DesTeller: A System for Destination Prediction Based on Trajectories with Privacy Protection

Andy Yuan Xue
The University of Melbourne
Victoria, Australia
andy.xue@unimelb.edu.au

Rui Zhang
The University of Melbourne
Victoria, Australia
rui@csse.unimelb.edu.au

Yu Zheng
Microsoft Research Asia
Beijing, P.R.China
yuzheng@microsoft.com

Xing Xie
Microsoft Research Asia
Beijing, P.R.China
xingx@microsoft.com

Jianhui Yu
South China Normal University
Guangzhou, P.R.China
jianhui_yu@qq.com

Yong Tang
South China Normal University
Guangzhou, P.R.China
ytang@scnu.edu.cn

ABSTRACT

Destination prediction is an essential task for a number of emerging location based applications such as recommending sightseeing places and sending targeted advertisements. A common approach to destination prediction is to derive the probability of a location being the destination based on historical trajectories. However, existing techniques suffer from the "data sparsity problem", i.e., the number of available historical trajectories is far from sufficient to cover all possible trajectories. This problem considerably limits the amount of query trajectories whose predicted destinations can be inferred.

In this demonstration, we showcase a system named "DesTeller" that is interactive, user-friendly, publicly accessible, and capable of answering real-time queries. The underlying algorithm *Sub-Trajectory Synthesis* (SubSyn) successfully addressed the data sparsity problem and is able to predict destinations for almost every query submitted by travellers. We also consider the privacy protection issue in case an adversary uses SubSyn algorithm to derive sensitive location information of users.

1 Introduction

As the usage of smartphones and in-car navigation systems becomes part of our daily lives, we benefit increasingly from various types of location based services (LBSs) such as route finding and location based social networking. A number of new location based applications require *destination prediction* technique, for example, to recommend sightseeing places, to send targeted advertisements based on destination, and to automatically set destination in navigation systems. Figure 1 provides a schematic with the lines representing roads and the circles representing locations of interests (They may be road intersections, sightseeing places, shopping centres, etc.). If one drives from l_1 to l_4, an LBS provider may predict the most probable destinations to be l_7, l_8 and l_9 based on past popular routes taken by other drivers. As a result, LBS providers can push advertisements of products currently on sale at those locations.

A common approach to destination prediction is to make use of public historical spatial trajectories, available from trajectory sharing websites, or large sets of taxi trajectories. If an ongoing trip matches part of a popular route derived from historical trajectories, the destination of the popular route is very likely to be the destination of the ongoing trip (we refer to the ongoing trip as the *query trajectory*). Shown in Figure 1 are five historical trajectories: $T_1 = \{l_1, l_2, l_5, l_6, l_9\}$, $T_2 = \{l_6, l_3, l_2\}$, $T_3 = \{l_4, l_5, l_8\}$, $T_4 = \{l_9, l_8, l_7\}$, and $T_5 = \{l_1, l_4, l_7\}$. Each trajectory is represented by a different type of line. For instance, a trip is taken from l_1 to l_4, and this query trajectory $\{l_1, l_4\}$ matches part of the historical trajectory T_5. Therefore, the destination of T_5 (i.e., l_7) is the predicted destination of the query trajectory. In practice, each trajectory here may be associated with a weight denoting the number of historical trajectories that exactly match this one, and the most popular trajectories are used for destination prediction. This idea has been described in further details in [7].

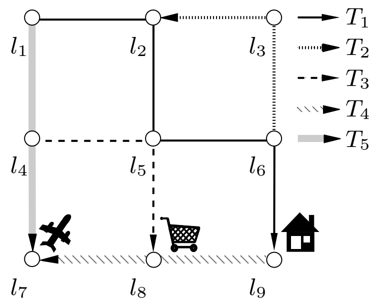


Figure 1: An example of destination prediction

However, the above method has a significant drawback. A location l can be predicted as a destination only when the query trajectory matches a historical trajectory and the destination of the historical trajectory is l. In practice, l_8 and l_9 are also very likely to be the destination of the query trajectory, but will not be recommended to the user due to the limitation of the historical dataset. Moreover, if the query trajectory continues to l_5, the above method will not be able to predict any destination since no historical trajectory contains the trajectory $\{l_1, l_4, l_5\}$. We refer to this phenomenon as the **data sparsity problem**. This problem is inevitable in practice due to the following reasons. First, given a road network, the number of possible routes between all pairs of origin-destination is prohibitively large (exponential to the number of edges in the network), and currently the largest available real-life trajectory dataset covers only a tiny portion of it. Second, even trips with the same origin-destination pair may vary on their routes, making it unlikely to have identical trajectories.

