Truth Discovery for Spatio-Temporal Events from Crowdsourced Data

Daniel A. Garcia-Ulloa
Math & CS Department
Emory University
dgarci8@emory.edu

Li Xiong
Math & CS Department
Emory University
lxiong@emory.edu

Vaidy Sunderam
Math & CS Department
Emory University
vss@emory.edu

ABSTRACT

One of the greatest challenges in spatial crowdsourcing is determining the veracity of reports from multiple users about a particular event or phenomenon. In this paper, we address the difficulties of truth discovery in spatio-temporal tasks and present a new method based on recursive Bayesian estimation (BE) from multiple reports of users. Our method incorporates a reliability model for users, which improves as more reports arrive while increasing the accuracy of the model in labeling the state of the event. The model is further improved by Kalman estimation (BE+KE) that models the spatio-temporal correlations of the events and predicts the next state of an event and is corrected when new reports arrive. The methods are tested in a simulated environment, as well as using real-world data. Experimental results show that our methods are adaptable to the available data, can incorporate previous beliefs, and outperform existing truth discovery methods of spatio-temporal events.

1. INTRODUCTION

Spatial Crowdsourcing or Mobile Crowdsourcing refers to a variety of data collecting models in which individuals with sensing or computing devices are tasked to report and share information for a common good [17]. There has been a recent increase in the amount of social sensing activities due to the proliferation of smart devices with sensors. One of the first works on social sensing is [2], which presented different applications in urban planning, public health, cultural expression, and natural resource management. As more applications for social sensing arise, so does the need to determine the veracity of the reports.

Determining the accuracy of reports in spatial crowdsourcing is particularly difficult if the reports correspond to events that might change over time. Sources often present uncertain or even contradicting reports which can result in considerable loss of accuracy [1]. In general, crowdsourcing users may not visit all target locations, and even if they do, may not necessarily report the state of an event. Simply ignoring the missing reports has repercussions for the accuracy of any truth-discovery method [7, 10].

Problem Setting. Truth discovery or fact-finding [21, 12] seeks to integrate data from different sources and determine the true values of target events. Truth-finding has gained attention due to the wide variety of possible implementations and the increasing interest in crowdsourcing applications. We consider a set of spatio-temporal events which can be constantly changing from a true state to false state and vice versa, while a set of users report the state of these events at different times. Our goal is to determine or estimate the true state of the event at each time point based on the users' reports.

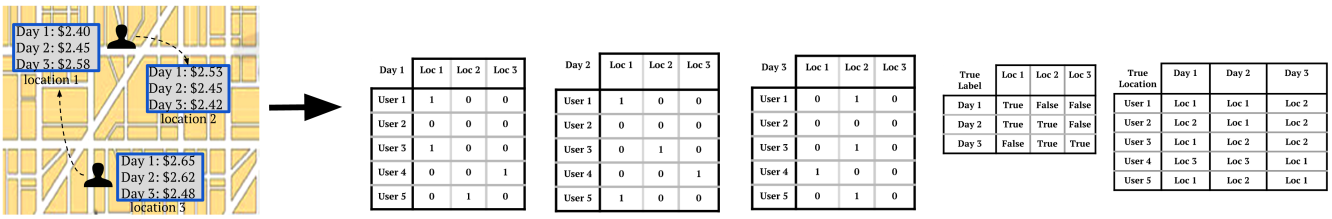
Consider a running example as shown in Figure 1. Users could be asked to report if the gas price at specific gas stations is below $2.50 a gallon. The reports of the users will depend on whether they visit the specific location at the specific time of the event as well as their reliability. A passive user could be at the specific location but decides not to send a report, while a malicious user could send a report about a location different from their own. Additionally, seemingly inconsistent reports for a single event could be correct if they were made at different times. The price of gas at a gas station could be below $2.50 one day and above $2.50 the next. Figure 1(a) shows this example for location 1 between Day 2 and Day 3. Another motivating example is reporting whether there is high traffic at different locations, the way it is done through the app Waze[1]. In this scenario, the traffic in one location could be correlated to traffic in another location. In sections 3 and 7 we discuss other applications under the scenario described above, and how this analysis can be extended to include more states.

Existing solutions and limitations. While there have been many recent studies on truth discovery in various crowdsourcing applications (e.g. crowdsourced labeling of online websites [5]), most of them [25, 10] do not consider the sources of data to be mobile, or do not consider that the events could change their state over time. Few methods [14, 3] consider spatial events but do not handle streaming data, and would need to re-run the algorithm each time new data arrives. Other methods [13, 24] handle streaming data and changing truth, as in the case of weather and stock prices,

Proceedings of the VLDB Endowment, Vol. 10, No. 11

[1] http://www.waze.com

Research supported in part by AFOSR grant FA9550-12-1-0240 and NSF grant CNS-1618932



(a) Users moving around locations and sending reports

Day 1	Loc 1	Loc 2	Loc 3
User 1	1	0	0
User 2	0	0	0
User 3	1	0	0
User 4	0	0	1
User 5	0	1	0

Day 2	Loc 1	Loc 2	Loc 3
User 1	1	0	0
User 2	0	0	0
User 3	0	1	0
User 4	0	0	1
User 5	1	0	0

Day 3	Loc 1	Loc 2	Loc 3
User 1	0	1	0
User 2	0	0	0
User 3	0	1	0
User 4	1	0	0
User 5	0	1	0

(b) Reports from users (observable).

True Label	Loc 1	Loc 2	Loc 3
Day 1	True	False	False
Day 2	True	True	False
Day 3	False	True	True

True Location	Day 1	Day 2	Day 3
User 1	Loc 1	Loc 1	Loc 2
User 2	Loc 2	Loc 1	Loc 2
User 3	Loc 1	Loc 2	Loc 2
User 4	Loc 3	Loc 3	Loc 1
User 5	Loc 1	Loc 2	Loc 1

(c) True labels of the events and true locations of the users (not observable).

Figure 1: Example of spatio-temporal crowdsourced task. Users send true reports ('1') if the price at different locations is less than $2.50. Based on the reports, our goal is to determine the true labels of the events.

but do not consider the location of the sources and their mobility.

Contributions. In this paper, we present the challenges that arise in truth-discovery of spatio-temporal events from crowdsourcing tasks and propose an iterative method based on a graphical model that maximizes the probability of correctly assessing the state of the events, given the reported data from users. The method also yields a reliability model for each user that determines their probability of correctly reporting the state of an event in the future. We further enhance our method with a model based on the Kalman filter, which incorporates a prediction and correction model of the state of the events and can incorporate historical data. We summarize our contributions below.

1. We present a dynamic graphical model that describes the dependencies of the hidden (true labels of the events, reliability of the users) and observed variables (reports from moving users) in a spatio-temporal setting. We present a recursive Bayesian estimation (BE) method for training the parameters for inferring the true state of the events. Our method incorporates a reliability model for users, which improves as more reports arrive while increasing the accuracy of the model in labeling the state of the event.
2. We further enhance the graphical model with an event model that explicitly describes the spatio-temporal correlations between the events and present a Kalman Filter based approach (BE+KE) for improved inference of the true states.
3. We perform experimental validation of the model and algorithms using simulated and real-world data. The experimental results show that our methods are adaptable to the available data, can incorporate previous beliefs, and outperform existing truth discovery methods of spatio-temporal events from crowd sourced data.

2. RELATED WORK

Truth discovery methods in spatial crowdsourcing can be classified into iterative, optimization-based, and probabilistic graphical models, although overlaps are possible [12]. In the case of iterative methods, Dong et. al. [7] present a novel method that consider a possible relation between the sources where the value provided by one source is copied by other sources. They use an iterative Bayesian approach to determine these dependencies and infer the true value. Truthfinder [22] presents an approach to use the relationship between different facts and the reliability of the sources of information to determine the true facts from a large amount of conflicting data. An optimization method is provided in [13] where they develop an optimization problem to infer both source reliability and trustworthiness of the information. A probabilistic graphical model can be found in Zhao et.al [23], where the authors proposed an unsupervised Bayesian method that takes advantage of the generation process of false positives and false negatives to infer the true records and the source quality from different databases.

With respect to spatio-temporal methods, Wang et.al. [19] proposed an expectation-maximization algorithm that is specifically interested in short-lived crowdsourcing campaigns, where there is no time to build a reputation for the users. Their applications on social sensing deal with events that usually do not vary much over time (e.g. open gas stations after a disaster). Wang et.al [21] have developed an algorithm based on expectation maximization to determine which observations were correct and which ones were not, with no prior knowledge of the source reliability nor about the correctness of prior individual observations. A posterior work [20] includes a sensitivity analysis of the optimal expectation - maximization estimator to understand the accuracy trade-offs in source and claim credibility in social sensing applications. Another approach for dealing with spatial data is the Truth for Spatial Events algorithm (TSE) [15], where not only the truthfulness and reliability of the sources is determined, but also the probability that the users visited locations to improve the accuracy. Since there are some similarities between this work and ours (although our work differs from TSE since we also consider that the state of the events are changing as a function of time), TSE is one of the methods we use to compare with our own.

