Inferring Movement Trajectories from GPS Snippets

Mu Li
Carnegie Mellon University
muli@cs.cmu.edu

Amr Ahmed
Google Strategic Technologies
amra@google.com

Alexander J. Smola
Carnegie Mellon University
Google Strategic Technologies
alex@smola.org

ABSTRACT

Inferring movement trajectories can be a challenging task, in particular when detailed tracking information is not available due to privacy and data collection constraints. In this paper we present a complete and computationally tractable model for estimating and predicting trajectories based on sparsely sampled, anonymous GPS land-marks that we call GPS snippets. To combat data sparsity we use mapping data as side information to constrain the inference process. We show the efficacy of our approach on a set of prediction tasks over data collected from different cities in the US.

Keywords

GPS; Movement trajectories; Motion modeling

1. INTRODUCTION

Smart mobile devices are becoming ubiquitous due to ready availability of bandwidth and low entry cost. By now most mobile phones and many tablets carry a Global Positioning System (GPS) or equivalent sensor that can be used to locate the devices in high accuracy, even without using the mobile network. This allows app-designers and website developers to provide hyper-localized services, e.g. for shopping, restaurant recommendation (Yelp, Urban Spoon, ...), context-aware assistance (Square, Siri), device location (Find My Friends, Foursquare), and contextual metadata for user generated content (Twitter, Facebook).

Unfortunately, continuous logging of location data is very costly in terms of energy, hence it is inadvisable to use GPS location information beyond need, especially on devices that are as energy-constrained as mobile phones. As a result, GPS location records are only available in the form of *GPS trajectory snippets* and sporadically, e.g. when a navigation software is activated (e.g. Bing Maps, Apple Maps, Mapquest), or whenever there is abundant power (e.g. while plugged into a car charger). While this data is obviously biased, it provides us with valuable observations regarding travel times, and it allows us to infer properties of individual road segments at a spatial resolution exceeding that offered by the GPS itself.

Modeling trajectories is a highly desirable task since it allows one to improve the prediction of future locations, that is, to *extrapolate future behavior*. This is crucial for saving battery on mobile phones, and to establish geofences that can be exploited by location based services (Square, Remember the Milk). Moreover, it allows us to fill in the blanks between intermittent observations, that is, to *interpolate between past actions*. Note that the second task is considerably easier since we know both approximate start and endpoints of the trajectory. Both of these tools are important to assess the popularity of places and to establish the preferred trajectory between two locations. Numerous challenges arise when working with such GPS snippet data:

- The data collected is sampled non-uniformly, based on a number of decisions influenced by power constraints, business logic, and context.
- The data available to such algorithms needs to respect appropriate privacy policies to retain the trust of the users of the location based service. This may affect e.g. distribution, length, accuracy, location, identity and quantity.
- The data is often noisy, in particular when recording it in cities, where urban canyons may bias the inferred locations generated by the GPS.
- People are not always rational in their path planning. That is, given a GPS snippet of observed locations, they do not always follow the shortest or quickest path between them for numerous reasons. At a minimum, the latent motivations of leisure, convenience, bounded knowledge are poorly understood and unobserved.
- The travel speeds are highly variable, depending on time, roads, driving style, and context. Moreover, waiting times on road intersections also fluctuate widely.
- The inference framework needs to be scalable to large collections of trajectories, many locations, and many GPS snippets. It helps if the problem decomposes hierarchically and if there is only a need to share parameters locally (i.e. for adjacent observations), besides a succinct set of global parameters.

Outline: We begin by a description of the model in Section 2 and details on an efficient inference algorithm in Section 3. We then give experimental results in Section 6 and a summary. Finally we give an overview of related work in Section 7 and contrast those approaches to ours and then conclude in Section 8.

WSDM 2015, February 02 - 06 2015, Shanghai, China

http://dx.doi.org/10.1145/2684822.2685313.

2. MODEL

Modeling movement trajectories requires a rather delicate compromise between computational efficiency and fidelity of the model. On one hand, a fair amount of details is required for the model to be truthful. On the other hand, this can lead to considerable expense in the dynamic programs executed for inference. In the following we describe a model that, as we believe, offers a balanced compromise between these two aims.

2.1 Trajectory Data

Observations are in the form of sets of sequences of GPS location snippets with an approximate level of accuracy and a timestamp added to each such observation. The diagram below depicts the relationship between reported locations o_i, also referred-to as observations, true device locations, s_i, and possible paths ξ taken between the latter. We assume that the paths are constrained by roads, i.e. the GPS snippet will only follow paths that are considered fit for this purpose.

We assume that the data have the following structure:

- A GPS observation o consists of a (latitude, longitude) pair, a direction heading, and a timestamp.
- We denote by $O = (o_1, \ldots, o_n)$ a GPS snippet of a given length n
- $\mathcal{O} = \{O_1, O_2, \ldots\}$ contains all GPS snippets.
- The state s is a location on a road segment. Its position can be determined by which segment it lies in and the offset to the segment beginning. A state shares properties of the road segment, such as direction heading, and also properties of the observation it maps to, such as the timestamp.
- $S = (s_1, \ldots, s_n)$ denotes a sequence of states.
- The index variables i, j usually indicate road segments.
- The index variable k is an indicator for points in a trajectory and segments in a path.
- The variable ξ denotes a path between locations s_i, s_{i+1} that the GPS snippet might have taken. Note that ξ is typically the composition of several road segments.

Denote by θ the parameters, then $p(O, S|\theta)$ is the probability of mapping a observed trajectory snippet O into a sequence of states S. Our goal is to maximize the following log-likelihood

$$\log p(\mathcal{O}|\theta) = \sum_{O \in \mathcal{O}} \log \sum_{S} p(O, S|\theta), \tag{1}$$

where the summations are over all available trajectory snippets and all possible state sequences, respectively.

We make a first order Markov assumption to simplify the inference. More specifically, we assume that $p(O, S|\theta)$ is a product of observation models and motion models, each of which only depends on the previous observation.

$$p(O, S|\theta) = \prod_{k=1}^{n} p(o_k|s_k, \theta) p(s_{k+1}|s_k, \theta) \tag{2}$$

where $p(s_{n+1}|s_n, \theta) = 1$ for the virtual state s_{n+1}.

Given this data our goal is to map the observations to an actual path that a GPS snippet might have taken and to infer future propensities of following a given path. This will allow us to infer actual travel times on road segments. In other words, we aim to infer a distribution over paths ξ that is consistent with the locations, timings and direction headings observed via a sequence of GPS records.

2.2 Observation Model

We assume that observed locations and directions are drawn independently. This is reasonable, when taking into account that direction inference on the device may involve not only past locations but also additional observables such as acceleration and magnetic field.

