2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology

Hierarchical Bayesian Nonparametric Approach to Modeling and Learning the Wisdom of Crowds of Urban Traffic Route Planning Agents

Jiangbo Yu[†], Kian Hsiang Low[†], Ali Oran[§], Patrick Jaillet[‡]
Department of Computer Science, National University of Singapore, Republic of Singapore[†]
Singapore-MIT Alliance for Research and Technology, Republic of Singapore[§]
Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, USA[‡]
{yujiang, lowkh}@comp.nus.edu.sg[†], *aoran@smart.mit.edu*[§], *jaillet@mit.edu*[‡]

***Abstract*—Route prediction is important to analyzing and understanding the route patterns and behavior of traffic crowds. Its objective is to predict the most likely or "popular" route of road segments from a given point in a road network. This paper presents a hierarchical Bayesian non-parametric approach to efficient and scalable route prediction that can harness the wisdom of crowds of route planning agents by aggregating their sequential routes of possibly varying lengths and origin-destination pairs. In particular, our approach has the advantages of (a) not requiring a Markov assumption to be imposed and (b) generalizing well with sparse data, thus resulting in significantly improved prediction accuracy, as demonstrated empirically using real-world taxi route data. We also show two practical applications of our route prediction algorithm: predictive taxi ranking and route recommendation.**

***Keywords*-wisdom of crowds; crowdsourcing; sequential decision making; route prediction; intelligent transportation systems; hierarchical Dirichlet and Pitman-Yor process**

I. Introduction

Knowing and understanding the route patterns and behavior of traffic crowds has grown to be critical to the goal of achieving smooth-flowing, congestion-free traffic [**?**], especially in densely-populated urban cities. Of central importance to analyzing the traffic route patterns and behavior is that of route prediction whose objective is to predict the most likely or "popular" route of road segments from a given point[1] in a road network. Such a capability is useful in many real-world traffic applications such as studying drivers' route preferences, enhancing their situational awareness, supporting drivers' safety and assistance technologies, among others (see Section V). In practice, it is often not possible to make perfect route predictions because (a) the quantity of sensing resources (e.g., loop detectors) that can be allocated to measure traffic flow in a dense urban road network is cost-constrained, and (b) the actual traffic flow is usually unknown or difficult to formulate precisely in terms of other traffic-related variables since it varies depending on the driver types (e.g., truck or taxi/cab drivers, tourists) and road conditions/features (e.g., tolls, congestion, traffic lights, speed limit, road length, travel time). In the absence of accurate traffic flow information to determine the most popular route(s) exactly, we propose to rely on crowdsourcing for route prediction in this paper, specifically by consulting the wisdom of traffic crowds.

The proliferation of low-cost GPS technology has enabled the collection of massive volumes of spatiotemporal data on large crowds of human-driven vehicles such as taxis/cabs, from which we can exploit to learn about the patterns of their planned routes to be used for route prediction. To elaborate, though these human agents have different levels of expertise and criteria affecting their individual ability to compute and plan routes, consensus patterns can generally be observed among their planned routes in real-world traffic networks [1], [2]. If the crowds of route planning agents are sufficiently large, these consensus patterns potentially approximate the actual most popular routes well, the latter of which are not available as mentioned earlier. Hence, we will be able to harness the wisdom of such traffic crowds for learning the route prediction capability.

The "wisdom of crowds" phenomenon refers to the observation that the aggregation of solutions from a group of individual agents often performs better than the majority of individual solutions [3]. Traditionally, such a phenomenon has been widely applied to simple point estimation of continuous-valued physical quantities [3] and discrete labeling [4], [5]. In contrast, our challenge here is to aggregate more complex structured outputs (in particular, sequential decisions) from the crowds of route planning agents for performing route prediction. There are several non-trivial practical issues confronting this challenge:

- *Real-time prediction and scalable learning*. For the consensus patterns to closely approximate the actual most popular routes, a huge amount of traffic data (i.e., planned routes) is needed to filter out the noise and variation arising from different agents' individual planning abilities. Furthermore, prediction has to be performed in real time in order to be useful for the applications described above. How then can an aggregation model be built to learn scalably and predict efficiently?
- *Sequential nature of routes*. An agent's history of traversed road segments can potentially affect its future route. For example, suppose that two agents originating from different locations share some common route and subsequently split up to go to different destinations. This example raises an important trade-off question: How much history is needed to predict the future route

[1] Besides a given origin, a destination is sometimes also specified (see Section V-B).

978-0-7695-4880-7/12 $26.00 © 2012 IEEE
DOI 10.1109/WI-IAT.2012.216

accurately? Existing literature have assumed a Markov property [6], [7], that is, the future route of an agent depends only on its current road segment. This is clearly violated in the example; longer route histories have to be considered in order to predict the future routes of the two agents correctly.
- *Data sparsity*. Though the amount of traffic data is huge, it may only sparsely populate a dense urban road network. As a result, many road segments are not traversed by any agent. How can an aggregation model still be built to predict accurately when the data is sparse?

