Frugal Traffic Monitoring with Autonomous Participatory Sensing

Vladimir Coric* Nemanja Djuric*† Slobodan Vucetic*

Abstract

As mobile devices are becoming pervasive, participatory sensing is becoming an attractive way of collecting large quantities of valuable location-based data. An important participatory sensing application is traffic monitoring, where GPS-enabled smartphones can provide invaluable information about traffic conditions. In this paper we propose a strategy for frugal sensing in which the participants send only a fraction of the observed traffic information to reduce costs while achieving high accuracy. The strategy is based on autonomous sensing, in which participants make decisions to send traffic information without guidance from the central server, thus reducing the communication overhead and improving privacy. We propose to use traffic flow theory in deciding whether or not to send an observation to the server. To provide accurate and computationally efficient estimation of the current traffic, we propose to use a budgeted version of the Gaussian Process model on the server side. The model is tuned to be robust to missing observations, which occur quite often in frugal sensing. The experiments on real-life traffic data set indicate that the proposed approach can use up to two orders of magnitude less samples than a baseline approach when estimating traffic speed on a highway network, with only a negligible loss in accuracy.

1 Introduction

It has been estimated that traffic congestion costs the world economy hundreds of billions of dollars each year, increases pollution, and has a negative impact on the overall quality of life in metropolitan areas. In order to solve this emerging problem, transportation departments have been relying on systems for real-time traffic control and monitoring. One of the main goals of traffic monitoring systems is to inform travelers about current and future traffic and to motivate them to modify travel plans during congested periods, which could in turn relieve the congestion. In order to successfully achieve this objective, it is crucial to collect data about current traffic conditions and use it to accurately estimate current and future traffic conditions within a region of interest.

To acquire traffic data, traditional traffic monitoring systems require installation of expensive traffic sensors and the supporting infrastructure. The most common traffic sensors are loop detectors, sensors buried in the road pavement which report traffic volume and traffic speed. However, loop detectors are very expensive to deploy and maintain, and are typically installed only on major highways or on critical intersections monitored by traffic lights.

About a decade ago, an idea to use *probe vehicles* was introduced [19]. This involved installation of specialized GPS-enabled devices in vehicles such as buses and taxis to monitor position of the vehicles and their speed. The collected traffic data from multiple vehicles was integrated and used to provide estimates of traffic conditions on roads on which probe vehicles operated. However, this approach required installation of specialized hardware and software, and real-time traffic estimation was hampered by networking issues and low road coverage. A few years back, the Mobile Millennium project heralded the era of participatory sensing for traffic monitoring [10]. This project allowed volunteers to install monitoring software on their GPS-enabled cellular phones to gather traffic information sent to a centralized server that performed traffic speed estimation.

Many participatory sensing projects, including the Mobile Millennium, have been suffering from the participation problem [14], where they either could not attract enough users or because the existing users stopped providing data. For people to participate, they need to be incentivized either by receiving valuable services (e.g., navigation, social networking) or by rewards such as coupons [5] and virtual credits [6]. In [14] and [15], auction-based systems were proposed to determine appropriate pricing for data provided by participants. While those systems focused on motivating users to continue sharing their data, they paid less attention to minimizing their operational costs, either through reductions in communication and computational overhead or through reduction in the number of received samples.

The objective of this paper is to design a participatory sensing system for traffic monitoring that has low operational cost, low communication overhead, and a high degree of location privacy protection of the participants. To achieve these objectives, we propose a

*Department of Computer and Information Sciences, Temple University, {vladimir.coric, nemanja, slobodan.vucetic}@temple.edu

†Currently at Yahoo! Labs.

strategy that allows client software in the cars to decide whether to send a sample based on the observed traffic conditions, without guidance from the centralized server. We call such a strategy the *autonomous participatory sensing*.

The proposed system is modeled after the idea of Virtual Trip Lanes (VTLs) [11], an autonomous participatory sensing approach in which cars send samples only when they cross virtual sensors placed sparsely on the roads within a region of interest. It has been shown that VTLs can guarantee location privacy while providing accurate estimation of the current traffic conditions. Unlike the original system [11], where cars send samples every time they cross a VTL, cars in our system send samples selectively by exploiting traffic flow theory.

As we describe in the remainder of the paper, our approach for sampling at the client side can result in long intervals when no samples are sent to the central server. The open challenge is how to estimate current traffic conditions given the samples produced by such clients. While we found Gaussian Process models very appropriate for this purpose, we had to modify them to make them applicable for real-time traffic estimation and to make them robust to missing observations.

Contributions of our work are as follows: (1) we propose several sampling strategies to reduce the number of samples transmitted to the server, (2) we utilize traffic flow theory to develop autonomous sampling which adjusts the probability of transmitting a sample based on the current traffic conditions, (3) we modify the Gaussian Process model to become robust to intervals of missing data and to provide fast and accurate real-time traffic estimates, and (4) we empirically evaluate the proposed sampling strategies on a large real-life traffic data set.

2 Related Work

Here we give a brief overview of the traffic estimation problem, and explain two major types of sampling for participatory sensing together with their implications to user privacy.

