeneous Poisson process): The intensity is assumed to be constant over space and time: $\lambda(\mathbf{x}) = \lambda_0$. The optimization can be solved in closed form.
- LGCP (Log Gaussian Cox process) [7]: LGCP is a kind of Poisson process with varying intensity, where the log-intensity is assumed to be drawn from a Gaussian process (See Appendix C). For LGCP, we performed the comparison only on event number prediction, since the log-likelihood of this model is computationally intractable. The inference is based on the Markov chain Monte Carlo (MCMC) approach (see [32] for details). For event number prediction, we sampled events from LGCP using the thinning method [22], and compared the aggregated number of events within predefined spatial regions and time periods with the ground truth.
- RMTPP (Recurrent Marked Temporal Point Process) [8]: RMTPP uses RNN to describe the intensity of the marked temporal point process; it assumes a partially parametric form for the intensity, and can capture temporal burst phenomena. This model is primarily intended to model event timing; to allow comparison, we mapped latitude and longitude values into location names and treating them as marks, using Neighborhood Names GIS data [3,4] (details are provided in Appendix C). The following hyper-parameters are tuned on the validation set: Number of units per layer in {16, 32, 64} and mini-batch size in {8, 16, 32}. The optimal settings

[1] Map data copyrighted OpenStreetMap contributors and available from https://www.openstreetmap.org

[2] https://www.511ny.org

[3] NYC Neighborhood Names GIS data, https://data.cityofnewyork.us/City-Government/Neighborhood-Names-GIS/99bc-9p23

[4] Chicago Neighborhood Names GIS data, https://data.cityofchicago.org/Facilities-Geographic-Boundaries/Boundaries-Neighborhoods/bbvz-uum9

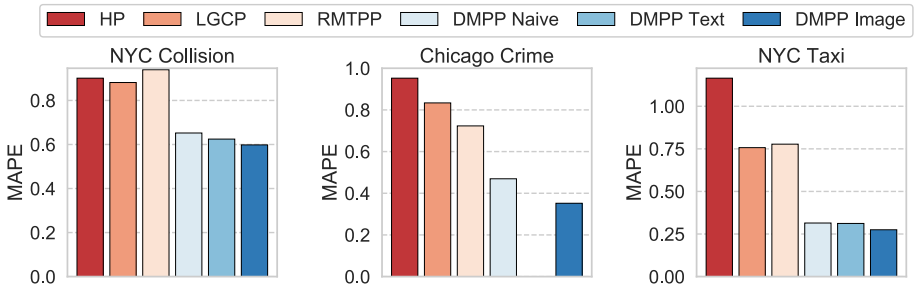


Figure 1: MAPE for event number prediction from six methods on three data sets: NYC Collision data (left); Chicago Crime data (middle); NYC Taxi data (right). Lower is better. DMPP is the proposed method. The error bars are omitted as the deviations are negligible.

Table 1: Comparison of the proposed method and its variants to the baselines. The number indicates the predictive log-likelihood per event (LogLike) for the test data. Higher is better.

	NYC Collision	Chicago Crime	NYC Taxi
HP	8.232	9.384	10.473
DMPP *Naive*	8.296	9.614	11.311
DMPP *Text*	8.297	NA	11.318
DMPP *Image*	**8.409**	**9.621**	**11.333**

are given in Appendix C. Note that the likelihood of RMTPP is for the location names, not for latitude and longitude values. For event number prediction, we sequentially predicted next event (See Appendix C). The predicted location names are mapped to latitude and longitude coordinates by simply using their centroids; and then the generated events are aggregated into counts.

We introduce three variants of DMPP below.

- DMPP *Naive*: The simplest variant of DMPP, it does not incorporate any contextual features. The neural network of DMPP accepts the location and time of each representative point $\mathbf{u}_j$.
- DMPP *Image*/*Text*: The DMPP variants that incorporate either map images or the social/traffic event descriptions, as well as the locations and times of the representative points.