This paper presents a hierarchical Bayesian non-parametric approach to efficient route prediction (Section III) that, in particular, can exploit the wisdom of crowds of route planning agents by aggregating their sequential routes of possibly varying lengths and origin-destination pairs. It resolves the above-mentioned issues in the following ways:

- *Incremental/Online learning*. Learning is made scalable in an incremental/online fashion. In this manner, our aggregation model can be trained with a small amount of data initially and updated as and when new training data arrives. We show that our incremental/online learning algorithm incurs linear time in the size of training data (Section III-D) and can satisfy the real-time requirement (Section IV-D).
- *Non-Markovian property*. Our route prediction algorithm does not require imposing a Markov assumption (Section III-C). Consequently, the future route of an agent can depend on the complete history of traversed road segments. Without imposing a Markov assumption, the prediction accuracy can be improved considerably, as demonstrated empirically using real-world taxi route data (Section IV-B).
- *Smoothing*. Smoothing (Section III-D) is an effective way to deal with the issue of sparsity. The key idea is to add some pseudo agents and make them travel along road segments that are not traversed by any agent in the traffic data. As a result, the prediction accuracy can be significantly improved, as shown empirically in Section IV-C.

Section V shows two practical applications of our route prediction algorithm: predictive taxi ranking and route recommendation.

II. Background

A. Notations and Preliminaries

Definition 1 (Road Network): The road network $G = (V, E)$ is defined as a directed graph that consists of a set V of vertices denoting intersections or dead ends and a set E of directed edges denoting road segments.

Definition 2 (Route): A route R of length i is defined as a walk in the road network G, that is, $R \triangleq (r_1, r_2, \ldots, r_i)$ where the road segments $r_k, r_{k+1} \in E$ are adjacent for $k = 1, \ldots, i-1$.

B. Problem Formulation

Recall from Section I that the objective of route prediction is to predict the most likely future route given a history of traversed road segments. We achieve this in a probabilistic manner: Let the route R' denote the sequence of road segments to be traversed in the future, that is, $R' \triangleq (r_{i+1}, \ldots, r_m)$ for some $m \geq i+1$. Given a set $\mathcal{D}$ of agents' routes of possibly varying lengths and origin-destination pairs, the most likely future route conditioned on a past route R is derived using

$$\max_{R'} P(R'|R, \mathcal{D}) \ .$$

It is computationally challenging to compute the posterior probability $P(R'|R, \mathcal{D})$ for every possible past route R because the number of possible past routes is exponential in the length of history of traversed road segments. To reduce this computational burden, one can decrease the length of history by imposing a Markov property:

$$P(R'|R, \mathcal{D}) = P(R'|R_n, \mathcal{D}) \tag{1}$$

where R_n denotes a reduced route of R consisting of only the past n road segments. That is, $R_n \triangleq (r_{i-n+1}, \ldots, r_i)$ for $1 \leq n \leq i$ and R_0 is an empty sequence. Note that n is usually known as the order of Markov property.

C. Maximum Likelihood Estimation

A simple widely-used approach to model the posterior probability $P(R'|R_n, \mathcal{D})$ is that of *maximum likelihood estimation* (MLE) [6]. Let c_{R_n} and $c_{R_n \cdot R'}$ denote, respectively, the count of R_n and the count of R_n concatenated with R' observed within the routes in the dataset $\mathcal{D}$. So, $c_{R_n} = \sum_{R'} c_{R_n \cdot R'}$. Then, $P(R'|R_n, \mathcal{D})$ (1) can be approximated using MLE by

$$P_{\text{MLE}}(R'|R_n, \mathcal{D}) = \frac{c_{R_n \cdot R'}}{c_{R_n}} \ .$$

In practice, MLE often performs poorly, especially when the data is sparse. If a route R' has not been observed within the routes in the dataset $\mathcal{D}$, then $P_{\text{MLE}}(R'|R_n, \mathcal{D}) = 0$, which may not be the true value of $P(R'|R_n, \mathcal{D})$ (1). It may be due to sparse data that such a route has not been observed. In reality, it may still be associated with a small probability in the true distribution. Researchers from linguistic community have developed smoothing methods to resolve this issue. We will discuss them next.

D. Smoothing Methods

To tackle sparsity, the simplest form of smoothing makes a prior assumption that every possible future route R' is observed once more than it already has. Then, this smoothing method approximates $P(R'|R_n, \mathcal{D})$ (1) by

$$P_{\text{S}}(R'|R_n, \mathcal{D}) = \frac{c_{R_n \cdot R'} + 1}{c_{R_n} + |\mathcal{R}'|} \tag{2}$$

where $\mathcal{R}'$ is the set of all possible future routes R' of the same length. It can be observed from (2) that every route R' has at least a small probability $1/|\mathcal{R}'|$ of being traversed even if it is not observed in the dataset (i.e.,

$c_{R_n \cdot R'} = 0$). Hence, this smoothing method mitigates the issue of sparsity.

To improve prediction accuracy, a more advanced method called interpolated smoothing can be used. It performs a weighted average of smoothing probabilities (2) with different orders of Markov property and is formulated as follows:

$$P_{\text{I}}(R'|R_n, \mathcal{D}) = \lambda_n P_{\text{S}}(R'|R_n, \mathcal{D}) + (1-\lambda_n) P_{\text{I}}(R'|R_{n-1}, \mathcal{D})$$

where $\lambda_n \in [0, 1]$ is a fixed weight applied to the n-order smoothing probability. To understand why interpolated smoothing can work well, supposing R_n cannot be observed within the routes in dataset $\mathcal{D}$ but some shorter routes $R_m (m < n)$ within the route R_n can, it is reasonable then to involve smoothing probabilities with these lower m-order Markov property to yield more accurate prediction.