Observed locations are assumed to be normally distributed around the true locations. Moreover, since directions are constrained to $[0, 2\pi]$ we cannot model directional data as explicitly Gaussian. However, we assume that the log-likelihood is a quadratic function of the deviation between observed and true heading. As a result, the observation model is given by

$$p(o|s) \propto \exp\left(-\frac{1}{2\sigma_{\mathrm{d}}^2} \left\| o^{\mathrm{loc}} - s^{\mathrm{loc}} \right\|^2 - \frac{1}{2\sigma_{\mathrm{l}}^2} \left\| o^{\mathrm{dir}} - s^{\mathrm{dir}} \right\|^2 \right), \tag{3}$$

where $\left\| o^{\mathrm{dir}} - s^{\mathrm{dir}} \right\|$ is meant to denote the angle on the ring $[0, 2\pi]$, i.e. we assume that we have an approximately Gaussian distribution over relative headings.

To ground observations we assume that true locations and true directions are predominantly constrained by the directions and locations of the underlying road network. For this to be feasible, we assume that we have access to underlying mapping data. In other words, our aim is also to supplement the mapping data with GPS snippet data, as inferred from GPS traces.

2.3 Motion Model

The key to our analysis is a detailed motion model. We will discuss the numerous challenges posed by an efficient inference algorithm for it. It incorporates travel times and corresponding distributions over alternative paths that a GPS snippet might have taken to reach a destination. Moreover, it incorporates the aggregate probability of certain trajectories by explicitly modeling the distribution over turns at intersections.

Intuitively we capture the joint distribution as follows: a given trajectory follows a sequence of turns at any given time. Each of the associated road segments and intersections take some time t to traverse. Moreover, segments need to be consecutive in order to constitute a valid path. This yields the following likelihood model for a sequence of observations O and locations S:

$$\begin{aligned} p(O, S|\theta) &= \prod_k p(o_k|s_k, \theta) p(s_{k+1}|s_k, \theta) \\ &= \prod_k p(o_k|s_k, \theta) \sum_{\xi} p(\xi|s_k, \theta) p(s_{k+1}|s_k, \xi, \theta) \end{aligned} \tag{4}$$

In other words, we need to sum over all paths ξ that could have led from s_k to s_{k+1}. The propensity of taking a cer-

tain path ξ will depend on s_k, simply via direction heading, location, speed and other context.

Furthermore, the state s_{k+1} is only reachable from s_k via ξ. This is encoded as follows: Let $\pi(i, j)$ be the turning probability from road segments i into j, where i and j are adjacent. Assume that the path ξ consists of n sequential road segments $(i_1, \ldots, i_n)$. If ξ starts with s_k, then

$$p(\xi|s_k, \theta) = \prod_{\iota=1}^{n} \pi(i_\iota, i_{\iota+1}), \tag{5}$$

and it will be 0 otherwise.

Note that our model uses a first order Markov assumption, namely that $s_{k+1}|\text{past}$ follows $s_{k+1}|s_k$. A more advanced model could incorporate longer sequence histories, albeit at the expense of a significantly more expensive dynamic program. In fact, a longer history would only be solvable by means of sampling, hence we focus on the more directly accessible first order Markov assumption in the present paper.

2.4 The Inverse Gaussian Distribution

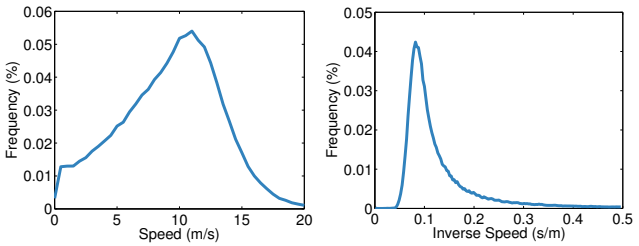

Figure 1: Left: histogram of speeds reported by GPS records; right: distribution of inverse speeds. Note that the time distribution is distinctively non-Gaussian.

Besides requiring spatially contiguous trajectories we also need to ensure that the travel occurs within the constraints of available travel times. This means that the GPS snippet needs to start and finish near well-defined locations at a well-defined point in time.

At first glance, this would suggest that the time to travel from s to s' would follow a Normal distribution. However, when analyzing road segments we do not condition on the distance traveled but rather on the start and stop location. Correspondingly, the distribution is over travel times which are inversely related to speed. An empirical inspection of speed distributions on a segment, as in Figure 1 confirms that a Normal distribution would be a terrible fit for the observed data. Under the assumption that the *velocities* follow a Normal distribution this means that the *travel times* follow an *inverse Gaussian distribution*. Before going into specifics on how this is used, we briefly review its properties here.

The inverse Gaussian distribution $\text{IG}(\mu, \lambda)$ is a member of the exponential family. Hence parameter inference is straightforward and can be solved as a convex minimization problem. Its probability density is given by

$$p(x; \mu, \lambda) = \left(\frac{\lambda}{2\pi x^3}\right)^{\frac{1}{2}} \exp \frac{-\lambda(x-\mu)^2}{2\mu^2 x}, \tag{6}$$

with $\mu, \lambda > 0$. It has mean μ and variance $\frac{\mu^3}{\lambda}$. From its definition we can see that it has reference measure $x^{-\frac{3}{2}}$ and sufficient statistics $\phi(x) = (x, x^{-1})$. Its normalization can be computed efficiently by a variable transform which yields a Gaussian integral. It captures the first passage time of a Brownian random walk, hence it is quite appropriate for modeling the time to reach a given location.

2.5 Modeling Travel Times

We now discuss how to model the travel times between observations. In the following we assume that the path ξ contains road segments $i_1, \ldots, i_n$. The k-th segment has length ℓ_{i_k} and is traversed at speed v_{i_k} with variance in travel time δ_{i_k}.

A straightforward way is to model the travel time on each road segment as an inverse Gaussian distribution. However, this is not suitable for our purpose. The issue is that we have a sum over many segments and the inverse Gaussian distribution is not closed under addition. This complicates inference and leads to a potentially less faithful model.

Instead, we now make the following simplified assumption: the average speed along a path ξ is given by a length-weighted average of speeds over road segments. Moreover, we assume that likewise, the variance is a linear combination of per-segment variances:

$$v_\xi = \frac{1}{\ell_\xi} \sum_{k=1}^{n} \ell_{i_k} v_{i_k} \text{ and } \delta_\xi = \frac{1}{\ell_\xi} \sum_{k=1}^{n} \ell_{i_k} \delta_{i_k} \tag{7}$$

where $\ell_\xi = \sum_{k=1}^{n} \ell_{i_k}$ is the total length of the path.

The advantage of this approach is that we can now model both time and variance as functions that are given by linear combinations of per-segment attributes. An equivalent view would be that we construct a Reproducing Kernel Hilbert Space into which we map all segments and perform estimation in this space [2].

