Predicting Urban Dispersal Events: A Two-Stage Framework through Deep Survival Analysis on Mobility Data

Amin Vahedian, Xun Zhou, Ling Tong, W. Nick Street
The University of Iowa
{amin-vahediankhezerlou,xun-zhou,ling-tong,nick-street}@uiowa.edu
And
Yanhua Li
Worcester Polytechnic Institute
yli15@wpi.edu

Abstract

Urban dispersal events are processes where an unusually large number of people leave the same area in a short period. Early prediction of dispersal events is important in mitigating congestion and safety risks and making better dispatching decisions for taxi and ride-sharing fleets. Existing work mostly focuses on predicting taxi demand in the near future by learning patterns from historical data. However, they fail in case of abnormality because dispersal events with abnormally high demand are non-repetitive and violate common assumptions such as smoothness in demand change over time. Instead, in this paper we argue that dispersal events follow a complex pattern of trips and other related features in the past, which can be used to predict such events. Therefore, we formulate the dispersal event prediction problem as a survival analysis problem. We propose a two-stage framework (DILSA), where a deep learning model combined with survival analysis is developed to predict the probability of a dispersal event and its demand volume. We conduct extensive case studies and experiments on the NYC Yellow taxi dataset from 2014-2016. Results show that DILSA can predict events in the next 5 hours with F1-score of 0.7 and with average time error of 18 minutes. It is orders of magnitude better than the state-of-the-art deep learning approaches for taxi demand prediction.

Introduction

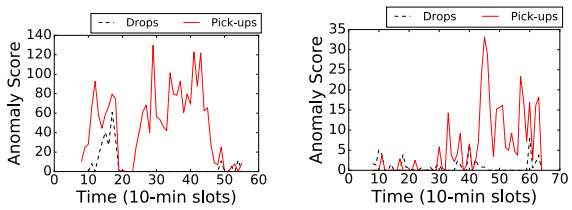
An urban dispersal event is the process where an abnormally large crowd leaves the same area within a short period. Dispersal events can be observed after large gathering events, such as concerts, sporting events, or protests. Unexpected dispersal events often cause public safety risks, congestion, and high demands of public transportation resources (e.g., taxis) within a short period. Therefore, early prediction of large dispersal events as well as the crowd size are of great importance to a number of different parties. (1) Public safety officials and traffic administrators can benefit from such a technique since they could allocate resources and make plans to mitigate potential risks or congestion. (2) Transportation stakeholders such as taxi drivers and fleet managers are enabled to improve profit by dispatching more taxis to such events if they can be predicted in advance.

Thus, dispersal event prediction is **non-trivial** and **necessary**. While most of the large events have schedules, the time of dispersion often has a high level of uncertainty due to varying occasions, attendees, and other factors such as weather. Moreover, many events are not planned or have much higher attendance than expected, such as social protests and gatherings. In addition, many events are not public and only known to special interest groups and it is not possible for the public to collect schedules of such events. For example, large groups of Pokemon Go game players gather for special events in the game. Social activities organized through instant messaging tools are not public, either. Finally, collecting and verifying schedule information from various sources often requires costly human labor and cannot be done in a fully automated manner.

In recent years many big mobility datasets such as taxi trip data and For-Hire Vehicle (FHV) requests (e.g., Uber), have become available. Automatic dispersal event prediction, therefore, has become feasible. A typical solution is to build models to predict taxi demand using the above datasets and identify high-demand locations as dispersal events. Related research shows taxi demand has a highly predictable pattern, when predicting near future (Zhao et al. 2016; Xu et al. 2017; Zhang et al. 2016; Zhao et al. 2016; Moreira-Matias et al. 2013; Davis, Raina, and Jagannathan 2016). However, that is often true only for regular demand prediction rather than abnormally high demand. In such a case, the assumptions of pattern repetitiveness are violated and the methods will fail to provide a timely and accurate prediction. In particular, their ability to make long-term forecast of abnormally high demand is weak due to assumptions that demand changes smoothly (auto-correlated) over time.

In this paper, we focus on predicting such non-periodic and unexpected large dispersal events with abnormally high taxi demand. Specifically, given the historical taxi trip records and other relevant features (e.g., weather, POI), we predict (1) when and where abnormally high taxi demand will occur, and (2) the volume of demand in the predicted time span of the dispersal event.

To address the limitations of related work, we propose an alternative solution. Firstly, we treat dispersal event prediction as a "Survival Analysis" problem, where we learn a model to predict the probability of "death" (event occurs) at each location in the future. Secondly, we argue that there is evidence of demand abnormality during the time leading to it, which can be used to train the survival analysis



(a) An anomalously high pick-up, preceded by an anomalously high number of arrivals.

(b) An anomalously high pick-up, not preceded by a drop event.

Figure 1: Examples of abnormally large number of pick-ups.

model. The intuition is that dispersal events are often caused by some forms of gatherings, which can indicate future abnormally high demand. Figure 1 (a) shows an example of abnormally high pick-ups for a concert event at Madison Square Garden in New York City. The dashed and solid lines represent the anomaly scores (Neill 2009) of the drops and the pick-ups, respectively. The large anomaly in pick-ups towards the end of the event follows an anomaly in the drops earlier. However, such patterns are not always as explicit. Fig. 1 (b) shows such a case around McKittrick Hotel in Manhattan, where there are no abnormal drops preceding the dispersal event. In such cases, finding the right signals for predicting dispersal events is more challenging.