Another advanced method called *interpolated Kneser-Ney* (IKN) smoothing estimates the probability of the future route by discounting the count $c_{R_n \cdot R'}$ by a fixed weight d and interpolating with lower-order probabilities:

$$P_{\text{IKN}}(R'|R_n, \mathcal{D}) = \frac{\max(0, c_{R_n \cdot R'} - d)}{c_{R_n}} + \frac{d\ t_n}{c_{R_n}} P_{\text{IKN}}(R'|R_{n-1}, \mathcal{D}) \tag{3}$$

where $t_n \triangleq |\{R' | c_{R_n \cdot R'} > 0\}|$ denotes the number of distinct future routes R' after traversing the past route R_n in the dataset. The role of t_n is to make the probability estimates sum to 1. *Modified Kneser-Ney* (MKN) smoothing generalizes IKN smoothing by making the discounting weight depend on the length of R_n. Interested readers are referred to [8] for more details. In the next section, we will describe our route prediction algorithm that exploits some form of smoothing to resolve the issue of sparsity.

III. Sequential Model for Aggregating Routes

A. Non-Markovian Aggregation Tree

Since a Markov property is not imposed, we need to address the challenge of aggregating and storing an enormous set $\mathcal{D}$ of agents' sequential routes efficiently. To achieve this, our sequential model utilizes the data structure of a suffix tree[2] [9] which we call a non-Markovian aggregation tree (Fig. 1). Specifically, every possible prefix of an agent's sequential route in $\mathcal{D}$ when reversed is represented by some path in the tree starting from its root. Consequently, the tree can aggregate the sequential decisions within the routes of all the agents. Since the tree's height is determined by the length of the longest route, the tree can maintain information on all past sequential decisions made within the routes of all agents, thus preserving the non-Markovian property. As an illustration (Fig. 1), supposing there are two routes $R_1 = (r_3, r_{12}, r_5, r_1)$ and $R_2 = (r_6, r_3)$, every possible prefix of R_1 and R_2 when reversed is represented as a path in the tree. For example, after updating the tree (i.e., assuming that the tree has been updated using R_1) with R_2, two solid nodes corresponding to (r_6) and (r_6, r_3) are added.

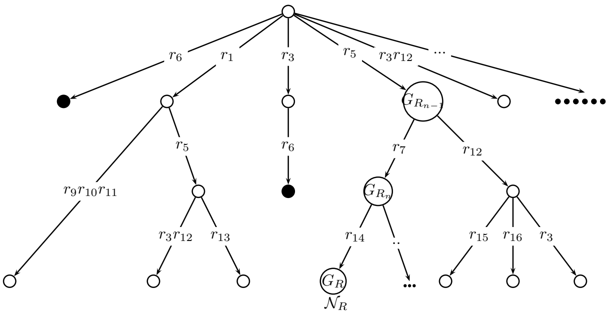


Figure 1: Non-Markovian aggregation tree (refer to text for more details).

To perform route prediction, the key step is to learn the posterior probability $P(\cdot|R, \mathcal{D})$ for every possible past route R found in $\mathcal{D}$ and store them efficiently. To simplify the notation, let $G_R \triangleq P(\cdot|R, \mathcal{D})$. As shown in Fig. 1, G_R is stored in the node $\mathcal{N}_R$ that is indexed by a context representing some past route R. A node's context can be formed by appending the edge labels along the path back to the root node; it therefore appears as the suffix of its descendants' contexts.

The tree construction algorithm is shown in Algorithm 1. The tree is constructed dynamically from the training data in $\mathcal{O}(|\mathcal{D}|\ell^2)$ time and $\mathcal{O}(|\mathcal{D}|\ell)$ space where ℓ is the length of longest route in the dataset $\mathcal{D}$.

```
Function UpdateTree($Tree, R$)
Input: Current tree $Tree$, newly observed route
       $R = (r_1, \ldots, r_i)$
Output: Updated tree $Tree$
for $k = 1$ to $i$ do
    Search $Tree$ for the node $\mathcal{N}_{R'}$ of context $R'$ that
    shares the longest suffix with $\widetilde{R} \triangleq (r_1, \ldots, r_k)$
    if $R'$ is suffix of $\widetilde{R}$ then
        Insert $\mathcal{N}_{\widetilde{R}}$ as a child of $\mathcal{N}_{R'}$
    else
        $R'' \leftarrow$ longest suffix of both $\widetilde{R}$ and $R'$
        Replace $\mathcal{N}_{R'}$ with $\mathcal{N}_{R''}$
        Insert $\mathcal{N}_{R'}$ and $\mathcal{N}_{\widetilde{R}}$ as children of $\mathcal{N}_{R''}$
    end
end
return $Tree$
```

Algorithm 1: Tree construction algorithm.