Therefore the expected time to traverse the segment is given by $t_\xi = \ell_\xi / v_\xi$. Note that we need to scale the variance with the total length of the segment, since it is reasonable to assume that it should scale $O(\ell_\xi)$ with longer intervals. The travel time of this path is then modeled as

$$T(\ell_\xi) \sim \text{IG}\left(v_\xi^{-1} \ell_\xi, (\delta_\xi \ell_\xi)^2\right) \tag{8}$$

As per the properties of the inverse Gaussian distribution this amounts to a mean travel time of $v_\xi^{-1}\ell_\xi$ and a variance of $v_\xi^{-3}\delta_\xi^{-2}\ell_\xi$. This allows us to measure the probability of reaching s_{k+1} if traveling from s_k along ξ within a given travel time, which is obtained by from the GPS timestamps of the observations that adjacent steps map into.

2.6 Covariates

Both travel speed and variance of different segments are correlated. For example, nearby road segments often share similar values, so do roads in the same city, roads of the same type, or traffic at different times of the day. This challenge can be addressed by building a hierarchical model with a broad range of covariates governing the relationship between these parameters. We use the following attributes in our model:

- Information regarding the type of road (e.g. highway, major road, or residential) is highly indicative of the travel time on a given segment.
- The number of lanes and information regarding traffic signs convey information about travel speeds.

Algorithm 1 Inference Algorithm

1: **repeat**
2: Randomly sample a set of trajectories $\mathcal{O}$
3: For each trajectory in $\mathcal{O}$ do dynamic programming discussed in Section 3.2.
4: Update transition probability π by (13)
5: Update inverse Gaussian distribution parameters ω and γ by (15) and (16)
6: **until** converged

- Information about usage type (bicycle, number of lanes, pedestrians, speed limit).
- Location (street name, city, state, country, ZIP code).
- Traffic is highly time-varying, e.g. the time of rush-hours may differ between days of the week and weekend traffic patterns may be different yet. Hence it is worth incorporating temporal information (time, week day, holiday) into the traffic estimates.
- We control for endogenous effects.

The dataset used in the experiments is anonymous. This means that we have no information as to whether two snippets are generated by the same user. Recall that we made the somewhat simplified assumption that speed v and dispersion δ are linear combinations of the per-path contributions. In this case, we define the feature vector for segment ξ by the linear combination of per-segment features:

$$\phi_\xi := \frac{1}{\ell_\xi} \sum_{k=1}^{n} \ell_{i_k} \phi_{i_k}. \tag{9}$$

Then the average speed and variance of this segment, which are defined in (7), can be rewritten as

$$v_\xi = \langle \phi_\xi, \omega \rangle \quad \text{and} \quad \delta_\xi = \langle \phi_\xi, \gamma \rangle, \tag{10}$$

where both ω and γ are parameters will be learned.

Lastly, for computational convenience, we treat intersections as road segments with a fixed virtual length. This allows us incorporate all parts into a common inference framework without specific per-segment accounting.

3. INFERENCE

Our goal is to find a suitable set of parameters $\theta = \{\omega, \gamma, \pi\}$ that allow us to capture both the distribution over road segments, the probability of turns, and the variance of travel times. One possible way is to maximize the log-likelihood defined in (1). That is, we aim to solve the nonconvex optimization problem

$$\underset{\omega,\gamma,\pi}{\text{maximize}} \ \log p(\mathcal{O}|\omega, \gamma, \pi) \tag{11}$$

subject to the positive speeds and variances constraints:

$$\langle \omega, \phi_i \rangle > 0 \text{ and } \langle \gamma, \phi_i \rangle > 0 \text{ for all segments } i.$$

Algorithm 1 summarizes the sketch of the inference. Specifically, on each iteration, we first randomly sample a set of GPS snippets, and then sequentially update the parameters according to their subgradients. The constraints can be easily achieved by nonnegativity constraints on ω and γ, given that ϕ_i has only nonnegative entries.

The key challenge is on calculating the gradients. We first introduce auxiliary results on deriving partial derivatives, next we explain how to calculate the conditional probabilities via dynamic programming, and finally we present how to update the parameters.

3.1 Subgradients

We have a number of observed (GPS locations) and hidden (true locations, paths, intersection times) variables and joint inference is a nonconvex problem. Let us briefly consider the general problem of computing gradients of a probability distribution $p(x, y; \theta)$ that consists of observed x and unobserved y random variables. Moreover, assume that p factorizes into terms

$$p(x, y; \theta) = \prod_{c \in \mathcal{C}} \psi_c(x_c, y_c; \theta)$$

where x_c and y_c denote the corresponding (typically overlapping) subsets of (x, y) that are involved in the function ψ_c. This holds for the likelihood of the sequence of observations. In this case we have [1]:

$$\partial_\theta \log p(x; \theta) = \sum_{c \in \mathcal{C}} \mathbf{E}_{y_c|x} \left[\partial_\theta \log \psi_c \left(x_c, y_c; \theta \right) \right]. \tag{12}$$

This can be seen via

$$\begin{aligned}
&\partial_\theta \log p(x; \theta) \\
&= \frac{1}{p(x; \theta)} \sum_{c \in \mathcal{C}} \int dy \prod_{c' \neq c} \psi_{c'}(x_{c'}, y_{c'}; \theta) \partial_\theta \psi_c(x_c, y_c; \theta) \\
&= \frac{1}{p(x; \theta)} \sum_{c \in \mathcal{C}} \int dy p(x, y; \theta) \frac{\partial_\theta \psi_c(x_c, y_c; \theta)}{\psi_c(x_c, y_c; \theta)} \\
&= \sum_{c \in \mathcal{C}} \int dy p(y|x; \theta) \partial_\theta \log \psi_c(x_c, y_c; \theta)
\end{aligned}$$

The claim follows from integrating out all hidden variables except for y_c. The advantage of this strategy is that it suffices to compute expectations with respect to subsets y_c of variables at a time. For instance, in our case this involves only variables for adjacent road segments and transition probabilities.

3.2 Dynamic Programming

The key in computing the gradients is the ability to compute the expectation in (12), which is essentially calculating the conditional probabilities over the latent variables s_k and ξ. However, this task is potentially quite expensive. For instance, naively we would have to take all paths from s_k to s_{k+1} into account, regardless of how improbable and far they may be. Moreover, a naive application of $p(o_k|s_k)$ equally yields a near infinite number of possible latent states that could have led to the observation o_k, assuming an improbably inaccurate GPS measurement. In practice, these eventualities need to be ignored to keep the algorithm feasible.

