CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	Inferring destination from mobility data
Authors	Naeem, Ali A.;Brown, Kenneth N.
Publication date	2017
Original Citation	Naeem, A. A. and Brown, K. N. (2017) 'Inferring destination from mobility data', Proceedings of the 25th Irish Conference on Artificial Intelligence and Cognitive Science, Dublin Institute of Technology, 7 - 8 December, pp. 153-165
Type of publication	Conference item
Link to publisher's version	http://ceur-ws.org/Vol-2086/AICS2017_paper_37.pdf
Rights	© 2017, the Authors. Copying permitted for private and academic purposes.
Download date	2026-11-16 19:27:05
Item downloaded from	https://hdl.handle.net/10468/6891



UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Inferring Destination from Mobility Data

Ali A. Naeem & Kenneth N. Brown

Insight Centre for Data Analytics, Department of Computer Science
University College Cork, Ireland
{ali.naeem, ken.brown}@insight-centre.org

Abstract. Destination prediction in a moving vehicle has several applications such as alternative route recommendations even in cases where the driver has not entered their destination into the system. In this paper a hierarchical approach to destination prediction is presented. A Discrete Time Markov Chain model is used to make an initial prediction of a general region the vehicle might be travelling to. Following that a more complex Bayesian Inference Model is used to make a fine grained prediction within that destination region. The model is tested on a dataset of 442 taxis operating in Porto, Portugal. Experiments are run on two maps. One is a smaller map concentrating specificially on trips within the Porto city centre and surrounding areas. The second map covers a much larger area going as far as Lisbon. We achieve predictions for Porto with average distance error of less than 0.6 km from early on in the trip and less than 1.6 km dropping to less than 1 km for the wider area.

1 Introduction

The ubiquity of smart phones and other GPS devices has increased interest in location-based services such as location-based social networking and real-time information updating. Destination prediction is an integral part of these services as it enables, for example, sight-seeing recommendations, more efficient route recommendations based on current conditions, etc. Most of the previous work is based on data mining of historical trajectories of known individuals to provide personalised destination prediction. For unknown users or new users, the problem is significantly harder since we have a much larger set of a priori probable destinations. In this paper, the question addressed is: can the destination of a vehicle during its journey be correctly and reliably predicted using a history of previous trip trajectories from a wider population?

We propose a hierarchical model which first predicts a general region for the destination and then refines it by making a second more fine grained prediction within that higher destination region. The initial prediction to select the destination region is made by a Discrete Time Markov Chain (DTMC) model. The states in the markov chain are the discrete locations a car might occupy. The second prediction is made by a more complex Bayesian Inference model which calculates the probability of the destination location given some observed features of the trip such as start location and last two observed locations.

The model inference is evaluated on actual datasets of taxis in Porto, Portugal. If traffic is limited to the Porto urban region, our best model predicts the destination coordinates to within 0.5 km on average across the entire trip. This accuracy is reached very early into the trip in the smaller map. For the more general case covering a much larger area of Porto and surrounding regions, the best model predicts to an accuracy of 1.6 km early on in the trip and the error dips to below 1 km as the trip progresses.

2 Related Work

The Predestination algorithm [2] uses a Bayesian Inference model combined with geographic data such as types of land cover. This allows the model to preemptively rule out, for example, areas of water as possible destinations. The algorithm implements an open world model which attempts to capture the possibility of a vehicle going to destinations which have not yet been observed in the dataset. It does this by assuming that there is a certain likelihood of a driver going to a destination which is near some other destination that they have visited before (for instance, they may stop off at the supermarket or restaurant close to their home or place of work. The algorithm also includes the idea of driving efficiency. It supposes that the driver will take a reasonable route to their destination and thus gives higher preferences to those destinations which can be reached within reasonable travel.

The PROCAB algorithm [7] improves on Predestination by using Markov Decision Processes to learn preferences on the part of drivers (for instance some road types are more preferable and hence have higher utility). This allows the algorithm to learn contextual information which increases performance. PROCAB is shown to perform similarly to Predestination when a large portion of the trip is observed and do better than Predestination earlier in the trip.