5.5 Quantitative results

Figure 1 shows the overall MAPE of the six different methods on the three data sets for event number prediction. In this figure, the error bars are omitted as the deviations are negligible. The results indicate the superiority of our approach. HP performs worse than the other methods across almost all data sets, as it does not consider the spatio-temporal variation of the rate of event occurrence. We can see this in Figure 2, which depicts events generated by four different methods from the Chicago Crime data. We simulated events with the thinning algorithm [22], using the learned intensity of each method. LGCP presents better performance for NYC Collision and NYC Taxi data, as it captures spatio-temporal variations. RMTPP achieves relatively better performance than LGCP only for the Chicago Crime data. The result suggests that the assumption of RMTPP, the temporal burst phenomena, holds for Chicago Crime data, but not for NYC Collision data and NYC Taxi data. Even our simple model, DMPP *Naive*, largely surpasses all existing methods. The result implies that the parametric assumptions of the existing methods are too restrictive, and do not capture real urban phenomena. Also, RMTPP intensity is influenced by all the past events, regardless of how spatially far away, so it is not suited for spatio-temporal events. The differences between DMPP *Naive* and the best among the existing methods are significant (two-sided t-test: p-value$<$ 0.01) for all data sets. DMPP *Naive* outperforms LGCP in terms of MAPE by 0.229 for the NYC Collision data, from 0.778 to 0.315 for the NYC Taxi data. DMPP *Naive* outperforms RMTPP in terms of MAPE by 0.254 for the Chicago crime data. DMPP *Text* further improves DMPP *Naive* to 0.624 for the NYC Collision data, to 0.312 for NYC Taxi data. We can clearly see that DMPP *Image* offers significantly improved prediction performance. From these results, we can conclude that considering urban contexts is very effective in improving event prediction performance. The results also suggest that our proposal, DMPP, effectively utilizes the information provided by urban contexts.

Table 1 lists the LogLike values (predictive log-likelihood) of the four different methods for the three data sets, i.e., NYC Collision (vehicle collision) Data, Chicago Crime Data and NYC Taxi (taxi pick-up) Data. Note that DMPP *Text* is not applicable to Chicago Crime data, as no text data is available for Chicago. The proposal, DMPP, outperforms HP. Even the simplest variant of DMPP, DMPP *Naive* explains the observed event sequences better than these existing methods, which demonstrates the expressiveness of DMPP. DMPP *Text* also outperforms HP. DMPP *Image* achieves the best performance among all methods. This again shows the effectiveness of incorporating the urban contexts and our point process formulation with the deep neural network.

5.5.1 Sensitivity study. Here we analyze the impact of parameters on DMPP, including (1) number of representative points; (2) kernel function (3) map style; and (4) neural network structure.

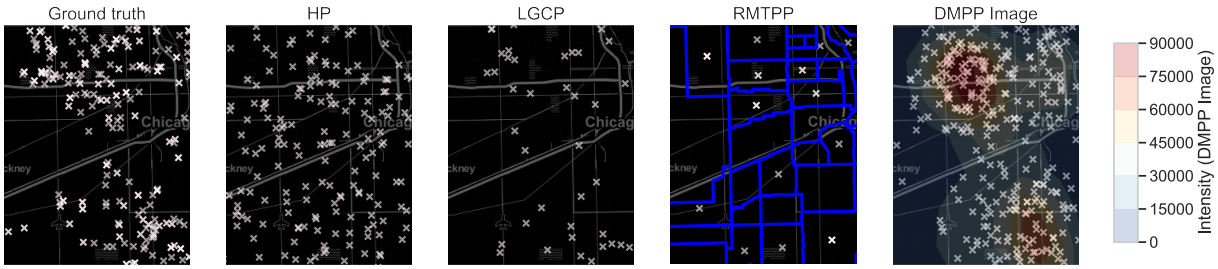
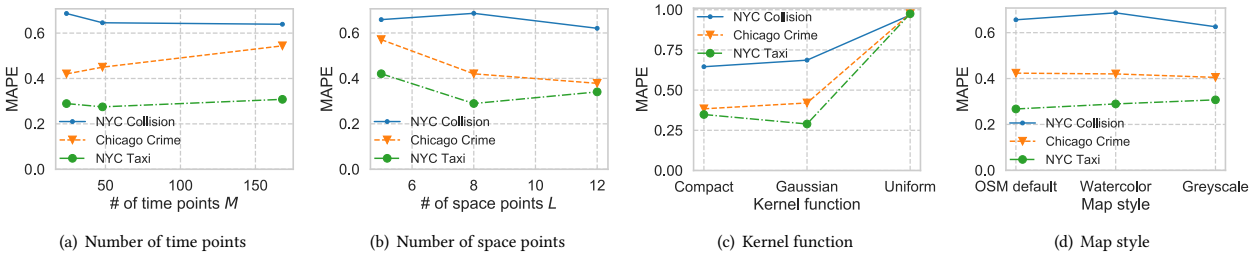
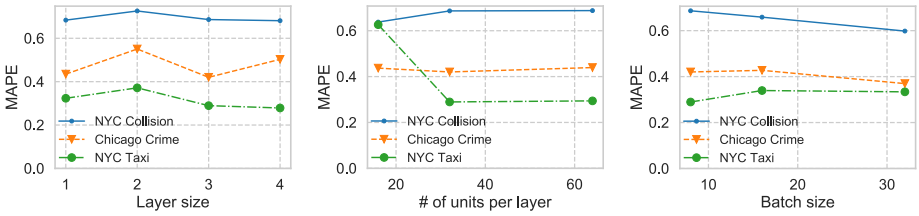


Figure 2: Events generated by four of the implemented methods for Chicago Crime data in Central and West Chicago between 0:00 am and 24:00 pm, on Mar 31th. The cross markers (x) denote the events generated by simulations. In the fourth plot, the blue lines denote the location boundaries used for RMTPP in the experiment. In the right-most plot, we overlaid the estimated intensity for DMPP *Image* at Mar 31th 12:00 am.