We do so by imposing a hard constraint on the distance between o_k and s_k. Since there are typically only a relatively modest number of streets, this limits the number of possible locations s_k to tens rather than millions. Likewise, while we allow for arbitrarily slow movement, we limit the maximum speed in which the trajectory will neither violate the laws of physics nor violate the laws of traffic substantially.

Denote by $\text{Paths}(s_k, s_{k+1})$ the set of admissible paths. The transition probability between states is then given by

$$p(s_{k+1}|s_k, \theta) = \sum_{\xi \in \text{Paths}(s_k, s_{k+1})} p(\xi_k|s_k, \theta) p(s_{k+1}|\xi_k, s_k, \theta).$$

Note that dynamic programming would be more appropriate if we had a substantially larger set of Paths(s_k, s_{k+1}). However, it was computationally more efficient to perform the above computation in a brute-force fashion in our case, since there are relatively few admissible paths between adjacent states.

Next consider the trajectory of a path, as observed by O. Here we need to resort to the forward-backward algorithm to compute the likelihoods along the trellis of admissible states.

$$\alpha(s_k) = \sum_{s_{k-1}} p(o_{k-1}|s_{k-1}) p(s_k|s_{k-1}, \theta) \alpha(s_{k-1})$$

$$\beta(s_k) = \sum_{s_{k+1}} p(o_k|s_k) p(s_{k+1}|s_k, \theta) \beta(s_{k+1})$$

where we define $\alpha(s_1) = 1$ and $\beta(s_n) = p(o_n|s_n, \theta)$. Hence marginals and pairwise probabilities are given by

$$p(s_k|O) \propto \alpha(s_k)\beta(s_k)$$
$$p(s_k, s_{k+1}|O) \propto \alpha(s_k) p(s_{k+1}|s_k, \theta) p(o_{k+1}|s_{k+1}, \theta) \beta(s_{k+1}).$$

Using these probabilities we can compute the expectation over latent variables s_k, ξ as required for the gradients. Note that the algorithm is $O(m)$ due to the simple recursion in the dynamic program, where m is the number of observations.

3.3 Updating Parameters

Recall that the objective (1) is given by a sum over trace likelihoods $\log \sum_S p(O, S|\theta)$ for all available GPS snippets. Moreover, note that $p(O, S|\theta)$ can be expanded via (4). In turn, $p(\xi|s, \theta)$ can be expanded further via (5). Putting everything together we obtain the objective:

$$\begin{aligned}
\log p(\mathcal{O}|\omega, \gamma, \pi) &= \sum_{O \in \mathcal{O}} \log \sum_S p(O, S|\theta) \\
&= \sum_{O \in \mathcal{O}} \log \sum_S \left\{ \prod_k p(o_k|s_k, \theta) p(s_{k+1}|s_k, \theta) \right\} \\
&= \sum_{O \in \mathcal{O}} \log \sum_S \left\{ \prod_k p(o_k|s_k, \theta) \left[\sum_\xi p(\xi|s_k, \theta) p(s_{k+1}|s_k, \xi, \theta) \right] \right\} \\
&= \sum_{O \in \mathcal{O}} \log \sum_S \left\{ \prod_k p(o_k|s_k, \theta) \right. \\
&\qquad \left. \times \left[\sum_\xi \left(\prod_{\iota=1}^{n} \pi(i_\iota, i_{\iota+1}) \right) p(s_{k+1}|s_k, \xi, \theta) \right] \right\}
\end{aligned}$$

Updating Transition Probability π

Note that $\pi(a, b) \neq 0$ only if the locations (a, b) are adjacent to another since otherwise there is quite a noticeable difference between the speeds and heading cannot be any transition between them. This simplifies computing expectations greatly. The gradient with respect to π can be computed by taking the expectation over adjacent states

$$\psi(a, b) := \sum_{O \in \mathcal{O}} \sum_{o_i \in O} \mathbf{E}_{\xi|O} \left[\sum_{k=1}^{n_\xi} \{(i_k, i_{k+1}) = (a, b)\} \right]$$

using dynamic programming. Here the sum over the sequence is due to the fact that we assumed that $\pi(a, b)$ is stationary and by subsequent application of the product rule for differentiation. Consequently we can update π to

$$\pi(a, b) = \psi(a, b) / \sum_a \psi(a, b). \tag{13}$$

The normalization by $\psi(a, b)$ is needed to ensure that π encodes proper conditional probabilities. An analogous result holds if we make $\pi(a, b)$ dependent on additional covariates, such as time. Moreover, $\psi(a, b)$ can be modified easily using a conjugate prior.

Updating Inverse Gaussian Distribution Parameters γ, ω

To update the associated parameters recall the probability density (6) and moreover that we model the parameters via $\mu = \ell_\xi / v_\xi$ and $\lambda = \ell_\xi^2 \delta_\xi^2$. Plugging this into the appropriate time distribution we obtain

$$\begin{aligned}
-\log p(t|v_\xi, \delta_\xi, \ell_\xi) =& \frac{1}{2} \log 2\pi + \frac{3}{2} \log t - \log \delta_\xi - \log \ell_\xi \\
&+ \frac{t}{2} \delta_\xi^2 v_\xi^2 - \ell_\xi \delta_\xi^2 v_\xi + \frac{1}{2t} \ell_\xi^2 \delta_\xi^2
\end{aligned} \tag{14}$$

Note that due to our specific choice of parametrization, the problem is nonconvex in v_ξ and δ_ξ. Nonetheless, it has a unique minimum. We have

$$-\partial_\gamma \log p(t|v_\xi, \delta_\xi, l_\xi) = \phi_\xi \left[\frac{\delta_\xi}{t} (t v_\xi - \ell_x)^2 - \frac{1}{\delta_\xi} \right]$$
$$-\partial_\omega \log p(t|v_\xi, \delta_\xi, l_\xi) = \phi_\xi \left[\delta_\xi^2 (t v_\xi - \ell_\xi) \right]$$

Since the times are defined via the distance between observations o_k and o_{k+1}, it suffices if we are able to take the expectation over segments $\xi|o_k, o_{k+1}$ and aggregate over all observation pairs (o_k, o_{k+1}) to obtain proper gradients. Finally, note that $p(\xi|s_k, \theta)$ only depends on s_k insofar as segments that start too far from s_k or that result in discontiguous paths are omitted. Hence we need not concern ourselves with any explicit parameters inherent to $p(\xi|s_k, \theta)$.

Finally, to ensure nonnegativity of the velocity and dispersion, and to exploit the fact the features are typically very sparse, we use nonnegative feature maps ϕ_ξ and multiplicative updates, i.e. exponentiated gradient [6]. That is, we perform coordinate-wise updates

$$\omega^{(t+1)} = \omega^{(t)} .* \exp\left(\eta t^{-\frac{1}{2}} \partial_\omega \log p(\mathcal{O}|\theta) \right) \tag{15}$$

$$\gamma^{(t+1)} = \gamma^{(t)} .* \exp\left(\eta t^{-\frac{1}{2}} \partial_\gamma \log p(\mathcal{O}|\theta) \right) \tag{16}$$

where η is the learning rate, t is an iteration counter, and both $.*$ and exp are carried out element by element.

