Inferring Human Mobility Patterns from Taxicab Location Traces

Raghu Ganti, Mudhakar Srivatsa[*], **Anand Ranganathan**
IBM T J Watson Research Center
rganti,msrivats,arangana@us.ibm.com

Jiawei Han
University of Illinois, Urbana-Champaign
hanj@illinois.edu

ABSTRACT
Taxicabs equipped with real-time location sensing devices are increasingly becoming popular. Such location traces are a rich source of information and can be used for congestion pricing, taxicab placement, and improved city planning. An important problem to enable these application is to identify human mobility patterns from the taxicab traces, which translates to being able to identify pickup and dropoff points for a particular trip. In this paper, we show that while past approaches are effective in detecting *hotspots* using location traces, they are largely ineffective in identifying trips (pairs of pickup and dropoff points). We propose the use of a graph theory concept - *stretch factor* in a novel manner to identify trip(s) made by a taxicab and show that a Hidden Markov Model based algorithm can identify trips (using real datasets from taxicab deployments in Shanghai and partially simulated datasets from Stockholm) with precision and recall of 90-94% −, a significant improvement over past approaches that result in a precision and recall of about 50-60%.

Author Keywords
Human mobility patterns; Trajectory analysis; Taxi cab occupancy; Hidden Markov Models

ACM Classification Keywords
H.4.m. Information Systems Applications: Miscellaneous

INTRODUCTION
Applications that are driven by the analysis of geo-tagged sensor data generated by everyday mobile devices such as smartphones, tablets, and cars have become quite popular in the recent past [5, 10, 13]. A class of such applications [7, 16, 22] rely on being able to obtain human mobility patterns (typically obtained from location sensors on mobile phones) and are used for urban planning [22], carpooling [20], traffic forecasting, and the spread of biological and mobile viruses [7]. Such mobility patterns have also been used to design and develop routing algorithms in human-driven mobile networks [16].

[*]The first two authors equally contributed to this work

A major source of data for deriving human mobility patterns in large cities are taxicabs, which are one of the primary means of transportation. In the recent past, several taxicab companies [14, 23, 17, 1, 2, 21] have deployed mobile sensing infrastructure that provide rich fine grained location sensor data from taxicabs. In this paper, we propose to leverage this location data to infer human mobility patterns and thus enable a large set of novel applications and services.

The main challenge in drawing such inferences translates to being able to identify when the taxicab is occupied by a passenger and the corresponding pickup and dropoff points − henceforth called the endpoints of a *trip*. A simple solution to identify such trips (namely, pairs of pickup and dropoff points) would be for the taxicab driver to *manually* mark them when a passenger is picked up and dropped off (e.g., by turning on the meter, assuming that the existing sensing infrastructure is integrated with the meter). Unfortunately, many existing systems lack such explicit marking capabilities [24, 17]. Thus, our goal in this paper is to identify taxicab trips (i.e., corresponding pickup and dropoff points) and enrich several existing taxicab traces [24, 17]. These trips aggregated across a fleet of taxicabs determine human mobility patterns.

In this paper we show that simple methods (e.g., speed based classification, spatial clustering based classification) are effective in identifying pickup/dropoff points, but not essentially good at identifying trips. For example, using spatial clustering algorithms, outliers may be classified as candidate endpoints for a trip; infrequently visited suburban/residential regions are candidates for both pickup and dropoff points; also, highly frequented regions may be classified as landmarks (e.g., tourist spots) that are candidates for both pickup and dropoff points. In general, given a location l (of the taxicab), these simple approaches are effective in determining the likelihood of l being a pickup **or** dropoff point. However, we recall that a trip is a pair of pickup **and** dropoff points; hence we are interested in determining the likelihood of l_p being a pickup point **and** l_d being the corresponding dropoff point. We show that naïvely using past knowledge discovery and data mining approaches to identify trips (e.g., marking likely pickup/dropoff points that appear in succession in a taxicab trace as trip endpoints) can result in very poor performance (precision and recall of about 30%). The key intuition here is as follows. Let us consider three locations `home`, `walmart`, and `train station` and let us suppose that `walmart` appears on the route from `home` to `train station`. A classifier may learn that `home` is an outlier and a possible trip endpoint and that `walmart` and `train station` are highly frequented places and hence likely trip endpoints. But given a location trace that includes `home`, `walmart`, and `train station` (in that order) classical approaches fail to determine whether the trip dropoff point is `walmart` or `train station`.

In order to tackle the above problem, we propose the use of a graph theoretical concept, *stretch factor* [4], which is defined as the ratio of the actual distance traversed on a given trip to that of the shortest distance between the endpoints of that trip (*distance stretch factor*) and the *time stretch factor* is the ratio of the actual time traveled to that of the shortest time (for travel between these two points).

DEFINITION 1. *Formally, let $m(\cdot,\cdot)$ be a metric that satisfies triangular inequality: $m(a,b)+m(b,c) \geq m(a,c)$, for all a, b and c. Then the stretch factor of a cab trip from location l_0 to l_n via $l_1, l_2 \cdots l_{n-1}$ is given by $\sum_{i=0}^{n-1} m(l_i, l_{i+1})/m(l_0, l_n)$. There are two such metrics $m(a,b)$: one based on the geographical shortest distance between points a and b (distance stretch factor) and the second based on the shortest time of travel between points a and b (time stretch factor).*

The key intuition is that taxicabs tend to travel on the shortest path routes (either based on distance or expected commute time), when occupied. Whereas, they tend to travel on sub-optimal routes (relative to the shortest path), when unoccupied. We note that this approach does not apply to detection of personal trips [19], where the movement patterns are significantly different from that of the taxicabs. We use the above intuition to partition a taxicab trace into multiple segments that correspond to shortest path routes (occupied state) and non-shortest path routes (unoccupied state). We show that this approach can significantly improve identification of taxicab occupancy status over past approaches [24], where traditional machine learning techniques (e.g., decision tree) were applied. In particular, we show that combining the concept of stretch factor (in a novel manner) with traditional machine learning techniques can significantly improve occupancy detection. We evaluate this idea using two extensive datasets, one from a deployment in Stockholm, Sweden and another from Shanghai, China to show that the stretch factor can be used to identify endpoints of a trip with 90-94% accuracy (in most cases).

DEPLOYMENT AND CHALLENGES

In this section, we will first describe taxicab deployments and datasets used in this paper and then illustrate the challenges involved in determining trip endpoints.

Datasets

The datasets used in this paper are from two independent taxicab based sensor deployments in two cities: Stockholm and Shanghai. Both these deployments collect periodic location sensor data from the movements of taxicabs using a GPS device with cellular network connectivity installed on each taxicab. Each sample comprises of *taxi ID*, *location (latitude/longitude)*, and *timestamp (UTC ms)*. The characteristics of both these datasets are summarized in Table 1, with location samples being acquired and recorded as long as the taxicab is operational.