(a) Number of time points (b) Number of space points (c) Kernel function (d) Map style



(e) Layer size (f) Number of units per layer (g) Batch size

Figure 3: Impact of hyper-parameters on MAPE performance.

Number of Representative Points. Figure 3a and 3b show the impact of the numbers of representative points on the performance of DMPP *Image*. Figure 3a shows that MAPE is slightly improved when $M = 168$ for NYC Collision data, $M = 48$ for Chicago Crime data, and $M = 24$ for NYC Taxi data. As shown in Figure 3b, the prediction performance of DMPP generally tends to increase with number of time points M. Overall, DMPP is moderately robust to variations in the numbers of representative points. In this experiment, we fixed the network depth n_l to 4, the number of units per layer n_u to 32 and batch size $|\mathcal{I}|$ to 16.

Choice of Kernel Function. Figure 3c presents the prediction results with three kernel functions: Uniform, Gaussian, compactly supported Gaussian(definitions are provided in Appendix B). We can observe that the compactly supported kernel offers similar accuracy to the Gaussian kernel, while affording a computational advantage (See Appendix B). The uniform kernel performs worst. Throughout this paper, we use the compactly supported Gaussian kernel as the default setting. The hyper-parameters were set to $n_l = 4$, $n_u = 32$, $|\mathcal{I}| = 16$, $M = 24$ and $L = 20$ in this experiment.

Choice of Map Style. Figure 3d demonstrates the effect of map style. The predictive performance appears to be insensitive to the style of map images. For NYC Taxi data, MAPE is slightly improved when using the OSM default style. This may because the OSM default map distinguishes minor and major roads by color (as shown in the original map image of Figure 4b). As the default setting, we use OSM default style for NYC Collision and NYC Taxi data, and Watercolor for Chicago Crime data. In this experiment, we set $n_l = 4$, $n_u = 32$, $|\mathcal{I}| = 16$, $M = 24$ and $L = 20$.

Network Structure. We show the impact of network structures in Figure 3e-3g. The prediction performance slightly improves when layer size is 4, for NYC Collision data and NYC Taxi data, layer size 3 for Chicago Crime data. DMPP performs robustly for large number

of units $n_u >= 32$ across all the data sets. The prediction accuracy is likely to become better for larger batch size (Figure 3f). In this experiment, we fix $M = 24$ and $L = 12$, respectively. The optimal value of network hyper-parameters correspond to $n_l = 4$, $n_u = 16$, $|\mathcal{I}| = 32$ for NYC Collision data, $n_l = 3$, $n_u = 64$, $|\mathcal{I}| = 32$ for Chicago Crime data, $n_l = 4$, $n_u = 64$, $|\mathcal{I}| = 8$ for NYC Taxi data.

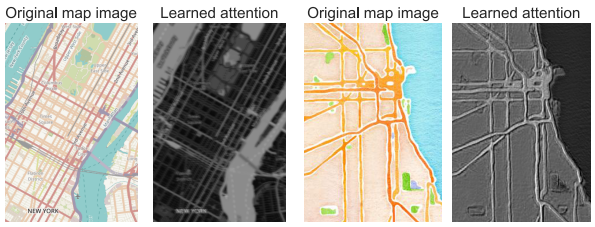
In conclusion, DMPP is moderately robust to variations in the hyper-parameters, and so can yield steady performance under different conditions.

5.6 Qualitative results

To demonstrate that our model provides useful insights as to why and under which circumstances events occur, we analyze what was learned by our method.