B. Bayesian Nonparametric Model

In this subsection, we will describe how to infer G_R from the dataset $\mathcal{D}$. Following the Bayesian paradigm, G_R is defined as a random variable such that a prior distribution can be placed over G_R before observing data. After observing data, G_R is updated to a posterior distribution. Prediction can then be performed by sampling from the posterior distribution. Formally, a nonparametric

[2] A suffix tree of a set of strings is a compacted trie of all suffixes of all the strings.

prior called *Pitman-Yor process* [10] is placed over G_R with prior/base distribution H:

$$G_R \sim \mathcal{PY}(d, \alpha, H) \tag{4}$$

where $d \in [0, 1)$ and $\alpha > -d$ are discount and concentration parameters, respectively. Both d and α control the amount of variability around the base distribution H.

The Pitman-Yor process can be better understood by interpreting it as a *Chinese Restaurant Process* (CRF) [10]: The analogy is that a sequence $(x_1, x_2, x_3, \ldots)$ of customers go to visit a restaurant which has an unbounded number of tables. The first customer sits at the first available table while each of the subsequent customers sits at the k-th occupied table with probability proportional to $(c_k - d)$ where c_k is the number of customers already sitting at that table, and he sits at a new unoccupied table with probability proportional to $(d\ t + \alpha)$ where t is the current number of occupied tables. From the CRP, it can be observed that a Pitman-Yor process has a useful rich-get-richer property: If more people sit at one table, then more people will sit at it. Using the CRP interpretation, the predictive/posterior distribution for the next customer can be derived as follows:

$$P(x_{c+1} = k | x_1, \ldots, x_c) = \frac{c_k - d}{c + \alpha} + \frac{d\ t + \alpha}{c + \alpha} H(k) \tag{5}$$

where $c = \sum_{k=1}^{t} c_k$. Note that when little data is available, prior knowledge H plays a significant role. But, after we have enough data, the predictive probability depends primarily on the data and becomes less affected by the prior distribution. Furthermore, (3) is similar to (5), which explains why IKN smoothing is an approximate version of CRP. We adapt the same idea to perform route prediction by treating each customer as a road segment and extend it to the sequential case, as discussed next.

C. Hierarchical Bayesian Nonparametric Model

Inspired by the hierarchical structure of the aggregation tree (Section III-A), we extend the Pitman-Yor process to a hierarchical model. To do this, we start by placing the base distribution H in (4) as a prior at the root node of the tree. Pitman-Yor process priors can then be placed recursively in each node along the path descending from the root of the tree as follows: Recall that R_n denotes a reduced route of R consisting of only the past n road segments. Let $G_{R_{n-1}}$ be the prior/base distribution for G_{R_n} (see Fig. 1), that is, $G_{R_n} \sim \mathcal{PY}(d_n, \alpha_n, G_{R_{n-1}})$. We can recursively do this to derive the hierarchical version of the Pitman-Yor process:

$$\begin{aligned} G_{R_n} \sim\ & \mathcal{PY}(d_n, \alpha_n, G_{R_{n-1}}) \\ G_{R_{n-1}} \sim\ & \mathcal{PY}(d_{n-1}, \alpha_{n-1}, G_{R_{n-2}}) \\ \cdots\ & \\ G_{R_0} \sim\ & \mathcal{PY}(d_0, \alpha_0, H) \end{aligned}$$

where the parameters d_n and α_n depend on the length of R_n, R_0 is the empty route with length 0, and H is often set to be a uniform distribution. Such a hierarchical structure is inherently consistent with that of the aggregation tree.

The key insight behind the hierarchical structure is that the predictive distribution G_{R_n} of a node $\mathcal{N}_{R_n}$ uses the predictive distribution $G_{R_{n-1}}$ of its parent node $\mathcal{N}_{R_{n-1}}$ as its prior/base distribution. Hence, this structure encodes the prior knowledge that the predictive distributions $G_{R_n}, G_{R_{n-1}}, \ldots, G_{R_0}$ with routes of different lengths are similar to each other. In particular, predictive distributions with routes sharing longer suffixes are more similar to each other. As a result, our hierarchical model can interpolate and promote sharing of statistical strength between predictive distributions with higher-order (i.e., longer route) and lower-order (i.e., shorter route) Markov properties. When there is no data for a route R_n, our model considers shorter routes $R_m (m < n)$ that appear in the training data by removing the earliest road segments from R_n first. This makes sense if we assume that the earliest road segments are least important to determining the next road segment, which is often the case.

D. Model Inference

Exact inference over Pitman-Yor processes is intractable. Instead, we employ an approximate inference method called the *Lossless Compression Sequence Memoizer* (LCSM) [11] which is well-suited for real-time, scalable learning. Different from [11], the input to our model is not a single long sequence, but a set of agents' routes. Hence, we modify it to be updated after observing each new agent's route.