The Kalman filter is a recursive algorithm that uses the available, noisy observations and produces an estimate for the current, hidden state [18, 11]. It can run in real time and consists of two phases: prediction and correction. In the prediction phase, it uses the observations from previous time-steps as input for a system model to predict the state of the system. In the correction phase, the filter trains and uses a parameter (called Kalman gain) to combine the prediction from the previous phase, and the current observations. The Kalman filter has been successfully implemented in applications such as simultaneous localization and mapping, and monitoring web browsing behavior [4, 9].

3. PROBLEM FORMULATION

Consider the spatial crowdsourcing scenario, in which a set of n_{users} different users join the task of reporting specific events at n_{locs} different locations and at n_{times} different and

consecutive time slices. We denote the set of these users as $U = \{u_i | i = 1, ..., n_{users}\}$, the set of locations of interest as $L = \{l_j | j = 1, ..., n_{locs}\}$ and the set of relevant times as $T = \{t_k | k = 1, ..., n_{times}\}$. At any time $t_k \in T$, an event in location $l_j \in L$ could be true (denoted by '1') or false (denoted by '0'). Let Z be the set of events, where each element $z_{jk} = 1$ iff the event at time t_k and location l_j is true. We present our problem and solutions for events with a binary state here, but we note that they can be extended to events with multiple states.

For example, a set of $n_{users} = 5$ users could be asked to report if there is high traffic at $n_{locs} = 10$ different street intersections during the $n_{times} = 24$ hours of a certain day, or they could be asked to report if gas is lower than \$2.50 at $n_{locs} = 3$ different gas stations for $n_{times} = 3$ days. The price at location 1 for the three days could be \$2.40, \$2.45, and \$2.58 respectively (Figure 1(a)), so the true states of the event are $z_{11} = 1, z_{12} = 1$, and $z_{13} = 0$ for location 1.

The users would simply report '1' if the event is true. We assume there is a default state (e.g. gas price is \$2.50 or more, light traffic) and consider therefore that the reports could be of two kinds: 1) *Positive report*, where the user reports the event as true (denoted as '1'); and 2) *Missing report* where the user did not send a report (denoted as '0'). Missing reports could occur in a variety of circumstances, such as the user not being at the specific place and time, lack of participation from the user, or the event being false. In other words, a missing report could mean both absence of report or report of a negative state. To distinguish between these scenarios, we take into consideration both the probability that a user was at the specific time and place of the event and the reliability traits of the user, although these parameters are not known beforehand. Our main goal is to determine the label of z_{jk} as true or false for all of the relevant locations and times.

Figure 1 shows an example of a spatio-temporal crowdsourced task. Users are moving and report if an event is true at different locations of interest. The event could be gas price below \$2.50. Figure 1(b) shows an example of received reports throughout three different days. Based on these reports and without knowledge of the users' true locations, the goal is to infer the true label of the events over time (figure 1(c)). We consider different types of users: 1) trustworthy users (e.g. user 1 and 3) who report the events true when they are true; 2) passive users (e.g. user 2) who never send reports despite being in the correct location and time; 3) untrustworthy users (e.g. user 4) who send the wrong reports; and 4) malicious users (e.g. user 5) that send reports about events they are not observing.

Our assumption of a default state is motivated by a variety of applications where users normally do not report a negative state. For example, users do not normally send a report when there is no traffic, since that is the default state. Other examples with a default state include sending a report when there is graffiti on a wall, reporting potholes or trash on the street, locations with free Wi-Fi, and accidents on a road.

Our model can be extended to include multiple states (e.g. report on low, medium, or high traffic), discretize continuous variables (e.g. the price of gas is either below \$1.50, between \$1.50 and \$3, or higher than \$3), or have a "negative" report (e.g. there is no accident at this location). We further discuss these scenarios in section 7.

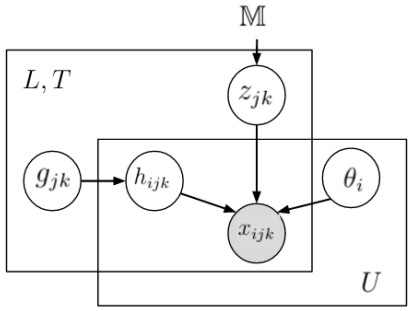


Figure 2: Graphical Model showing the dependency of the variables. U, L, and T are the set of Users, Locations, and Time steps, respectively

Table 1: Notation for the Graphical Model

Symbol	Meaning
n_{users}, n_{locs}, n_{times}	Number of users, event locations, and time slices, respectively
U, L T	Set of Users, Locations, and Time slices, respectively
$\mathbb{M}$	Event model for the events Z
z_{jk}	Binary variable with the state of the event at location l_j and time t_k
x_{ijk}	Binary variable with the report of user u_i at location l_j at time t_k
h_{ijk}	Binary variable that indicates if user u_i was at location l_j at time t_k
g_{jk}	Popularity of location l_j at time t_k
θ_i	Reliability model for user u_i

4. PROPOSED MODEL

In this section we present our proposed graphical model that specifies the dependence at a given time point between the report of a user, the true state of the event at a given location, and other factors. Figure 2 shows our graphical model and Table 1 summarizes its notations. The model consists of the following elements:

Reports from the users. The reports from the users are collected in the form of a three dimensional binary variable $X = \{x_{ijk}\}$, where $x_{ijk} = 1$ iff user u_i reported the event at location l_j and time t_k as true. Since a user u_i cannot be in two locations at a same time t_k, $x_{i\mathbf{j}k} = 1$ for some $\mathbf{j} \in \{1, ..., n_{locs}\} \Rightarrow x_{ijk} = 0 \quad \forall j \neq \mathbf{j}$.

On the other hand, $x_{ijk} = 0$ could be because the user u_i was not at location l_j at time t_k, the event z_{jk} was not true, or because the user decided not to participate. Our method establishes a probability model to explain the different possibilities, which are detailed in Section 4.

Label of the event. The label of the event at location l_j and time t_k is denoted as z_{jk} and it is a latent binary variable and one we ultimately seek to estimate based on the reports from the users. We assume that there is an *event model* $\mathbb{M}$ that models the spatio-temporal correlations between the event states at consecutive time slices and determines the value of z_{jk}. In Section 5.1, we will assume that $\mathbb{M}$ is not known and present a method based on Bayesian Estimation, and then we present an enhanced method that explicitly specifies and estimates $\mathbb{M}$ in Section 5.2.

Popularity of the locations. One of the factors that influences the reports from the users is if the user is actually at the specified location. Establishing the popularity of each place will be important to establish the probability that a user was actually at a determined location at a certain time. The popularity of a place is defined as the probability that a user chosen at random will be at that place at any time. The popularity usually follows a power law distribution, so that roughly 20% of the places have 80% of the popularity [15]. Let g_{jk} be the popularity of location l_j at time t_k. For certain applications, and depending on the time periods, it is reasonable to assume the popularity of a location is independent of time. In such cases, we can drop the time index and $g_{jk} = g_j$.

User's location indicator. We use a binary variable $H = \{h_{ijk}\}$ to indicate if user u_i was at location l_j at time t_k. This variable is determined by the popularity of the locations g_{jk}. Since we assume that we are not tracking the users at all times, this variable is hidden from us, but our method is capable of determining a probabilistic approximation of H that will ultimately be useful to establish the reliability model for the users. Similarly to the reports, $h_{i\mathbf{j}k} = 1$ for some $\mathbf{j} \Rightarrow h_{ijk} = 0 \quad \forall j \neq \mathbf{j}$, since a user cannot be in two locations at a same time. On the other hand, $h_{ijk} = 0$ directly implies that the user u_i was not at location l_j at time t_k. This is different from the reports, where $x_{ijk} = 0$ could be for a series of reasons. Although some mobile apps send GPS location together with the report, we are assuming that this data is not specifically sent. On the other hand, if GPS data is available, we could incorporate this information.

User's reliability model. An important factor that determines the reports from the users is their reliability traits. We assume that a user is going to report an event with the same probability regardless of the location they are in or the time slice. Therefore, the reliability model does not vary as a function of the locations L or the time slices T. We model the user's reliability in the following way. For each user $u_i \in U$, for all $l_j \in L$, and $t_k \in T$,

$$\alpha_i = P(x_{ijk} = 1 | h_{ijk} = 1, z_{jk} = 1) \quad (1)$$

$$\beta_i = P(x_{ijk} = 1 | h_{ijk} = 1, z_{jk} = 0) \quad (2)$$

$$\gamma_i = P(x_{ijk} = 1 | h_{ijk} = 0) \quad (3)$$

The probability that a user u_i will report an event as true given that the user is in the correct location and time and that the event is actually true is represented by α_i. The probability that user u_i will report an event as true given that u_i is in the correct location and time, but the event is false is represented by β_i. Finally, γ_i is the probability that a user reports an event as true, given that the user was not at the specified time and location and regardless of whether the event was true or not.