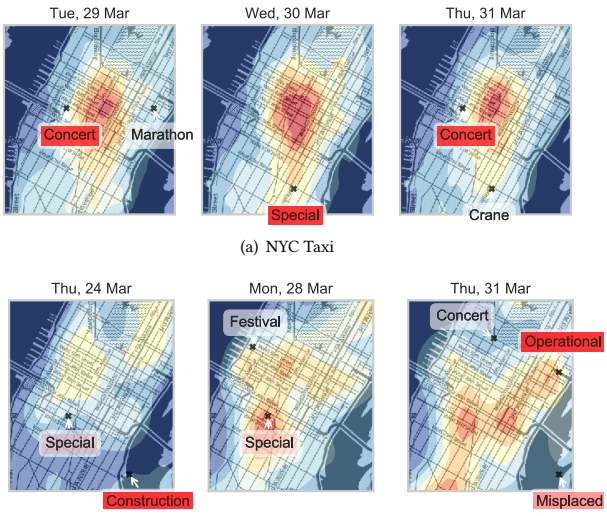
Figure 4 visualizes the learned attention for the map images from NYC Taxi data (Figure 4a) and Chicago Crime data (Figure 4b). Here we fed the map images (left) into the learned *image attention network*, and plot the output attention weights (right). In the attention heatmaps (right), the light and dark regions correspond to high and low attention weights, respectively. We can see that DMPP assigns high attention weights to major roads (depicted by pink in the original map image) for the NYC Taxi data in Figure 4a. The minor roads (depicted by light orange) draw less attention. This indicates that taxi pick-ups take place mostly on roads, especially on major roads. Interestingly, for Chicago Crime data (Figure 4b), the attention mechanism weights both the roads and land cover. Apparently, the roads have the highest attention weights. This may because crime is found both on roads and in building. These results suggest that our method can elucidate the key spatial components related to event occurrence.

Figure 5 shows the attention weights for the event descriptions overlaid with the learned intensity around Midtown Manhattan from Mar 24th to 31th. For the sake of clarity, only the first word of each sentence is depicted. The word *Special* appears usually with *event*; *Operational* stands for *Operational (activity)*. The darker shade of red for the texts indicates higher attention values. In the heatmaps, red corresponds to high intensity value, while blue represents low value. For NYC Taxi data (Figure 5a), the learned intensity yields high values in Midtown and Murrey Hill of Manhattan. Seemingly, the attention mechanism also highlights the words associated with these regions. Mainly words related to the social events, e.g., *Concert* and *Special (event)*, are highlighted. In contrast, the words associated with traffic events, such as *Construction* and *Operational (activity)*, gain more attention, in the NYC Collision data (Figure 5b). The above results suggest that traffic events affect the collision rate, whereas social events drive the taxi pick-up demand. Figure 6 further supports this. It visualizes the top 15 words ranked by attention weight learned from each data set; larger size denotes higher attention. For NYC Taxi data, social events (e.g., *special (event)* and *concert*), as well as traffic event (e.g., *construction*), tend to receive attnetion. For NYC Collision data, traffic events, e.g., *construction* and *operational (activity)*, seem to draw more attention. These results demonstrate that DMPP identifies important words that affect event occurrence. The descriptions thus found help us explain why and in which contexts events occur.



(a) NYC Taxi (b) Chicago Crime

Figure 4: Attention weights for the map images learned from NYC Taxi data and Chicago Crime. Left: original map image. Right: learned attention weights; lighter shade indicates stronger attention values.



(a) NYC Taxi

(b) NYC Collision

Figure 5: Learned attention weights for social/traffic descriptions with the learned intensity around Midtown Manhattan from Mar 28th to Apr 4th. Darker shade of red for the texts denotes higher attention weight.

6 CONCLUSION AND FUTURE WORK

In this paper, we studied the problem of event prediction in urban areas. Our solution, DMPP (Deep Mixture Point Process), is a novel point process model based on a deep learning approach. DMPP models the point process intensity by a deep mixture of kernels. The key advantage of DMPP over existing methods is that it can utilize the highly-dimensional and multi-sourced data provided by rich urban contexts, including images and sentences, and automatically discover their complex effects on event occurrence. Moreover, by taking advantage of the mixture model-based approach, we have developed an effective learning algorithm. Using real-world data

(a) NYC Taxi (b) NYC Collision

Figure 6: Word cloud of top 15 words by attention weight; larger size denotes higher attention.

sets from three different domains, we demonstrated that the proposed method outperforms existing methods in terms of prediction accuracy.

REFERENCES

[1] Yacine Aït-Sahalia, Julio Cacho-Diaz, and Roger JA Laeven. 2015. Modeling financial contagion using mutually exciting jump processes. *Journal of Financial Economics* 117, 3 (2015), 585–606.

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

[3] Patricia L. Brantingham and Paul J. Brantingham. 1981. Mobility, Notoriety, and Crime: A Study in the Crime Patterns of Urban Nodal Points. *Journal of Environmental Systems* 11, 1 (1981), 89–99. https://doi.org/10.2190/DTHJ-ERNN-HVCV-6K5T

[4] Longbiao Chen, Daqing Zhang, Leye Wang, Dingqi Yang, Xiaojuan Ma, Shijian Li, Zhaohui Wu, Gang Pan, Thi-Mai-Trang Nguyen, and Jérémie Jakubowicz. 2016. Dynamic cluster-based over-demand prediction in bike sharing systems. In *Proceedings of the 18th Ubicomp*. ACM, 841–852.