In this paper, we make the following novel contributions: **(1) We propose a two-stage framework**, using deep neural networks to predict dispersal events. We incorporate various features including spatial, temporal, weather, and Point of Interest (POI) features, in addition to recent taxi pick-up and drop observations. **(2) We formulate the dispersal event prediction problem as a survival analysis problem** (Miller Jr 2011).In the two-stage prediction framework, we predict the time of abnormal demand using survival analysis and then predict the demand volume. We call our method **DILSA**, DIspersaL event prediction using Survival Analysis. We evaluate our methods using real-world data from New York City. Our evaluations show our method identifies dispersal events with F1-score of 0.7, while the error for predicting the time of the dispersal event is 18 minutes for a 5-hour prediction period. Also, our method predicts the pick-up demand in case of anomaly with superior accuracy compared to a baseline.

The rest of the paper is organized as follows: In the next section we discuss the related work, followed by problem formulation and our proposed computational solution. Then, we present the evaluations and conclude the paper.

Related Work

Prior related work include (1) event detection and forecasting, (2) taxi demand prediction, and (3) survival analysis.

Event Detection and Forecasting: Event detection has been widely studied in various domains, including public health, urban computing, and social network analysis. The works (Kulldorff 1997; Kulldorff et al. 2005; Neill 2009) and other recent works on event detection (Li, Xiong, and Liu 2012; Hong et al. 2015) use already observed counts. An event is defined as a region with significantly higher counts, such as disease reports or number of taxi drops. Social media posts and geo-tagged tweets have been used as well to detect and forecast events such as social unrests and protests (Zhou and Chen 2014; Chen and Neill 2014; Liu et al. 2016; Zhang et al. 2017). Regions and time windows where the frequency of certain keywords exhibit abnormal changes are identified as events. These works do not use mobility data. The **dynamic patterns** of the events such as gathering or dispersing **are not captured**.

Works (Zhou et al. 2016; Khezerlou et al. 2017; Hoang, Zheng, and Singh 2016) use traffic flow data to detect gathering events. Vahedian et al. use destination prediction to predict gathering events (Vahedian et al. 2017). However, **such methods are not applicable** to dispersal events, as trajectories and traffic flow are **observed only after such events**.

Taxi demand prediction has been studied closely in recent years, due to access to public taxi datasets (Zhao et al. 2016). To the best of our knowledge none of the proposed methods directly address the prediction problem in case of anomaly. State-of-the-art methods for predicting taxi demand use historical data and time series analysis. (Yao et al. 2018) propose a deep learning framework which captures the spatial and temporal dependencies to predict taxi demand. (Xu et al. 2017) formulate an LSTM Network to learn the regular pattern of taxi demand. (Zhao et al. 2016) show the regular taxi demand is highly predictable and test different algorithms to approach the maximum accuracy. (Zhang et al. 2016) used spatial clustering to predict demand hotspots. They predict areas with high density of demand using DBSCAN. Such areas, despite having high demand, are part of the regular pattern. (Moreira-Matias et al. 2013) used streams of taxi data as time series to predict taxi demand in the next 30-minute period. (Davis, Raina, and Jagannathan 2016) used time series analysis to solve the demand prediction problem, giving recommendations to drivers. (Mukai and Yoden 2012) used a simple multi-output ANN to predict demand, using features created from recent demand, time and weather information.

The above-mentioned research aims at learning the **regular pattern** of taxi demand in absence of anomaly. Considering the regular demand is highly predictable, in this paper, we take on the harder **challenge of predicting anomalous taxi demand**, which we believe is of greater importance.

Survival analysis is the analysis of duration of time until an event. It has been applied in engineering as well as health practices (Street 1998), for which it was originally developed (Miller Jr 2011). To the best of our knowledge, this is the first time survival analysis is used in the context of urban event prediction. In this paper, we propose to use a deep Artificial Neural Network to predict the probabilities of survival. Predicting the probabilities of survival at different time points using a common internal representation (the hidden nodes of a deep ANN) allows the learned model to share information across the time points, resulting in better predictive results.

Problem Formulation

Concepts and Definitions

We define a spatio-temporal field $Z = (S, T)$ as a two-dimensional geographical region S paired with a period of time T. S is partitioned by a grid. Each grid cell $l_1, l_2, ..., l_{|S|}$ represents a distinct location in the geographical region. T is partitioned into fixed-length time-steps. Given Z, the location of any moving object, can be mapped into a grid cell in S and a time-step in T. For instance, pick-up and drop location and time of a taxi trip can be represented by (l^s, t^s, l^d, t^d), where (l^s, t^s) are source location and time and (l^d, t^d) correspond to the destination. **Pick-up count** $\left(C^p_{l,t}\right)$ of grid cell l at time t is the number of trips with source (l, t). Similarly, **drop count** $\left(C^d_{l,t}\right)$, is the number of trips with destination (l, t). Since the drop and pick-up counts demonstrate a periodic pattern, we define baseline counts to represent the expected counts. **Pick-up baseline** $\left(B^p_{l,t}\right)$ of grid cell l at time t is the average of pick-up counts at l at the same time of day. **Drop baseline** $\left(B^d_{l,t}\right)$ is defined similarly. A **spatio-temporal region** $R = (S_R, T_R)$ is a rectangular sub-field of S paired with a continuous subset of T. The counts and baselines can be obtained for any spatio-temporal region, defined bellow. To study the abnormally high taxi demands, in this paper, we are interested in regions, where there are significantly higher counts than expected, i.e., when C^p_R is significantly higher than B^p_R. We assume C^p_R follows a Poisson distribution and test the following hypotheses: H_0: C^p_R is from a Poisson distribution of parameter B^p_R, H_1: C^p_R is from a Poisson distribution of a parameter larger than B^p_R. We use the Expectation-based Likelihood Ratio Test of (Neill 2009):

$$LLR(R) = \begin{cases} C^p_R \log \frac{C^p_R}{B^p_R} + (B^p_R - C^p_R) & \text{if } C^p_R \geq B^p_R \\ 0 & otherwise \end{cases} \quad (1)$$