We discuss several kinds of typical users which can be modeled based on different values for α, β, and γ. A user with γ_i close to 1 would be a malicious user, since the user is reporting an event that is not being observed. In general, we expect that the users have no incentive to be malicious and that γ_i is close to zero. Assuming γ_i close to zero, a trustworthy user u_i would have α_i close to one and a low β_i, and would send reports iff u_i observes the event as true. An aggressive user would have both α_i and β_i close to one, and would send reports if the event is true and sometimes also when it is false. This could be due to a misinterpretation of the event. A passive user would have low α_i and β_i, and would not send reports often, regardless of the label of the event. Finally, an untrustworthy user would have low α_i and high β_i, and would report events as true when they are false, and not report events when they are true.

Table 2: Probabilities of reports given H, Z, and θ. For example, $P(x_{ijk} = 1 | h_{ijk} = 1, z_{jk} = 1) = \alpha_i$

$h = 1$		x_{ijk}	
		0	1
z_{jk}	0	$(1-\beta_i)$	β_i
	1	$(1-\alpha_i)$	α_i

$h = 0$		x_{ijk}	
		0	1
z_{jk}	0	$(1-\gamma_i)$	γ_i
	1	$(1-\gamma_i)$	γ_i

For ease of notation, let $\theta_i = (\alpha_i, \beta_i, \gamma_i)$ and $\Theta = \{\theta_i | i = 1, ..., n_{users}\}$. Variable Θ completely determines the reliability traits of the users, and is able to describe their trustworthiness, willingness to cooperate, and maliciousness. Table 2 is a summary of the probabilities of X given Z, H, and Θ under all circumstances.

The model presented here allows a **probabilistic explanation for missing reports** in the following way. A report could be missing ($x_{ijk} = 0$) if the user was not at the correct time and place, if the state of the event was the default (i.e. not true), or if the user preferred not to participate. Now that we have defined our model, we calculate the probability of each of these cases, and determine that:

$$\begin{aligned} P(x_{ijk} = 0) =& P(h_{ijk} = 0)(1 - \gamma_i) + \\ & P(h_{ijk} = 1)(1 - P(z_{jk} = 1))(1 - \beta_i) + \\ & P(h_{ijk} = 1)P(z_{jk} = 1)(1 - \alpha_i) \\ =& (1 - \hat{h}_{ijk})(1 - \gamma_i) + \\ & \hat{h}_{ijk}(1 - \hat{z}_{jk})(1 - \beta_i) + \\ & \hat{h}_{ijk}\hat{z}_{jk}(1 - \alpha_i) \end{aligned}$$

Where $\hat{z}_{jk}$ is an estimation of the true value of z_{jk} and is defined as $\hat{z}_{jk} = P(z_{ij} = 1)$. Analogously, $\hat{h}_{ijk} = P(h_{ijk} = 1)$. The above equation clearly separates the reasons why a report might be missing into three different terms. A similar analysis could be used to determine the reasons for a positive report.

5. TRUTH-INFERENCE ALGORITHM

We first discuss the Bayesian Estimation part of the Truth Inference algorithm, which infers the labels of the events without using the event model, and then we improve upon it with the Kalman Estimation, which explicitly models the event with a state-space model $\mathbb{M}$.

5.1 Bayesian Estimation

We discuss how to obtain the latent variables Z, H, and Θ through a continuous approximation in a recursive way based on the reports X of the users.

5.1.1 Setting the initial values.

We start the algorithm by setting initial values for the latent variable Θ. In general, any random number would work, but we can make some assumptions that will speed up the convergence of the method. We assume that the users do not have any incentives to report an event as true when they are not in the location and time of interest, so we set $\hat{\gamma}_i$ close to zero.[2] On the other hand, a value of β_i

[2] we use the conventional "hat" notation to mean an estimated value

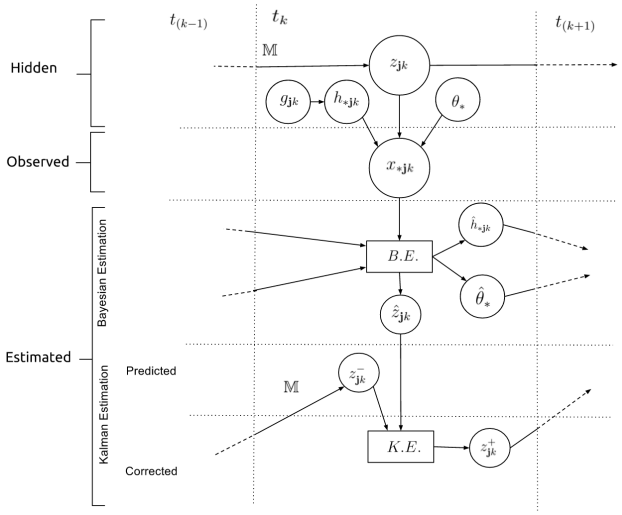


Figure 3: Bayesian Estimation and Kalman Estimation

close to 1 would indicate that the user tends to misinterpret the label of the event. In the case of clear-cut events (e.g. the price of gas less than $2.50) there is not much room for misinterpretation, so we could assign a value of $\hat{\beta}_i$ close to zero. A value of α_i close to 1 could be used for applications where there is an incentive for cooperating. The application to report traffic in Waze, for example, has the incentive of helping others avoid traffic jams and a sense of community. Depending on the application at hand, we could assign values of $\hat{\alpha}_i$ close to 1, where $\hat{\alpha}_i = 1$ implies perfect participation and trustworthiness.

Since we are assuming that there is no incentive to report from a far-away location, we set $\hat{h}_{\mathbf{ijk}} = 1$ whenever $x_{\mathbf{ijk}} = 1$. In such case, we also set $\hat{h}_{\mathbf{ijk}} = 0\ \forall j \neq \mathbf{j}$ since a user can only be at one location at one time. The rest of the values of $\hat{H}$, where there were no reports at location l_j at time t_k are estimated as follows. Since $\hat{h}_{ijk}$ is interpreted as the continuous variable that determines the probability that user u_i was at location l_j at time t_k, and g_j is the probability that a randomly chosen participant will visit the location at least once in $t = 1, ..., n_{times}$, it follows that for a particular user u_i and location l_j, $\prod_{k=1}^{n_{times}}(1-h_{ijk})$ is the probability that user u_i will not visit l_j in any time period, which is equal to $(1 - g_j)$. If we assume that all time periods are equally popular for the same location, then $h_{ijk} = h_{ij1}$ for all k, and $\prod_{k=1}^{n_{times}}(1-h_{ijk}) = \prod_{k=1}^{n_{times}}(1-h_{ij1}) = (1-h_{ij1})^{n_{times}} = (1-g_j)$.We then find that the value of $h_{ijk} = 1-(1-g_j)^{1/n_{times}}$ for all k where $x_{ijk} = 0$.

For the initial values of $\hat{Z}$, we set $\hat{z}_{jk} = P(z_{jk} = 1) = (1/\sum_{u_i \in U} x_{ijk}) \sum_{u_i \in U} x_{ijk}\hat{h}_{ijk}$. This first approximation to $\hat{Z}$ is a "weighted majority voting" approach where the weights are determined by the probability of the users being there. However, it does not consider all of the reasons previously discussed for missing reports.

5.1.2 Updating the variables

Once we have initial values for the latent variables, we continue to update their values with the available reports by using equations of total probability and Bayes' theorem recursively given our dependency graph in Figure 2.

Updating $\hat{H}$. In the case of H, for each time t_k, location l_j and user u_i, we update the value of $\hat{h}_{ijk}$ using the available reports X and the equation for total probability:

$$P(h_{ijk} = 1|X) = P(h_{ijk} = 1|X, z_{jk} = 1)P(z_{jk} = 1) + P(h_{ijk} = 1|X, z_{jk} = 0)P(z_{jk} = 0) \quad (4)$$

Taking the first term when the true label of the event is 1 ($z_{jk} = 1$), and assuming that the reported value is 1 ($x_{ijk} = 1$), we can use Bayes' theorem and obtain that:

$$\begin{aligned} & P(h_{ijk} = 1|x_{ijk} = 1, z_{jk} = 1) \quad (5)\\ =& \frac{P(x_{ijk} = 1, z_{jk} = 1|h_{ijk} = 1) \times P(h_{ijk} = 1)}{P(x_{ijk} = 1, z_{jk} = 1)} \\ =& \frac{P(x_{ijk} = 1|h_{ijk} = 1, z_{jk} = 1)P(z_{jk} = 1|h_{ijk} = 1)P(h_{ijk} = 1)}{P(x_{ijk} = 1|z_{jk} = 1)P(z_{jk} = 1)} \\ =& \frac{\hat{\alpha}_i \times P(z_{jk} = 1) \times \hat{h}_{ijk}}{(\hat{\alpha}_i \times \hat{h}_{ijk} + \gamma_i(1 - \hat{h}_{ijk})) \times P(z_{jk} = 1)} = \frac{\hat{\alpha}_i\hat{h}_{ijk}}{\hat{\alpha}_i\hat{h}_{ijk} + \hat{\gamma}_i(1 - \hat{h}_{ijk})} \end{aligned}$$

Likewise, if we now assume that $x_{ijk} = 0$, we analogously determine that:

$$P(h_{ijk} = 1|x_{ijk} = 0, z_{jk} = 1) = \frac{(1 - \hat{\alpha}_i)\hat{h}_{ijk}}{(1 - \hat{\alpha}_i)\hat{h}_{ijk} + (1 - \hat{\gamma}_i)(1 - \hat{h}_{ijk})} \quad (6)$$