[5] Edward Choi, Nan Du, Robert Chen, Le Song, and Jimeng Sun. 2015. Constructing disease network and temporal progression model via context-sensitive hawkes process. In *Proceedings of the 15th ICDM*. IEEE, 721–726.

[6] David R Cox. 1992. Regression models and life-tables. In *Breakthroughs in statistics*. Springer, 527–541.

[7] Peter J Diggle, Paula Moraga, Barry Rowlingson, Benjamin M Taylor, et al. 2013. Spatial and spatio-temporal log-Gaussian Cox processes: extending the geostatistical paradigm. *Statist. Sci.* 28, 4 (2013), 542–563.

[8] 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 *Proceedings of the 22nd KDD*. ACM, 1555–1564.

[9] Seyda Ertekin, Cynthia Rudin, and Tyler H. McCormick. 2015. Reactive point processes: A new approach to predicting power failures in underground electrical systems. *Annals of Applied Statistics* 9, 1 (2015), 122–144. https://doi.org/10.1214/14-AOAS789 arXiv:1505.07661

[10] Mehrdad Farajtabar, Nan Du, Manuel Gomez Rodriguez, Isabel Valera, Hongyuan Zha, and Le Song. 2014. Shaping social activity by incentivizing users. In *Advances in Neural Information Processing Systems*. 2474–2482.

[11] Song Gao, Yaoli Wang, Yong Gao, and Yu Liu. 2013. Understanding urban traffic-flow characteristics: a rethinking of betweenness centrality. *Environment and Planning B: Planning and Design* 40, 1 (2013), 135–153.

[12] Roch Giorgi et al. 2003. A relative survival regression model using B-spline functions to model non-proportional hazards. *Statistics in medicine* 22, 17 (2003), 2767–2784.

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

[14] Junichiro Hayano et al. 2011. Increased non-gaussianity of heart rate variability predicts cardiac mortality after an acute myocardial infarction. *Frontiers in physiology* 2 (2011), 65.

[15] Dave Higdon. 2002. Space and space-time modeling using process convolutions. In *Quantitative methods for current environmental issues*. Springer, 37–56.

[16] Minh X Hoang, Yu Zheng, and Ambuj K Singh. 2016. FCCF: forecasting citywide crowd flows based on big data. In *Proceedings of the 24th ACM SIGSPATIAL*. ACM, 6.

[17] Tomoharu Iwata, Amar Shah, and Zoubin Ghahramani. 2013. Discovering latent influence in online social activities via shared cascade poisson processes. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge Discovery and Data mining*. ACM, 266–274.

[18] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[19] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[20] Herbert K H Lee, Bruno Sanso, Weining Zhou, and David M Higdon. 2008. Inference for a proton accelerator using convolution models. *J. Amer. Statist. Assoc.* 103, 482 (2008), 604–613.

[21] Ricardo T Lemos and Bruno Sansó. 2009. A spatio-temporal model for mean, anomaly, and trend fields of North Atlantic sea surface temperature. *J. Amer. Statist. Assoc.* 104, 485 (2009), 5–18.

[22] PA W Lewis and Gerald S Shedler. 1979. Simulation of nonhomogeneous Poisson processes by thinning. *Naval research logistics quarterly* 26, 3 (1979), 403–413.

[23] Liangda Li, Hongbo Deng, Anlei Dong, Yi Chang, and Hongyuan Zha. 2014. Identifying and labeling search tasks via query-based hawkes processes. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 731–740.

[24] Zhouhan Lin, Minwei Feng, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130* (2017), 1–15. arXiv:arXiv:1703.03130v1

[25] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. 2017. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *Proceedings of the 30th CVPR*, Vol. 6. 2.

[26] George O Mohler, Martin B Short, P Jeffrey Brantingham, Frederic Paik Schoenberg, and George E Tita. 2011. Self-exciting point process modeling of crime. *J. Amer. Statist. Assoc.* 106, 493 (2011), 100–108.

[27] Yosihiko Ogata. 1998. Space-time point-process models for earthquake occurrences. *Annals of the Institute of Statistical Mathematics* 50, 2 (1998), 379–402.

[28] Michael D Porter, Gentry White, et al. 2012. Self-exciting hurdle models for terrorist activity. *The Annals of Applied Statistics* 6, 1 (2012), 106–124.

[29] Frederic Schoenberg, Marc Hoffmann, and Ryan Harrigan. 2017. A recursive point process model for infectious diseases. *arXiv preprint arXiv:1703.08202* (2017).