In the Shanghai dataset, ground truth (endpoints) is determined for a portion of the dataset by manually marking trip endpoints. In the Stockholm deployment, due to privacy constraints, the installed devices did not acquire GPS samples when the taxicab is hired (occupied by a passenger). In this paper, we will use this lack of data to infer the ground truth, that is, the endpoints of a particular trip[1]. For the purpose of evaluation, we build a synthetic trace by filling in missing portions of the Stockholm taxicab traces using map-matched routes between the trip endpoints. This will bias our evaluation results, but will provide an upper bound on the accuracy of identifying occupancy of taxicabs. Also, in our evaluation we vary the stretch factor (i.e., we add progressively longer routes between trip endpoints) of the synthetic trace to study the effectiveness of our approach.

Characteristic	Stockholm	Shanghai
Sampling rate	1/min	2/min
Number of cabs	$\sim$ 2000	$\sim$ 10,000
Privacy	No sampling when taxi hired	None
Timeline	1 month	1 month
Total number of trips	570,690	1,335,360

Table 1. Characteristics of the Stockholm and Shanghai taxicab datasets used in this paper

For our analysis in this paper, we will use a month's data (as specified in Table 1) from each of Shanghai and Stockholm deployments. Also, wherever we utilize machine learning techniques, the evaluation metrics for determining the accuracy of these algorithms are *precision* and *recall*, definitions of which can be found in [9].

Challenges

This section introduces fundamental concepts and notations used in this paper and formalizes the problem of identifying trips from taxicab traces. We first introduce simple solutions to solve this problem – based on average taxicab speed and spatial clustering and classification. We then show that these simple solutions are not accurate in determining the endpoints and provide an intuition as to why these solutions are inadequate.

Problem Formulation

First, we present a more formal problem definition.

DEFINITION 2 (TRANSPORTATION NETWORK). *A transportation network is modeled as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V}$ representing the set of nodes, each corresponding to a GPS location, and $\mathcal{E}$ representing the set of directed edges over $\mathcal{V}$, each corresponding to a road link.*

The location of a taxi cab is mapped to the nearest point in the transportation network using the map-matching algorithm developed in [12].

[1]It appears that this privacy protection mechanism is inadequate for protecting trip privacy; however, it does hide the route taken by an occupied taxicab

DEFINITION 3 (NETWORK EVENT). *Each event e is a tuple of the form $x = \langle id, l, t\rangle$, where id denotes the taxi cab identifier, l represents a node in the transportation network (GPS location), and t denotes the timestamp of the event.*

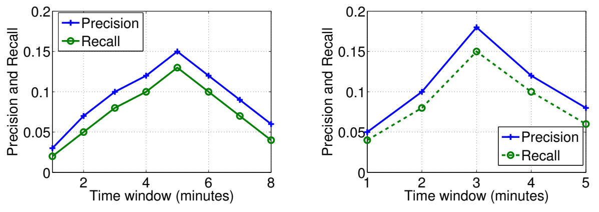
In the rest of this paper, we use id_x, l_x and t_x to denote the taxi cab identifier, location and timestamp of an event x, respectively.

Problem Definition. Given a sequence of network events $(x_1, x_2, \cdots, x_n)$ for a given taxicab, ordered by timestamp (i.e., $t_{x_i} \leq t_{x_{i+1}}$ for all $0 < i < n$), the goal is to identify all the trip endpoints (for that taxicab). Each trip endpoint is denoted by x_a, $1 \leq a < n$ (pickup point) and x_b, $1 < a < b \leq n$ (dropoff point).

Speed-Based Classification

A simple algorithm to identify endpoints from a continuous stream of location samples is to check if the *average speed* within a given time window is less than a threshold. First we normalize the speed of a taxicab at location l by historical speeds of all taxicabs on neighboring road segments that contain l. Historical speed at a location l at time t is computed as an exponentially weighted moving average (EWMA) that is biased towards recent speed measurements on road segments that adjoin location l. The intuition here is to: (i) compensate for different speed limits in different road segments, (ii) remove false positives – misclassifying slow down due to traffic light or stop signs at a junction as a candidate trip endpoint, and (iii) ensure that the normalization is robust against transient congestion in road traffic.

We train an SVM [9] (using Linear Discriminant Analysis) to learn a classifier over a taxicab's normalized speed; given a location trace $x_1, \cdots, x_n$ the classifier labels each x_i as an endpoint or not. Successive endpoint markings on a taxicab trace correspond to trips undertaken by the taxicab. Figure 1 plots the precision and recall in identifying trip endpoints when the time window size (over which average speed is computed) is increased for both the Shanghai and Stockholm datasets. We observe that in both the figures, the precision and recall are quite low, with the highest being around 0.16. The reason for such low precision and recall is that average speed often fails to be strong enough differentiator between slow moving traffic and endpoints of a particular trip.



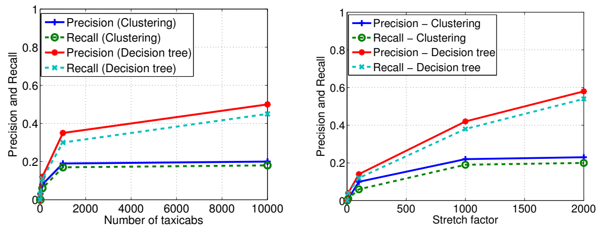
(a) Shanghai dataset (b) Stockholm dataset

Figure 1. Precision and recall for the speed based classification algorithm when time window size is increased

Clustering-Based Classification

Yet another solution for identifying endpoints of a trip is to use a spatial clustering algorithm, where the key hypothesis is that the expected number of cabs situated at a trip's endpoint (e.g., a tourist spot) is likely to be larger than that in the immediate neighborhood. In order to efficiently determine these endpoints, we insert taxicab location information into an R-tree [8] (an efficient hierarchical data structure composed of bounding boxes for indexing and retrieval of spatial locations).

We denote the count on the number of taxicab sightings in a spatial bounding box b as $count(b)$. Similar to the approach used in speed-based classification, an exponentially weighted moving average is applied when count is updated across time windows. We determine that a location is a likely trip endpoint if $count(b) \geq \gamma * count(parent(b))$, where $\gamma \leq 1$ is a tunable parameter and $parent$ is defined based on the spatial inclusion relationship on the R-tree. We determine the optimal γ (γ^*) empirically and use that for our subsequent analysis. We build and merge spatial clusters based on the optimal parameter γ^*; subsequently, we learn a SVM for classifying a cluster into a trip endpoint. Similar to the speed based classification algorithm, we train the model using data from all the taxicabs and test it on individual taxicabs[2]. Successive endpoint markings on a taxicab trace correspond to trips undertaken by the taxicab. Figure 2 plots the precision and recall of the SVM based classifier. We observe from Figure 2 that both the precision and recall are pretty low (less than 0.25) for both the datasets, showing that the clustering based classification approach does not work well. We also plot (in Figure 2) the performance of the approach presented in [24], which uses a decision tree based classification mechanism followed by a HSMM (Hidden Semi-Markov Model) based smoothing. We will refer to this approach as HSMM. We observe from Figure 2 that the HSMM approach performs with a precision/recall of 40-60% (which is consistent with the 40-70% range reported in [24]).