For the second term, where we have $z_{jk} = 0$, and assuming $x_{ijk} = 1$:

$$P(h_{ijk} = 1|x_{ijk} = 1, z_{jk} = 0) = \frac{\hat{\beta}_i\hat{h}_{ijk}}{\hat{\beta}_i\hat{h}_{ijk} + \hat{\gamma}_i(1 - \hat{h}_{ijk})} \quad (7)$$

Finally, when $z_{jk} = 0$ and assuming $x_{ijk} = 0$, then

$$P(h_{ijk} = 1|x_{ijk} = 0, z_{jk} = 0) = \frac{(1 - \hat{\beta}_i)\hat{h}_{ijk}}{(1 - \hat{\beta}_i)\hat{h}_{ijk} + (1 - \hat{\gamma}_i)(1 - \hat{h}_{ijk})} \quad (8)$$

Therefore, if a report $x_{ijk} = 1$, then we use equations (4),(5), and (7) to update H, and equations (4), (6), and (8) if $x_{ijk} = 0$. They are shown respectively as follows:

$$P(h_{ijk} = 1|x_{ijk} = 1) = \frac{\hat{\alpha}_i\hat{h}_{ijk}\hat{z}_{jk}}{\hat{\alpha}_i\hat{h}_{ijk} + \hat{\gamma}_i(1 - \hat{h}_{ijk})} + \frac{\hat{\beta}_i\hat{h}_{ijk}(1 - \hat{z}_{jk})}{\hat{\beta}_i\hat{h}_{ijk} + \hat{\gamma}_i(1 - \hat{h}_{ijk})} \quad (9)$$

$$P(h_{ijk} = 1|x_{ijk} = 0) = \frac{(1 - \hat{\alpha}_i)\hat{h}_{ijk}\hat{z}_{jk}}{(1 - \hat{\alpha}_i)\hat{h}_{ijk} + \hat{\gamma}_i(1 - \hat{h}_{ijk})} + \frac{(1 - \hat{\beta}_i)\hat{h}_{ijk}(1 - \hat{z}_{jk})}{(1 - \hat{\beta}_i)\hat{h}_{ijk} + (1 - \hat{\gamma}_i)(1 - \hat{h}_{ijk})} \quad (10)$$

Updating $\hat{Z}$. Updating the values of $\hat{Z}$ follows a very similar procedure, where we will be updating its values using the values of the recently calculated $\hat{H}$ and our observations of X. We follow an analogous reasoning as for equation (4):

$$P(z_{jk} = 1|X) = P(z_{jk} = 1|X, h_{ijk} = 1)P(h_{ijk} = 1) + P(z_{jk} = 1|X, h_{ijk} = 0)P(h_{ijk} = 0)$$

Analogously to equations (5-8), we construct the equations depending on the observed reports:

$$P(z_{jk}=1|x_{ijk}=1) = \frac{\hat{\alpha}_i \hat{z}_{jk}\hat{h}_{ijk}}{\hat{\alpha}_i \hat{z}_{jk} + \hat{\beta}_i(1-\hat{z}_{jk})} + \frac{\hat{\gamma}_i \hat{z}_{jk}(1-\hat{h}_{ijk})}{\hat{\gamma}_i \hat{z}_{jk} + (1-\hat{\gamma}_i)(1-\hat{z}_{jk})} \quad (11)$$

$$P(z_{jk}=1|x_{ijk}=0) = \frac{(1-\hat{\alpha}_i) \hat{z}_{jk}\hat{h}_{ijk}}{(1-\hat{\alpha}_i) \hat{z}_{jk} + \hat{\beta}_i(1-\hat{z}_{jk})} + \frac{(1-\hat{\gamma}_i) \hat{z}_{jk}(1-\hat{h}_{jk})}{(1-\hat{\gamma}_i)\hat{z}_{jk} + \hat{\gamma}_i(1-\hat{z}_{jk})} \quad (12)$$

The final estimation for z_{jk} is an aggregation over all the reports from the users.

Updating $\hat{\Theta}$. Finally, for updating $\hat{\Theta}$, we follow the definitions of α_i, β_i, and γ_i from section 4 and applying Bayes theorem, arrive at the following update equations:

$$\hat{\alpha}_i = \frac{\hat{z}_{jk}\hat{h}_{ijk}}{\hat{z}_{jk}\hat{h}_{ijk} + (1-\hat{z}_{jk})(1-\hat{h}_{ijk})} \quad (13)$$

$$\hat{\beta}_i = \frac{(1-\hat{z}_{jk})\hat{h}_{ijk}}{(1-\hat{z}_{jk})\hat{h}_{ijk} + \hat{z}_{jk}(1-\hat{h}_{ijk})} \quad (14)$$

$$\hat{\gamma}_i = \frac{\hat{z}_{jk}(1-\hat{h}_{ijk})}{\hat{z}_{jk}(1-\hat{h}_{ijk}) + (1-\hat{z}_{jk})\hat{h}_{ijk}} + \frac{(1-\hat{z}_{jk})(1-\hat{h}_{ijk})}{(1-\hat{z}_{jk})(1-\hat{h}_{ijk}) + (1-\hat{z}_{jk})\hat{h}_{ijk}} \quad (15)$$

This is essentially an expectation maximization algorithm, since we are updating the variables in the expectation phase and maximizing the value of $P(X|\hat{H}, \hat{\Theta}, \hat{Z})$.

5.1.3 Convergence.

We iteratively continue updating the variables $\hat{H}, \hat{Z}$, and $\hat{\Theta}$ until convergence, which is determined by the relative change of the variables $\hat{Z}$ between one iteration and the next. In other words, we stop if at some iteration r, $||\hat{Z}^{(r-1)} - \hat{Z}^{(r)}||/||\hat{Z}^{(r-1)}|| \leq \epsilon$ for some tolerance ϵ. The convergence of $\hat{Z}$ considers all the locations and all the times. The resulting $\hat{Z}$ has values in the interval $[0, 1]$, which are interpreted as the probability of an event to be true. If we need to decide a True/False label for the event, we set a threshold τ (say $\tau = \frac{1}{2}$) and label the event at location l_j at time t_k as true iff $\hat{z}_{jk} \geq \tau$. Algorithm 1 summarizes the method.

Algorithm 1 Truth Inference from Spatio-Temporal reports using Bayesian Estimation (BE)

Input: Spatio-temporal reports from the users: X
Output: 1) Labels of the events: $\hat{Z}$;2) Reliability Model $\hat{\Theta}$

1: Initialize $\hat{\Theta}$ with random numbers. It is useful to incorporate beliefs to speed up convergence.
2: Initialize $\hat{H}$ in the following way:
   - For all **i,j,k** such that $x_{\mathbf{ijk}} = 1$, set $\hat{h}_{\mathbf{ijk}} = 1$ and $\hat{h}_{\mathbf{ijk}} = 0\ \forall j \neq \mathbf{j}$
   - For all **i,j,k** such that $x_{\mathbf{ijk}} = 0$, set $\hat{h}_{\mathbf{ijk}}$ according to the popularity of l_j at time t_k
3: Initialize $\hat{Z}$ by setting $\hat{z}_{jk} = \frac{1}{|U|}\sum_{u_i \in U} x_{ijk}$ for each $l_j \in L, t_k \in T$
4: Set $r = 1$, $\hat{H} = \hat{H}^{(0)}$, $\hat{Z} = \hat{Z}^{(0)}$, $\hat{\Theta} = \hat{\Theta}^{(0)}$
5: **while** has not converged **do**
6: Update $\hat{H}^{(r)}$ with equations (9-10)
7: Update $\hat{Z}^{(r)}$ with equations (11-12)
8: Update $\hat{\Theta}^{(r)}$ with equations (13, 14, 15)
9: Check for convergence using $\hat{Z}^{(r)}$ and $\hat{Z}^{(r-1)}$
10: $r = r + 1$
11: **end while**
12: For all $l_j \in L$, $t_k \in T$, if $\hat{z}_{jk} \geq \tau$ for some threshold τ, label event at location l_j, time t_k as true. Otherwise, label it as false.

5.2 Kalman Estimation

The recursive Bayesian method obtains the probability that an event is true given the reports from the crowdsourced data. When we consider the mathematical or probabilistic model for the behaviour of the events (event model), we can obtain a better approximation to Z. We use the Kalman filter, which is a recursive algorithm that uses data from the previous state of the variables (i.e. the previous time-step, or t_{k-1}) together with observations from the current state (at t_k) and makes a prediction based on the underlying model to estimate the unknown or latent variables [11].

Event Model. The event model is used in the *Prediction phase* of the Kalman Estimation. We assume that there exists an event model $\mathbb{M}$ that determines the values of the labels of the events z_{jk}. For example, the value of z_{jk} could be determined by a Markov model if its value depended only on the value of $z_{j(k-1)}$:

$$z_{jk} = \mathbb{M} z_{j(k-1)} \quad (16)$$

It could also be determined by time series models if the value of z_{jk} depended on $z_{j(k-1)}, z_{j(k-2)}, ..., z_{j1}$. The value of z_{jk} could also be determined by the values of $z_{\mathbf{jk}}, \mathbf{j} \neq j$ if there is a correlation among the locations. For example, the traffic at location $l_{\mathbf{j}}$ could be determined by the surrounding locations since the traffic could start spreading to nearby areas. In general, the event model $\mathbb{M}$ for the values of Z is not known.