4. EVALUATION TASKS

Two tasks are designed to evaluate the proposed algorithm. We test how well the model fits the data by estimating a past location or travel time to reach an internal point on a given trajectory (interpolation task). The second task tests how well the model generalizes to future events by estimating either the GPS snippet location in the future at a given time or the time needed to reach a future location. Future prediction is a challenging task considering that the GPS snippet could take various paths in the future, thus the quality of the prediction depends on how well the model estimates the transition probabilities and how well the model estimates the speed of each road segment.

4.1 Inferring the Past

In this task, we are interested in predicting a point, s, within a trajectory. Let s_- and s_+ be the previous and next observed points of s respectively in the trajectory, and let t be the travel time between s_- and s_+. Two tasks can be performed.

Time inference. Assume the location of s is given, and the goal is to estimate the travel time from s_- to s. Denote by ξ is the most likely path from s_- to s_+ passing through s, and by ξ_- the path section from s_- to s. Then we estimate the travel time by interpolation $t_- = \frac{|\xi_-|}{|\xi|}t$, where $|\xi|$ is the length of path ξ.

Location inference. The goal is to estimate the location given the travel time t_- from $s-$ to s. Let ξ be the most likely path from s_- to s_+. We make an assumption that s lies in ξ and denote by ξ_- the subpath. Then ξ_- is determined by the interpolation $|s_-| = \frac{t_-}{t}|\xi|$.

4.2 Predicting the Future

In this section we focus on predicting a future event beyond the boundaries of the observed trajectory. Again, let s be the point of interest in the future, and s_- be the last observed point in a given trajectory. Unlike the case in Section 4.1, s_+ is unknown here, so that s can not obtained by interpolation. Instead, we will use the learned *motion model*. As in Section 4.1, we consider two tasks: predicting the travel time, given a future location and predicting the future location after a given time period.

Predicting travel time t. Since the location of s is known, we could find all possible paths between s_- and s. Let ξ be one of them. By the inverse Gaussian model, the expected time reaching s via ξ is $v_\xi^{-1}\ell_\xi$. Then we sum over all possible paths from s_- to s to predict the travel time t:

$$\left(\sum_\xi p(s|s_-,\xi)\right)^{-1} \sum_\xi p(s|s_-,\xi)\frac{\ell_\xi}{v_\xi}, \qquad (17)$$

where $p(s|s_-,\xi)$ contains both transition probability of path ξ and the probability to arrive at location s after time given by t_ξ.

Predicting future location s. Now we describe how to predict the future location s given the travel time t. To accomplish this, we first find all possible paths starting from s_-. Let ξ be one of such paths. Then we predict the most likely future location after traveling along this path for time t. We let the travel speed be v_ξ and the time variance of this speed is σ_ξ. Thus, the most likely traveled distance ℓ is

$$t = \operatorname*{argmax}_x \; p(x; v_\xi^{-1}\ell, \delta_\xi^2\ell^2). \qquad (18)$$

where p is the probability density of the inverse Gaussian distribution defined in (6). This equation has a closed form solution, usually called the mode and is given by:

$$t = \frac{\ell_\xi}{v_\xi}\left[\left(1+\frac{3}{2}\frac{1}{\ell_\xi v_\xi \sigma_\xi^2}\right)^{\frac{1}{2}} - \frac{3}{2}\frac{1}{\ell_\xi v_\xi \sigma_\xi^2}\right]. \qquad (19)$$

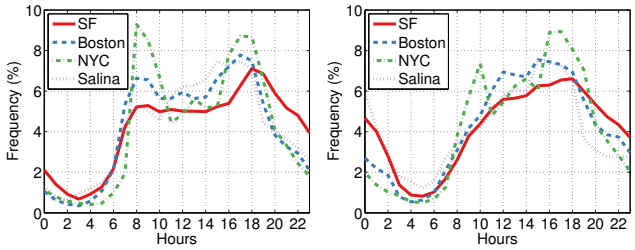

Figure 2: Histogram of traffic frequency. Left: weekdays, right: weekends. Note the rather pronounced double rush hour in NYC and Boston.

Substituting (19) back into (18) we obtain the solution:

$$\ell = t_g v_\xi \left(1 + 3/t v_\xi^2 \delta_\xi^2\right)^{\frac{1}{2}}, \qquad (20)$$

This solution comprises two terms, the first is the distance traveled with speed v_ξ and time t. The second term takes into account the variance of the speed.

5. DATASETS

To demonstrate the efficacy of our approach we sampled GPS trajectory data in 2013 from four US cities: San Francisco (CA), New York City (NY), Boston (MA) and Salina (KS) and corresponding map data. The resulting dataset comprises around 20 million trajectories, 50,000 road segments and 100,000 intersections.

	SF	Boston	NYC	Salina
segments	17,602	6,639	17,409	9,041
intersections	34,989	9,902	29,453	23,622
trajectories	8.1M	6.8M	3.8M	3.3M

Figure 2 shows temporal patterns of these trajectories. There are clear peaks corresponding to rush hours at 8am and 6pm during weekdays. As expected, there is a temporal shift and smoothing on weekends when rush hours are not quite as prominent.

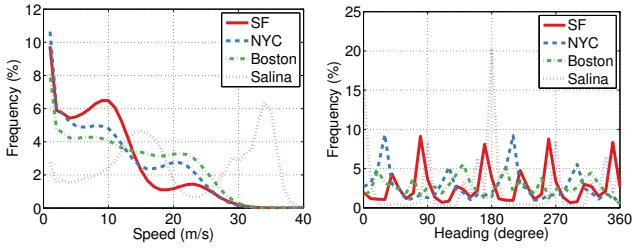

Figure 3: Left: histogram of travel speeds. Right: distribution over heading directions.

There is quite a noticeable difference between the speeds and heading directions in different cities, as can be seen in Figure 3. More specifically, traffic in San Francisco is much slower than in New York and Boston. Moreover, note the pronounced bimodality for Salina. This is likely due to the fact that one of the highways is a major thoroughfare for interstate transport. Also note the strong directionality of traffic in all cities with the exception of Boston. This arises from a grid-layout of the roads.

Figure 4: Road Segments and Intersections

The road segments and intersections are obtained from mapping data, as represented in Figure 4. Both segments and intersections are directional.

6. EXPERIMENTS

We present the experimental results of the two challenge tasks described in Section 4 : inferring the past and predicting the future. However, we first present the features used to learn the speed of road segments.