In this demonstration, we present a novel method to address the data sparsity problem based on our previous paper [5]. To begin with, we decompose all the trajectories into sub-trajectories comprising two neighbouring locations, then the sub-trajectories are connected together into "synthesised" trajectories. As long as the query trajectory matches part of any synthesised trajectory, the destination of the synthesised trajectory can be used for destination prediction. By this means, the coverage of trips on which we can make destination predictions is exponentially increased. We call the above method the ***Sub**-Trajectory **Syn**thesis* (**SubSyn**) algorithm. For the aforementioned query trajectory $\{l_1, l_4, l_5, l_6\}$, SubSyn algorithm will be able to predict other destinations such as l_8 and l_9 since they can be synthesised using sub-trajectories of T_1, T_3, and T_5.

While SubSyn algorithm largely enhances the destination prediction capability of LBS providers, it can also be used by a malicious party to derive destinations which users do not wish to disclose such as homes and hospitals. Therefore, we also investigate on how to counter any privacy breach caused by destination prediction using algorithms like SubSyn. In particular, a user may choose not to check in (or publish) certain locations to prevent adversaries from deriving her destination. In the previous example, the user may manually choose not to check in at l_6 in order to reduce the probability of l_9 being the destination below a certain threshold. To mitigate the disturbance to the check-in service, we study which locations in the trajectory the user should not check in such that the number of locations that the user does not check in is minimised.

We make the following specific contributions in this demonstration:

- We identify the data sparsity problem in destination prediction and propose a novel *Sub-Trajectory Synthesis* (SubSyn) algorithm, which successfully addressed this problem while achieving competitive prediction accuracy and much higher runtime efficiency.

- We build an interactive demonstration program (*DesTeller*)[1] that simulates a trip along user-specified route and displays predicted destinations on a map. Whilst the simulation is running, predicted destinations are displayed to the users in real time. The users are given options to customise several parameters for accuracy, information and aesthetic purposes (such as the number of predicted destinations returned and the speed of the car).

2 Related Work

In this section, we discuss existing work on destination prediction and methods of protecting users' privacy.

Although most destination prediction studies make use of historical trajectories, their focuses have mainly followed two streams: (i) using external information in addition to historical trajectories to help improve the accuracy of predicted destinations; (ii) personalised destination prediction for individual users. Examples of external information include distributions of different districts (i.e., ground cover), travel time, trajectory's length, accident reports, road condition, and context information such as time-of-day. Our work considers a generic setting where only a historical trajectory set is assumed. Personalised destination prediction trains prediction models using historical trajectories from an individual and then predicts destinations for this same individual, whereas our work considers a query trajectory from an unknown individual (without available personalised information).

Amongst the above studies, Bayesian inference is the most popular framework used for deriving the probability of destinations based on historical trajectories [4, 7, 2]. Our approach also follows this framework.

On protecting the users from privacy leak caused by trajectory based spatial-temporal queries, there are mainly four approaches: (i) by *clustering* trajectories within the same time period; (ii) by further picking *atomic* points from the group; (iii) by iteratively *suppressing* (deleting) locations in trajectories; and (iv) by grouping neighbouring cells in a *grid* into groups. Our study on privacy protection against destination prediction lies in the third group. The difference between our study and existing studies is that our algorithm utilises distinct properties of SubSyn algorithm. Privacy protection and trajectory mining have also been studied in moving KNN query in [3] and predictive KNN query in [6].

3 Our Methodology

In this section, the general idea of our algorithms will be explained. For an elaborated description of the algorithms, please refer to [5].

3.1 Destination Prediction

In order to overcome the data sparsity problem, we propose a novel ***Sub**-Trajectory **Syn**thesis* (SubSyn) algorithm that uses a Markov model to offline prepare the probabilities needed to efficiently predict destinations for any given query trajectory online.

First of all, we partition the entire interested area (e.g., metropolitan area of a major city) into nodes in a grid graph in order to simplify the unnecessary complexity. In this way, we have converted the representation of trajectories from list of GPS points to list of grid cell coordinates. We decompose each historical trajectory into sub-trajectories and index them using *Markov model*. Specifically, a Markov model based transition matrix M is constructed and filled with probabilities of travelling from a node to its adjacent node. This process, when inspected from another approach, is effectively the process of decomposing each trajectory in D into a set of sub-trajectories with length 2 (i.e., ordered pairs of neighbours). For instance, the trajectory T_1 in Fig. 1 is decomposed into $T^p_{1,2}$, $T^p_{2,5}$, $T^p_{5,6}$, and $T^p_{6,9}$ which in turn contribute to the transition probabilities p_{12}, p_{25}, p_{56}, and p_{69}, respectively. To synthesise sub-trajectories is to utilise the transition matrix M to compute the probability of being a destination of a query trajectory. The following are several formulae used in the algorithm of predicting destination.