(Zhou et al. 2016) showed that $LLR(R)$ is at α-level significance if $1 - Pr(X \leq C^p_R) \leq \alpha$, where $X \sim Poisson(B^p_R)$. Therefore, we define dispersal events:

Definition 1 *There is a **dispersal event** at spatio-temporal region R, if $LLR(R)$ is significant at α-level.*

Locations have specific attributes other than the pick-up and drop counts. We consider two of them: weather and Point of Interest (POI) vector. Locations have a daily maximum and minimum temperature, average wind speed and total precipitation, which impact the traffic and people's movement. In addition, locations consists of several POIs that can be categorized into functions. For instance, one grid cell in S might contain many hotels and few shopping centers, while another grid cell might contain many shopping centers. The distribution of categories of POIs over the space impacts people's movement. Therefore, we define a POI vector $V^l = (v^l_1, v^l_2, ..., v^l_n)$, where v^l_i is the number of places in category i at l.

Problem Statement

Given: Spatio-temporal field $Z = (S, T)$, historical trip records and weather information in Z, POI vectors of S, significance threshold α, current time t_c and target period $(t_c, t_g]$, **Find**: All the dispersal events and their (1) Start time $t_e \leq t_g$ of the dispersal event and (2) Demand volume $C^p_{T_g}$, in case of a predicted dispersal event, where $T_g = [t_e, t_g]$. Our objective is to improve accuracy of t_e and $C^p_{T_g}$.

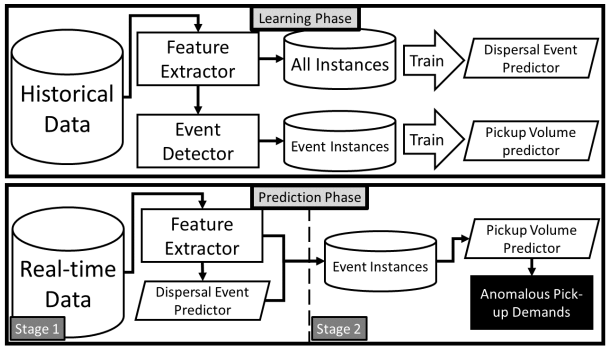


Figure 2: Dispersal event prediction framework.

Computational Solution

Overview

Per problem statement, we predict (1) start time of dispersal events, (2) demand during dispersal events. We propose the framework in Fig. 2. In the learning phase, we extract features from historical data and use them to train an event predictor based on Survival Analysis and a demand predictor. In the prediction phase, we follow two steps. First, we use the event predictor to predict the start time of the event. Then, we predict the pick-up count for the period of the event.

Survival Analysis

Survival analysis analyzes the expected time until an event happens (Miller Jr 2011). The event could be death or failure, or in this paper, a dispersal event. The analysis is primarily done using the survival function defined as follows:

$$S(t) = Pr(E > t). \quad (2)$$

In Eq. 2, $S(t)$ is the probability of the event not happening until t (subject has survived at t). Another commonly used function is the hazard function $h(t)$, which is the rate of event at time t, given that it has not occurred by then. Hazard function is defined as follows:

$$h(t) = \frac{-S'(t)}{S(t)}. \quad (3)$$

$-S'(t)$ is the rate with which $S(.)$ decreases at t. It is divided by $S(t)$, the remaining mass of survival probability, because it is conditional to the survival of the subject at t. We use this analysis to calculate the remaining time to dispersal events.

Feature Extraction

To do supervised learning, we need to have a training set with instances of inputs and outputs. In this section, we define the input variables, or the building blocks of the feature vector of the supervised learning framework. Let (l, t_c) be

the current location and time. We build the variables through following definitions:

Definition 2 ***Time profile*** *of* (l, t_c) *is* $Q^l_{t_c} = \langle d_y, d_w, t_c - t_d \rangle$*, where* d_y *and* d_w *are the day of the year and day of week for* t_c*, and* t_d *is the first time-step of current day.*

Definition 3 ***Weather profile*** *of* (l, t_c) *is* $W^l_{t_c} = \langle \omega, \eta, \zeta, \theta_{max}, \theta_{min} \rangle$*, where* ω *is average daily wind speed,* η *is total rain fall of the day,* ζ *is total snowfall of the day and* θ_{max} *and* θ_{min} *are the maximum and minimum temperatures of* l *at* t_c*.*

Definition 4 ***Daily profile*** *of* (l, t_c) *is defined as:*

$$M^l_{t_c} = \left\langle \sum_{t \in [t_d, t_c)} C^p_{l,t}, \sum_{t \in [t_d, t_c)} B^p_{l,t}, \sum_{t \in [t_d, t_c)} C^d_{l,t}, \sum_{t \in [t_d, t_c)} B^d_{l,t} \right\rangle. \tag{4}$$

The daily profile is a vector containing the sum of pick-up and drop counts and baselines since the start of current day. It is important, because a gradual gathering during the day can result in an accumulation of people in l at t_c, which might not be obvious in individual time-steps. Next, we define the recent profile of $\mathbf{x}$.