[30] Laura Serra, Marc Saez, Jorge Mateu, Diego Varga, Pablo Juan, Carlos Díaz-Ávalos, and Håvard Rue. 2014. Spatio-temporal log-Gaussian Cox processes for modelling wildfire occurrence: the case of Catalonia, 1994–2008. *Environmental and Ecological Statistics* 21, 3 (2014), 531–563.

[31] Masamichi Shimosaka, Keisuke Maeda, Takeshi Tsukiji, and Kota Tsubouchi. 2015. Forecasting urban dynamics with mobility logs by bilinear Poisson regression. In *Proceedings of the 17th Ubicomp*. ACM, 535–546.

[32] Benjamin M Taylor, Tilman M Davies, Barry S Rowlingson, Peter J Diggle, et al. 2013. lgcp: an R package for inference with spatial and spatio-temporal log-Gaussian Cox processes. *Journal of Statistical Software* 52, 4 (2013), 1–40.

[33] Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton. 2015. Chainer: a next-generation open source framework for deep learning. In *Proceedings of workshop on Machine Learning Systems (LearningSys) in the 22th NIPS*, Vol. 5. 1–6.

[34] Pengfei Wang, Yanjie Fu, Guannan Liu, Wenqing Hu, and Charu Aggarwal. 2017. Human mobility synchronization and trip purpose detection with mixture of hawkes processes. In *Proceedings of the 23rd KDD*. ACM, 495–503.

[35] Senzhang Wang, Lifang He, Leon Stenneth, Philip S Yu, and Zhoujun Li. 2015. Citywide traffic congestion estimation with social media. In *Proceedings of the 23rd SIGSPATIAL*. ACM, 34.

[36] Holger Wendland. 1995. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics* 4, 1 (1995), 389–396.

[37] Shuai Xiao, Mehrdad Farajtabar, Xiaojing Ye, Junchi Yan, Le Song, and Hongyuan Zha. 2017. Wasserstein learning of deep generative point process models. In *Proceedings of the 30th NIPS*. 3247–3257.

[38] 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 *Proceedings of the 31th AAAI*, Vol. 17. 1597–1603.

[39] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction.. In *Proceedings of the 30th AAAI*. 1655–1661.

[40] Tian Zhou, Lixin Gao, and Daiheng Ni. 2014. Road traffic prediction by incorporating online information. In *Proceedings of the 23rd WWW*. ACM, 1235–1240.

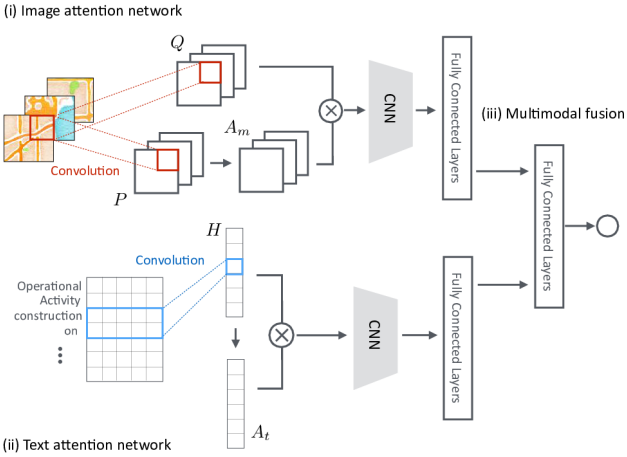


Figure 7: The architecture of the neural network used in the proposed method.

APPENDIX

A NEURAL NETWORK ARCHITECTURE

This section details the architecture of the neural network used in our experiment, see Figure 7. Our neural network consists of three components: (i) *the image attention network*, (ii) *the text attention network*, (iii) *the multimodal fusion module*. This section describes each component in detail. In this paper, we describe the proposal assuming the use of two types of features: map images and social/traffic event descriptions. Note that the proposed method can be easily extended to handle other types of features.

(i) Image attention network. We construct the *image network* by combining CNN with a self-attention mechanism, which extracts attention for regions of the image. Suppose we have a collection of map images $\{I_j\}_{j=1}^{J}$, $I_j \in \mathbb{R}^{N_w \times N_h \times N_c}$, where N_w, N_h, N_c represent width, height, and the number of image features (e.g., three color channels), respectively. In the following discussion, we omit index j for the sake of simplicity. *The image attention network* accepts images I, and passes them through convolutional transformation followed by pooling and activation layers.