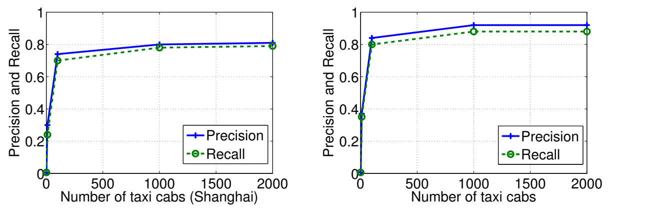


(a) Shanghai dataset (b) Stockholm dataset

Figure 2. Precision and recall for identifying trip endpoints when the number of cabs are increased for the clustering-based classification approach and the HSMM approach

In order to understand the reason for the clustering based classification algorithm's poor performance, we plot the accuracy with which this algorithm can classify a location as a candidate endpoint in Figure 3. We observe from Figure 3 that identifying endpoints is accurate (0.7-0.8 precision and recall in both the Stockholm and Shanghai datasets). We recall that a trip is a pair of endpoints, namely, the pickup point and the dropoff point. Hence, being able to classify a location as a candidate endpoint would not necessarily result in detecting trips from a taxicab trace.

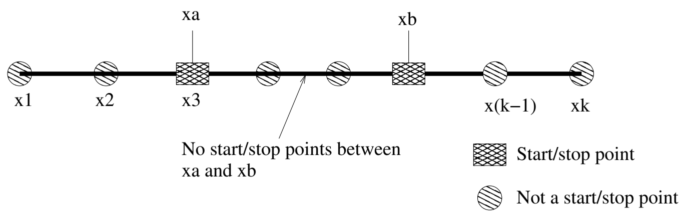
[2]In all our experiments, the training data set is different from the test data set



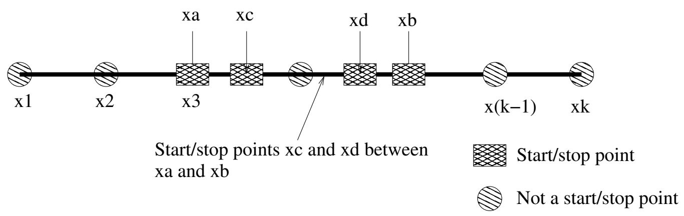
(a) Shanghai dataset (b) Stockholm dataset

Figure 3. Precision and recall for identifying endpoints (not part of a particular trip) when the number of cabs are increased for the clustering based classification approach

A closer investigation offered more insights into this problem. With reference to Figure 4(b) we observe that a segment of a taxicab trace x_a to x_b may include several candidate endpoints (e.g., x_c and x_d are candidate endpoints from other trips). Recall the example from the Introduction, wherein a taxicab makes a trip from `home` to `train station` and a popular endpoint `walmart` happens to be en route. Even though the actual trip is from x_a to x_b, a clustering based classification technique will identify multiple trips: x_a to x_c, x_c to x_d and x_d to x_b. Hence, naïve classification of locations from a taxicab trace into candidate endpoints is insufficient to identify trips, and is thus insufficient to infer human mobility patterns.



(a) Trip x_a to x_b has no other endpoints



(b) Trip x_a to x_b has other endpoints

Figure 4. Illustration of challenges involved in identifying endpoints from a stream of location data $(x_1, x_2, \ldots, x_k)$

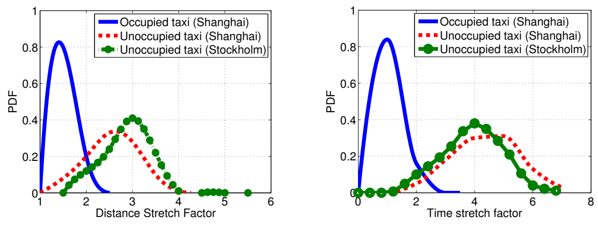
In the next Section, we will describe the novel concept of *stretch factor* in further detail and show how it can be used to overcome the challenges described above to determine trip endpoints.

HUMAN MOBILITY PATTERN INFERENCE

In this section, we will develop novel algorithms to identify trip endpoints. As described earlier in the Introduction, the algorithms we develop are based on the idea of *stretch factor*, that taxicabs when occupied traverse a route that has a stretch factor of nearly 1 and when unoccupied traverse a route with a stretch factor significantly greater than 1 (about 3-4). In what follows, we will show that indeed our above hypothesis is empirically valid for both the Stockholm and Shanghai datasets and then introduce our algorithms that use this stretch factor in determining trip endpoints. Note that the Stockholm dataset lacks inputs to compute the stretch factor of an occupied taxi; nonetheless, we show that the stretch factor for an unoccupied taxi is significantly greater than one.

Stretch Factor

The stretch factor forms the basis for our algorithms presented in the rest of this section. In order to provide the reader with an understanding about the use of stretch factor, we plot the probability distribution function (PDF) of the stretch factor (distance and time) for the cases when the taxicab is occupied and unoccupied for the Shanghai dataset in Figures 5(a) and 5(b), respectively. In the Stockholm dataset, we note that GPS samples were not acquired when the taxicab was occupied and hence, we plot the stretch factor for the unoccupied scenario only. For the Stockholm dataset, the PDF plot is generated for all the trips from all of the taxicabs (about 2000) over the course of one month. For the Shanghai dataset, the PDF plot is generated for about 10,000 trips (over the course of one month).



(a) Distance stretch factor (b) Time stretch factor

Figure 5. PDF of the stretch factor (distance and time) for taxi cabs in Shanghai and Stockholm when they are occupied and unoccupied

We observe from Figure 5 that the stretch factors differ significantly for the occupied case (in Shanghai) when compared to the unoccupied case. The stretch factor varies between 1 and 2.5 for the occupied case (Shanghai). On the other hand, when the taxicab is unoccupied, it is between 1 and 5, with 70% of the trips for Shanghai having a stretch factor greater than 2 and 95% of the trips with a stretch factor greater than 2 for Stockholm. We also believe that the stretch factor distribution for the occupied case in Stockholm will follow a similar distribution as that of Shanghai (occupied scenario). The difference is more pronounced for the time stretch factor (Figure 5(b)); in fact later, we will show that the time stretch factor results in better prediction accuracy. The algorithms we develop next will build on this stretch factor to identify trip endpoints (and thus the human mobility patterns). The hierarchical segmentation algorithm detects "maximal subsequences of shortest paths" and the HMM-stretch factor algorithm tracks the shortest path vector and detects significant deviations in this vector (the larger the deviation from the shortest path, the higher the stretch factor).