6.1 Feature Sets for Learning Speeds

We constructed a set of binary features, ϕ for each road segment to learn the speed of road segments as detailed in Section 2.6. Note that ϕ denotes the features of a road segment instances, i.e. the appearance of a road segment within a given trajectory at a given time. These features can be categorized into three sets:

Road features. These features capture several facets of the road attributes such as road type "major road", "high way", etc.

Temporal features. We sliced time into workday and weekend hours to obtain 48 features. A given road segment instance was assigned to an hour based on the time of the majority of GPS points that fell into it.

Individual Speed. We used trajectory ID as the feature to model individual speed preference along the trajectory. This produced tens of millions of unique features.

We combined road and time features to obtain cross-features. In other words, a feature from the road feature group was paired with a feature from the time feature group to form a new feature. This new feature has value 1 if and only if the former two are both equal to 1. This allows us to model non-linear interactions between features since we employ a linear model as described in Section 2.6. This feature combination is also equivalent to a hierarchical model. Due to the large size of the trajectory feature group, we did not combine it with other features.

6.2 Models Compared

To our best knowledge, most of the work in the literature (as we will discuss in Section 7) focuses on on personalized long term trajectory prediction with *continuous* GPS recording [16, 17]. In contrast, our data consists of a large number of anonymous snippets without user IDs. Hence their algorithm does not apply directly. Instead, we compared the proposed algorithm against several variants to understand the contribution of each component in our model.

`Full-Model`. This model used the full set of features, and parameters were learned by Algorithm 1.

`GPS-Speed`. The recorded GPS speeds was directly used without learning ω. In other words, we modeled the speed of a trajectory by the average of its recorded speeds from GPS points. The remaining components were the same as `Full-Model`. Probability π were modeled as usual.

`Common`. The individual speed feature group was removed compared to `Full-Model`. In this model, the speed and time variance only depend on the location and time.

`Shortest-Path`. Only the shortest path between two states was considered, the remaining components were the same as in `Full-Model`. This variant has a computational advantage compared to other variants above. However, it restricts the allowed behavior by assuming people always choose the shortest path to the destination.

6.3 Experimental Setup

To carry out the tasks described above we randomly chose 30% of the trajectories as the test set. We held out a randomly selected internal GPS point of each trajectory in this set to accomplish the task of inferring the past. To predict the future, we elided the last point in the trajectory. To use these removed points for evaluation, their true location, which is required in estimating the travel time, is chosen to be the nearest point in a road segment.

We trained a single model on all trajectories from the four cities. We ran the optimization algorithm for a set of iterations until convergence. In each iteration, we randomly sampled 1,000 trajectories from a random zip code area for processing. The search diameter, namely the maximal distance from possible true locations (states) to observed locations, was limited by 50 meters. For computing expectations over hidden paths ξ, we also limited ourselves to paths containing at most 15 road segments, which was roughly 1.5 kilometer long. Those paths were computed by the breadth-first search algorithm and stored at the beginning before optimization starts. Hence all possible paths between two locations could be fetched during training. The running time of the training procedure took several hours on a single machine with a parallel implementation. In other words, the dynamic program on each trajectory was parallelized.

We fixed $\delta_d = 100$ and $\delta_l = \frac{\pi}{4}$ for the motion model. The learning rates were chosen from the interval $[1, 0.01]$ by examining the convergence rate. Empirically we found that the features constructed from the trajectory IDs (in the Individual speed group) are much sparser than the other two feature groups. This imbalance slows the convergence of the stochastic gradient descent. Instead of performing feature normalization, we placed different learning rates η_1 and η_2 of these two kinds of features respectively. In addition, we only tune η_1 by fixing $\eta_2 = 10\eta_1$. Lastly, we used the top 5 candidate paths when predicting future locations.

Table 1: Average errors of estimating past locations and travel times.

	time error (sec.)				location error (m)				heading direction error (·°)			
	SF	NYC	Boston	Salina	SF	NYC	Boston	Salina	SF	NYC	Boston	Salina
`Full-Model`	2.27	3.23	1.21	0.82	26.9	40.1	16.0	19.2	20.8	19.3	12.6	16.8
`common`	2.38	3.36	1.43	0.80	25.9	38.0	17.0	18.3	17.6	16.6	11.0	15.2
`GPS-speed`	2.99	4.51	1.85	0.98	31.5	50.3	22.3	21.7	17.2	19.1	12.4	14.1
`Shortest-path`	2.28	3.27	1.25	0.81	26.1	39.7	16.2	19.2	21.8	20.0	12.9	18.3

Table 2: Average error of predicting a future location and the travel time. The top 5 locations candidates are considered.

	time error (sec.)				location error (m)				heading direction error (·°)			
	SF	NYC	Boston	Salina	SF	NYC	Boston	Salina	SF	NYC	Boston	Salina
`Full-Model`	3.56	4.83	2.41	1.19	57.8	70.5	44.4	33.5	20.1	19.3	12.9	16.3
`common`	3.95	5.20	2.81	1.27	67.1	77.6	50.9	36.3	17.4	16.2	11.2	15.5
`GPS-speed`	4.71	6.62	3.41	1.42	59.8	77.8	50.7	31.2	19.8	19.4	12.9	17.0
`Shortest-path`	4.40	6.03	2.73	1.46	68.7	83.4	49.2	40.6	22.1	21.5	12.9	19.3

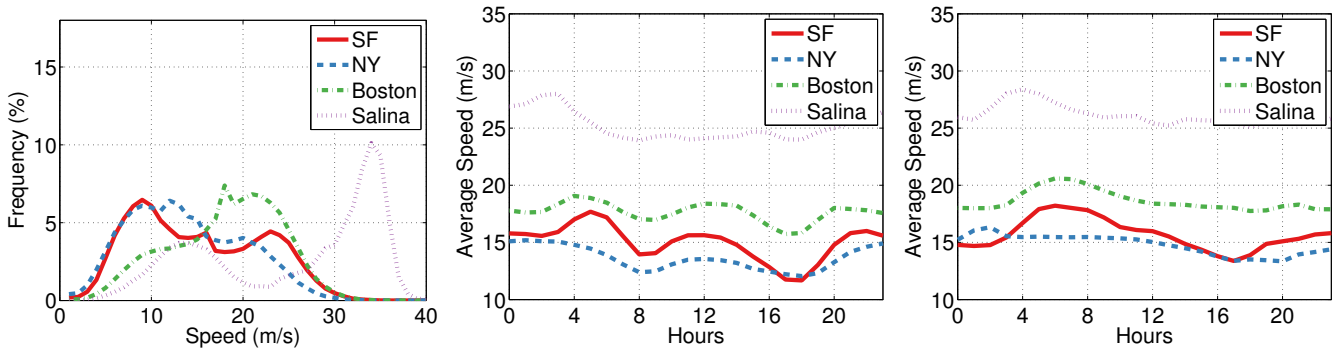

Figure 5: Left: histogram of learned travel speeds. There is a reduction of low speeds portion comparing to the recorded GPS speeds in Figure 3. Middle and right: travel speed as a function of the time of day on weekdays and weekend. As before, the effect of rush hours are quite visible in their reduction of travel speeds.