$$P = g_p(C_p * I), \;\; Q = g_q(C_q * I) \tag{10}$$

where $*$ denotes the convolution; C_p and C_q are the parameter matrices to be learnt; $g_p(\cdot)$ and $g_q(\cdot)$ are a set of activation and pooling operations. For our experiment, we use 3×3 same convolution so as to straightforwardly visualize the attention weights developed for the image features. Subsequently, we process P through a spatial attention model consisting of a single self-attention layer followed by a softmax function:

$$A_m = \text{softmax}\Big(M_2 \tanh(M_1 P^\top)\Big), \tag{11}$$

where $M_1 \in \mathbb{R}^{d \times d_a}$ and $M_1 \in \mathbb{R}^{r \times d_a}$ are parameter matrices. In the experiment, we set $d = N_c$, $d_a = 32, r = 1$. The attention weights $A_m \in \mathbb{R}^{N_h \times N_w}$ indicate which regions of the image were focused on during training. Then, we multiply the intermediate map P by the attention A_m. The output of the self-attention layer, B, is processed by a three layer CNN with a set of 3×3 convolutions. The output is then processed by two fully connected layers, with size of 512, and the rectified linear unit (ReLU) activation functions.

(ii) Text attention network. Social/traffic event descriptions are represented as a sequence of words $\{W_j\}_{j=1}^{J}$, $W_j \in \mathbb{R}^{N_s \times N_v}$, where N_s is the length of the sentence and N_v is the vocabulary size. We design the *text network* on the CNN designed for sentences [18] and an attention mechanism [24]. First, *the text attention network* reads the input sequence of 1-of-K word vectors $W = [\mathbf{w}_1, ..., \mathbf{w}_{N_s}]$, $\mathbf{w} \in \{0, 1\}^{N_v}$ and transforms it into a set of hidden vectors $H = [\mathbf{h}_1, ..., \mathbf{h}_{N_s}]$, where $\mathbf{h}_i$ is a r-dimensional vector. We then feed the vectors into the attention network. In particular, we transform the set of hidden vectors $H = [\mathbf{h}_1, ..., \mathbf{h}_{N_s}]$ into new vectors with dimension d_c such that

$$A_t = \text{softmax}\Big(T_2 \tanh(T_1 H^\top)\Big), \tag{12}$$

where $T_1 \in \mathbb{R}^r$ and $T_1 \in \mathbb{R}^r$ are parameter matrices. We set $r = 1$ in our experiment. $A_t \in \mathbb{R}^{N_s}$, the attention weight for each word, reflects the importance of the word. We multiply the hidden vectors H with the attention weights A_t, and feed the results through a three layer CNN. The output of the CNN is transformed by two fully connected layers with size of 8, ReLU activation functions, and dropout of 0.1.

(iii) Multimodal fusion module. The positions of the representative points, $\mathbf{u}$, are processed by two fully connected layers with 32 units and relu activations. Their output and the outputs of the attention network (*the image attention network* or *text attention network*) are concatenated. This is followed by fully connected layers with relu activation functions. The hyper-parameters of the last fully connected layers for *the multimodal fusion module* are tuned on the validation set.

B IMPLEMENTATION

B.1 Kernel function

As mentioned in 5.2, we explored three kinds of kernel functions: uniform, Gaussian, and compactly supported Gaussian. We define each kernel below.

Uniform kernel.

$$k(\mathbf{x}, \mathbf{u}_j) = \mathbb{1}(||\mathbf{x} - \mathbf{u}_j|| < w),$$

where $\mathbb{1}(\cdot)$ is an indicator function.

Gaussian kernel.

$$k(\mathbf{x}, \mathbf{u}_j) = \exp\Big(-(\mathbf{x} - \mathbf{u}_j)^\top \Sigma^{-1} (\mathbf{x} - \mathbf{u}_j)\Big),$$

where Σ is a 3×3 covariance matrix.

Compactly supported Gaussian kernel.

$$k(\mathbf{x}, \mathbf{u}_j) = \exp\Big(-(\mathbf{x} - \mathbf{u}_j)^\top \Sigma^{-1} (\mathbf{x} - \mathbf{u}_j)\Big) \cdot \mathbb{1}(||\mathbf{x} - \mathbf{u}_j|| < w),$$

where Σ is a 3×3 covariance matrix, $\mathbb{1}(\cdot)$ is an indicator function, and w is a positive parameter that thresholds the kernels, $||\mathbf{x} - \mathbf{u}_j|| \geq w$, to zeros. This means that $k(\mathbf{x}, \mathbf{u}_j)$ will be zero when $\mathbf{x}$ and $\mathbf{u}_j$ are far enough away. The use of the compactly supported kernel allows for an effective learning algorithm, especially for large data size N and for large numbers of representative points, J. The objective (6) involves kernel evaluations for all pairs of $\mathbf{x}_i$ and $\mathbf{u}_j$, resulting in an $N \times J$ kernel matrix K with elements $K_{ij} = k(\mathbf{x}_i, \mathbf{u}_j)$. The back-propagation is carried out by taking the

derivation of K, which requires $O(NJ)$ operations at each iteration ($O(|\mathcal{I}|J)$ for the mini-batch optimization). The computation burden can be impractically heavy when the data size N (the mini-batch size $|\mathcal{I}|$) or the number of representative points J is large. The use of the compactly supported kernel allows us to scale up the learning algorithm. Such a kernel ensures that the kernel element K_{ij} is zero whenever the distance between $\mathbf{x}_i$ and $\mathbf{u}_j$ is above a certain threshold. This leads to a sparse structure in the kernel matrix, K, thus allowing for fast sparse matrix computations.