Hierarchical Segmentation

Given a stream of network events (e.g., segments of the transportation network, see definition 3) from one taxicab, this approach seeks to progressively merge segments with the goal of identifying maximal subsequences of taxicab locations that correspond to shortest paths over the transportation network. The algorithm starts with a sequence of atomic shortest path segments[3]: $(l_{x_1}, l_{x_2}), \cdots, (l_{x_i}, l_{x_{i+1}}), \cdots, (l_{x_{n-1}}, l_{x_n})$ and progressively merges two overlapping shortest path segments, namely, segments (l_{x_i}, l_{x_j}) and $(l_{x_{j'}}, l_{x_k})$ such that $i < j' \leq j < k$ if the stretch factor of the actual path taken by the taxi $(l_{x_i}, l_{x_{i+1}}, \cdots, l_{x_k})$ is smaller than a threshold β (for some $\beta \geq 1$). Intuitively, two consecutive trips are combined if the combined trip has a stretch factor that is smaller than a given threshold. Note that by construction the candidate trips that are being combined also satisfy the stretch factor requirement. Candidate trips are progressively combined until they violate the stretch factor constraint. The algorithm terminates when no two overlapping segments can be merged. The resulting shortest path segments extracted from the taxicab trace are purportedly indicative of regions where the taxicab was occupied; hence, the beginning and end locations of such segments correspond to trip endpoints. We remark that the computational complexity of this algorithm is $O(d * n)$ shortest path computations on the underlying road network, where n is the size of the taxicab trace and d is the length of the longest taxicab trip (measured in terms of the number of location samples). In general, $d \ll n$ and thus the overall running time of the algorithm is quasi-linear in n.

An extension to the hierarchical segmentation approach is to post-process the output of the above algorithm. We observed earlier that the clustering based algorithm can classify if a point is *an* endpoint or not with high accuracy (80-90% precision and recall). We use this algorithm (with a small variation) to post-process the output of the hierarchical segmentation. If the output of the hierarchical segmentation algorithm is x_i (for an endpoint of the trip), we consider x_{i-1} and x_{i+1} and query the clustering algorithm as to which of the three points (x_{i-1}, x_i, x_{i+1}) are the most likely endpoints. This post-processing reduces off-by-one errors and improves the accuracy of our algorithm.

Hidden Markov Model

A Hidden Markov Model (HMM) [15] is a statistical Markovian model where the system being modeled is assumed to be a Markovian process with *hidden* states. The challenge then is to determine these unknown (hidden) states from observable parameters. A HMM is characterized by the following: (i) N: the number of hidden states, (ii) M: the number of distinct observation symbols per state, (iii) α_{NXN}: state transition probabilities, (iv) B_{NXM}: observation symbol probability distribution for each state (also, known as emission probability), and (v) Π_{NX1}: initial state distribution. In this particular problem, the occupancy state of a taxicab is modeled as its hidden state, that is, hidden states $Y = \{o$: occupied, u: unoccupied$\}$. The taxi is assumed be at an unoccupied state at 4am (local time); hence, the initial probabilities are set as Π_u = 1 and Π_o = 0. We will first describe a straight-forward approach to use HMM and then describe our algorithm that combines HMM in a novel manner with the distance and time stretch factors.

[3]we assume that the taxicab takes the shortest path between two successive location samples – since sampling intervals are small this assumption has minimal impact in practice

HMM-Naïve

The transition probabilities $\alpha_{u,u}$, $\alpha_{u,o}$, $\alpha_{o,o}$ and $\alpha_{o,u}$ can be estimated in a naïve manner by computing the likelihood of transitioning between the occupied and unoccupied states (for each taxi cab) based on whether the next point is *a candidate endpoint* or not (similar to the clustering approach described in the previous Section). We will call this the *HMM-naïve* approach and use it as a baseline to measure the performance of our algorithm (HMM-Stretch factor). We also compare with past work [24] that utilizes HMM in a similar manner. In that paper, a decision tree was used to determine if the cab is occupied. This decision tree was trained on three categories of features, (i) trajectory, (ii) Points of Interest (PoI), and (iii) statistics from historical trajectories. The output of the decision tree is then smoothed using a HSMM (a variant of HMM).

HMM-Stretch factor

In our algorithm, we compute the likelihood of transitioning between the occupied and unoccupied states based on the stretch factor. We refer to this as the *HMM-Stretch factor* based approach. In what follows, we describe how to determine the occupancy of the cab for our HMM based approach.

The observable outputs from this Markov process are the sequence of network events $(x_1, x_2, \cdots, x_n)$. The goal is to determine the hidden state sequence $(y_1, y_2, \cdots, y_n)$ that is most likely to have produced the observations $(x_1, x_2, \cdots, x_n)$. We incorporate the stretch factor into the HMM by computing the emission probability as follows (for some tunable parameter $\triangle \geq 1$): $Pr(\{x_{i-\triangle}, \cdots, x_i, \cdots, x_{i+\triangle}\}|y_i = o) \propto 1/sf(l_{x_{i-\triangle}}, \cdots, l_{x_{i+\triangle}})$, where $sf(\cdot)$ denotes the stretch factor of a location trace.

In order to estimate the most likely hidden state, one can use the classic Viterbi decoding algorithm [15] where the emission probability matrix is given by $Pr(x_i|y_i)$, i.e., the probability of *one observation* given the hidden state. However, in our problem structure, emission probabilities are defined over a sequence of observations, namely, $Pr(\{x_{i-\triangle}, \cdots, x_i, \cdots, x_{i+\triangle}\}|y_i)$. Conceptually, this can be accommodated by modeling the i^{th} observation as $\widetilde{x}_i = \{x_{i-\triangle}, \cdots, x_i, \cdots, x_{i+\triangle}\}$. However, naively doing so results in a blowup in the size of the emission probability matrix because the size of the set of observations grows from $|O|$ to $|O|^{2\triangle+1}$. Given that $|O|$ can be in the order of thousands (each taxi cab location is a possible observation) and both computing and materializing (e.g., using an in-memory array) the emission probability matrix is infeasible for even small values of $\triangle$.

We circumvent this problem by not supplying the Viterbi decoding algorithm with a pre-computed emission probability matrix. Instead the algorithm is modified to invoke our method for every observation $\widetilde{x}_i = \{x_{i-\triangle}, \cdots, x_i, \cdots, x_{i+\triangle}\}$ it encounters. Our method computes and returns the $1/sf(\{x_{i-\triangle}, \cdots, x_i, \cdots, x_{i+\triangle}\})$ as a probability measure. A keen reader may recognize that $1/sf(\{x_{i-\triangle}, \cdots, x_i, \cdots, x_{i+\triangle}\}$ is not a well formed probability, since it does not essentially sum up to one. In order

to compute the true probability one would have to normalize the returned value as sf($\{x_{i-\triangle}, \cdots, x_i, \cdots, x_{i+\triangle}\}$) / SF, where
$SF = \sum_{\{x_{i-\triangle},\cdots,x_i,\cdots,x_{i+\triangle}\}} sf(\{x_{i-\triangle}, \cdots, x_i, \cdots, x_{i+\triangle}\})$. Note that computing the normalization factor SF incurs a time complexity of $|O|^{2\triangle+1}$. Fortunately, the Viterbi decoding algorithm yields identical results as long as all probabilities are scaled by the same factor – even if it is an *unknown* factor SF.