Synthesis of All Paths Towards Destination: A useful value that can be generated from M is the sum of the probabilities of all possible paths between two nodes n_i and n_k. We define this concept as the *total transition probability*:

Definition 1. The **Total Transition Probability** of travelling from one node n_i to another node n_k with ℓ_1 distance $L_1 = r$, denoted by $p_{i \to k}$, is the r-step transition probabilities of all possible paths between n_i and n_k.

Synthesis of Path for Query Trajectory We provide a definition of *path probability* which will be used to compute the posterior probability. The path probability is the probability of a person

[1]The demonstration program is publicly accessible at [1].

travelling from one location to another via a specific path. Typically the path is the query trajectory provided by a user. In general, given any query trajectory $T^p_{1,2,\cdots,k}$, the definition of path probability is:

$$P(T^p) = P(T^p_{1,2,\cdots,k}) = \prod_{i=1}^{k} p_{i(i+1)}. \qquad (1)$$

Computing the Posterior Probability We apply Bayer's rule as the prediction tool, i.e., using

$$P(d \in n_j) = \frac{|T_{d\in n_j}|}{|D|}, \qquad (2)$$

to compute the prior probability and

$$P(d \in n_j|T^p) = \frac{P(T^p|d \in n_j)P(d \in n_j)}{\sum_{k=1}^{g^2} P(T^p|d \in n_k)P(d \in n_k)}, \qquad (3)$$

to predict the probability of being a destination. Here $d \in n_j$ denotes that the destination lies in node n_j, $|T_{d\in n_j}|$ indicates the number of historical trajectories whose destinations lie in node n_j, $|D|$ gives the cardinality of historical trajectory database, and T^p represents the query trajectory.

After defining the total transition probability and the path probability, given a query trajectory T^p, we calculate the posterior probability of a user travelling from the staring node n_s to the current node n_c via T^p conditioned on the destination being in node n_j:

$$P(T^p|d \in n_j) = \frac{P(T^p) \cdot p_{c\to j}}{p_{s\to j}}, \qquad (4)$$

where $P(T^p)$ is the path probability of the given query trajectory T^p; $p_{c\to j}$ is the total transition probability of moving from the current node of T^p, n_c, to a predicted destination $d \in n_j$; and $p_{s\to j}$ is the total transition probability of travelling from the starting node of T^p, n_s, to a predicted destination $d \in n_j$.

The posterior probability is used when a user issues a query to compute destination probabilities. Given a transition matrix M, a total transition matrix M^T generated from taxi trajectories in a grid graph, and a query trajectory T^p, the overview of the *SubSyn-Prediction* algorithm is as follows: Compute values for (2), (1), and (4), hence compute (3) and output the top k results. It is clearly observed that the online prediction algorithm is extremely straightforward because of the offline training stage *SubSyn-Training* (i.e., computing the transition matrix and the total transition matrix).

3.2 Privacy Protection

We present an efficient method, *End-Points Generation Method*, which extensively reduces the running time (i.e., number of sub-trajectories that we need to examine) compared with an exhaustive search algorithm where each location is removed from the trajectory to see if the resulting trajectory satisfies the requirement. Consequently we are able to process privacy protection queries in real time.

THEOREM 1. *Using a first order Markov model based SubSyn algorithm and given a query trajectory, the probability of any potential destination depends only on the starting node n_s and the current (i.e., most recent) node n_c of this query trajectory.*[2]

The above theorem indicates that, given a first order Markov model and a query trajectory, the destination probability of the user's private destination can be altered by only deleting the two *end points* (i.e., n_s and n_c) from this query trajectory.

[2]The proof can be found in [5].

4 Experimental Study

Extensive experiments were conducted and presented in [5] using real-world dataset (i.e., real taxi trajectories in the city of Beijing). The experiments confirmed the advantages of SubSyn algorithm in both coverage (i.e., solves data sparsity problem) and prediction accuracy than those of the existing algorithm. Our algorithms also achieve high runtime efficiency and are capable of processing real-time online queries.