6.4 Learned Travel Speeds

The learned travel speeds are demonstrated in Figure 5. The positions of the modes in the histogram of the learned speeds are similar to the ones from GPS which were shown in Figure 3. However, a noticeable differences is the significant reduction of low speeds (less than $10m/s$). These speeds might have happened near a red traffic light or a stop sign. The proposed model has a smoothing effect because it uses a smoothed constant speed between two locations and thus misses these range of speeds.

There is a strong pattern of the fall and rise of the traveling speed along time as shown in Figure 5. As expected, the traveling speed decreases during the two rush hours in weekdays in all cities. However this effect is less obvious in Salina, whose traffic is mainly on the interstate highways.

6.5 Time and Location Prediction

We first present the results of estimating past locations and travel times. The average test errors are summarized in Table 1. The average test errors of travel time, location, and heading direction are below 5 seconds, 50 meters, and 20 degrees, respectively. Several reasons contributed to the errors, such as the complexity of city roads, e.g. frequent turning and waiting, and the degradation of data precision due to urban canyons. We believe that our models fit the data reasonably well.

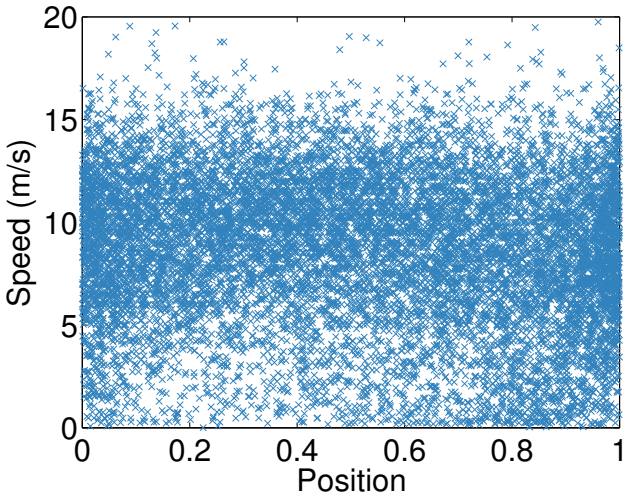

Figure 6: Recorded speeds in different positions of a road segment during the time between 7pm and 9pm.

The estimation errors differ among cities, which can be better observed in the top of Figure 7. As expected, the existence of major arterial highways in Boston and Salina makes the estimation problem easier than in SF and NYC because the speed has lest variation in highways than local

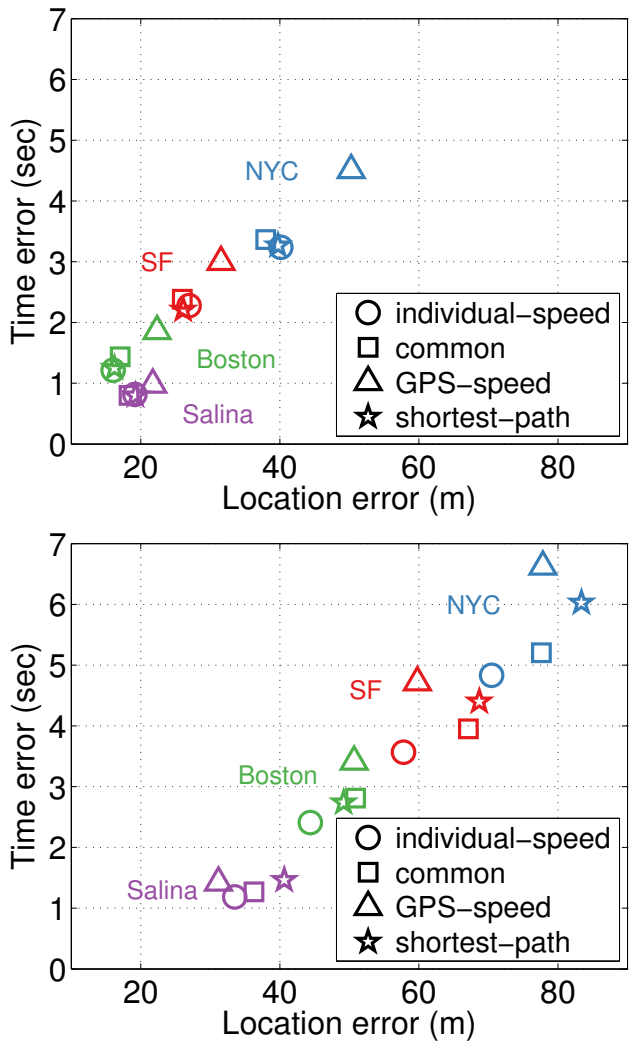

Figure 7: Top: location and time error for different cities when performing interpolation. Bottom: corresponding extrapolation errors.

city roads. Moreover, Manhattan has a higher density of roads and people than the San Francisco area. As expected it has larger estimation error than the latter.

Next we turn to comparing the different variants of the models. The best (statistically significant) results are highlighted via underlines in Table 1, which are also visualized in Figure 7. As can be seen, the model `GPS-speed` performs worse than the other models. We believe this happens because the infrequently sampled GPS points induce highly variable recorded speeds which in turn increases the estimation errors. To see this point, refer to to Figure 6 which shows the recorded speeds at a particular road segment have a large variation. In addition, the gap between `GPS-speed` and the best two other models increase from Salina to NYC (left to right in Figure 7), where the traffic environment becomes more complex.

Finally we show the results of predicting future locations and travel time in Table 2 and also in the bottom of Figure 7. Comparing to the result obtained when inferring past locations, there is an increase on both location error and time error as expected (since we have less constraints). However, the errors are still within reasonable ranges. The increase of errors in time, location and heading directions are less than 2 second, 40 meters, and almost 0, respectively.

The same conclusion observed from estimating past locations are still applicable when estimating future locations. In addition, the model `Full-Model` with all functionality performs better than the slightly simplified model `common` and then the model that only takes shortest paths into consideration. There are two reasons for this. First, the assumption that people always choose the shortest path may be too restrictive. People take a longer path for several possible reasons: easier to drive, personal preference, or even due to turning mistakes. Taking into account a few more possible paths take into considerations this variation better. Second, the travel time and location are predicted by the motion model while in the previous task (predicting the past) we simply interpolate the values. Hence a better motion model like `Full-Model` that takes into consideration individual preference, is more likely to have better individual motion models and therefore gives better results in the future-prediction task.