We remark that the computational complexity of this algorithm is $O(e * n)$ shortest path computations on the underlying road network, where n is the length of the taxicab trace and e is the cost of computing shortest path lengths on the underlying (weighted) road network. In contrast, the hierarchical segmentation approach is far more efficient at $O(d*n)$ (where d is the length of the longest taxicab trip, which in turn is bounded by the diameter of the underlying road network). Because of the hierarchical nature of the segmentation algorithm, its complexity is limited by the diameter d of the network (in fact, it is limited by the length of the longest taxicab trip which is typically shorter than d). In the HMM-based approach the state space of the Markovian model is linear in the number of nodes in the network and thus the complexity of applying Viterbi decoding is quadratic in the size of the network the optimizations suggested above can reduce this to linear in the number of edges in the network (edges in a sparse graph is typically much smaller than the square of the number of nodes).

Finally, we remark that both the HMM algorithms can be extended by post-processing the output (endpoints) of the HMM using the clustering based classifier (similar to the extension that we described for the hierarchical segmentation approach). We will refer to this as the *HMM-clustering* approach. We consider three points x_{i-1}, x_i, x_{i+1}, where x_i is the output of the HMM (an endpoint of a particular trip), and query the classifier as to which of the three points is the most likely endpoint. This is the output of the post-processing step, which we evaluate in the next section.

EVALUATION

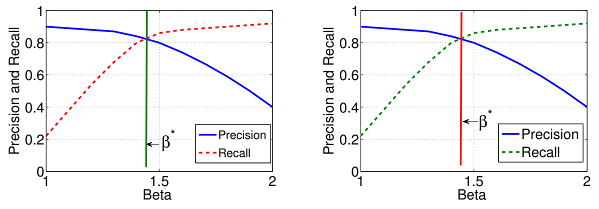
We divide this evaluation section into two subsections, the first one evaluates the performance of our algorithms based on stretch factor and the second provides an end-to-end evaluation of the mobility pattern detection algorithms. For our evaluation, we use one month's data from each of the datasets and the results are derived (wherever applicable) by averaging across all the trips made by all the taxicabs (for the corresponding dataset). We use two weekdays and one weekend trace for training and the rest of the dataset is used for evaluation. We build separate models for weekdays and weekends to capture inherent differences in human mobility patterns: some notable differences include, changes to hotspots (frequently visited locations) during weekends, increase in the length of an average commute distance (by over 60% in Stockholm and 50% in Shanghai datasets) during weekends, and shift in morning busy hour from 8am on weekdays to 10am on weekends. We note again that the Stockholm dataset is partially synthetic (when the taxicab is occupied), but the stretch factor of the synthetic trace is varied in a controlled manner. This synthetic dataset (wherein all taxicab trips have a stretch factor of one) is used to provide an upper bound on the accuracy (precision and recall).

Trip Endpoints Determination

We have shown earlier that the human mobility pattern identification problem is equivalent to identifying a trip and its corresponding endpoints (pickup and dropoff locations). We proposed two algorithms, one that is based on hierarchical segmentation and the second that is based on HMM, whose evaluation results we present below.

Hierarchical Segmentation

In the hierarchical segmentation algorithm, the primary tunable parameter is the stretch factor threshold β. In order to determine the optimal value of β (β^*), we plot the precision and recall (averaged over all the taxicabs) for varying β for both the datasets in Figure 6.



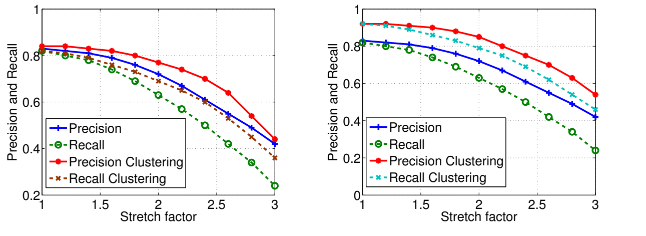
(a) Shanghai dataset (b) Stockholm dataset

Figure 6. Precision and recall for varying β, to determine optimal $\beta(\beta^*)$ for the hierarchical segmentation approach

We observe from Figure 6 that in order to achieve a high precision and recall, the optimal value of β (β^*) is around 1.4 for both the datasets. As per this criterion, taxicab location traces whose stretch factor is under β=1.4 will be inferred as trips. We will use this empirically derived value of β in our evaluation (for the hierarchical segmentation algorithm).

For our next experiment, we vary the stretch factor of an occupied taxicab between the pickup and dropoff points of a particular trip for both the datasets. In the Stockholm dataset, since the data when a taxicab is occupied is absent, we artificially generate the data. The method we use to do this is as follows: given a stretch factor, we enumerate all the possible paths between the pickup and dropoff locations that are within a certain threshold (τ) times the stretch factor and randomly choose one of these paths for evaluation. The reason for introducing τ is that there may not exist any routes with the exact stretch factor between two arbitrary points. In our experiments we configure τ to permit no more than 10% error on the stretch factor. On the other hand, for the Shanghai dataset, we do not need to generate these artificially, as the actual route is available. We plot precision and recall for the case of using only the hierarchical segmentation and the case where we post-process the output to determine if a particular point is an endpoint, in Figure 7.

We observe from Figure 7 that the precision and recall fall when the stretch factor of the occupied taxicab is increased. In particular, if the stretch factor of an occupied cab is greater than 2, the precision and recall begin to fall below 70%, resulting in poor identification of trips. Nonetheless, we observe from the Figure 5(a) in Section *Stretch Factor* that in 96% of the cases (for Shanghai dataset), the stretch factor of



(a) Shanghai dataset (b) Stockholm dataset

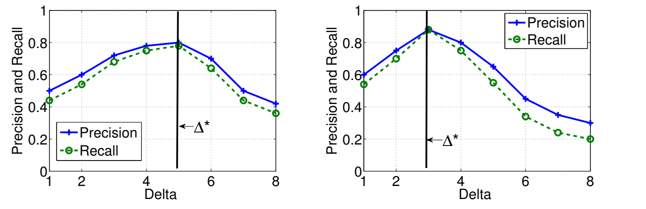
Figure 7. Precision and recall for varying stretch factor for the simple hierarchical segmentation and the hierarchical segmentation with a clustering based post-processing approach

an occupied taxicab is less than 2; hence, the resulting precision and recall for identifying trips is fairly high in practice.

In the next section, we present results from the HMM based algorithm. Intuitively, the HMM stretch factor algorithm should perform better than a simple hierarchical segmentation algorithm as the HMM considers the time-series nature of the location data traces.