In such cases, we can use the observed data from the previous time-steps to train the model $\mathbb{M}$. To continue our example with gas prices, assume we do not have previous data for the prices, and we only have our estimation based on observations at t_1. A prediction for t_2 could be made by assuming $\mathbb{M} = \mathbb{I}$, so that the gas prices at t_2 would be the same as in t_1. If we observe any changes at t_2 after making an estimation and a correction, then we start adjusting the model. Since the gas prices could be modeled as a time series, we can use an autoregressive moving average to make our next prediction at t_3 [16]. The model would be adjusted accordingly as more observations are made.

Initialization of the recursive implementation. In our recursive implementation, for each location j, and at t_1, the approximation obtained from the Bayesian estimation $\hat{z}_{j1}$ is used to predict the value of $\hat{z}_{j2}$ using the underlying model $\mathbb{M}$ (prediction phase). Let z_{j2}^{-} be the *prior* or predicted value.[3] On the other hand, at time t_2 we have our Bayesian estimation $\hat{z}_{j2}$. Using these two values z_{j2}^{-} and $\hat{z}_{j2}$, we use the Kalman estimation to obtain a corrected estimation of z_{j2} (correction phase). The next subsection details the prediction and correction phase. Let z_{j2}^{+} be the *poste-*

[3] We use the conventional $^{-}$ notation to denote the a priori prediction

rior[4] or the value approximated by the Kalman estimation. We can now apply the underlying mathematical model to the posterior value to obtain a prior for the next time step. With this initialization, this recursive algorithm can be applied at $t_2, ..., t_{n_{times}}$. We now explain in more detail the two phases of the Kalman Estimation part of the algorithm.

Prediction Phase. In the prediction phase at time t_k, we use the underlying event model $\mathbb{M}$ to make a prediction for time t_{k+1}.

In general, if we consider T to be the training data, either obtained offline, online, or both, and $z^+_{j(k-1)}$ the corrected approximation for the label at a previous time-step, then the a priori prediction for time t_k is

$$z^-_{jk} = \mathbb{M}(T, z^+_{j(k-1)}) \tag{17}$$

Correction Phase. In this phase, we use the prediction from the previous phase together with the estimated variable from the observed reports at the current time-step to update the approximation to the true label of the event. If z^-_{jk} is the a priori prediction, and $\hat{z}_{jk}$ is the approximation obtained from the observations at location l_j and time t_k, then

$$z^+_{jk} = z^-_{jk} + K_k \times (\hat{z}_{jk} - z^-_{jk}) \tag{18}$$

is the posterior or corrected approximation to z_{jk}, where K_k is the *Kalman gain* at time t_k. Details on the derivation of the Kalman gain can be found in the seminal work by R.E. Kalman [11]. In the unidimensional case, the posterior is a convex linear combination of the prior and the estimated values. If $K_k = 0$ then $z^+_{jk} = z^-_{jk}$ and $K_k = 1$ implies $z^+_{jk} = \hat{z}_{jk}$, so the Kalman gain is a term that "corrects" the prediction using the estimated value from the observed data. We also consider the multidimensional case when there is a temporal-spatial correlation of the events. Such cases occur, for example, if high traffic in one location correlates with high traffic in another location.

Algorithm 2 Truth Inference from Spatio-Temporal reports using Bayesian Estimation and Kalman Estimation (BE+KE)

Input: Spatio-temporal reports from the users X; Event model $\mathbb{M}$
Output: Labels of the events: Z^+;

1: Apply algorithm 1 (BE) to x_{ij1} to obtain $\hat{z}_{j1}$ for all j
2: $z^+_{j1} = \hat{z}_{j1}$
3: **for** $k = 2, ..., n_{times}$ **do**
4: Apply algorithm 1 (BE) to x_{ijk} to obtain $\hat{z}_{jk}$
5: $z^-_{jk} = \mathbb{M}(z^+_{j(k-1)})$ (Prediction)
6: $z^+_{jk} = \hat{z}_{jk} + K \times (z^-_{jk} - \hat{z}_{jk})$ (Correction)
7: Retrain $\mathbb{M}$
8: **end for**

Figure 3 shows the algorithm at a location j and at time t_k. The hidden variables include our objective variable z_{jk}, the observed variable is the data obtained from all the users at location j and time k, and the Bayesian estimation (BE) obtains an approximation to θ, H, and $\hat{z}_{jk}$. This last variable is used together with the prior prediction from the previous step ($z^-_{j(k-1)}$) in the Kalman estimation (KE) to obtain the posterior estimation (z^+_{jk}). The posterior is used together with the underlying model $\mathbb{M}$ to obtain a prior estimation for time t_{k+1}. The additional estimated variables from Bayesian Estimation ($\hat{h}_{*jk}$ and $\hat{\Theta}_*$) are used to improve the estimations of the Bayesian Estimation at the next time step. Algorithm 2 summarizes the whole process.

[4]Likewise, the $^+$ notation denotes the posterior estimation

6. EXPERIMENTAL RESULTS

In this section we describe the experiments performed. First, we describe a baseline method against which our methods were compared, and then present both a real-world case study and extensive simulations evaluating the impact of parameters and settings.

Voting. For our baseline method, we consider the classic approach of *Voting*, where for each location and time, we simply count the number of reports and divide over all users. Since the number of users is in general much larger than the number of reports at a specific location and time, this ratio provides a very low number. We then need to set a threshold above which the method considers the label as true. In the simulation study, we optimized the threshold to have as many true positives as possible while for our real-world example, we set the threshold to 0, so that any positive report would make the method consider the event as true.

Truth Finder for Spatial Events (TSE). Truth Finder for Spatial Events [15] is a recent method that incorporates the probability of a user to be at a certain location, as well as the user's reliability. This method can effectively handle positive and missing reports to infer the truth through crowdsourced data of spatial events. However, TSE does not consider a time dimension so it does not consider a time-dependent model for the changes in the events. To compare with our work, the whole TSE algorithm was run at each time-step with only the data available at that time step.

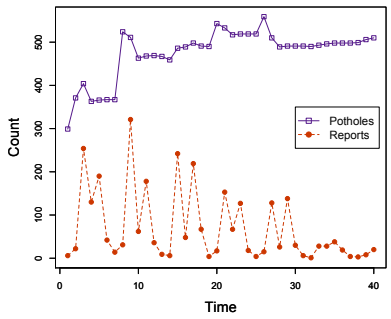
Bayesian Estimation (BE). In both the real-world case study and the simulation experiments, the Bayesian estimation method uses the graphical model presented in figure 2 and the algorithm described in Section 5.1. It does not assume knowledge of an event model and it is purely based on the reports from the users, but it takes into consideration the reports of previous time-steps.

Bayesian Estimation and Kalman Estimation (BE+KE). This method is described in Section 5.1 and 5.2, and can train the event model $\mathbb{M}$ in real-time (BE+KE), or it can use existing historical data to train $\mathbb{M}$ (BE+KE pretrained). In the case of the simulation experiment, we also included a multidimensional model that considers the correlation between the locations as part of the event model $\mathbb{M}$ (BE+MD KE pretrained)

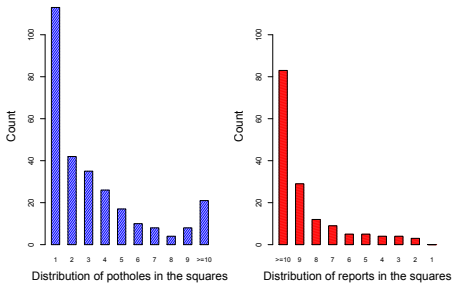
6.1 Real-World Case Study

Waze Reports. Waze is a navigation app where users can send reports from a predefined set of spatio-temporal events. The set of events includes potholes, accidents, and high traffic, and the reports are typically done while the users are driving, so in many occasions, users prefer not to send a report. For our real-world case study, we took a dataset of alert reports sent to the Waze application in the city of Boston, from 2/23/2015 to 3/1/2015. The dataset is publicly available[5] and includes date, time, latitude, longitude, anonymized user id, and type of report. This dataset provides a real-world application of spatio-temporal events with reports from users of varying and unknown reputation and undisclosed locations.