2.1 Traffic State Estimation Interest in traffic estimation in the data mining community has increased in the last few years, with two recent competitions [3, 4] and several related papers [17, 18]. Transportation research community developed and evaluated many algorithms for estimation of critical traffic variables (e.g., speed, volume), modeled using physically-based traffic flow models [9] and estimated using Kalman filter (KF) algorithms. However, Kalman filters suffer from several limitations such as low accuracy for small penetration rates, as well as cumbersome calibration of parameters

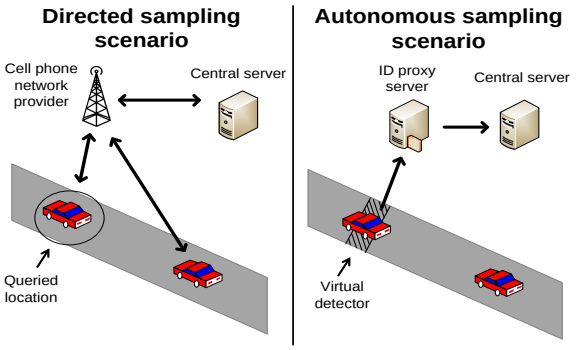


Figure 1: Participatory traffic sensing: Directed (left), Autonomous (right)

that could lead to overfitting [10]. One of the most appropriate alternatives to Kalman filtering is the Gaussian Process (GP) algorithm [13, 18], which we use for traffic speed estimation in this work. However, the original GP algorithm can be computationally costly in online settings such as traffic estimation, and its accuracy might suffer in presence of missing observations. In this work we propose a modified GP algorithm in order to address these shortcomings.

2.2 Directed and Autonomous Participatory Sensing In *directed sensing*, the central server actively seeks samples from participants. A representative system architecture proposed in [13] is shown in Figure 1 (left). The central server estimates the best locations in the traffic network from which to sample, provides a list of these locations to a third-party server (e.g., cell phone provider), which contacts vehicles at selected locations and collects their location and speed. The third-party server reports the collected information back to the central server while preserving user anonymity, which then employs traffic estimation algorithm such as KF or GP to estimate traffic for the entire region of interest. A major drawback of directed sampling is that it requires clients to continually broadcast their location to the third-party server and listen to its requests. This adds a significant communication overhead and could also pose a significant privacy problem if the third-party server becomes compromised (e.g., see security breach at Adobe [2] and similar incidents).

In *autonomous sensing*, the central server passively collects samples sent by the clients who decide when the samples are sent. A representative system architecture called VTLs was proposed in [11] and is shown in Figure 1 (right). VTLs are placed along a traffic network, and when a car with the participatory sensing client app crosses a VTL (VTLs need to be stored in the smartphone app), the current car speed and direction are recorded. The recorded information can be

transmitted to the ID proxy server, which de-identifies the sample and forwards it to the central server. Under this setup, and assuming the VTLs are sufficiently far apart from each other (i.e., no closer than 0.15 miles), it has been shown that VTLs provide a significant level of location privacy [12]. VTLs improve privacy, but they could produce more samples than necessary, thus resulting in large operational costs. In this paper we argue that it could be possible to significantly reduce number of samples submitted by the clients while maintaining advantages of the VTL approach.

3 Problem Statement

In this section we describe the problem of autonomous participatory sensing illustrated in Figure 1 (right). Let us consider a road network R within a geographical region of interest, and assume that M locations along the road network, $l_i, i = 1, \ldots, M$, are selected to represent VTLs. We denote the set of all VTLs as $\mathcal{L}$, where $\mathcal{L} = \{l_i, i = 1, \ldots, M\}$. Let us assume that a certain fraction r of all cars currently driving along the road network R have a working mobile app capable of sensing current location and speed of a car, as well as sending this information to the central server. We call such cars the *participating cars* and r the *penetration rate*. Further, let us assume that the participating cars only send their speed to the estimation server at the moment when they drive over one of the VTLs. We denote a data set of all VTL samples observed by the participatory cars up to time t as $D_t = \{s_i = (t_i, l_i, v_i), t_i \leq t, i = 1 \ldots N_t\}$, where t_i is the time when a participating car drives over a VTL, l_i is its location at time t_i, v_i is its speed at time t_i and N_t is the number of samples collected until time t. Lastly, we assume that the client app can decide autonomously from the server which of the observed samples to send to the server. We denote a subset of those samples that are sent as $D_t^{sent} = \{s_i, i = 1, \ldots, N_t^{sent}\}$, $D_t^{sent} \subseteq D_t$.

In this paper, we address two questions: (1) how should client app decide which samples to send to the central server, and (2) how should the server use the received samples to accurately estimate current traffic speed at any location along R. For the first question, we want to explore whether a strategy could be designed to reduce the number of samples sent to the server while maintaining accuracy of the traffic estimation. This is an important objective if we assume that each submitted sample has a monetary cost to the participatory sensing system. To simplify presentation, we assume that each sample sent to the server incurs the same cost. Since, to the best of our knowledge, there is no prior work done on cost-effective autonomous sensing for traffic estimation, in the remainder of this paper we will propose several sampling strategies. For the second question, we need to design a computationally efficient algorithm that is appropriate for real-time traffic estimation and is robust in the presence of missing observations. To that end we will propose several modifications to the well-known GP model.

4 Client-side: Cost-Efficient Sampling

In this section we propose three strategies that the client app can use to select which sample to send to the server. The first strategy will be used as a baseline, while the remaining two result in frugal sampling which adopts to the current traffic conditions.

4.1 Random Sampling (RS) In the RS strategy, the participating cars decide whether or not to send a sample to the server probabilistically. Each time a participating car passes a VTL, a sample is sent with some probability $\mathbb{P}_{rand}$.