Hidden Markov Model

In this section, we evaluate the performance of the HSMM approach [24] and the three proposed HMM approaches, HMM-naïve, HMM stretch factor, and HMM-clustering (HMM stretch factor with output postprocessing). For the HSMM approach, we implement the corresponding decision tree with all the features that were listed in the paper [24] and choose the time parameter for smoothing the output empirically (that gives the best result). Similarly, for our HMM stretch factor based approach we tune the parameter Δ, that defines the time window of observations over which emission probabilities are defined. In order to obtain the optimal Δ (Δ^*), we vary Δ and plot the precision and recall in identifying trip endpoints for both the datasets in Figure 8 (for HMM and HMM-clustering approaches). We compute the Δ^* in a similar fashion for the HMM-naïve approach. We observe from Figure 8 that the Δ^* for Stockholm is around 3 and that of Shanghai is about 5. In the following portions of this section, we use the above values for Δ.



(a) Shanghai dataset (b) Stockholm dataset

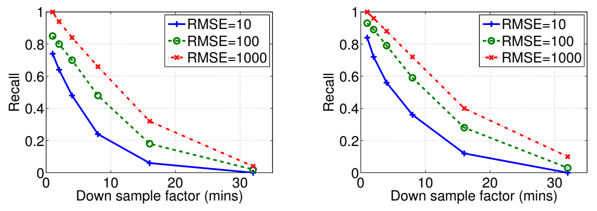
Figure 8. Precision and recall when Δ is varied, to determine the optimal $\Delta(\Delta^*)$ for the HMM based algorithm

In Figure 9, we plot the precision and recall for the three variants of the HMM based algorithm along with the HSMM (decision tree) algorithm when the stretch factor of the occupied taxicab is increased. We follow a similar approach to obtain these results as in the hierarchical segmentation case. The first is a naïve-HMM approach, which will be a baseline for measuring the performance of our algorithms, the second is based on a previous approach [24], the third simply uses the stretch factor hypothesis to determine trip endpoints and the fourth applies a post-processing step that refines the trips generated by the HMM stretch factor model based on the likelihood of a point being an endpoint.

We observe from Figure 9 that the latter two approaches (HMM, HMM-clustering) outperform the HMM-naïve approach (by a factor of 4-5) and the HSMM approach (by a factor of about 2), suggesting that the stretch factor is indeed key to effectively solve this problem. We further observe from Figure 9 that while the precision and recall drops as the stretch factor of the occupied taxicab is increased, the drop is not as marked as in the hierarchical segmentation approach (Figure 7). The HMM based algorithm with the clustering based post-processing falls below 70% (precision and recall) for both the datasets at around a stretch factor of 2.5. In practice (as observed from Figure 5(a)), the stretch factor is typically around 1.6, which gives a precision and recall of about 90-95% in the Stockholm and Shanghai datasets.

We also remark that the performance of the HMM-naïve approach (which is not based on stretch factor) also falls when the stretch factor is increased. The reason for this performance degradation is the way we increase the stretch factor, by progressively selecting longer (sub-optimal) routes between trip endpoints. With increasingly longer routes, the number of possible endpoints (that are not essentially end points of this trip – e.g., they may be landmarks en route on this trip) encountered by the baseline algorithm also increases. In summary, we note that the HMM based approach outperforms the hierarchical segmentation based approach in identifying trip endpoints, the best among the four approaches being the HMM-clustering approach.

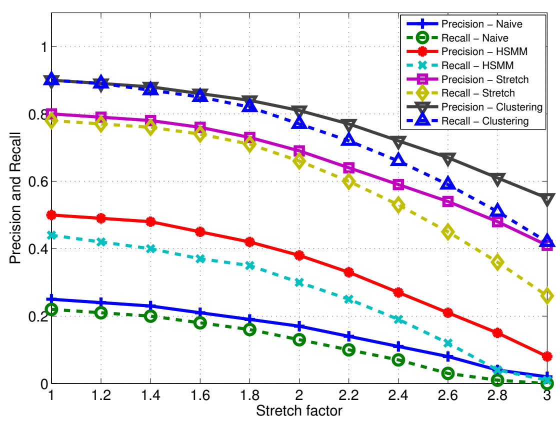
We now study the performance of HMM-clustering algorithm when the sampling rate (the number of samples obtained per minute) for the datasets are increased. We fix distance precision, that is, the error (RMSE in Figure 10) in meters in identifying the location of the trip endpoints. For example, an error of 100 meters indicates that the trip endpoints are identified within a 100 meter radius of the actual location of the endpoint. We vary the sample time (in minutes) and plot the recall for both the Stockholm and Shanghai datasets for different errors (precision) in Figure 10.



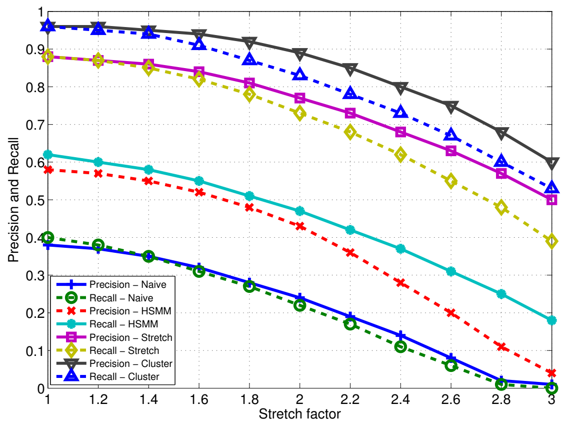
(a) Shanghai dataset (b) Stockholm dataset

Figure 10. Recall when downsampling is increased for three different fixed precisions using the HMM combined with clustering based post-processing algorithm

We observe from Figure 10 that the recall quickly drops when the sampling rate is increased beyond once every 3 minutes. This means that the sampling rate has to be maintained at around once every two minutes in order to be able to identify trip endpoints accurately.



(a) Shanghai dataset



(b) Stockholm dataset

Figure 9. Precision and recall when the stretch factor is varied for the HSMM and HMM based approaches, the first HMM is a baseline approach, the second uses only the stretch factor and the third applies a post-processing step based on clustering

Time-based Stretch Factor

We evaluate the performance of the HMM based algorithm when the time stretch factor (as defined in Section) is varied for both Shanghai and Stockholm data sets in Figure 11(a) and Figure 11(b), respectively. We compute the fastest routes between two given points based on historical speed averages.

We observe from Figure 11 that the stretch factor based approaches (HMM and HMM-clustering) outperform stretch factor agnostic approaches (HMM-naïve and HSMM). We further observe (on comparing the Figures 11 and 9) that the performance of HMM and HMM-clustering improves when using the time stretch factor as opposed to using the distance based stretch factor. In conclusion, these results indicate that the best approach for identifying trip endpoints is using a HMM based algorithm combined with a clustering based post-processing step (to identify the likelihood of a point being *an* endpoint) and the time stretch factor. We further note that there exist certain drawbacks of using the concept of stretch factor to determine end points - in the situation that the taxicab picks passengers consecutively, the multiple trips may be identified as a single trip. We observe that this will occur only when the shortest path between trip 1's start point and trip 2's end point is a combination of trip 1's and trip 2's shortest paths. Further, the above approach will also be influenced by the behavioral patterns of taxicab drivers, e.g., certain cab drivers may always choose to take the shortest path irrespective of whether they are occupied or not.