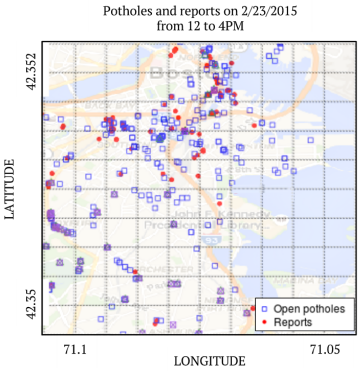
[5]https://data.cityofboston.gov/Transportation/Waze-Alert-Data



(a) Potholes and user reports from 2/23/2015 to 3/1/2015. Each time slice is 4 hours long

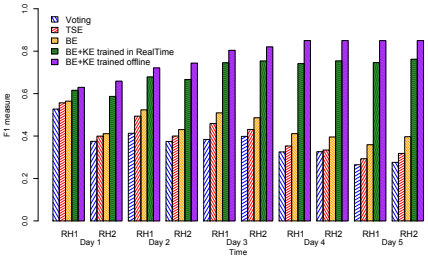


(b) Using a 40x40 grid, most non-empty grid cells have only 1 pothole (blue bar on the left-hand side of the graph) while several grid cells have a considerable amount of reports (red bar on the right)

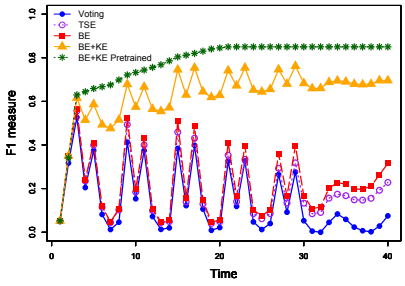


(c) Map of Boston with officially reported potholes and reports by Waze users. The new and closed potholes relate only to this time period, while the open potholes were reported before and closed after this 4 hour period.

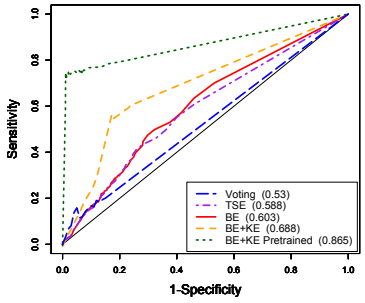
Figure 4: (a)Time Series of potholes and reports; (b) distribution of potholes and reports on a 40x40 grid; (c) open potholes and reports from real-world data.



(a) Comparison of the different methods at rush hours along 5 different days, where RH1 is 8am to 12pm and RH2 is 4pm to 8pm



(b) Different methods compared througout the whole timeline



(c) ROC curve comparing the different methods. The AUC is shown in parenthesis for each method

Figure 5: Comparison of the methods using real-world data of potholes and Waze reports

City Pothole data. For our ground truth, we took a dataset of closed pothole cases, which is publicly available in the city government website of Boston[6]. The dataset includes the date and time when the case was opened, the latitude, longitude, and the date and time when the case was closed. This dataset is updated every day since 2014, but we were only interested in those cases that intersected with the Waze reports. This includes cases that were opened before 2/23/2015 and closed on or after 3/1/2015, as well as those cases that were opened or closed during that time.

Figure 4(a) shows a time series of the potholes and the reports from the users in this period. The time series of the potholes was obtained from the Boston government website and the time series of the reports is from the Waze dataset. Each time slice represents a 4 hour period and we can see that the reports have peaks during the rush hours, and go down at night and on the weekends. The time series representing the potholes did not show such variation. The potholes that were not fixed during working hours remained open during the night.

Matching the reports with potholes on map. Taking the two potholes that were furthest away as corners of a square, we drew an $n \times n$ grid on the city of Boston. The number of grid cells was adjusted so that the cells were small enough to have most non-empty grid cells containing only one pothole, but big enough so that most non-empty grid cells had several reports in it. Figure 4(b) shows the distribution of potholes and reports on a 40 by 40 grid. The horizontal axis shows how many potholes and reports per individual cells there are, and the vertical axis is the count of such cells. The figure shows that there are several grid cells with only one pothole (bar plot on the left) and several grid cells with a large amount of reports (bar plot on the right). All of the non-empty grid cells were considered

[6] https://data.cityofboston.gov/city-services/closed-pothole-cases

Table 3: Precision, Recall, and F1 measure for the real-world data corresponding to figure 5(a)

		Voting			TSE			BE			BE+KE			BE+KE Pre		
Time		Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
Day 1	RH1	0.55	0.50	0.53	0.62	0.52	0.57	0.57	0.56	0.56	0.70	0.55	0.62	0.66	0.60	0.63
	RH2	0.43	0.33	0.38	0.49	0.36	0.41	0.46	0.37	0.41	0.62	0.56	0.59	0.67	0.64	0.66
Day 2	RH1	0.49	0.36	0.41	0.54	0.53	0.53	0.60	0.47	0.52	0.77	0.61	0.68	0.76	0.68	0.72
	RH2	0.39	0.36	0.37	0.47	0.41	0.44	0.52	0.37	0.43	0.67	0.67	0.67	0.77	0.72	0.74
Day 3	RH1	0.45	0.33	0.38	0.54	0.51	0.52	0.55	0.48	0.51	0.83	0.67	0.75	0.84	0.77	0.80
	RH2	0.48	0.34	0.40	0.60	0.43	0.50	0.50	0.47	0.49	0.79	0.72	0.75	0.82	0.82	0.82
Day 4	RH1	0.37	0.29	0.33	0.46	0.40	0.43	0.44	0.38	0.41	0.79	0.70	0.74	0.89	0.81	0.85
	RH2	0.34	0.31	0.33	0.44	0.38	0.41	0.40	0.39	0.40	0.80	0.72	0.75	0.89	0.81	0.85
Day 5	RH1	0.34	0.22	0.26	0.41	0.34	0.37	0.39	0.33	0.36	0.79	0.71	0.75	0.89	0.81	0.85
	RH2	0.32	0.24	0.28	0.49	0.36	0.42	0.42	0.38	0.40	0.83	0.70	0.76	0.89	0.81	0.85

as locations and the rest were discarded from the analysis. Figure 4(c) shows a 10×10 close-up of the map of Boston with the officially reported potholes and the reports from the Waze users on a 4 hour period[7]. The 284 grid cells containing potholes or reports were considered as the locations ($n_{locs} = 284$) and the rest were discarded from the analysis. The dataset considered 2,396 different users (n_{users}) and 8,492 reports.

Compared Methods. We implemented the Bayesian and the Kalman Estimation model. For the event model in the Kalman Filter, we trained the method both in real time, and using previous data from the city government website of Boston. For the real-time training, we modeled the opening and closing of potholes as a time series and used an autoregressive moving average model (ARMA) to predict the next time-step [16]. This model has the advantage that it gives higher weight to the latest observations and less weight to older observations, and the number of observations to consider can be adjusted to the observations that are available at the time. On the other hand, the historical data provided by the city government showed that potholes are usually open for a few days, and that they are only fixed during working hours. With this data, we built a Markov model that showed that if a pothole was open, then with a very high probability it would remain open within the next 4 hours. If a pothole was closed, then with a very high probability it would remain closed. This gave a very accurate a priori prediction for each of the next time steps.

Evaluation metrics. To evaluate the performance of the methods, we used *precision*, *recall*, and the *F1 measure*, which takes into account both precision and recall since $F1 = 2\frac{precision \times recall}{precision + recall}$. These metrics are commonly used to evaluate the performance of truth inference algorithms [12, 25, 8, 6].

Results. Table 3 shows the precision, recall, and F1 measure for the different methods at rush hours (RH). The table shows that precision tends to be higher, presumably because the missing data increases the number of false negatives, while false positives are less common. Figure 5(a) is a graph of the F1 measure, and we observe that the BE and TSE methods outperform in all cases the Voting method, but it is still very dependent on the data provided by the users. The seemingly low values for the performance, and the higher precision than recall is due to the missing values (e.g. fewer reports at night or low transited locations with few reports overall), which increases the number of false negatives, while false positives are less common. The Bayesian Estimation and Kalman Filter (BE+KE) method trained in real time has a performance very similar to the Bayesian Estimation in the first days, but once the time series has more data to perform the predictions and corrections, it performs better than Voting, TSE, and the BE method. If we use the historical data to train the model offline (BE+KE pretrained) then the method starts similarly to the other three, but then it tends to give more weight to the a priori prediction over the observed and incomplete data.

Figure 5(b) shows a comparison of the methods throughout the process and it shows how the methods are dependent on the reports. In the case of Voting, there is a clear similarity with the reports of figure 4(a). Although the Bayesian Estimation and TSE outperform the Voting method, they are still very dependent on the available reports. The drops in the performance of Voting, BE, and TSE can be explained by the lack of reports during night time and the weekend. Both BE+KE and BE+KE pretrained are more robust to this lack of reports due to the a priori estimate based on prediction.

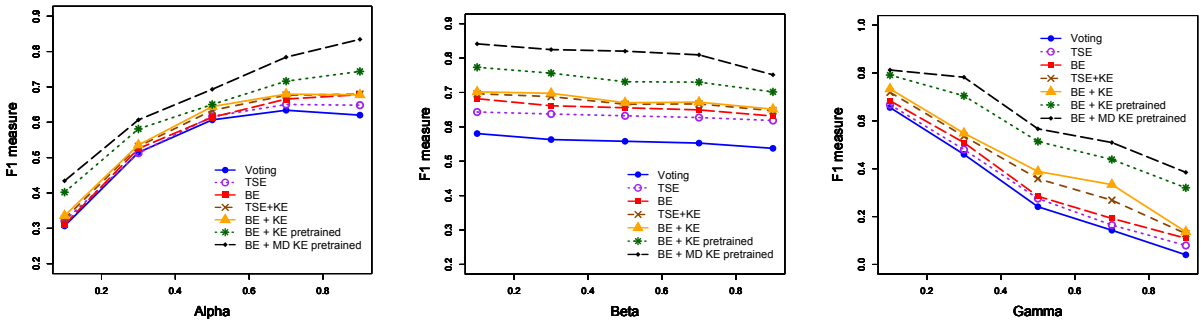
Finally, a ROC curve in figure 5(c) shows a comparison of the different methods as we vary the threshold to determine true or false events. The area under the curve (AUC) is shown in parenthesis in the legend of the graph. We observe that the AUC of the BE+KE pretrained method is 0.861, which makes this a very reliable method under these conditions. The graph shows how BE+KE pretrained outperforms BE+KE, which in turn outperforms BE and TSE. Although BE is still dependent on the reports from the users, it outperforms the Voting method.