5 Demonstration Scenarios

In this section, we present the demonstration program (DesTeller), which can be accessed from [1], and describe use cases and scenarios enabled by DesTeller. Firstly, a screenshot of the demonstration program is shown in Figure 2a, which illustrates DesTeller in the form of a dynamic webpage. The demonstration program was built to achieve objectives including interactivity (i.e., users are given flexible and multiple ways to tune parameters and control the simulation), easy-to-use (i.e., clear instructions are given, and procedures are concise), accessibility (i.e., accessible from anywhere in the world where internet is available), and instantaneous response time (i.e., able to perform query processing in real time). Users can run simulations for destination prediction and privacy protection using this demonstration. We also offer an option named *heatmap* (cf. Figure 2c) to reveal all popular destinations in *Beijing* with multiple grid resolutions to choose from.

The following subsections will show two use cases of DesTeller.

5.1 Destination Prediction

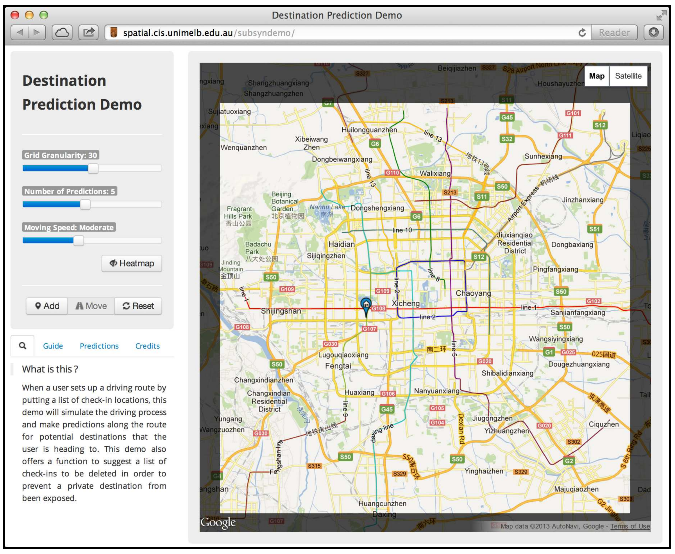
The use case described in this subsection corresponds to Figure 2b. Consider the following scenario in which a user, John, drives from his hotel (cf. the first waypoint, with a "house" icon, at the bottom left corner in Figure 2b) to the airport (cf. the top right corner). John started his trip by following traffic signs, which direct to the airport. Soon after he left the hotel, the GPS device in John's car recommended several likely destinations (cf. blue squares in Figure 2b-Left) including the airport to him based on his trip so far from the hotel to his current location (cf. waypoint with a "car" icon)[3]. John selected the airport from the list of predicted destinations and continued to drive to the airport without the need to manually search and input the location "airport" into his GPS device. During his entire trip, the system continued to predict destinations using updated trip travelled so far, and send recommendations and targeted advertisement along the route to destination to John, who may find a particular sightseeing place attractive and visit during his leisure time.

5.2 Privacy Protection

The use case described in this subsection corresponds to Figure 2d. Consider the following scenario in which a user takes advantage of our privacy protection solution to avoid privacy leak. Jane has a geo-social application on her smartphone, and during a Sunday afternoon she is on her way home from a public event (cf. waypoint with "house" icon). Right before arriving at her house, four locations (cf. four waypoints in Figure 2d)[4] are recorded from auto check-ins when uploading a photo taken by her smartphone and when posting a message, and manual check-ins at the event venue and in a restaurant. Since our privacy protection solution is in place,

[3]Ignore the waypoints with numbers (i.e., W_1 and W_2 in Figure 2b). They only serve an auxiliary purpose to set the route for the simulation to run.

[4]Here the waypoints are actual check-in locations instead of auxiliary route mark as in the destination prediction simulation.