The main idea of the inference procedure is to view Pitman-Yor process in terms of CRP. The distribution of Pitman-Yor process is captured by the counts $\{c_k, t\}$ of CRP (5). The predictive/posterior probability can be computed by averaging probabilities over all samples of counts $\{c_k, t\}$ which is often performed by the computationally intensive Markov chain Monte Carlo sampling method. LCSM simply updates the counts information for a newly observed route and removes the resampling step for the counts. It can be viewed as a particle filter with only one particle. For example, supposing there is a newly observed road segment $R' = r_{i+1}$ after traversing the route R_n, the counts information along the path from the root node $\mathcal{N}_{R_0}$ to the node $\mathcal{N}_{R_n}$ is updated. Let $t_{R_n \cdot R'}$ be the number of tables labeled with R' in the restaurant R_n in terms of the CRP representation. Then, the predictive/posterior probability is given by

$$P(R'|R_n, \mathcal{D}) = \begin{cases} \frac{c_{R_n \cdot R'} - d_n t_{R_n \cdot R'}}{c_{R_n} + \alpha_n} + \\ \frac{d_n t_{R_n} + \alpha_n}{c_{R_n} + \alpha_n} P(R'|R_{n-1}, \mathcal{D}) & \text{if } c_{R_n \cdot R'} > 0, \\ P(R'|R_{n-1}, \mathcal{D}) & \text{otherwise} \end{cases} \tag{6}$$

where $t_{R_n} = \sum_{R'} t_{R_n \cdot R'}$. It is important to note that when the parameters α_n and d_n are set to 0, (6) reduces to that of MLE without smoothing [6] (Section II-C). We will empirically compare the prediction accuracies of LCSM vs. MLE in Section IV-C.

As the model is updated with each new route, it may incur more space. However, coagulation and fragmentation properties of the hierarchical Pitman-Yor process [12] make it possible to maintain constant space overhead. Learning can be done in $\mathcal{O}(|\mathcal{D}|\ell^2)$ time. Since the length ℓ of the longest route is constant, the time complexity is linear in the size of training data $\mathcal{D}$.

IV. Experiments and Discussion

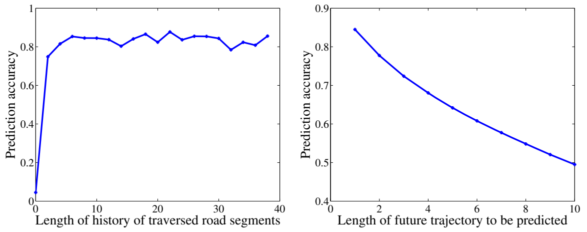
A. Traffic Data

In this section, we show experimental results based on real-world taxi route data in Singapore. Our data is provided by the Comfort taxi company. The data is collected in the following way: The Comfort taxi company has over $10,000$ taxis equipped with GPS roaming in the city. The GPS location, velocity, and status information of these taxis are sent to and stored at a central server. The dataset is collected in the whole month of August, 2010. But, the GPS data is noisy and sampling frequency is relatively low. The data cannot be used directly since it is in GPS format. Hence, the GPS data must be mapped onto routes of road segments. We apply a map matching technique [13] to preprocess the data and assume that the map matching result is perfect. After preprocessing the GPS data, we obtain a set of taxi routes such that each route's origin and destination correspond to the pick-up and drop-off locations, respectively. In this paper, we focus mainly on routes that are associated with the Bugis area. The reason for choosing Bugis is because of its dense road network and high volume of taxi traffic which make the route prediction problem even more challenging.

B. Prediction Accuracy

In the first experiment, we evaluate how the prediction accuracy varies in terms of varying length of history of traversed road segments and length of future route to be predicted. The size of training data is 6000 routes. Fig. 2 shows the results averaged over 300 test instances. Fig. 2a reveals that increasing the length of history of traversed road segments improves the average prediction accuracy. When the history increases beyond the length of 10, there is no significant improvement. The prediction accuracy stabilizes at around 85%. This implies the order n of Markov property can be set to be 10 with little or no performance loss. In turn, the maximum depth of the aggregation tree can be limited to 10, which significantly reduces the space overhead. Fig. 2b shows that the prediction accuracy decreases almost linearly in the length of future route to be predicted. For predicting future routes of length 10, our model can achieve 50% accuracy.

The second experiment evaluates the prediction accuracy with increasing size of training data. Our sequential model is learned incrementally with 100 routes each time, and its resulting prediction accuracy is shown in Fig. 3. It can be observed that the prediction accuracy stabilizes after the model has learned from about 5000 observed routes.



(a)

(b)

Figure 2: Graphs of average prediction accuracy vs. (a) length of history of traversed road segments and (b) length of future route to be predicted: The latter result is based on using the history of traversed road segments of length 10.

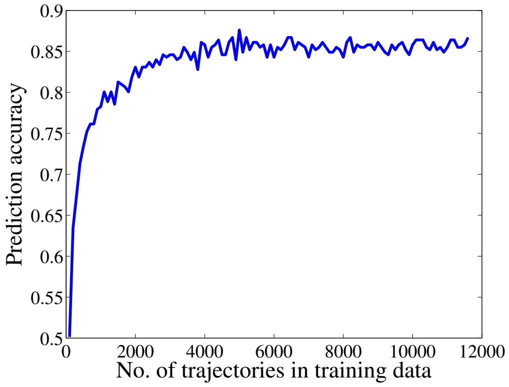


Figure 3: Graph of prediction accuracy vs. size of training data $\mathcal{D}$ (i.e., number of observed routes): The result is based on using the history of traversed road segments of length 10.

The third experiment evaluates how the prediction accuracy changes at different times of the day. In the previous experiments, the training data covers the whole day. In this experiment, the training and test data are obtained from 4 different times of the day (specifically, 2 a.m., 8 a.m., 1 p.m., 9 p.m.), each lasting 45 minutes. The results are shown in Fig. 4. It can be observed that the time of the day has minimal effect on the prediction accuracy except during $2 - 2$:45 a.m. This time of the day offers higher prediction accuracy when the history of traversed road segments is short (i.e., of length 2 to 4). This may be due to fewer, more easily distinguished routes during this time.