Computation Time

Finally, Table 2 compares the computation time of various schemes for identifying taxicab occupancy. As expected, we observe that hierarchical segmentation is the fastest (about one order faster than HMM-based approaches), since it does not account for smoothness constraints in HMM models. HMM-naive and HSMM are faster than HMM-stretch factor and HMM-clustering because the former two are provided with hardcoded emission probability matrices; on the other hand, in HMM-stretch factor the emission probabilities are too large to be materialized, and thus computed on the fly (by evaluating shortest paths on the underlying transportation network). Finally, we note that the post processing step (namely, refining the endpoint based on its likelihood) adds very little overhead. In summary, our approach incurs about 20% additional computation time; but results in 30-40% improvement in accuracy.

Mobility Patterns

So far, we have evaluated the performance of our algorithms in identifying trip endpoints. In the rest of this section, we will illustrate an end-to-end evaluation of our system. We show the mobility patterns inferred by applying the HMM with clustering based post processing algorithm (using time stretch factor) to identify trip endpoints. We identified the top ten most traveled routes for both the datasets from the trip endpoints to infer the frequently traveled routes in the cities of Shanghai and Stockholm (see Figure 12 and Figure 13).

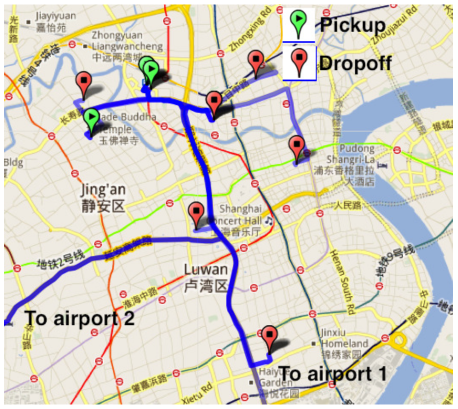
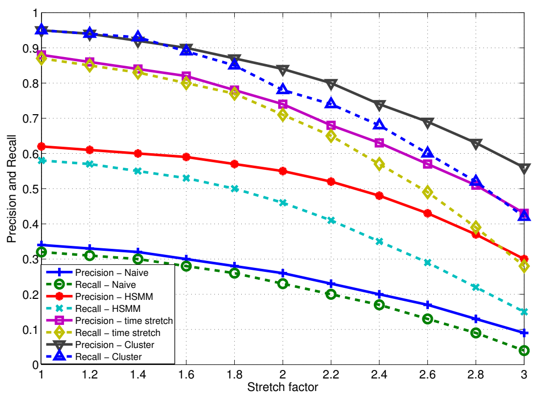
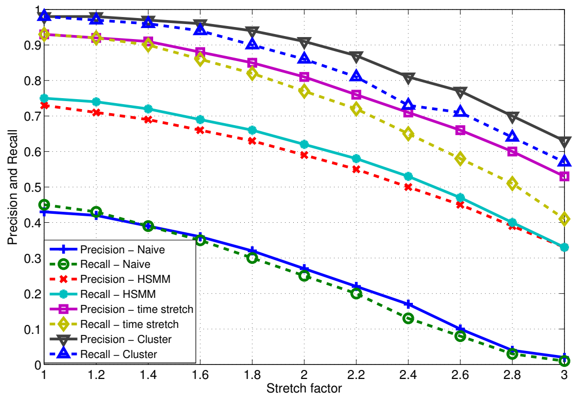


Figure 12. Top ten most traveled routes for Shanghai

We observe from Figure 12 that the origin of the top ten trips are very few and the two main origins are the airports in Shanghai. We do not explicitly show the markers for the airport for map-readability (the airports are located in the south and south-west portions of the map). Further, most endpoints are in the downtown area. There are quite a few similarities in the Stockholm map as well, which is shown in Figure 13. For



(a) Shanghai dataset



(b) Stockholm dataset

Figure 11. Precision and recall when the time based stretch factor is varied for the HSMM and HMM based approach, the first HMM is a baseline approach, the second uses only the stretch factor and the third applies a post-processing step based on clustering

Dataset	Hierarchical Segmentation	HMM naive	HSMM [24]	HMM Stretch Factor	HMM Clustering
Stockholm	2.1	23.4	35.2	41.1	41.3
Shanghai	3.2	29.4	51.4	63.4	63.7

Table 2. Computation Time (seconds): Taxicab Trace Length 10,000

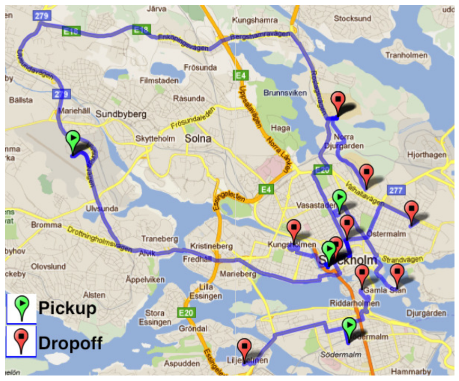


Figure 13. Top ten most traveled routes for Stockholm

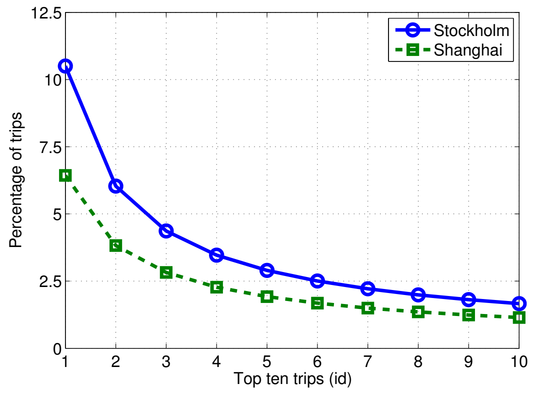


Figure 14. Number of trips in one month on the top ten most traversed roads as a percentage of the total trips made for Shanghai and Stockholm

example, the origin of the trips are also only a few (only five distinct origins amongst the top ten trips), whereas the destinations are different. Typically, most endpoints of a trip are again in the downtown area. Further, the pickup point in the west area of the map in Figure 13 is the Stockholm airport. A main difference between these maps is that the most frequent pickup point in Stockholm is the Central Station (not shown explicitly on the map), whereas it is the airports in Shanghai.