Definition 5 ***Recent profile*** *of* (l, t_c) *is defined as:*

$$N^l_{t_c} = \langle C^p_{l,t_c-\tau}, C^d_{l,t_c-\tau}, ..., C^p_{l,t_c}, C^d_{l,t_c} \rangle \tag{5}$$

where τ *is a parameter.*

The recent profile contains all the pick-up and drop counts of the recent τ time-steps at current location. We define the target profile as follows:

Definition 6 ***Target profile*** *of* (l, t_c) *is defined as:*

$$G^l_{t_g} = \langle B^p_{l,t_c+1}, ..., B^p_{l,t_g} \rangle. \tag{6}$$

where $(t_c, t_g]$ *is the target period, i.e. the time period for which we are going to make predictions.*

The target profile is the expected pick-up counts of the prediction target time period in the future. We define the anomaly profile as follows:

Definition 7 ***Anomaly profile*** *of* (l, t_c) *is defined as:*

$$F^l_{t_g} = \langle LLR(l, t_c + 1), ..., LLR(l, t_g) \rangle. \tag{7}$$

where $(t_g - 1, t_g]$ *is the target period, same as Definition 6.*

The anomaly profile is consisted of the anomaly scores of l during the target period based on Eq. 1. These values are available during training, but not during testing. We will use predicted anomaly scores instead, while testing.

Building the Training Sets

As in Fig. 2, we train three estimators: survival function estimator (f_s), anomaly profile estimator (f_a) and dispersal event pick-up predictor (f_e). In this section, we describe how their training sets are obtained. We propose to use estimators that are maintain an internal state, such as recurrent neural networks. Thus, the order of instances in the training set matters. This order must match the order of real-time data. Ensuring this requirement is straightforward for f_a and f_s, since they are trained using all the instances. However, it is not straightforward for f_e, because it is not trained on all instances. Later in this section, we will demonstrate how this requirement is satisfied.

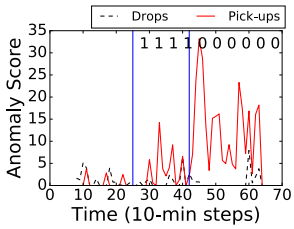


Figure 3: An example of the values of the survival function $S(.)$ in case of a dispersal event.

As mentioned earlier, we treat the dispersal event prediction problem as a survival problem. Therefore, the output vector for f_s is the survival probabilities. In this case, the dispersal event is the death event in the survival problem. To this end, the survival function is defined as follows:

$$S(t) = Pr(E^p > t) \tag{8}$$

where E^p is the time of dispersal event. In our proposed framework, we train a model to predict $S(t)$. For location l and time t_c, we use the following input vector for f_s:

$$\mathbf{x}_s = \langle Q^l_{t_c}, W^l_{t_c}, M^l_{t_c}, G^l_{t_g}, F^l_{t_g}, V^l, N^i_{t_c}, S(t_c) \rangle, i \in l^*. \tag{9}$$

We call l^* the surrounding area of $l = (a, b)$ defined as the rectangular area bounded by grid cells $(a - \lambda, b - \lambda)$ and $(a+\lambda, b+\lambda)$, where λ is a parameter. Input vector $\mathbf{x}_s$ consists of time, weather, daily, target and anomaly profiles and the POI vector of (l, t_c) and the recent profile of (l^*, t_c). $\mathbf{x}_s$ plus the current value of the survival function $S(t_c)$.

For each input vector $\mathbf{x}_s$ at location l and time t_c, we use the following output vector:

$$\mathbf{y}_s = \langle S(t_c + 1), ..., S(t_g) \rangle. \tag{10}$$

Ideally, we would like to have a labeled event list for our training phase. However, such lists are not available. Therefore, we use an algorithm to obtain $S(.)$ for a given time and location (l, t_c) by determining if any dispersal event has occurred, or is underway in the future of t_c. This procedure is presented in Alg. 1. We put a limit on the length of a dispersal event, assuming the events that are shorter than e_{min} or longer than e_{max} are not interesting. Then, we test every sub-period between $t_c - e_{max}$ and t_g that are longer than e_{min}, using Def. 1. The survival value will be set to one before and zero after the start of the dispersal event. For example, consider Fig. 3, which shows the dispersal event of Fig. 1 (b). The first vertical line is the current time, the second vertical line is the starting time of the dispersal event. The survival function is set to 1 before the start of the event and is set to zero afterwards. Alg. 1 calculates the survival function. $(t_c - e_{max}, t_c + t_g)$ has exponential number of sub-periods. However, we are only interested in the earliest dispersal event, because the survival function will

be zero afterwards. Alg. 1 takes advantage of this fact and runs in $O(nm)$, where n is the length of time being searched ($end - start$) and m is the number of different lengths sub-periods can have ($e_{max} - e_{min}$).

$\mathbf{x}_s$ and $\mathbf{y}_s$ are obtained for every spatio-temporal grid cell in Z. They constitute the training set for $\mathbf{y}_s = f_s(\mathbf{x}_s)$ for estimating the survival function. We will discuss how we use f_s to predict the start time of dispersal events.