C. Effect of Smoothing

To evaluate the effect of smoothing, the prediction accuracy of LCSM with smoothing is compared to that of MLE without smoothing [6], as described previously in Section II-C. Besides considering the Bugis data, we also experiment with the more sparse data taken from the whole Singapore. The size of training data for each region is 6000 routes. The results are shown in Fig. 5. Firstly, it can be observed that LCSM with smoothing generally performs better than MLE without smoothing for both Singapore and Bugis data, with the performance improvement being more significant for the sparse Singapore data. The latter demonstrates that smoothing can achieve larger performance improvement for more sparse

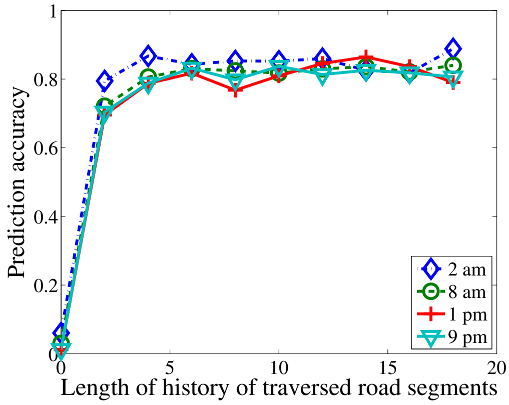


Figure 4: Graphs of prediction accuracy vs. length of history of traversed road segments at different times of the day.

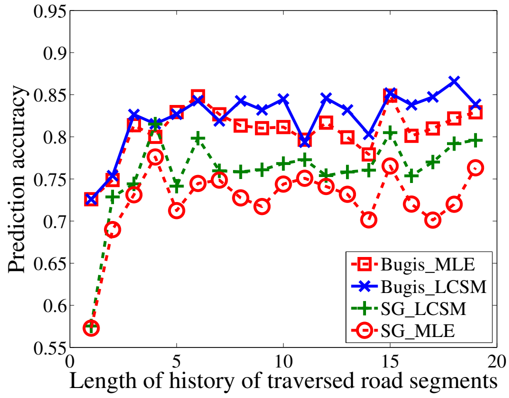
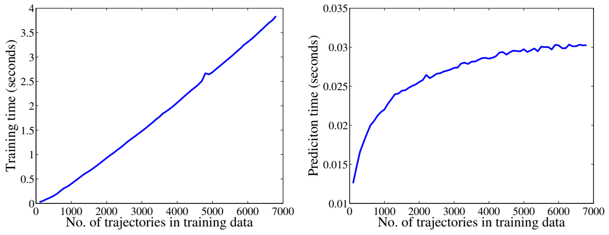


Figure 5: Graphs of prediction accuracy vs. length of history of traversed road segments: "SG_LCSM" and "SG_MLE" denote results that correspond to data taken from the whole Singapore while "Bugis_LCSM" and "Bugis_MLE" denote results that correspond to data taken from the Bugis area.

data. Secondly, the prediction accuracy is lower for the larger Singapore road network, which is expected due to data sparsity.

D. Time and Space Overhead

As mentioned in Section III-D, the learning time is linear in the size of training data $\mathcal{D}$. Fig. 6a shows that the incurred learning time is linear, and it only takes about 3 seconds to learn from 6000 routes, which is sufficient for achieving a high prediction accuracy, as reported earlier in Section IV-B. The prediction time is linear in the depth of the aggregation tree. Fig. 6b shows that the incurred prediction time stabilizes at about 0.03 seconds after learning from 6000 routes because the depth of the tree has more or less stabilized as well. The space overhead is also very reasonable. After learning from 6000 routes, our sequential model uses about 50MB of space.



(a) Learning time

(b) Prediction time

Figure 6: Graphs of time overhead vs. size of training data.

V. Practical Applications

In this section, we demonstrate two practical applications of our route predictive algorithm.

A. Predictive Taxi Ranking

The taxi dispatch system assigns taxis to the callers/ passengers waiting at different locations. Efficient taxi dispatching is essential to reducing passengers' waiting time and consequently, greater customer satisfaction.

A key factor that determines the system efficiency is that of ranking of the taxis to be dispatched. The state-of-the-art way of ranking [14] is based on the shortest path or duration from the taxi to the passenger, which can be retrieved by providing the GPS locations of the taxi and passenger to Google Maps API. In practice, such a ranking algorithm often performs poorly: The GPS sampling frequency is low, thus resulting in a discrepancy between the GPS location of the taxi and its actual location at the time of booking. Furthermore, there is a latency between the receipt of the booking message and the response of the taxi driver to it, during which the taxi would have moved along its original intended route for some distance to a new location. As a result, the taxi may no longer be "closest" to the passenger in terms of path distance or waiting time. This problem is exacerbated when the taxi crosses an intersection or moves into a congested road or expressway during this latency period. If there are many other taxis roaming in the same area, then some may have moved to be even closer to the passenger. This motivates the need to predict the future locations of the taxis using our route prediction model and exploit them for the ranking instead.