7. RELATED WORK

There is increasing interest in estimating and predicting travel times as the GPS devices become more available on recent years. There are mainly two research directions. The first is mapping an observed noisy GPS location to a real location, and recovering the trajectory from temporal data. There is a rich of work in this topic, such as [11, 9, 3]. [8] models the whole trajectory path by taking account of the road speed constraint while [10] adopts a HMM model. A related topic is inference the map from the GPS trajectories [7].

The second direction of research is more focused on measuring and predicting the travel time. Most of the work in the literature is focused on high frequency, long sequence of GPS data [14] or highway traffic estimation [15]. [4] presented a probabilistic model of travel time in arterial network based on taxi GPS data. The travel time of each road segment is modeled independently as a Normal and log-normal distribution. A lower-bound of the log-likelihood is solved by an EM-like algorithm to avoid the computational intractable integrals. [13] used a similar model but from a Bayesian approach with an MCMC algorithm for inference. Very recently, [12] used a tensor decomposition method on Beijing taxi GPS data.

By observing that travel times are usually correlated between adjacent roads, [5] proposed modeling the travel times by general features which are related to the road and temporal information. However, their work assume the true position and paths are available. [16, 17] also used general features to modeling taxi driver preference from the inverse reinforcement learning approach.

Our work is different on several aspects. First our model can not only map trajectories to real locations but also modeling road speed and variance, and predicting future travel time and position. Second, we focus on noisy and sparsely sampled anonymous GPS sequences, while most of the work in the literature focus on long, personalized and densely sampled GPS sequences or high precision high-wag data. Third, we model the travel speeds and variance as a function of road, time and personal preference and we consider

non-linear models. Fourth, the travel time is modeled as an inverse Gaussian distribution with the path speeds and variance as parameters. During inference the gradient can be computed in a simple close form. Thus this gives a more faithful approximation of the data as we shown in Figure 1 and is more computationally efficient due to the existence of a closed form solution. Fifth, we perform a joint inference over mapping observations to true locations, discovering possible paths, and inferring travel times. We achieved this by a probabilistic model that considers all possible mapping sequences, potentially paths between two locations, and road transition probabilities. Finally, most importantly, we use millions of trajectories from different cities to demonstrate the scalability of the proposed algorithm. This data is an order of magnitude larger than previous work.

8. CONCLUSION

Nowadays there are new challenges to the task of inferring movement from GPS data. The data may be short and anonymous due to increasing demand of privacy control. The limited power capacity of mobile devices places extra constraints on the sampling frequency of locations, inducing temporal sparse GPS recordings. The spatial coverage and volume of these GPS snippets is quite high due to increasing availability of smart phones and other mobile GPS devices.

In this paper we presented an efficient probabilistic model to analyze this challenging GPS snippet data to perform the following three tasks simultaneously: location mapping, path discovery and travel time estimation. We give an efficient scalable inference algorithm and demonstrated its efficiency by using tens of millions of GPS trajectories snippets from four different cities. The experimental results showed that this algorithm performed well on both estimating past and predicting future locations and travel times.

9. REFERENCES

[1] Christopher Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] T. Gärtner, P. A. Flach, A. Kowalczyk, and A. J. Smola. Multi-instance kernels. In *Proceedings of the International Conference on Machine Learning*, pages 179–186. Morgan Kaufmann Publishers Inc., 2002.

[3] Timothy Hunter, Pieter Abbeel, and Alexandre M Bayen. The path inference filter: model-based low-latency map matching of probe vehicle data. In *Algorithmic Foundations of Robotics*, pages 591–607. Springer, 2013.

[4] Timothy Hunter, Ryan Herring, Pieter Abbeel, and Alexandre Bayen. Path and travel time inference from gps probe vehicle data. *NIPS Analyzing Networks and Learning with Graphs*, 2009.

[5] Erik Jenelius and Haris N Koutsopoulos. Travel time estimation for urban road networks using low frequency probe vehicle data. *Transportation Research Part B: Methodological*, 53:64–81, 2013.

[6] J. Kivinen and M. K. Warmuth. Additive versus exponentiated gradient updates for linear prediction. In *Proc. 27th Annual ACM Symposium on Theory of Computing*, pages 209–218. ACM Press, New York, NY, 1995.

[7] Xuemei Liu, James Biagioni, Jakob Eriksson, Yin Wang, George Forman, and Yanmin Zhu. Mining large-scale, sparse gps traces for map inference: comparison of approaches. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 669–677. ACM, 2012.

[8] Yin Lou, Chengyang Zhang, Yu Zheng, Xing Xie, Wei Wang, and Yan Huang. Map-matching for low-sampling-rate gps trajectories. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 352–361. ACM, 2009.

[9] Tomio Miwa, Daisuke Kiuchi, Toshiyuki Yamamoto, and Takayuki Morikawa. Development of map matching algorithm for low frequency probe data. *Transportation Research Part C: Emerging Technologies*, 22:132–145, 2012.

[10] Paul Newson and John Krumm. Hidden markov map matching through noise and sparseness. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 336–343. ACM, 2009.

[11] Mahmood Rahmani and Haris N Koutsopoulos. Path inference of low-frequency gps probes for urban networks. In *IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1698–1701. IEEE, 2012.

[12] Yilun Wang, Yu Zheng, and Yexiang Xue. Travel time estimation of a path using sparse trajectories. In *Proceeding of the 20th SIGKDD conference on Knowledge Discovery and Data Mining*, 2014.

[13] Bradford S Westgate, Dawn B Woodard, David S Matteson, and Shane G Henderson. Travel time estimation for ambulances using bayesian data augmentation. *Annals of Applied Statistics*, 2013.

[14] Daniel B Work, O-P Tossavainen, Sébastien Blandin, Alexandre M Bayen, Toch Iwuchukwu, and Ken Tracton. An ensemble kalman filtering approach to highway traffic estimation using gps enabled mobile devices. In *IEEE CDC*, pages 5062–5068. IEEE, 2008.

[15] Yufei Yuan, JWC Van Lint, R Eddie Wilson, Femke van Wageningen-Kessels, and Serge P Hoogendoorn. Real-time lagrangian traffic state estimator for freeways. *IEEE Trasaction on Intelligent Transportation Systems*, 13(1):59–70, 2012.

[16] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, pages 1433–1438, 2008.

[17] Brian D Ziebart, Andrew L Maas, Anind K Dey, and J Andrew Bagnell. Navigate like a cabbie: Probabilistic reasoning from observed context-aware behavior. In *Ubiquitous Computing*, pages 322–331. ACM, 2008.