6.2 Simulation Experiment

To evaluate the impact on our methods caused by the different parameters, we also performed simulation experiments by generating synthetic data. We observed the effects in performance caused by changes in 1) number of users, 2) number of locations, 3) number of time slices, and 4) reliability traits of users.

Simulation setup. We tried different parameters for the n_{users} and n_{locs}. We fixed $n_{times} = 24$ to represent a scenario where each time period represents an hour of the day. For each user u_i, we generate their reliability traits $\theta_i = (\alpha_i, \beta_i, \gamma_i)$ where each of these parameters is a value between 0 and 1. We simulate the reliability traits of the users with a Beta distribution (the conjugate prior of the Bernoulli distribution) as was done in [15]. Then we simulated the popularity of the locations (g_j with $j = 1, ..., n_{locs}$) by drawing from a power law distribution [15] and generated variable H using this data and the definition of popularity of a location (section 5.1.1). Any user had a g_j chance of visiting location l_j at some point in time, and once a user

[7]Due to slight imprecision in the juxtaposition of images, there is a misalignment between the map and the points



(a) Performance when varying α and fixing $\beta = 0.25$, $\gamma = 0.05$.

(b) Performance when varying β and fixing $\alpha = 0.85$, $\gamma = 0.05$.

(c) Performance when varying γ and fixing $\alpha = 0.85$, $\beta = 0.25$.

Figure 6: Results of simulations as a function of the reliability model for the users

was at a certain location, then the same user could not be at any other location at the same time.

To model the true label of the events, we used two models. Under the first model, we assume that the locations are independent of each other and we use a different Markov model for each location, where the states of the model are *True* or *False* and we simulate with different transition probabilities. In general, these probabilities can be learned in real time **(BE+KE)** as well as using historical data **(BE+KE pretrained)**. The second model corresponded to the case in which the events at the locations are correlated. Such a scenario could be for example modelling traffic, where high traffic on one location could be correlated with high traffic on a nearby location. The correlation matrix is learned offline and the true label of the events is simulated accordingly. We refer to this multidimensional method as **BE+MD KE pretrained**. Finally, to compare the effect that the Kalman estimation method has on our BE method, we replaced the BE with another method for truth inference (the previously mentioned TSE) together with the Kalman estimation and used it with the simulated data **(TSE+KE)**

Once we have simulated H and Z, we use Θ_i to simulate the reports from the users, according to equations 1, 2, and 3. For each user u_i at location l_j and time t_k, the probability that $x_{ijk} = 1$ is drawn from a Bernoulli distribution with parameter α_i, β_i, or γ_i, depending on the values of h_{ijk} and z_{jk}. Algorithm 3 summarizes the simulation procedure.

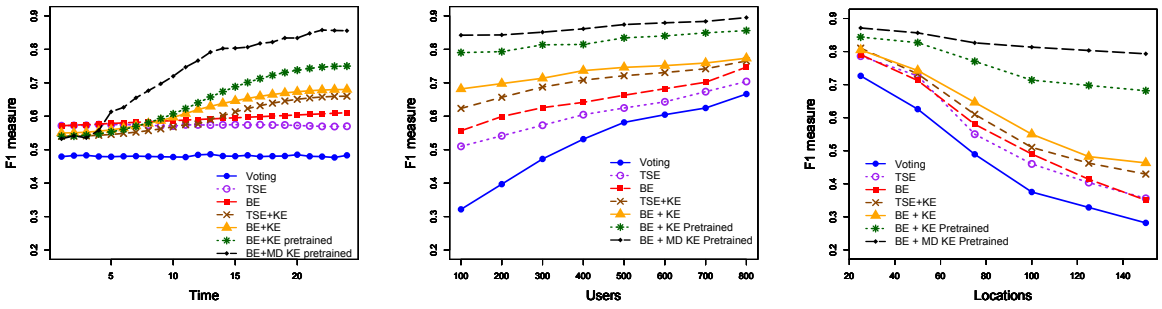
Impact of user reliability. Figure 6 shows the results of the simulations as we varied different values of Θ. We tried different values for α, β, and γ, which test different degrees of reliability (passiveness, trustworthiness, maliciousness), and the figures in 6 are representative of all the different scenarios. In figure 6(a), we fixed the value of β to 0.25 and γ to 0.05 (i.e. low values) and observed the performance of the different values as a function of α. A low value of both β and α represents the situation in which the users are not very participative, and they do not often report the event regardless of it being true or false. In this case, we have several missing values. As α increases with a fixed and low β, the users become more trustworthy, and only report events when the event is true, increasing the number of true positive values without increasing the false positives or false negatives, and therefore increasing the F1 score.

Algorithm 3 Algorithm to generate the simulation variables

1: Determine the values of n_{users}, n_{locs}, and n_{times}
2: For each user u_i, determine θ_i either by fixing the values or with a Beta distribution.
3: For each location l_j, determine its popularity g_j from a power-law distribution
4: Determine a transition model $\mathbb{M}$ and an initial probability for the true label of the events. The model can be unidimensional or multidimensional if there is a correlation between the events.
5: **for** $j = 1, ..., n_{locs}$ **do**
6: Set z_{j1} using the initial probabilities
7: **for** $k = 1, ..., n_{times}$ **do**
8: Use the location popularity to determine H
9: $z_{j(k+1)} = \mathbb{M}(z_{jk}, ..., z_{j1})$
10: **if** $z_{jk} = 1$ and $h_{ijk} = 1$ **then** $X_{ijk} \sim \text{Ber}(\alpha_i)$
11: **end if**
12: **if** $z_{jk} = 0$ and $h_{ijk} = 1$ **then** $X_{ijk} \sim \text{Ber}(\beta_i)$
13: **end if**
14: **if** $h_{ijk} = 0$ **then** $X_{ijk} \sim \text{Ber}(\gamma_i)$
15: **end if**
16: **end for**
17: **end for**

In figure 6(b), we fixed the values for α to 0.85, left γ as before, and observe the performance of the methods as we vary β. In this scenario, there is a slight decrease in the performance of the methods as β increases. A high value of α and β corresponds to the scenario of aggressive users who report events as true even if they are ambiguous or flat out false. There is an increase of true positives, but also of false positives, which makes the value of $F1$ to decrease slightly. For figure 6(c), we kept the values of α at 0.85 and β at 0.25, and modified the values of γ. For low values of γ, we observe a scenario with trustworthy users, but as γ increases, so do their reports when the users are not at the specific location and time. This corresponds to the scenario of malicious users, which increases the values of the false positives and false negatives, decreasing the overall performance of the methods.

In general, we can see that BE and BE+MD KE outperforms the other methods in all scenarios because it models the spatio-temporal correlations of the events explicitly.



(a) Evaluation of the performance of the methods as a function of time

(b) Number of users versus F1

(c) Number of locations versus F1

Figure 7: Results of simulations

TSE+KE also outperforms TSE, which highlights the benefit of using the event model and the Kalman Estimation.

Impact of number of users, locations and time periods. Next, we tried our methods using different number of users, locations, and time periods. We tried different values for each, and Figure 7(a) shows the performance of the methods as a time series with $n_{times} = 24$. For the first few time periods, we observe that the BE and TSE method outperform the rest. However, as we get more reports and are able to train the Kalman Estimation, the four methods that use the Kalman Estimation started performing better. In particular, we see that TSE+KE performs better in time (and better than TSE) since the Kalman estimation corrects the estimated values. TSE is not dependent on time and performs equally good at each time-step, while BE performs better in time since it uses previous reports to improve the estimation parameters at each time-step.

In figures 7(b) and 7(c), we again fixed the value of n_{times} to 24, and observed the performance of the methods as we varied the number of users and locations, respectively. Voting, BE, and TSE are the most sensitive to the reported data, while the methods that use the Kalman Estimation can depend more on the Prediction and Correction phases and hence outperform the classic and BE methods. Figure 7(b) shows that the more users we include, the methods will show an increase in performance. Figure 7 shows that if we keep all the other variables constant, and increase the number of locations, the performance will decrease.