Algorithm 1: Calculate survival function (get_St)

Input: Baselines and counts, current location l, current time t_c, target time t_g
Output: Survival function $S(t)$ where $t \in (t_c, t_g]$

1 $S(.) \leftarrow \{1\}$; $start \leftarrow t_c - e_{max}$; $end \leftarrow t_g$
2 **for** k *from* e_{max} *to* e_{min} **do**
3 **for** $t_0 \in [start, end - k]$ **do**
4 $t_1 \leftarrow t_0 + k$
5 **if** $LLR(l, [t_0, t_1])$ *is significant and* $t_1 > t_c$ **then**
6 **for** $t \in [max(t_0, t_c), t_g]$ **do**
7 $S(t) \leftarrow 0$
8 **return** $S(.)$
9 **return** $S(.)$

Although anomaly profile ($F^l_{t_g}$) values are available during training, they are not available during testing, because we do not have the true pick-up counts in the future. While, we train f_s using the true anomaly profile, we have to use predicted anomaly profile in the prediction phase. We use f_a to predict the anomaly profile. The input vector of f_a is denoted as $\mathbf{x}_a$ and is shown as follows:

$$\mathbf{x}_a = \langle Q^l_{t_c}, W^l_{t_c}, M^l_{t_c}, G^l_{t_g}, F^l_{(t_c-\tau,t_c]}, V^l, N^i_{t_c} \rangle, i \in l^*. \quad (11)$$

Where $F^l_{(t_c-\tau,t_c]}$ is the anomaly profile in the recent time period. Eq. 11 means that we use the time, weather, daily, recent profiles and the POI vector, in addition to recent anomaly values to predict the future anomaly profile.

Next, we predict the pick-up counts in case of dispersal events, using estimator f_e. We use an input vector with the same elements as $\mathbf{x}_s$. The output vector of f_e is as follows:

$$\mathbf{y}_e = \langle C^p_{l,t_c+1}, ..., C^p_{l,t_g} \rangle \quad (12)$$

Although f_s and f_e have the same feature vectors, they are not trained with the same sets. This is a key point in our proposed approach. In the training set of f_e we only include the data instances which correspond to a dispersal event. The reason is we will only use f_e to predict the pick-up counts in case of abnormally high pick-up counts. Thus, we train it with just those instances. Alg. 2 builds the training set for f_e. To make sure f_e learns a full cycle of a dispersal event in its internal state, for each event, we include all the instances starting from the time when the event is first observed in the target period (line 5). For example, let t be current time and the target period be 4 time-steps long. If the survival function is $\langle 1, 1, 1, 0 \rangle$, then the instances of time-steps $[t, t + 4)$ will be included in the training set (lines 6-9).

Algorithm 2: Build dataset for f_e (get_Xe)

Input: Baselines and counts, time, weather, daily, recent, target and anomaly profiles, POI vectors, $Z = (S, T)$
Output: Training set $\mathbf{X}_e$ and $\mathbf{Y}_e$ for f_e

1 $\mathbf{Y}_e$ = An empty list; $\mathbf{X}_e$ = An empty list
2 **for** $l \in S$ **do**
3 **for** $t \in T$ **do**
4 $S(t)$ = get_St(l,t)
5 **if** $S(t_g) = 0$ *AND* $S(t_g - 1) = 1$ **then**
6 **for** $t_c \in [t, t_g]$ **do**
7 $\mathbf{x}_e$ = $(Q^l_{t_c}, W^l_{t_c}, M^l_{t_c}, G^l_{t_g}, F^l_{t_g}, V^l, N^i_{t_c}), i \in l^*$
8 $\mathbf{y}_e$ = $(C^p_{l,t_c+1}, ..., C^p_{l,t_g})$
9 $\mathbf{X}$.push_back($\mathbf{x}_e$); $\mathbf{Y}$.push_back($\mathbf{y}_e$)
10 $t = t_g$
11 **return** $\mathbf{X}_e$, $\mathbf{Y}_e$

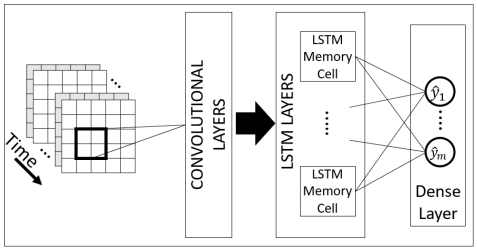


Figure 4: Deep Learning structure used to learn spatial and temporal dependencies.

DILSA: Dispersal Event Prediction using Survival Analysis

The training sets built in the previous section contain temporal and spatial dependencies. Thus, we use a Deep Artificial Neural Network that uses Convolutional layers to capture spatial dependencies and LSTM layers to capture temporal dependencies. Fig. 4 shows the employed structure.

First step in our framework is to obtain the anomaly profile of the target period using f_a, to be used in the input vector of f_s. Then, $1 - S(.)$ is the estimated cumulative probabilities of event. Assuming $S(0) = 1$, we calculate the probability of event at future time using the hazard function:

$$H(t) = \frac{S(t-1) - S(t)}{S(t)}. \quad (13)$$

Eq. 13 calculates the cumulative hazard of event happening between $t - 1$ and t given that it has not happened as of $t - 1$. This value is calculated by dividing the amount of drop in the survival function from time $t-1$ to t, by the total remaining amount, which is $S(t)$, given that $S(.)$ monotonically decreases. We predict an event, when value of $H(.)$ exceeds a threshold γ, which is tuned using a tuning set.