This sampling might be useful in scenarios where the number of participating cars is so large that every VTL is sampled unnecessarily frequently. This would happen, for example, if the penetration rate is near 1 and if VTLs are on congested highways or arterials. Given the current state of the technology, the assumption of such a high penetration rate may be too optimistic. More importantly, since within any region there may be a wide variety of roads (e.g., highways, arterials, side streets) where some are used heavily and others infrequently, using a fixed sampling rate for all roads equally might be suboptimal. Due to its simplicity, RS should be considered as a baseline approach.

4.2 Threshold Sampling (TS) The main idea behind TS is to send samples to the server only when traffic conditions deviate from normal conditions. For example, during nights or weekends the traffic is typically "free-flowing" at speed limit speeds. During these periods, there is practically no need for the participating cars to send data, because the central server could accurately predict traffic speed simply as the speed limit. Thus, cars should send traffic samples only when their speed is significantly far from the expected speed. We propose two variants of the threshold sampling: speed limit sampling and historical sampling.

Speed Limit Threshold Sampling (SL-TS). SL-TS represents a strategy where vehicles send information about their speed only when their current speed is below the traffic speed limit. More formally, in SL-TS, sample $s_i \in D_t$ is added to D_t^{sent} only if $(v_{sl} - v_i) > \theta$, where v_{sl} is the speed limit at the corresponding VTL and θ is a set threshold (e.g., $\theta = 5mph$). Note that in order to implement this proposed strategy, the client

app must be preloaded with the speed limits for all VTLs. In addition, it is clear that speed limit sampling may be combined with random sampling, thus further reducing the number of sent samples.

Historical Threshold Sampling (H-TS). Similarly to speed limit sampling, historical sampling is a strategy where the participating cars send their samples only when the difference between their current speed and the average historical speed at the given time and location is larger than a set threshold θ. Main difference between speed limit and historical sampling is that in cases where congestion is expected at certain VTLs, such as during morning or afternoon rush hours, historical sampling might decrease the number of samples sent to the estimation server compared to speed limit sampling. On the other hand, historical sampling can be inefficient when the traffic patterns are highly volatile. Similarly to speed limit sampling, averaged historical speeds for all VTLs at different times of day would need to be preloaded to the participatory sensing mobile app.

4.3 Fundamental Diagram Threshold Sampling (FD-TS) FD-TS is a sampling strategy which utilizes traffic flow theory to calculate the appropriate sampling probability for each observed sample. Desired sampling rates can differ significantly depending on the time of day and location. For example, for a 2-lane highway during a workday the traffic volume (i.e., number of cars per unit of time) can occasionally exceed 60 cars per minute. Interestingly, during the congestion conditions characterized by severely reduced traffic speed, the traffic volume tends to drop significantly. This behavior is well-known in traffic engineering and is typically described by the *fundamental diagram* [9], which represents traffic volume as a function of traffic speed. In Figure 2 we show a typical fundamental diagram fitted to the actual (speed, volume) data pairs obtained from single loop detectors on a typical highway lane in downtown Minneapolis, MN. We can observe that the largest traffic volumes are observed when the speed is around the speed limit. As traffic speeds drop due to congestion, the volume decreases until it eventually reaches zero at speed zero. The congestion is typically triggered during the bottleneck conditions when there are more cars than what the particular road can sustain. Conversely, when the number of cars on the road is below congestion-inducing, we have the so-called free-flow traffic behavior during which traffic speed is typically at the speed limit or slightly above.

The fundamental diagram can be very useful in determining an appropriate sampling rate. For example, if the penetration rate is $r = 1$ and the traffic speed observed by a participating car at a VTL on a one-lane

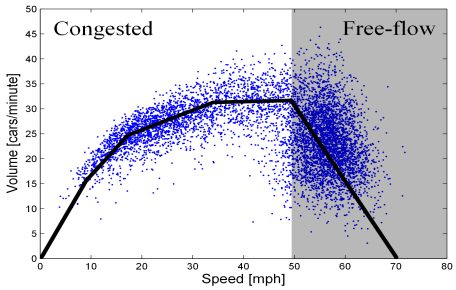


Figure 2: Example of a fundamental diagram, showing traffic volume as a function of traffic speed

highway is $40mph$, by considering Figure 2 it could be concluded that the traffic volume per minute is somewhere between 25 and 35 cars, with the mean about 30. Therefore, if a desired sampling rate to allow accurate traffic speed estimation is one per minute, the client app in the participating car could decide to send its current speed with probability of $1/30$. Thus, knowledge of traffic speed during the congested conditions can lead to an accurate estimation of traffic volume and be very useful for determining the sampling rate. Conversely, as can be seen from Figure 2, during the free-flow conditions the correlation between traffic speed and volume is much weaker. However, since free-flow can be considered as normal traffic state, we can borrow the idea from SL-TS described in the previous subsection and not send any samples during free-flow conditions.

To summarize, if we assume the penetration rate r is known, that the desired number of samples received from a VTL every minute is $n_{desired}$, and that the current traffic speed v_i corresponds to the congested conditions (i.e., $v_{sl} - v_i > \theta$), the client app can send its sample s_i with probability $\mathbb{P}_{fd}(s_i)$ calculated as

$$\mathbb{P}_{fd}(s_i) = \min(1, n_{desired}/(r \cdot fd(v_i) \cdot nl_i)), \tag{4.1}$$

where $fd(v_i)$ is the expected traffic volume according to the fundamental diagram when the current speed is v_i, and nl_i is the number of lanes at the given VTL.