A simple experiment is set up to empirically compare the performance of our *predictive taxi ranking* (PTR) algorithm with that of the state-of-the-art non-predictive taxi ranking (NTR) algorithm [14]: to simulate the taxi dispatch system, we pick 202 taxis roaming in the Bugis area, and place 11 passengers at random locations in Bugis calling for taxis. We assume the latency to be between 5 to 10 seconds and our model predicts the future locations of the taxis after this latency. Table I shows the waiting time of the 11 passengers. With a high prediction accuracy of about 85%, our PTR algorithm can achieve shorter waiting time for 5 passengers and the same waiting time as the NTR algorithm for the remaining ones, thus resulting in shorter average waiting time over all passengers. We

Algo.	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	$\sum_{i=1}^{11} P_i/11$
NTR	86	551	83	68	506	365	156	157	10	57	121	196
PTR	86	551	83	**41**	**424**	365	**9**	**31**	10	57	**33**	153

Table I: Waiting time (seconds) of passenger P_i for $i = 1, \ldots, 11$.

conjecture that the performance of our PTR algorithm will be even better than that of the NTR algorithm with greater latency and less available taxis.

B. Route Recommendation

Existing route planning methods such as that in Google Maps consider explicit criteria such as shortest distance or travel time (calculated based on speed limit[3]). In contrast, our route prediction algorithm recommends maximum-likelihood routes that implicitly account for all the (hidden) criteria (e.g., tolls, time of the day, etc.) affecting the crowds of route planning agents. This is illustrated in Fig. 7.

VI. Related Work

Researchers have developed mobility models to address route prediction problem. The most naive idea is to do prediction by matching current route to a set of observed routes using data mining methods [1]. But, they are not capable of predicting a route that has never appeared in the training data. The decision-theoretic work of [15] modeled a driver's route selection as a Markov Decision Problem (MDP) and employed a maximum entropy approach to imitate the driver's behavior. It suffered from computational inefficiency due to the MDP solver. The work of [16] adopted the kinematic model and mapped GPS locations to hidden velocity variables. They modeled the moving vehicles as a mixture of Gaussian processes (GP) with a Dirichlet process (DP) prior over the mixture weights. This approach works well, but has granularity problem: the prediction accuracy depends on the size of each disjoint cell/region. Since the approach directly maps each GPS location to a velocity, it cannot deal with predictions involving the same cells/regions such as u-turns or crossing the same intersections more than once.

The most closely related work to our sequential model is that of Markov models. Recall that our processed data is a set of road segment sequences. Markov models assume that the next road segment only depends on a few preceding road segments, which is called Markov assumption. [7] trained a hidden Markov model to predict a driver's future road segments while simultaneously predicting the driver's destination. Their model was restricted to a single-vehicle scenario. Although they achieved a next-segment prediction accuracy of up to 99% with 46 trips, there was only one possible next road segment to choose from (i.e., perfect prediction) for 95% of road segments. The work of [6] on Markov-based route prediction studied how the order of the Markov model affects the prediction accuracy. However, they did not account for the important issue of sparsity of data, which occurs in our real-world data. We show through empirical comparison (Section IV) that our algorithm can resolve the issue of sparsity and achieve better prediction accuracy than that of [6]. Other models are limited by some unrealistic model assumptions such as knowing the origins and destinations of vehicle routes [17], one unique destination in the problem [15], and fully observed routes [18].

The issues of real-time prediction and online learning are rarely considered in the literature. With high-order Markov assumption, the space complexity is exponential and learning time is quadratic in the length of the input sequence. The learning time for our model is also quadratic, but its space complexity is only linear. Our proposed incremental learning method also differs from those works performing batch learning [6], [16] (i.e., training the model with all the observed data). Our model can be updated incrementally/online as and when new data comes.

VII. Conclusion

This paper describes a hierarchical Bayesian non-parametric model to aggregate the sequential decisions from the crowds of route planning agents for performing efficient and scalable route prediction. By not imposing a Markov assumption, the results show that our model can achieve a prediction accuracy of around 85% for the Bugis area which has a dense road network and high volume of taxi traffic. The results also show that our model with smoothing can be effectively used to improve the prediction accuracy over the widely-used MLE with no smoothing for sparse traffic data. Our model is highly time-efficient in learning and prediction: It only takes about 3 seconds to learn from 6000 taxi routes and 0.03 seconds to perform prediction. In terms of practical applications, our model can be used in predictive taxi ranking to achieve shorter passengers' waiting time over many test instances as compared to the state-of-the-art non-predictive one. It can also be used to recommend routes that implicitly account for the criteria (e.g., tolls) affecting the crowds of route planning agents.