It is often the case that in approaches for destination prediction where the destination probability is calculated conditioned on certain observed features of the trip, there exists a data sparsity problem. Specifically the method is only feasible if there exists in the data a historical trajectory with the exact same characteristics as the current observed trip. One solution to this is to include extra information such as road maps, route planning, etc as seen in the Predestination algorithm with land cover [2]. An alternative method is Sub-Trajectory Synthesis (SubSyn) [5] which attempts to predict other destinations by synthesising sub-trajectories from the observed data. In the current work, we will be presenting a geometric model to address sparsity which estimates transition probabilities in specific trips not observed in the dataset.

The T-Drive algorithm [6] builds a landmark graph in which a node represents a road travelled often by taxis. The route prediction is made in a hierarchical fashion similar to the concept presented in this paper. Initially, a rough route is predicted using the landmark graph. In the second step, the rough route is refined to give a final prediction.

The problem of destination prediction with taxis specifically has also been attempted with neural networks [1] and regression based approaches [3]. A com-

mon thread in these approaches is some discretization of the locations whether by applying a uniform grid or clustering.

3 Data Preprocessing

The dataset utilised[1] consists of approximately 1.7 million taxi journeys from 442 taxis operating in the city of Porto in Portugal over a period of one year. The GPS readings are recorded every 15 seconds. Other data such as time of pickup, driver ID and trip ID are also available.

The prediction task was separated into two problems of increasing difficulty. In the first problem, we focus on modelling an area which captures the city centre and surrounding areas of Porto. Any trips that ventured outside this bounding box were removed. The area selected for modelling in problem 1 is shown in figure 1. This smaller map covers a distance of approximately 40 km measured diagonally. In the second problem, the area of the bounding box was increased so that it took in Porto and other surrounding cities such as Lisbon. This larger box captures the full extent of the trips recorded in the dataset. The area covered by the larger bounding box is shown in figure 2. This larger map covers a distance of 445 km measured diagonally. The raw data was cleaned by

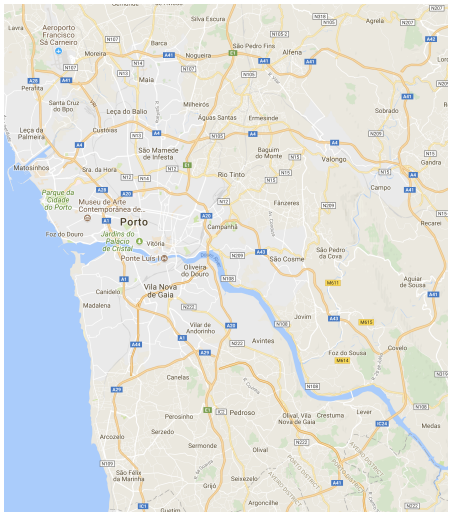


Fig. 1. Map of Area Modelled by smaller bounding box. Shown as the black box in the larger map.

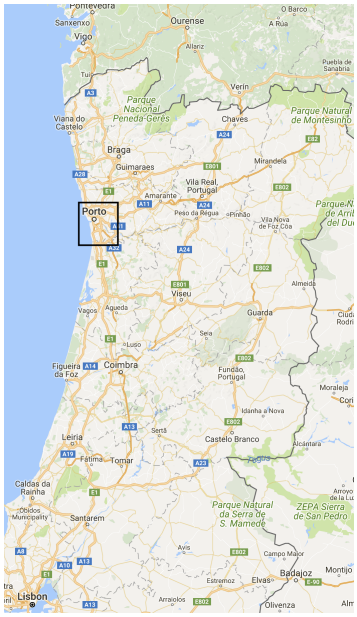


Fig. 2. Map of Area Modelled by larger bounding box.

applying a number of filters. Ten trips that were identified in the data itself as