If, on the other hand, the conditions are free-flowing (i.e., $v_{sl} - v_i < \theta$), the sample is not sent. In order to implement the proposed fundamental diagram sampling, the client app would need to know the fundamental diagram, number of lanes at each VTL, the desired number of samples per VTL per minute, and the penetration rate. We note that the fundamental diagram is a fundamental property of traffic and is quite stable. In the next section, by proposing the modification to the GP algorithm, we demonstrate how to provide fast and accurate traffic speed estimates using samples collected with the proposed sampling strategies.

5 Server-side: Traffic Speed Estimation

Here we describe a server-side of the traffic-speed estimation system, which employs Gaussian Process (GP) algorithm to infer current speeds over the entire region of interest R. We give an overview of GP, and point out its deficiencies for large-scale, online speed estimation due to its high computational overhead. We then propose a novel GP algorithm with near-constant time and space complexity, called k-Nearest Neighbors GP. Finally, we propose a modification to the GP algorithm that can handle missing data caused by the proposed client-side sampling algorithms.

5.1 Gaussian Process Let us assume the current time is t and the server has historical data set $D_t^{sent} = \{s_i, i = 1, \ldots, N_t^{sent}\}$ available to estimate the current traffic speed along the road network R. For notational convenience, in the following we will use N instead of N_t^{sent}. We assume that s_i, the i-th sample from the data set, contains information about traffic speed v_i and a corresponding feature vector $\mathbf{x}_i$ with features related to the i-th measurement, such as timestamp and location. We further assume that the measured traffic speed equals the true traffic speed ν_i corrupted by Gaussian noise $\epsilon_i \sim \mathcal{N}(0, \sigma^2_{noise})$, as

$$v_i = \nu_i + \epsilon_i, \tag{5.2}$$

where σ^2_{noise} denotes the noise variance. It follows that the conditional probability of samples $\mathbb{P}(v_i|\nu_i)$ follows Gaussian distribution $v_i \sim \mathcal{N}(\nu_i, \sigma^2_{noise})$. Further, we assume that the noise for each v_i is independent, such that we can write conditional distribution of samples $\mathbf{v} = [v_1, \ldots, v_N]$ given the true labels $\boldsymbol{\nu} = [\nu_1, \ldots, \nu_N]$,

$$\mathbb{P}(\mathbf{v}|\boldsymbol{\nu}) \sim \mathcal{N}(\boldsymbol{\nu}, \sigma^{-1}_{noise} \cdot \mathbf{I}_N), \tag{5.3}$$

where $\mathbf{I}_N$ is an $N \times N$ unit matrix.

Let us introduce a prior over true labels $\boldsymbol{\nu}$, and choose the Gaussian distribution defined by prior mean $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_N]$ and prior $N \times N$ covariance matrix $\mathbf{K}$. Then, it follows

$$\mathbb{P}(\boldsymbol{\nu}) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}), \tag{5.4}$$

where prior covariance matrix $\mathbf{K}$ is defined through user-defined kernel function $k(\cdot, \cdot)$, with elements $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. Given the conditional probability of samples $\mathbb{P}(\mathbf{v}|\boldsymbol{\nu})$ and a prior over true labels $\mathbb{P}(\boldsymbol{\nu})$, the marginal distribution of $\mathbf{v}$ follows normal distribution,

$$\mathbb{P}(\mathbf{v}) = \int \mathbb{P}(\mathbf{v}|\boldsymbol{\nu})\mathbb{P}(\boldsymbol{\nu}) d\boldsymbol{\nu} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{C}), \tag{5.5}$$

where $\mathbf{C}$ is an $N \times N$ covariance matrix with elements equal to $\mathbf{C}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) + \sigma^2_{noise} \cdot \delta_{ij}$, with binary indicator variable δ_{ij} returning 1 if i and j are equal, and 0 otherwise. For a new $\mathbf{x}_{N+1}$ (e.g., comprising current time and road segment location), the task is to estimate unknown label v_{N+1}. Conditional distribution of v_{N+1}, given all previously observed samples $\mathbf{v}$, is equal to

$$\mathbb{P}(v_{N+1}|\mathbf{v}) \sim \mathcal{N}(m_{N+1}, \sigma^2_{N+1}), \tag{5.6}$$

where mean and variance of v_{N+1} can be found as

$$\begin{aligned} m_{N+1} &= \mathbf{k}^{\mathrm{T}} \cdot \mathbf{C}^{-1} \cdot (\mathbf{v} - \boldsymbol{\mu}) + \mu_{N+1}, \\ \sigma^2_{N+1} &= k(\mathbf{x}_{N+1}, \mathbf{x}_{N+1}) + \sigma^2_{noise} - \mathbf{k}^{\mathrm{T}} \cdot \mathbf{C}^{-1} \cdot \mathbf{k}, \end{aligned} \tag{5.7}$$

with $\mathbf{k} = [k(\mathbf{x}_1, \mathbf{x}_{N+1}), \ldots, k(\mathbf{x}_N, \mathbf{x}_{N+1})]$, and where μ_{N+1} is the prior mean for the $(N+1)$-th example.

A drawback of GP is that the memory and time required to use (5.7) for a single prediction are $\mathcal{O}(N^2)$ and $\mathcal{O}(N^3)$, respectively. In fact, using the Sherman-Morrison formula, time for prediction on all N examples can scale as $\mathcal{O}(N^3)$. This scaling is unacceptable in an online setting, where estimation should be provided frequently and the historical data grow without bounds.