Runtime analysis. Figure 8 shows logarithmic plots of the runtime of the methods as a function of number of time slices, users, and locations. The experiments were run in a machine with Intel Core i7 CPU at 4.00GHz and 64 Gb of memory. In figure 8(a), we observe that Voting and TSE are the least affected by adding more time slices, since each time slice is evaluated independently and the increase in runtime is due to the increase of evaluation slices. BE performs faster than TSE since it considers all the time slices at the same time, but it is affected more by the increase of time slices since it requires convergence of more values. Similarly, the methods based on the Kalman estimation require evaluation of more time slices and convergence of the BE model at each time step. Figure 8(b) shows how the methods are affected by increasing the number of users and in general we see an increase in the runtime on all the methods. A similar situation occurs when increasing the number of locations (figure 8(c)). In all cases we observe that it is better to have a pre-trained model to use with the Kalman filter.

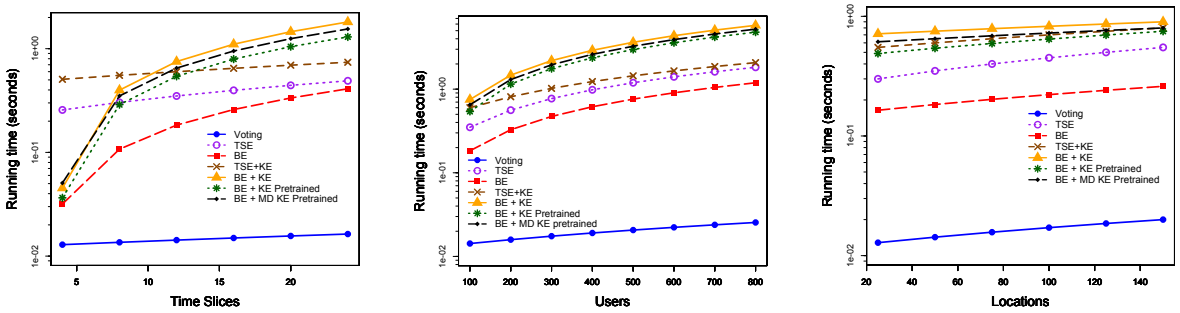
7. DISCUSSION

We have proposed a probabilistic graphical model for truth-inference in a spatio-temporal scenario that incorporates the user's reliability model and does not need to constantly track their location. The method also combines space-state model based prediction of a next time period and fusion with currently available reports. One of the advantages of this approach is that the estimations can be done in real time. If an observation has not arrived yet, we can use the a priori prediction as our estimation, making it suitable for streaming data, and once the information arrives we can update to get a better estimation. The method is versatile to the available data and can easily incorporate different prediction models. Experimental results on both real world data and simulated data show F1 scores that increase as more time periods are accessible and better prediction models are used, making it less dependable on the observations like the classic methods. The algorithm with a pre-trained model from historical data has a better performance, and it is preferable to use when the data is available.

The applicability of the model can be extended by considering the following:

Events could have more than two labels. For example, users could report if there is medium or high traffic. In such a case, users would need to be able to report on each of the different states and the model would need to be extended to include the probability for each state. If n_s is the number of states, then table 2 requires the computation of $n_s \times (n_s - 1) + n_s - 1 = n_s^2 - 1$, since when $h_{ijk} = 1$, for each of the n_s states we require $n_s - 1$ probabilities (since the last one can be inferred from the rest) and for $h_{ijk} = 0$ we require $n_s - 1$ probabilities. The reliability model for the users would also require more parameters to capture how a user would react under different circumstances. Depending on the application and the assumptions, the number of parameters can go from $2n_s - 1$ (assuming e.g. $P(x_{ijk} = s_1|z_{jk} = s_3) = P(x_{ijk} = s_2|z_{jk} = s_3)$ with s_1, s_2, s_3 non-default states) to $n_s^2 - 1$ (if no such assumptions are made).

Continuous variables could be discretized into ranges. For example, the users could report that price of gas is either below $1.50, between $1.50 and $2.50, or higher than



(a) Evaluation of runtime as a function on the number of time slices

(b) Evaluation of runtime as a function of the number of users

(c) Number of runtime as a function of the number of locations

Figure 8: Runtime of the simulations

$2.50. By having a default state (e.g. higher than $2.50), this scenario would be reduced to the previous one.

Users could report a "negative" state. For example, users could report that there is no accident at a certain location. This scenario eliminates the default state and changes the interpretability of the missing reports, since they would only be the result of lack of participation or because the user was not at the specified time and location. A negative report could be either because an event was incorrectly reported or because it is no longer true.

8. REFERENCES

[1] L. Berti-Equille, A. D. Sarma, X. Dong, A. Marian, and D. Srivastava. Sailing the information ocean with awareness of currents: Discovery and application of source dependence. In *CIDR*. Citeseer, 2009.

[2] J. A. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava. Participatory sensing. *Center for Embedded Network Sensing*, 2006.

[3] P. Cheng, X. Lian, Z. Chen, R. Fu, L. Chen, J. Han, and J. Zhao. Reliable diversity-based spatial crowdsourcing by moving workers. *Proceedings of the VLDB Endowment*, 8(10):1022–1033, 2015.

[4] N. Cressie and C. K. Wikle. Space-time kalman filter. *Encyclopedia of environmetrics*, 2002.

[5] G. Demartini, D. E. Difallah, and P. Cudré-Mauroux. Zencrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In *Proceedings of the 21st international conference on World Wide Web*, pages 469–478. ACM, 2012.

[6] X. L. Dong, L. Berti-Equille, Y. Hu, and D. Srivastava. Global detection of complex copying relationships between sources. *Proceedings of the VLDB Endowment*, 3(1-2):1358–1369, 2010.

[7] X. L. Dong, L. Berti-Equille, and D. Srivastava. Integrating conflicting data: the role of source dependence. *Proceedings of the VLDB Endowment*, 2(1):550–561, 2009.

[8] X. L. Dong, L. Berti-Equille, and D. Srivastava. Truth discovery and copying detection in a dynamic world. *Proceedings of the VLDB Endowment*, 2(1):562–573, 2009.

[9] L. Fan, L. Bonomi, L. Xiong, and V. Sunderam. Monitoring web browsing behavior with differential privacy. In *Proceedings of the 23rd international conference on World wide web*, pages 177–188. ACM, 2014.

[10] A. Galland, S. Abiteboul, A. Marian, and P. Senellart. Corroborating information from disagreeing views. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 131–140. ACM, 2010.

[11] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[12] Y. Li, J. Gao, C. Meng, Q. Li, L. Su, B. Zhao, W. Fan, and J. Han. A Survey on Truth Discovery. *ArXiv e-prints*, May 2015.

[13] Y. Li, Q. Li, J. Gao, L. Su, B. Zhao, W. Fan, and J. Han. On the discovery of evolving truth. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 675–684. ACM, 2015.

[14] R. W. Ouyang, A. Srivastava, P. Prabahar, R. Roy Choudhury, M. Addicott, and F. J. McClernon. If you see something, swipe towards it: crowdsourced event localization using smartphones. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 23–32. ACM, 2013.

[15] R. W. Ouyang, M. Srivastava, A. Toniolo, and T. J. Norman. Truth discovery in crowdsourced detection of spatial events. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 461–470. ACM, 2014.

[16] B. Pan, U. Demiryurek, and C. Shahabi. Utilizing real-world transportation data for accurate traffic prediction. In *2012 IEEE 12th International Conference on Data Mining*, pages 595–604. IEEE, 2012.

[17] L. Pournajaf, D. A. Garcia-Ulloa, L. Xiong, and V. Sunderam. Participant privacy in mobile crowd sensing task management: a survey of methods and challenges. *ACM SIGMOD Record*, 44(4):23–34, 2016.

[18] H. To, G. Ghinita, L. Fan, and C. Shahabi. Differentially private location protection for worker datasets in spatial crowdsourcing. *IEEE Transactions on Mobile Computing*, 16(4):934–949, 2017.

[19] D. Wang, T. Abdelzaher, L. Kaplan, and C. C. Aggarwal. Recursive fact-finding: A streaming approach to truth estimation in crowdsourcing applications. In *Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on*, pages 530–539. IEEE, 2013.

[20] D. Wang, L. Kaplan, T. Abdelzaher, and C. C. Aggarwal. On credibility estimation tradeoffs in assured social sensing. *Selected Areas in Communications, IEEE Journal on*, 31(6):1026–1037, 2013.

[21] D. Wang, L. Kaplan, H. Le, and T. Abdelzaher. On truth discovery in social sensing: A maximum likelihood estimation approach. In *Proceedings of the 11th international conference on Information Processing in Sensor Networks*, pages 233–244. ACM, 2012.

[22] X. Yin, J. Han, and P. S. Yu. Truth discovery with multiple conflicting information providers on the web. *Knowledge and Data Engineering, IEEE Transactions on*, 20(6):796–808, 2008.

[23] B. Zhao, B. I. Rubinstein, J. Gemmell, and J. Han. A bayesian approach to discovering truth from conflicting sources for data integration. *Proceedings of the VLDB Endowment*, 5(6):550–561, 2012.

[24] Z. Zhao, J. Cheng, and W. Ng. Truth discovery in data streams: A single-pass probabilistic approach. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1589–1598. ACM, 2014.

[25] Y. Zheng, G. Li, Y. Li, C. Shan, and R. Cheng. Truth inference in crowdsourcing: Is the problem solved? *Proceedings of the VLDB Endowment*, 10(5), 2017.