Once a dispersal event is predicted, we predict the pick-up count for the event using f_e. Since our estimators maintain

an internal state, we must make predictions in the same order as training. This is not a problem for estimators f_a and f_s, because they were trained using all the instances, which is the same order of real-time data. To train f_e, Alg. 2 establishes a specific order that must also be followed in the prediction phase. In the training phase, we included instances when the start of the event first appears in the target period, i.e. the survival function turns to 0 in the last time-step of the target period ($S(t_g) = 0$ and $S(t_g - 1) = 1$). Therefore, we must start predicting the pick-ups using f_e once Eq. 13 predicts the last time-step of the target period to be 0. However, Eq. 13 might not predict the occurrence of the event until the start time gets closer. In such a case, f_e will not have correct internal state. Therefore, to bring f_e to its correct internal state, we feed the input vectors of previous time-steps to f_e before the input vector of current time. For example, suppose we are at time t_c and target time period is 4 time-steps long. Then we predict a dispersal event at time $t_c + 2$. For f_e to make predictions for $t_c + 2$ and $t_c + 3$, we feed the input vectors of time $t_c - 2$, then $t_c - 1$ to f_e. Now f_e has the correct internal state to make predictions.

Alg. 3 shows the proposed dispersal event demand predictor. First, the anomaly profile is obtained and used to predict the survival function (lines 1-2). Then $H(.)$ is calculated for future periods and compared with threshold γ to predict the dispersal events (lines 4-9). A value of 1 in $\hat{\mathbf{y}}_s[t] = 1$ means a dispersal event is predicted for t time-steps after current time. In case of a predicted event, the internal state of f_e is corrected and pick-up counts are predicted (line 10-13).

Algorithm 3: Dispersal event predictor (DILSA)

Input: Estimators $f_s(.)$, f_a and $f_e(.)$, current time t_c, target time t_g, threshold γ
Output: Predicted dispersal events $\hat{\mathbf{y}}_s$, predicted counts of the predicted events $\hat{\mathbf{y}}_e$

1 **for** $l \in S$ **do**
2 $\hat{\mathbf{y}}_s[l] = \{0\}$; $\hat{\mathbf{y}}_e[l] = \{-1\}$; $\mathbf{x}_a$=construct_$\mathbf{x}_a(l,t_c)$
3 $\hat{\mathbf{F}}$=$f_a(\mathbf{X}_a)$; $\mathbf{x}_s$=construct_$\mathbf{x}_s(l,t_c,\hat{\mathbf{F}})$; $St = f_s(\mathbf{x}_s)$
4 is_event=**False**
5 **for** $i \in [1, t_g - t_c)$ **do**
6 $H = (S(i-1) - S(i))/S(i)$
7 **if** $H \geq \gamma$ **then**
8 **for** $j \in [i, t_g)$ **do**
9 $\hat{\mathbf{y}}^s_j[l] = 1$
10 is_event=**True**; event_time=i; break
11 **if** *is_event* **then**
12 Correct the internal state of f_e
13 $\mathbf{x}_e$=construct_$\mathbf{x}_e(l,t_c,\hat{\mathbf{F}})$
14 $\hat{\mathbf{y}}_e[l] = f_e(\mathbf{x}_e)$
15 Return $(\hat{\mathbf{y}}^s, \hat{\mathbf{y}}^e)$

Evaluations

Settings and Baseline Solutions

We use the trip records of Yellow Taxis in New York City from years 2014, 2015 and 2016. This dataset contains the pick-up and drop locations and is released by New York City Mayor's Office [1]. The weather data is obtained from the National Centers for Environmental Information [2] from two weather stations, Central Park and the La Guardia Airport. The Point of Interest data is obtained from Google Maps Places API [3], which assigns POIs into one or more of 129 categories. We partition the New York City area into a grid of 32×32 with cell size of 400×400 meters. We use 30-minute time-steps. Every record is mapped into the grid to obtain counts and baselines. The values of weather profile for each spatio-temporal grid cell is an average of the measurements reported by the two stations, weighted inversely by their distance. We train the models using year 2014 and evaluate on 2015 and 2016. All datasets are standardized by subtracting the minimum and dividing by the maximum value of each feature. The test sets are standardized using parameters from the training set. Table 1 shows our parameter settings.

Table 1: Parameter settings.

λ	α	$t_g - t_c$	τ	e_{min}	e_{max}
4	0.001	5 hrs	5 hrs	30 min	5 hrs

In table 1, $t_g - t_c$ is the duration of the target period. Our Deep Learning Network uses 4 convolutional layers with window size of 9×9, 2 LSTM layers of 69 memory cells and 10 output nodes. We compare f_e with state-of-the-art deep learning method for taxi pick-up prediction, **DMVST-Net** (Yao et al. 2018). Moreover, we use three additional baselines for comparison. First baseline is simple thresholding of the survival function instead of Eq. 13, i.e. if the survival function drops below a threshold (σ), the event is predicted. We call this baseline **DIL**. We tune both γ and σ, using a week's data in 2015 ($\gamma = 2.95$ and $\sigma = 0.1$). We also compare with Multi-Layer Perceptron (**MLP**) and Logistic Regression (**LgR**) models.

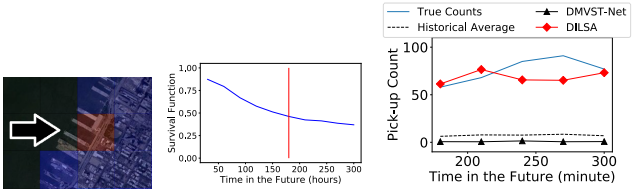
The estimators were trained using the stochastic gradient descent method proposed by (Kingma and Ba 2014), with 20 epochs for f_a and f_s and 40 epochs for f_e.