There are several existing approaches that can be used to lower the time and space requirements of the described GP model. For example, authors of [8] proposed an online GP on a fixed budget $B \leq N$, which observes a new example, stores it into a set of the so-called basis vectors, and updates the kernel matrix and other parameters as necessary. The proposed online method requires $\mathcal{O}(N \cdot B^2)$ time and $\mathcal{O}(B^2)$ memory to provide predictions for all N examples. However, although this method allows significant time and space gains, it is not suitable for traffic estimation as it attempts to model the distribution of all historical traffic conditions, which is suboptimal for real-time, online traffic estimation systems whose sole objective is to predict current traffic conditions.

5.2 k-Nearest Neighbor Gaussian Process In this section we propose a novel, efficient GP algorithm suitable for real-time traffic estimation. In traffic domain, the traffic conditions of two neighboring road segments and two neighboring time steps are very similar, and grow dissimilar as the time and space gap widens [16]. This motivates another approach to lower the time and memory requirements of GP, which we term k-Nearest Neighbors GP ($\mathrm{GP}_{k\text{-NN}}$). The approach maintains the most recent B samples. In addition, when computing GP prediction for a new example $\mathbf{x}_{N+1}$, we consider only its k-nearest neighbors in space and time in the training set, and find prediction and uncertainty for the new point as

$$\begin{aligned} m_{N+1} &= \mathbf{k}^{\mathrm{T}}_{k\text{-NN}} \cdot \mathbf{C}^{-1}_{k\text{-NN}} \cdot (\mathbf{v}_{k\text{-NN}} - \boldsymbol{\mu}_{k\text{-NN}}) + \mu_{N+1}, \\ \sigma^2_{N+1} &= k(\mathbf{x}_{N+1}, \mathbf{x}_{N+1}) + \sigma^2_{noise} - \mathbf{k}^{\mathrm{T}}_{k\text{-NN}} \cdot \mathbf{C}^{-1}_{k\text{-NN}} \cdot \mathbf{k}_{k\text{-NN}}, \end{aligned} \tag{5.8}$$

where $\mathbf{k}_{k\text{-NN}} = [k(\mathbf{x}_{\text{NN}(1)}, \mathbf{x}_{N+1}), \ldots, k(\mathbf{x}_{\text{NN}(k)}, \mathbf{x}_{N+1})]$, $\mathbf{C}_{k\text{-NN}}$ is $k \times k$ matrix with element in the i-th row and the j-th column equal to $k(\mathbf{x}_{\text{NN}(i)}, \mathbf{x}_{\text{NN}(j)}) + \sigma^2_{noise} \cdot \delta_{ij}$, with function $\text{NN}(i)$ returning an index in the training set of the i-th nearest neighbor of the new example.

The computational saving of the proposed approach as compared to [8] is that there is no need to maintain the kernel matrix for all B examples. Instead, each VTL maintains its own kernel matrix with k neighbors. Thus the memory scales as $\mathcal{O}(L \cdot k^2)$, where L is the number of VTLs. The time of the method scales as $\mathcal{O}(N \cdot k^2)$ because for each prediction we only have to consider the k-neighbors of the corresponding VTL. Considering the spatial and temporal ordering of incoming traffic data, note that we do not have to search through the entire training data set to find the k nearest neighbors. In particular, we can significantly speed up the search by considering only the most recently received samples in the spatial neighborhood of a given VTL, and disregard samples received too far in the past.

5.3 Traffic estimation robust to missing data

The threshold-based sampling techniques proposed in Section 4 make sampling decisions based on current traffic speed, which might result in long intervals without samples. However, in the case when the central server did not receive a sample, it is not obvious if it is because of low traffic volume or because the traffic conditions are free-flowing. In this subsection we propose a method for adjusting speed estimations of GP algorithms that makes them robust to missing samples.

Let us consider a situation when, for a particular VTL, the server did not receive a samples from the participants during the last time step. In that case, one of the following is true: either there were no participating vehicles that crossed over the VTL during the last time step, or there were participating vehicles that did not send samples because the speed was free-flow. We model this situation using a simple Bayes network shown in Figure 3, where event C stands for "VTL not observed by participating cars", event S stands for "traffic speed is free-flow", while event M stands for "no sample received by the server". Ultimately, we are interested in $\mathbb{P}(S|M)$, a probability that the conditions are free-flow given that the server did not receive any samples. Considering the Bayes net from Figure 3 and following simple derivation, we can obtain the following expression,

$$\mathbb{P}(S|M) = \frac{\mathbb{P}(S)}{\mathbb{P}(S) + (1 - \mathbb{P}(S)) \cdot \mathbb{P}(C)}. \tag{5.9}$$

In order to estimate $\mathbb{P}(S|M)$ using (5.9), we need to compute marginal probabilities of events C and S. To calculate $\mathbb{P}(C)$, we model the number of participating

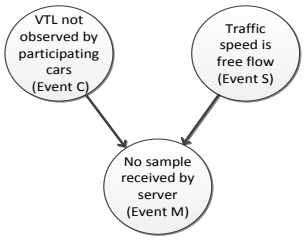


Figure 3: Bayes network for speed correction

vehicles using a Poisson distribution, with λ parameter equal to the expected count according to historical data. The probability $\mathbb{P}(S)$ equals $\mathbb{P}(v_{sl} - v_{N+1} < \theta)$ which can be calculated using equation (5.8). Once the probability $\mathbb{P}(S|M)$ is computed, we can adjust the estimate m_{N+1} of the $\text{GP}_{k\text{-NN}}$ algorithm when the server did not receive a sample for the corresponding VTL in the previous time step. Intuitively, high $\mathbb{P}(S|M)$ would indicate that the current speed is very likely to be near the speed limit, regardless of what GP estimate might be. More formally, in the case of speed-limit sampling, we estimate the current traffic speed at the VTL as

$$\hat{v}_{N+1} = (1 - \mathbb{P}(S|M)) \cdot m_{N+1} + \mathbb{P}(S|M) \cdot v_{sl}, \tag{5.10}$$

where $\hat{v}_{N+1}$ is the adjusted speed estimate.