B.2 Parameters

For each representative point, we extract a 20×20 image patch from the map image of the entire region-of-interest (i.e., Manhattan for NYC Collision data and NYC Taxi data, City of Chicago for Chicago Crime data), resize it to 10×10 pixels, and use its RGB vector as the input image. This corresponds to 290m× 500m square grid space for NYC Collision data and NYC Taxi data, and 290m× 600m square grid space for Chicago Crime data. Therefore, $N_w = 10$, $N_h = 10$, $N_c = 3$. In *the text network*, we only consider the first 200 most frequent words, and use the first 5 words of each sentence. For our input descriptions, we zero-pad to ensure a sentence length of 5 words. Thus, $N_v = 200$ and $N_s = 5$.

B.3 Environment

RMTPP and our DMPP are implemented using the Chainer deep network toolkit [33]. All the methods are run on a Linux server with an Intel Xeon CPU, and a GeForce GTX TITAN GPU. The GPU code is implemented using CUDA 9.

C COMPARED METHODS

In Section 5, we compared the proposed method with the existing methods in terms of the predictive log-likelihood per event (LogLike). Below we describe the setting and the procedure adopted for each method in detail.

HP (Homogeneous Poisson process). The intensity of HP is assumed to be constant over space and time: $\lambda(\mathbf{x}) = \lambda_0$. Given the test period $[T, T + \Delta T]$ and the region of interest $\mathbb{S}$, the likelihood of HP is written as

$$\log p(\mathcal{X}|\lambda(\mathbf{x}) = \lambda_0) = n \log \lambda_0 - \lambda_0 \Delta T |\mathbb{S}|, \tag{13}$$

where n is the number of test samples, ΔT is the length of the test period, and $|\cdot|$ is the operator providing the area of a spatial region.

LGCP (Log Gaussian Cox process). We implement the general LGCP method described in [32], where the intensity is defined as $\lambda(\mathbf{x}) = \mu(t)\psi(s) \exp\left(y(\mathbf{x})\right)$; $\mu(t)$ and $\psi(s)$ are temporal and spatial background rates, respectively. $y(\cdot)$ is a Gaussian process with the following covariate function:

$$\mathrm{cov}(\mathbf{x}, \mathbf{x}') = \sigma_{\mathrm{GP}}^2 \exp\left(-(\mathbf{x} - \mathbf{x}')^\top \theta_{\mathrm{GP}} (\mathbf{x} - \mathbf{x}')\right), \tag{14}$$

where σ_{GP} is the scale parameter, θ_{GP} is the bandwidth.

RMTPP (Recurrent Marked Temporal Point Process). As for DMPP, we used the Adam algorithm [19] with $\beta_1 = 0.01$, $\beta_2 = 0.9$ and learning rate of 0.01 for RMTPP. Following [8], we map the coordinates using NYC Neighborhood Names GIS dataset[5] for NYC Collision and NYC Taxi data. Similarly, we use Chicago Neighborhood Names GIS dataset[6] for Chicago Crime data. Finally, we obtained 48 unique locations for NYC Collision and NYC Taxi data, 44 for Chicago Crime data. We set unit size as 16 and batch size as 32 for NYC Collision data, unit size as 16 and batch size as 8 for Chicago Crime data, unit size as 16 and batch size as 16 for NYC Taxi data. We use the log-likelihood (LogLike) as defined in [8]. For event number prediction, we generate the sequence of events by sequentially predicting the timing of the next event. In particular, given the event sequence $\{t_1, ..., t_N\}$, we compute the timing of next event $\hat{t}_{N+1}$, using Equation (13) in [8]. Using $\hat{t}_{N+1}$ as known data, we then predict the timing of next event $\hat{t}_{N+2}$ based on the new sequence $\{t_1, ..., t_N, \hat{t}_{N+1}\}$. This procedure is repeated until $\hat{t}_{N+i} > T + \Delta T$.

[5] https://data.cityofnewyork.us/City-Government/Neighborhood-Names-GIS/99bc-9p23

[6] https://data.cityofchicago.org/Facilities-Geographic-Boundaries/Boundaries-Neighborhoods/bbvz-uum9