Case Studies

We apply the proposed method to a full dataset from 2016. Here, we present two of the predicted events.

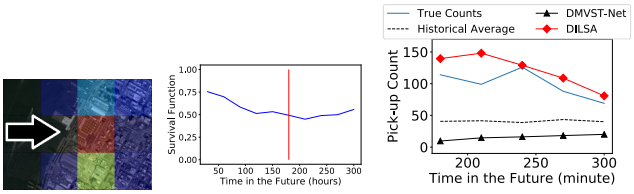
On March 19^{th}, 2016, we predicted a dispersal event at 1:00 PM around an exhibition center in Pier 92/94 in Manhattan. We predict the event 2.5 hours before (at 11:30 AM). Public records show a home design exhibition at the time [4]. Fig. 5 (b) shows the predicted survival curve at 11:30 AM. The red vertical line is the predicted time of the dispersal event, which is inferred by Alg. 3. Fig. 5 (c) shows the predicted counts by the baseline and the proposed method. The proposed method successfully predicts the increase, while DMVST-Net stays close to the historical average.

[1] https://opendata.cityofnewyork.us/overview/
[2] https://www.ncei.noaa.gov/
[3] https://developers.google.com/places/
[4] https://architecturaldigest.com/story/architectural-digest-design-show-video



(a) Event location. (b) Case 1 survival curve. (c) Predicted pick-up counts vs. true counts.

Figure 5: First case study (best viewed in color).



(a) Event location. (b) Case 2 survival curve. (c) Predicted pick-up counts vs. true counts.

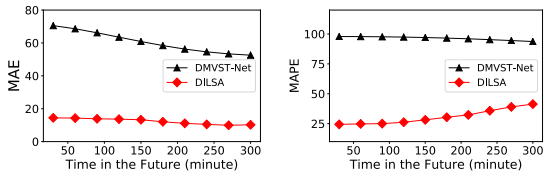
Figure 6: Second case study (best viewed in color).

We also predicted a dispersal event around 12:30 PM on June 26^{th}, 2016, at Jacob K. Javis Convention Center, 2.5 hours before. Public records show there was a food show at the convention center [5]. Fig. 6 (b) shows the predicted survival curve and the event prediction time, indicated by the vertical red line. Fig. 6 (c) shows the proposed method outperforms DMVST-Net in predicting the pick-up counts in this case. Fig. 5 (a) and 6 (a) show heatmaps of LLR scores based on the true counts in the predicted periods. The black arrows show the verified locations. The figures show a clear hotspot of pick-ups. Overall, these two case studies demonstrate examples of DILSA successfully predicting dispersal events and their corresponding demand.

Experiments

In this section, we first evaluate the prediction performance of DILSA, i.e. the performance of Alg. 3 to predict events. We compare our results with four baselines. Baseline DMVST-Net predicts taxi demand. We apply Def. 1 to the predicted value to determine if there is a dispersal event. Table 2 shows that DILSA out-performs all the baselines in terms of F1-score (0.7) and time error (18 minutes). Time error is the average difference between the true start time and predicted start time of the correctly predicted events. A prediction is considered a true positive if the predicted event period overlaps with the true event period. The results show the proposed survival analysis method predicts dispersal events with high accuracy. Although DMVST-Net demonstrates high precision, its recall is extremely low, meaning the regular patterns fail to predict accurately in case of abnormally high demand. Moreover, the results show using the cumulative hazard function of Eq. 13 in Alg. 3 has a considerable impact on model's performance.

[5] https://specialtyfood.com/news/article/2016-summer-fancy-food-show-largest-ever/



(a) MAE. (b) MAPE.

Figure 7: Performance of the proposed pick-up counts predictor vs. baselines, on events.

Second, we compare the demand predictor f_e to DMVST-Net in case of dispersal events. The baseline was trained on the same period as the previous experiment. f_e was trained on the dataset obtained using Alg. 2 on the same period of time in 2014. Fig. 7 shows Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) in future time-steps. Fig. 7 shows our proposed method out-performs the baseline in case of a dispersal event. This experiment shows methods proposed to capture the regular pattern of taxi demand are not reliable in case of dispersal events.

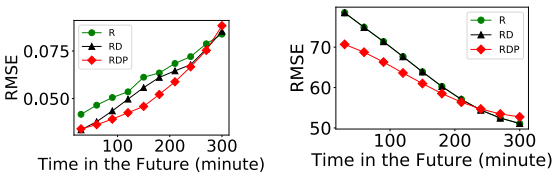
Table 2: Performance comparison, DILSA vs. baselines.

	DILSA	DIL	DMVST-Net	MLP	LgR
Precision	0.6	0.6	0.9	0.5	0.3
Recall	**0.9**	0.8	0.04	0.6	0.3
F1-score	**0.7**	0.7	0.08	0.5	0.3
Time error (min.)	**18.6**	29.1	60	59.6	80.9

Lastly, we evaluate the impact of different features on the performance of the models. We use Root Mean Squared Error (RMSE) as the measure. The x-axis represents future time-steps. The letters R, D and P represent the Recent and Daily profiles and the POI vector. The results show including the POI vector reduces the error. Including the daily profile does not have a significant effect on f_e while improves the performance of survival function predictor f_s.