Let us now consider the case when the server did not receive a sample from a VTL server during previous q time steps. As q increases, the probability that none of the participating cars observed it drops, and it becomes more likely that the traffic is free-flow. To account for this, we need to replace (5.9) with

$$\mathbb{P}(S|M) = \frac{\mathbb{P}(S)}{\mathbb{P}(S) + (1 - \mathbb{P}(S)) \cdot \mathbb{P}(C)^q}. \tag{5.11}$$

6 Experiments

6.1 Experimental setup We used a real-life data set obtained from the Minnesota DOT [1]. The data were collected between 7am and 8pm from March 1st to March 10th, 2003 over a highway network in Minneapolis, MN. There are 750 locations along the traffic network (called the traffic stations) where traffic sensors (called the single loop detectors) are installed on every lane. The stations are typically 0.2 to 0.6 miles apart and the single loop detectors report volume (number of cars) and occupancy (how long the sensor was occupied) every 30 seconds, resulting in more than 327 million samples during the considered 10-day period. Since the single loop detectors did not measure traffic speed directly, we used a standard approach from [7] to estimate traffic speed v_i. We set all speeds higher than $55mph$ to $55mph$ (speed limit in Minneapolis), since we were mostly interested in speeds during congestion peri-

ods. These preprocessed speeds are then used as ground truth. In addition to the 10 test days from March, we also used data from January and February in order to determine historical averages and fit parameters for the GP algorithm and the fundamental diagram.

We assumed that VTLs are placed at the locations of the actual loop detectors. We also assumed that the time step for traffic estimation is 30 seconds. If we assume that the penetration rate is 1 and that every observation is sent to the server, the server would collect exactly the same data as the Minessota DOT loop detector system. To simulate a range of participatory sensing scenarios, we experimented with penetration rates going from 0.05% to 100%. To simulate those penetration rates, we used a random numbers generator from the binomial distribution, where we used actual traffic volume as the number of trials and penetration rate as the probability of success for each trial.

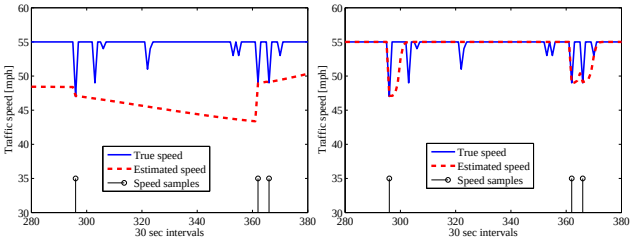
6.2 Accuracy measure To calculate the accuracy of the proposed sampling strategies, we used a mean absolute total travel time error (MAE_{tt}), defined as

$$MAE_{tt} = \frac{3600}{N} \sum |\frac{1}{\hat{v}_i + c} - \frac{1}{v_i + c}|, \tag{6.12}$$

where $\hat{v_i}$ and v_i are estimated and ground truth speed in miles per hour, respectively. The number 3,600 in the formula is used to give the following meaning to MAE_{tt}: it is approximately the difference in seconds between the predicted travel time needed to traverse one mile and the actual travel time for that distance. MAE_{tt} is not exactly such a difference because constant $c = 5$ was added for robustness of the measure, which would be otherwise dominated by errors incurred during heavy congestion events when speed approaches 0. Intuitively, the MAE_{tt} measure penalizes inaccurate traffic speed estimates during congestion periods more harshly than during free flow. For example, if traffic speed was $50mph$ and we estimated it to be $40mph$, the absolute travel time error in MAE_{tt} will be 15 seconds, while if traffic speed was $20mph$ and we estimated it as $10mph$, the absolute travel time error will be 96 seconds.

6.3 Parameter estimation for Gaussian Processes The parameters for GP algorithm described in Section 5 were estimated using cross-validation based on the training data from January and February. We used noise parameter $\sigma^2_{noise} = 0.001$ and budget $B = 2{,}000$, which retained in average about three most recent samples from each of the 750 VTLs.

As features $\mathbf{x}$ in GP algorithm we used time, latitude, and longitude. The most important task in the GP parameter estimation was how to determine spatial and temporal kernel widths for kernel function. As a first approach we tried to learn parameters using the marginal likelihood as suggested in [20]. However, this approach failed because the proposed sampling strategies resulted in bursty sampling with long periods of missing data. An alternative solution was to use the fact that temporal density of the available samples has a significant impact on accuracy and that the kernel width should be tuned to reflect the density. For temporally sparse sampling, kernel should be wider in both spatial and temporal direction to be robust to large periods of missing data. For temporally dense sampling, the kernel width should be small, such that GP should focus on the closest samples in both time and space. Instead of setting kernel widths to constant values for both sparse and dense sampling, we made the kernel width dependent on the number of received samples. Table 1 summarizes the kernel width in temporal and spatial dimensions, as learned from data in January and February. As the parameter k for the proposed GP$_{k\text{-NN}}$, we used all VTLs within a radius of 1 miles from the given VTL, since we observed that the correlation between two VTLs that are more than 2 miles apart is very small. In the following experiments we used proposed sampling strategies together with the k-NN Gaussian Processes.