Acknowledgments

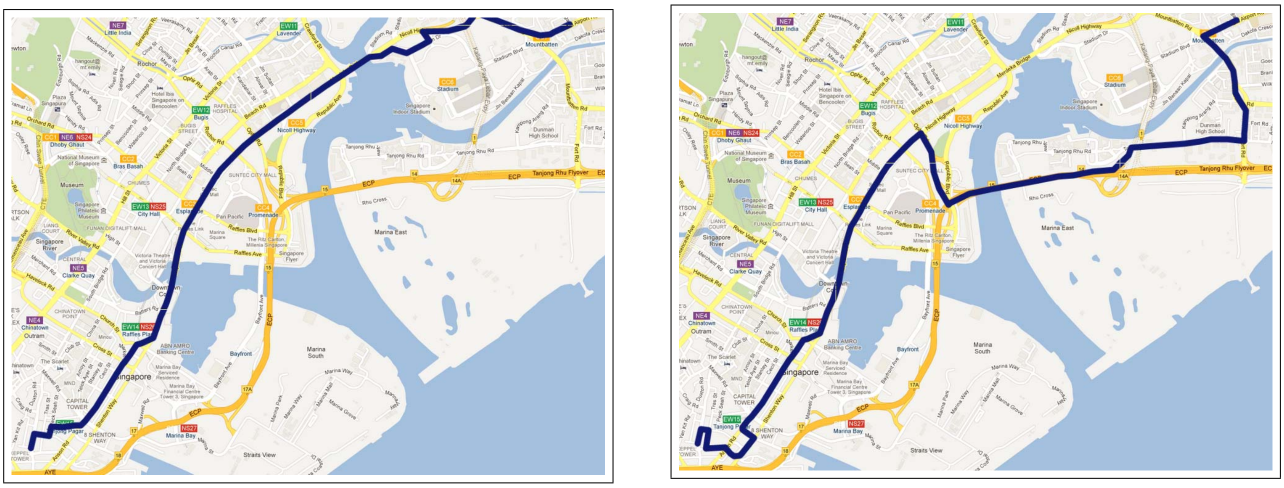
This work was supported by Singapore-MIT Alliance Research and Technology (SMART) Subaward Agreements No. 14 R-252-000-466-592 and No. 28 R-252-000-502-592.

References

[1] J. Froehlich and J. Krumm, "Route prediction from trip observations," in *Proc. Society of Automotive Engineers (SAE) 2008 World Congress*, 2008.

[2] K. Zheng, Y. Zheng, X. Xie, and X. Zhou, "Reducing uncertainty of low-sampling-rate trajectories," in *Proc. ICDE*, 2012.

[3] http://goo.gl/QEvqW, http://goo.gl/7nHi4



(a)

(b)

Figure 7: Given origin (Singapore General Hospital) and destination (Old Airport Road Food Center), our algorithm recommends two different routes depending on time of the day: (a) Take Nicoll highway during $3-6$ a.m. due to no electronic road pricing (ERP) fee, and (b) avoid Nicoll highway during 12 noon $-$ 2 p.m. as ERP is in effect.

[3] J. Surowiecki, *The wisdom of crowds*. Knopf Doubleday Publishing Group, 2005.

[4] S. Ertekin, H. Hirsh, and C. Rudin, "Learning to predict the wisdom of crowds," in *Proc. Collective Intelligence*, 2012.

[5] V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy, "Learning from crowds," *JMLR*, vol. 11, pp. 1297–1322, 2010.

[6] J. Krumm, "A Markov model for driver turn prediction," in *Proc. Society of Automotive Engineers (SAE) 2008 World Congress*, 2008.

[7] R. Simmons, B. Browning, Y. Zhang, and V. Sadekar, "Learning to predict driver route and destination intent," in *Proc. IEEE ITSC*, 2006, pp. 127–132.

[8] J. Goodman and S. Chen, "An empirical study of smoothing techniques for language modeling," *Computer Speech and Language*, vol. 13, no. 4, pp. 359–393, 1999.

[9] M. Farach, "Optimal suffix tree construction with large alphabets," in *Proc. FOCS*, 1997, pp. 137–143.

[10] Y. W. Teh and M. I. Jordan, "Hierarchical Bayesian nonparametric models with applications," in *Bayesian Nonparametrics*. Cambridge Univ. Press, 2010.

[11] J. Gasthaus, F. Wood, and Y. Teh, "Lossless compression based on the sequence memoizer," in *Proc. Data Compression Conference*, 2010, pp. 337–345.

[12] F. Wood, C. Archambeau, J. Gasthaus, L. James, and Y. Teh, "A stochastic memoizer for sequence data," in *Proc. ICML*, 2009, pp. 1129–1136.

[13] P. Newson and J. Krumm, "Hidden Markov map matching through noise and sparseness," in *Proc. 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2009, pp. 336–343.

[14] D. Lee, H. Wang, R. L. Cheu, and S. H. Teo, "A taxi dispatch system based on current demands and real-time traffic information," *Transportation Research Record*, vol. 1882, pp. 193–200, 2004.

[15] B. Ziebart, A. Maas, J. Bagnell, and A. Dey, "Maximum entropy inverse reinforcement learning," in *Proc. AAAI*, 2008, pp. 1433–1438.

[16] J. Joseph, F. Doshi-Velez, and N. Roy, "A Bayesian nonparametric approach to modeling mobility patterns," in *Proc. AAAI*, 2010.

[17] A. Karbassi and M. Barth, "Vehicle route prediction and time of arrival estimation techniques for improved transportation system management," in *Proc. IEEE Intelligent Vehicles Symposium*, 2003, pp. 511–516.

[18] D. Ashbrook and T. Starner, "Using GPS to learn significant locations and predict movement across multiple users," *Personal and Ubiquitous Computing*, vol. 7, no. 5, pp. 275–286, 2003.