Destination Prediction Demo

Grid Granularity: 30

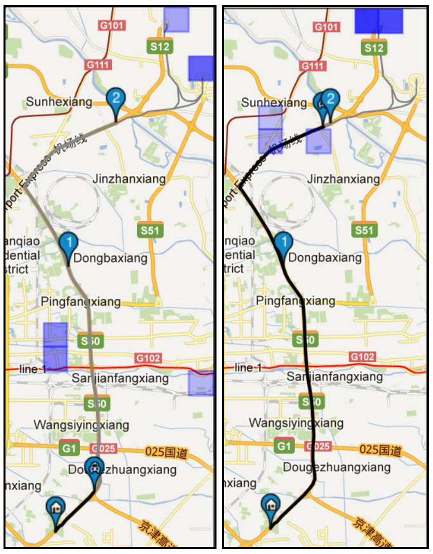
Number of Predictions: 5

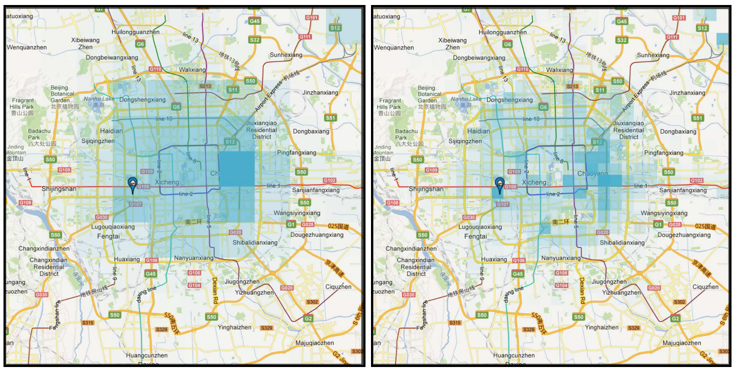
Moving Speed: Moderate

What is this ?

When a user sets up a driving route by putting a list of check-in locations, this demo will simulate the driving process and make predictions along the route for potential destinations that the user is heading to. This demo also offers a function to suggest a list of check-ins to be deleted in order to prevent a private destination from been exposed.

(a) Overview


(b) Destination Prediction


(c) Heatmap (grid granularities 10 and 30)

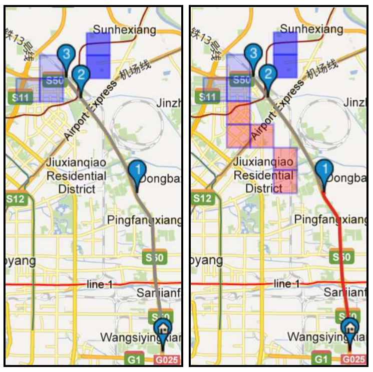

(d) Privacy Protection

Figure 2: Screenshots of the Demonstration System: *DesTeller*

before publishing each location, Jane receives a confirmation dialogue showing a list of predicted destinations (cf. blue squares) should the locations be published, among them are Jane's true residential place. Instructed by Jane, the program processes this privacy protection request and returns a new list of locations $\{W_0, W_1\}$ with predicted destinations (cf. red squares) which are the results of deleting the smallest number of locations (in this case W_2 and W_3 have been deleted) such that her true destination is not revealed. Jane is satisfied with the resulting list of locations, and publishes them without worrying the issue of privacy leak.

6 Conclusions

We proposed algorithms to perform destination prediction (SubSyn) and privacy protection tasks (End-Points Generation) robustly and efficiently. Enabled by these algorithms, we built and showcased a demonstration program (DesTeller) in the form of a dynamic webpage that interactively simulates how the SubSyn algorithm works in a visually appealing way. By entering a partial route, the program simulates the trip along the specified route and displays predicted destinations to the user. Users are given options to customise the behaviour of the simulation. Both the frontend (i.e., webpage) and the backend (i.e., web service) are publicly accessible where internet is available. The frontend can be easily adapted to other platforms such as smartphone operating systems without touching the backend web service. Therefore the usability and practicality of the demonstration program are high.

Acknowledgments: This work is supported by *Australian Research Council (ARC) Future Fellowships* Project FT120100832. Conference expenses are partially supported by *Google Ph.D. Travel Prize*.

7 References

[1] `http://spatial.cis.unimelb.edu.au/subsyndemo/`.

[2] J. Krumm and E. Horvitz. Predestination: Inferring destinations from partial trajectories. In *Proc. UbiComp*, pages 243–260, 2006.

[3] S. Nutanong, R. Zhang, E. Tanin, and L. Kulik. Analysis and evaluation of v*-*k*nn: an efficient algorithm for moving *k*nn queries. *VLDB J.*, 19(3):307–332, 2010.

[4] D. J. Patterson, L. Liao, D. Fox, and H. Kautz. Inferring high-level behavior from low-level sensors. In *Proc. UbiComp*, pages 73–89, 2003.

[5] A. Y. Xue, R. Zhang, Y. Zheng, X. Xie, J. Huang, and Z. Xu. Destination prediction by sub-trajectory synthesis and privacy protection against such prediction. In *Proc. IEEE 29th International Conference on Data Engineering*, ICDE, 2013.

[6] R. Zhang, H. Jagadish, B. T. Dai, and K. Ramamohanarao. Optimized algorithms for predictive range and knn queries on moving objects. *Information Systems*, 35(8):911 – 932, 2010.

[7] B. D. Ziebart, A. L. Maas, A. K. Dey, and J. A. Bagnell. Navigate like a cabbie: Probabilistic reasoning from observed context-aware behavior. In *Proc. UbiComp*, 2008.