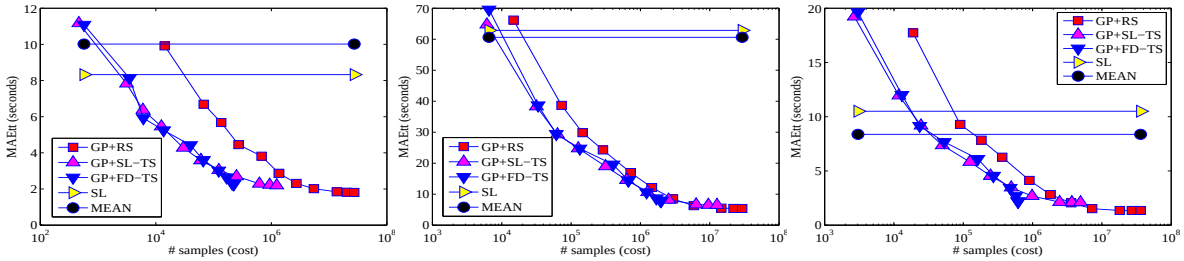
(a) No speed correction (NC) (b) Speed correction (C)

Figure 4: Speed correction example for sensor 1

Table 1: Gaussian Processes kernel width sizes for different number of samples

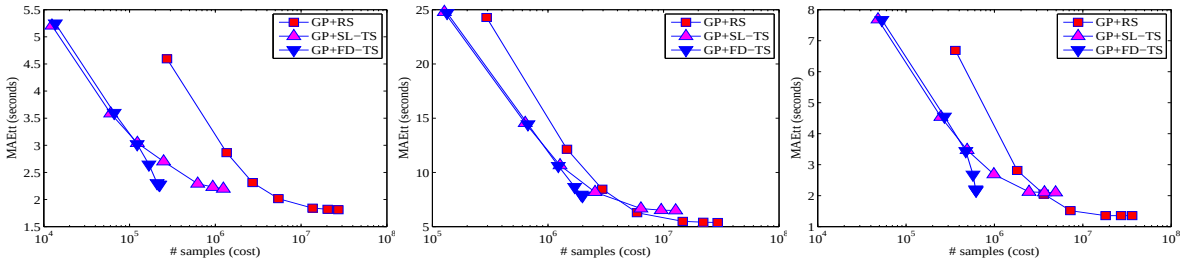
Number of samples	Spatial kernel (miles)	Temporal kernel(sec)
> 2 in last 2 min	0.001	100
> 5 in last 5 min	0.1	1000
> 5 in last 10 min	0.2	2000
> 5 in last 20 min	0.4	4000
> 5 in last 30 min	0.6	6000
> 5 in last 40 min	0.8	8000
> 5 in last 50 min	1	10000
otherwise	1.2	12000

6.4 Speed Correction In the first experiment we investigated whether method for correction of speed estimation in the presence of missing data described



(a) Sunday, March 2nd 2003 (b) Monday, March 3rd 2003 (c) Tuesday, March 4th 2003

Figure 5: MAE_{tt} for three representative test days



(a) Sunday, March 2nd 2003 (b) Monday, March 3rd 2003 (c) Tuesday, March 4th 2003

Figure 6: MAE_{tt} for random, speed limit and fundamental diagram strategies for high penetration rates

in Section 5.3 improves the accuracy. We started by estimating the expected number of cars for each VTL by taking an average historical volume separately for weekdays and weekends in order to account for different traffic patterns during weeks. We set the penetration rate to $r = 5\%$, which we found to be a reasonable value for the proposed system.

In Table 2 we show MAE_{tt} for speed limit SL-TS and historical sampling H-TS methods, calculated with (C) and without (NC) speed correction. Without the correction, error is much higher during the considered period, and the difference is particularly high during workdays. To further illustrate the difference, in Figure 4 we show the estimated speed for sensor 1 during a free-flow period on March 5th with and without correction. Since the results clearly show the advantages of speed correction, in the following experiments we will show only results with the correction.

We can also observe from Table 2 that speed limit sampling SL-TS produces more accurate results than historical sampling H-TS. Consequently, in the remaining experiments we only employ speed limit sampling.

6.5 Comparison of sampling methods In the next set of experiments, we measured performance of GP using $n_{desired} = 1$ and combined with sampling approaches: (1) RS with $\mathbb{P}_{rand} = 1$, (2) SL-TS with $\theta = 5mph$, and (3) FD-TS with $\theta = 5mph$. We compared these methods to the following baselines:

Table 2: Travel time MAE_{tt} for all days in the data set for speed limit(SL-TS) and historical(H-TS) sampling technique with(C)/without(NC) correction

Days	SL-TS NC	SL-TS C	H-TS NC	H-TS C
Sat. 1st	5.11	0.93	4.68	2.09
Sun. 2nd	8.83	1.79	6.45	2.89
Mon. 3rd	10.74	7.27	10.83	7.51
Tue. 4th	9.35	2.26	11.97	2.54
Wed. 5th	9.13	3.16	11.60	3.47
Thu. 6th	18.28	8.77	19.90	9.31
Fri. 7th	9.42	3.29	10.04	3.63
Sat. 8th	8.38	4.28	8.68	9.44
Sun. 9th	6.55	0.84	3.27	2.26
Mon. 10th	10.42	1.87	11.50	2.53

Speed limit estimator (SL). The speed limit estimator which assumes that traffic speed is always equal to the speed limit.