We also plot the number of trips made as a percentage of the number of total trips per day on the top ten routes in Figure 14 (averaged across all the days in a month). We observe from Figure 14 that the popularity of trips follow a heavy tailed distribution. Using standard distribution fitting methods we observe that the popularity of trips in Stockholm and Shanghai follows a Zipf distribution [3] with an exponent of 0.81 and 0.75 respectively. Indeed such end-to-end statistics may be leveraged in several city planning (e.g., new train routes, cell tower deployment) applications.

RELATED WORK

In this section, we summarize related work on human mobility patterns and various data mining techniques in this context. The closest related work is presented in [24], where a decision tree based approch is proposed. A HSMM based smoothing technique is applied to the output of the decision tree. Our algorithms perform 30-40% better than this approach, as shown in the evaluation section. A similar approach was used in [21], where the authors differentiate between a taxi cab that is parked and moving in slow traffic. This is different from identifying the occupancy status of a taxicab.

Human mobility patterns have been well studied in the past [7, 11, 16, 18, 20, 6, 22]. In [7], mobility patterns were analyzed from over 100,000 anonymized mobile phone location sensor data collected over six months. It was shown that human trajectories show a high degree of temporal and spatial regularity. This spatio-temporal regularities were substantiated in [18]. In [6], mobility traces from about 10,000 private cars in Italy were analyzed to obtain mobility patterns of people. A similar dataset (GPS traces from 2000 private cars in Italy) was used in [20] to extract mobility patterns by building mobility profiles. In the paper [22], human mobility patterns collected from taxicabs were applied to detect flaws in urban planning (in Beijing). In [11], location sensor data from taxicabs have been used to characterize human mobility patterns. In that paper, the pickup and dropoff points (endpoints of a trip) were manually marked. Our work focuses on inferring human (passenger) mobility patterns from taxicabs when explicit occupancy information is lacking and is complementary to the above papers.

CONCLUSIONS AND FUTURE WORK

In this paper, we developed several algorithms to identify human mobility patterns from taxicab location traces. We illustrated that taxicabs are a good platform for identifying human mobility patterns and that this problem is equivalent to identifying trips made by the taxicab. Our analysis showed that simple clustering techniques can accurately identify if a particular location is an endpoint, but in order to identify endpoints corresponding to a particular trip, it is inadequate. We developed algorithms to address this concern that use stretch factor in a novel way. The HMM based algorithm, which outperforms the hierarchical segmentation algorithm, achieves 90-94% precision and recall in identifying trips for both the Stockholm and Shanghai datasets. We also showed that the HMM based algorithm outperforms past work (HSMM) by nearly a factor of 2. We used the output of the HMM based algorithms to compute the top ten frequently traveled routes in both the cities. Further, we note that applying knowledge discovery algorithms in a naïve manner will not yield good results and new features will be necessary. As future work, we will explore the applicability of these concepts to other types of vehicle data such as delivery trucks (e.g., FedEx, UPS) to identify offloading and onloading of goods and ambulance movement to derive correlation between patient locations and hospitals/clinics.

ACKNOWLEDGMENTS

The authors would like to thank Shanghai Jiao Tong University for access to the Shanghai taxicab trace data, which was obtained from the Wireless and Sensor Networks Lab (WnSN). Further, we wish to thank Haris Koutsopoulos from KTH, Stockholm, Erling Weibust from IBM Sweden, Inga Maj Eriksson from the Swedish Road Administration and Tomas Julner from Trafik Stockholm for their support and provision of the data from the Stockholm taxicab deployment. We would also like to thank the anonymous reviewers and our shepherd Mikkel Baun Kjærgaard for their feedback that helped improve the paper. Research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

REFERENCES

1. New york city taxi and limousine commission. http://www.nyc.gov/html/tlc/html/home/home.shtml.
2. Trafik stockholm. http://www.trafikstockholm.com.
3. Barabasi, A.-L., and Albert, R. Emergence of scaling in random networks. In *Science* (1999).
4. Cheng, S.-W., Knauer, C., Langerman, S., and Smid, M. H. M. Approximating the average stretch factor of geometric graphs. *JoCG 3*, 1 (2012), 132–153.
5. Ganti, R. K., et al. Greengps: A participatory sensing fuel-efficient maps application. In *Proc. of MobiSys* (2010), 151–164.
6. Giannotti, F., et al. Unveiling the complexity of human mobility by querying and mining massive trajectory data. In *Proc. of VLDB* (2011), 695–719.
7. Gonzalez, M., Hidalgo, C., and Barbasi, A.-L. Understanding individual human mobility patterns. *Nature 453* (2008), 779–782.
8. Guttmann, A. R-trees: A dynamic index structure for spatial searching. In *Proceedings of ACM SIGMOD* (1984), 47–57.
9. Hastie, T., Tibshirani, R., and Friedman, J. The elements of statistical learning: Data mining, inference and prediction. In *Springer Series in Statistics, Second Ed* (2009).
10. Hull, B., et al. Cartel: a distributed mobile sensor computing system. In *Proc. of SenSys* (2006), 125–138.
11. Jiang, B., Yin, J., and Zhao, S. Characterizing human mobility patterns in a large street network. *Physical Review E 80*, 2 (2008).
12. Jones, K., Liu, L., and Alizadeh-Shabdiz, F. Improving wireless positioning with look-ahead map-matching. In *Proc. of Mobiquitous* (2007), 1–8.
13. Lu, H., et al. Soundsense: Scalable sound sensing for people-centric applications on mobile phones. In *Proc. of ACM MobiSys* (2009), 165–178.
14. Piorkowski, M., et al. A parsimmonious model of mobile partitioned networks with clustering. In *Proc. of COMSNETS* (2009), 1–10.
15. Rabiner, L. R. A tutorial on hidden markov models and selected applications in speech recognition. In *Proc. of IEEE* (1989), 257–286.
16. Rhee, I., et al. Human moblity patterns and their impact on routing in human-driven mobile networks. In *Proc. of HotNets-VI* (2007).
17. Shanghai Jiao Tong University. SUVnet-Trace Data. http://wirelesslab.sjtu.edu.cn.
18. Song, C., Qu, Z., Blumm, N., and Barbasi, A.-L. Limits of predictability in human mobility. *Science 327*, 5968 (2010), 1018–1021.
19. Stenneth, L., et al. Transportation mode detection using mobile phones and gis information. In *Proc. of GIS* (2011), 54–63.
20. Trasarti, R., Pinelli, F., Nanni, M., and Giannotti, F. Mining mobility user profiles for car pooling. In *Proc. of KDD* (2011), 1190–1198.
21. Yuan, J., et al. Where to find my next passenger? In *Proc. of UbiComp* (2011), 109–118.
22. Zheng, Y., Liu, Y., Yuan, J., and Xie, X. Urban computing with taxicabs. In *Proc. of UbiComp* (2011), 89–98.
23. Zhu, H., et al. Recognizing exponential inter-contact time in vanets. In *INFOCOM* (2010).
24. Zhu, Y., et al. Inferring taxi status using gps trajectories. Tech. Rep. MSR-TR-2011-144, Microsoft Research, Asia, March 2011.