Conclusions

In this paper we solved the problem of predicting dispersal events where a large number of people leave the same area in a short period. Predicting such events has managerial and business value for various stakeholders. We solved the problem as an abnormally high demand prediction problem. The



(a) Survival function predictor. (b) Abnormal demand predictor.

Figure 8: Impact of choice of features on accuracy.

taxi demands in unexpected dispersal events deviate from regular patterns and violate assumptions made by previous techniques (e.g., auto-correlation, periodic). In this paper we argued that dispersal events follow a complex pattern of trips and other related features. We formulated and learned such patterns to predict dispersal events. We formulated the dispersal event prediction as a survival analysis problem and proposed a two-stage framework (DILSA), where a supervised model predicted the probability of "death", i.e., the dispersal event. The demand was then predicted in case of a predicted event. We conducted extensive case studies and experiments on a real dataset from 2014-2016. Our method out-performed the baselines and predicted dispersal events with F1-score of 0.7 and time error of 18 minutes.

Acknowledgments

This work is partially supported by the NSF under Grant Number IIS-1566386. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. Yanhua Li is partly supported by NSF grant CNS-1657350, CMMI-1831140, and an industrial grant from DiDiChuxing Research.

References

Chen, F., and Neill, D. B. 2014. Non-parametric scan statistics for event detection and forecasting in heterogeneous social media graphs. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 1166–1175. ACM.

Davis, N.; Raina, G.; and Jagannathan, K. 2016. A multilevel clustering approach for forecasting taxi travel demand. In *Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on*, 223–228. IEEE.

Hoang, M. X.; Zheng, Y.; and Singh, A. K. 2016. Fccf: forecasting citywide crowd flows based on big data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 6. ACM.

Hong, L.; Zheng, Y.; Yung, D.; Shang, J.; and Zou, L. 2015. Detecting urban black holes based on human mobility data. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 35. ACM.

Khezerlou, A. V.; Zhou, X.; Li, L.; Shafiq, Z.; Liu, A. X.; and Zhang, F. 2017. A traffic flow approach to early detection of gathering events: Comprehensive results. *ACM Transactions on Intelligent Systems and Technology (TIST)* 8(6):74.

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

Kulldorff, M.; Heffernan, R.; Hartman, J.; Assunçao, R.; and Mostashari, F. 2005. A space-time permutation scan statistic for disease outbreak detection. *PLoS medicine* 2(3):216.

Kulldorff, M. 1997. A spatial scan statistic. *Communications in Statistics-Theory and methods* 26(6):1481–1496.

Li, Z.; Xiong, H.; and Liu, Y. 2012. Mining blackhole and volcano patterns in directed graphs: a general approach. *Data Mining and Knowledge Discovery* 25(3):577–602.

Liu, Y.; Zhou, B.; Chen, F.; and Cheung, D. W. 2016. Graph topic scan statistic for spatial event detection. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 489–498. ACM.

Miller Jr, R. G. 2011. *Survival analysis*, volume 66. John Wiley & Sons.

Moreira-Matias, L.; Gama, J.; Ferreira, M.; Mendes-Moreira, J.; and Damas, L. 2013. Predicting taxi–passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems* 14(3):1393–1402.

Mukai, N., and Yoden, N. 2012. Taxi demand forecasting based on taxi probe data by neural network. In *Intelligent Interactive Multimedia: Systems and Services*. Springer. 589–597.

Neill, D. B. 2009. Expectation-based scan statistics for monitoring spatial time series data. *International Journal of Forecasting* 25(3):498–517.

Street, W. N. 1998. A neural network model for prognostic prediction. In *ICML*, 540–546.

Vahedian, A.; Zhou, X.; Tong, L.; Li, Y.; and Luo, J. 2017. Forecasting gathering events through continuous destination prediction on big trajectory data. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL'17, 34:1–34:10. New York, NY, USA: ACM.

Xu, J.; Rahmatizadeh, R.; Bölöni, L.; and Turgut, D. 2017. Real-time prediction of taxi demand using recurrent neural networks. *IEEE Transactions on Intelligent Transportation Systems*.

Yao, H.; Wu, F.; Ke, J.; Tang, X.; Jia, Y.; Lu, S.; Gong, P.; Ye, J.; and Li, Z. 2018. Deep multi-view spatial-temporal network for taxi demand prediction. In *2018 AAAI Conference on Artificial Intelligence (AAAI'18)*.

Zhang, K.; Feng, Z.; Chen, S.; Huang, K.; and Wang, G. 2016. A framework for passengers demand prediction and recommendation. In *Services Computing (SCC), 2016 IEEE International Conference on*, 340–347. IEEE.

Zhang, C.; Liu, L.; Lei, D.; Yuan, Q.; Zhuang, H.; Hanratty, T.; and Han, J. 2017. Triovecevent: Embedding-based online local event detection in geo-tagged tweet streams. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 595–604. ACM.

Zhao, K.; Khryashchev, D.; Freire, J.; Silva, C.; and Vo, H. 2016. Predicting taxi demand at high spatial resolution: Approaching the limit of predictability. In *Big Data (Big Data), 2016 IEEE International Conference on*, 833–842. IEEE.

Zhou, X., and Chen, L. 2014. Event detection over twitter social media streams. *The VLDB journal* 23(3):381–400.

Zhou, X.; Khezerlou, A. V.; Liu, A.; Shafiq, Z.; and Zhang, F. 2016. A traffic flow approach to early detection of gathering events. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 4. ACM.