Historical estimator (MEAN). The historical estimator which assumes that traffic speed is equal to average historical speed at a given time and location.

We varied the penetration rate from an extremely small rate of 0.05% to an extremely high rate of 100%, which resulted in highly differing number of samples for each sampling strategy.

Figure 5 demonstrates results for three representative days, Sunday, March 2nd to Tuesday, March 4th.

MAE_{tt} was unusually high on Monday due to a severe snowstorm causing many traffic jams during that day, explaining large difference between results for Monday and Tuesday. We can observe that SL-TS was more successful than RS sampling and that it appeared similar to FD-TS sampling. All GP estimations were more accurate than MEAN and SL predictors on all but several lower penetration rates.

Figure 6 gives us a better look at the performance of the RS, SL-TS, FD-TS sampling strategies in the case of relatively large penetration rates. The advantage of SL-TS as compared to RS is due to RS sending information about every vehicle, while SL-TS sampled only congested traffic. However, when the penetration rate becomes high SL-TS starts over-sampling, since there may be multiple vehicles sending information for the same VTL at every time step. On the other hand, FD-TS successfully reduces the number of samples during congested traffic to the desired sampling rate. As a result, FD-TS is able to minimize the number of samples while retaining high accuracy of traffic estimation.

7 Conclusions

In this paper we proposed frugal autonomous participatory sensing strategies designed to reduce the number of samples sent to the server, while maintaining high accuracy of traffic estimation. This was achieved by allowing the participants to send samples depending on the observed current traffic conditions, and exploiting traffic flow theory. Since such sampling is biased and results in missing data, we also proposed an approach for correction of the GP algorithm. As a result, our approach allowed us to reduce sampling rates by almost two orders of magnitude as compared to the original VTLs approach, while incurring only a small loss in accuracy.

References

[1] *Dot: http://www.dot.state.mn.us/.*

[2] *Forbes: http://www.forbes.com/sites/alexkonrad /2013/10/09/how-safe-is-encrypted-card-data-adobe/.*

[3] *Kaggle: http://www.kaggle.com/rta/.*

[4] *Tunedit:http://tunedit.org/challenge/ieee-icdm-2010/.*

[5] A. Albers, I. Krontiris, N. Sonehara, and I. Echizen, *Coupons as monetary incentives in participatory sensing*, in Collaborative, Trusted and Privacy-Aware e/m-Services, Springer, 2013, pp. 226–237.

[6] C.-M. Chou, K.-c. Lan, and C.-F. Yang, *Using virtual credits to provide incentives for vehicle communication*, in ITS Telecommunications (ITST), 2012 12th International Conference on, IEEE, 2012, pp. 579–583.

[7] B. Coifman, *Improved velocity estimation using single loop detectors*, Transportation Research Part A: Policy and Practice, 35 (2001), pp. 863–880.

[8] L. Csató and M. Opper, *Sparse on-line gaussian processes*, Neural Computation, 14 (2002).

[9] C. Daganzo, *The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory*, Transportation Research Part B: Methodological, 28 (1994), pp. 269–287.

[10] A. Hofleitner, R. Herring, P. Abbeel, and A. Bayen, *Learning the dynamics of arterial traffic from probe data using a dynamic bayesian network*, (2012).

[11] B. Hoh, M. Gruteser, R. Herring, J. Ban, D. Work, J.-C. Herrera, A. M. Bayen, M. Annavaram, and Q. Jacobson, *Virtual trip lines for distributed privacy-preserving traffic monitoring*, in Proceedings of the 6th international conference on Mobile systems, applications, and services, ACM, 2008.

[12] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, *Achieving guaranteed anonymity in gps traces via uncertainty-aware path cloaking*, Mobile Computing, IEEE Transactions on, 9 (2010), pp. 1089–1107.

[13] A. Krause, E. Horvitz, A. Kansal, and F. Zhao, *Toward community sensing*, in Proceedings of the 7th international conference on Information processing in sensor networks, IEEE Computer Society, 2008.

[14] J. Lee and B. Hoh, *Sell your experiences: a market mechanism based incentive for participatory sensing*, in Pervasive Computing and Communications (PerCom), 2010 IEEE International Conference on, IEEE, 2010.

[15] D. Mendez and M. A. Labrador, *Density maps: Determining where to sample in participatory sensing systems*, in Mobile, Ubiquitous, and Intelligent Computing (MUSIC), 2012 Third FTRA International Conference on, IEEE, 2012, pp. 35–40.

[16] W. Min and L. Wynter, *Real-time road traffic prediction with spatio-temporal correlations*, Transportation Research Part C, (2011).

[17] B. Pan, U. Demiryurek, and C. Shahabi, *Utilizing real-world transportation data for accurate traffic prediction*, in Data Mining (ICDM), 2012 IEEE 12th International Conference on, 2012, pp. 595–604.

[18] L. S, Y. Y, and K. R, *Adaptive collective routing using gaussian process dynamic congestion models*, in In Proceedings of the ACM Conference on Knowledge Discovery and Datamining (KDD), 2013.

[19] K. K. Srinivasan and P. P. Jovanis, *Determination of number of probe vehicles required for reliable travel time measurement in urban network*, Transportation Research Record: Journal of the Transportation Research Board, 1537 (1996), pp. 15–22.

[20] C. Williams and C. Rasmussen, *Gaussian processes for regression*, (1996).