[1] https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i/data

having missing data were removed. However, our data exploration indicated that these weren't the only trips with missing data points. To identify these trips, the individual GPS data points were used to calculate the instantaneous speed of travel at every recording made on a trip. It is specified that the GPS data points are recorded once every 15 seconds. Hence, any trips with unreasonably large spikes in the instantaneous speed indicated missing or noisy data. Once a trip is identified as having noisy or missing data, the entire trip was excluded from the data. Secondly, the data is collected by mobile terminals carried by the drivers. Therefore, some 'trips' were in fact times where the driver had forgotten to turn off the data collection when outside the vehicle and as such represented pure walking trips. Those trips are often observed to have data points concentrated within a small geographical area such as a shopping mall. All such trips were identified and removed by excluding any trips that never exceeded a speed of 15 km/hr at any point.

In order to reason about similar trips and locations, we need to discretize the space. Rather than applying a uniform grid, we chose to cluster the GPS data points. This has the advantage that it creates tighter clusters where there was denser traffic hence giving finer predictions in those areas. We used K-Means Clustering at two levels of granularity: a coarse clustering with fewer clusters and a second much finer clustering. The specific values for the number of clusters k at each level were determined by clustering the data for a range of k values and carrying our silhouette analysis.

Silhouette analysis [4] can be used to analyse the separation distance between the clusters produced by a clustering algorithm. The analysis involves calculating a number known as the mean Silhouette coefficient of all the samples. This coefficient has a range between -1 and +1 and indicates how close each point in one cluster is to points in the neighbouring clusters. Silhouette coefficients near +1 indicate that the sample is far away from the neighbouring clusters and this is desirable. A value of 0 indicates that the sample is on or very close to the decision boundary between 2 neighbouring clusters. Negative values are an indication that the sample has been assigned to the wrong cluster.

The silhouette coefficient is calculated using the mean intra-cluster distance a and the mean nearest cluster distance b (the distance between the sample and the nearest cluster that the sample is not a part of). The formula for silhouette coefficient S then is:

$$S = \frac{b - a}{max(a, b)}$$

To determine the best value for k at each level, a range $[8, 12]$ at the coarse level and $[248, 260]$ at fine grained level was specified. The clustering algorithm was then run for each value of k which falls into the range. The mean silhouette coefficient was then calculated for each resultant clustering. The value for k which yielded the highest mean silhouette coefficient at each level was chosen. In the present case, for the smaller bounding box, the best values for k at the coarse level turned out to be 8 clusters and the fine level 259 clusters. A similar analysis was carried out for the data contained in the larger bounding box. The



Fig. 3. Coarse clustering with 8 clusters in smaller bounding box

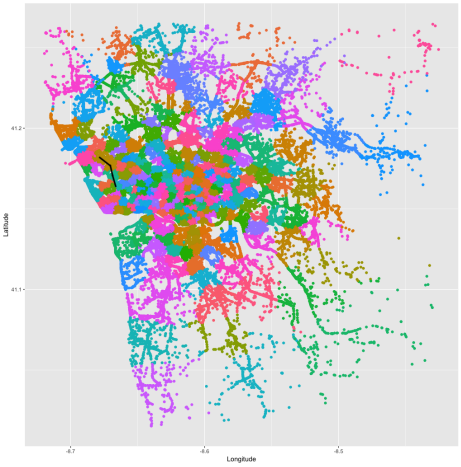


Fig. 4. Fine clustering with 259 clusters in smaller bounding box. Sample trip shown.

ideal clustering calculated was 18 clusters at the coarse level. Note that the dropoff behaviours were very different between the city center regions and the rural regions. For this reason, rather than attempt a global clustering at the fine level, a hierarchical clustering scheme was devised. In this case, the data within each of the 18 higher regions were isolated and a separate clustering carried out within each of the 18 higher regions. A visualisation of these fine clusters would be too small to be discernible. For illustration purposes, figure 5 shows a sample clustering in the larger bounding box. The actual clusters are much finer than this at the lower level. Following the clustering, the trips are now represented as a sequence of cluster visits with repeated cluster IDs for sequences of GPS locations that remain within the same cluster. When a prediction is made, the simplest level of prediction then predicts a destination cluster rather than coordinates. In figure 6, a cluster with radius r_2 is shown. (The actual clusters aren't perfectly circular but it is assumed these are average figures for the radius.) Assuming further that there is a uniform distribution for drop off locations inside the cluster, the average prediction error even in the most ideal case where the correct destination cluster is identified is given by a circle with radius r_1 whose area is $\frac{1}{2}$ of the circle with radius r_2. This gives the following expression for the areas of the circles:

$$\frac{A_1}{A_2} = \frac{\pi r_1^2}{\pi r_2^2} = \frac{1}{2} \implies r_1 = \frac{1}{\sqrt{2}} r_2$$

Therefore on average, for simple cluster prediction, an error factor of $0.707 r_2$ is expected even in cases where the correct cluster is identified due to the variability of the exact destination location within that cluster.

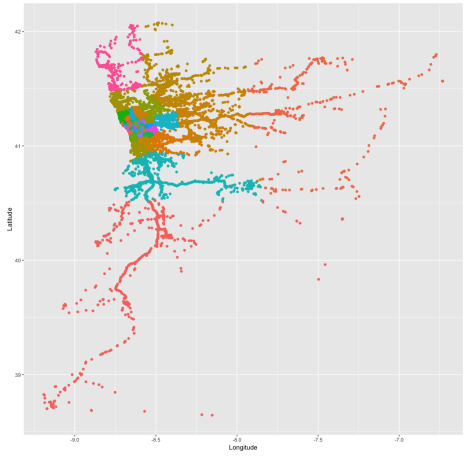


Fig. 5. Sample clustering in larger bounding box

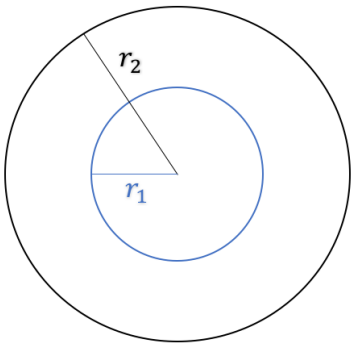


Fig. 6. Cluster Representation

4 The Model

To predict the destination of a taxi using an observed partial trajectory, we propose a hierarchical model. At the high level, the aim is to identify a general region or area the taxi is travelling to. The logic is that the early stages of a trip will be similar for any trip which terminates in destinations within the same neighbourhood. In the clustering scheme, this equates to picking a destination cluster from the coarse clustering. Once the higher destination region is identified, the algorithm will then 'zoom' down and make a fine grained destination prediction within the region identified in the previous step. At this stage a cluster from the fine grained clustering is picked.

4.1 The High Level Model

The high level model is based on Discrete Time Markov Chains (DTMCs). The states of the DTMCs are defined to be the clusters visited by the taxi during its trip. Before training, the trips are factorised by source cluster and destination cluster pairs. A separate DTMC is trained for every source/destination pair. During prediction, the source cluster is known. Only those DTMCs associated with this source are considered. The destination which gives maximum likelihood for the observed transition is then chosen as the predicted destination region.

4.2 The Low Level Model

The low level model is a Bayesian Inference model. We represent each journey as the sequence of clusters the taxi passed through. We calculate the destination cluster probability conditioned on the high level destination region, source cluster and last two clusters visited. The conditional probabilities are calculated as a set of probability tables by counting the transitions between clusters that are observed in the dataset. We represent the source cluster, destination cluster

and last two visited clusters x_i and x_{i-1} as unique cluster IDs. The notation $x_{i-1} \rightarrow x_i$ represents the last observed transition between two clusters in the most recent time step on the trip. Applying Bayes Rule, we have the following: $P(destination_{lower}|x_{i-1} \rightarrow x_i, source, destination_{higher}) =$

$$\frac{\begin{aligned} P(x_{i-1} \rightarrow x_i|source, destination_{higher}, destination_{lower})\times \\ P(x_{i-1}|source, destination_{higher}, destination_{lower})\times \\ P(source|destination_{higher}, destination_{lower})\times \\ P(destination_{higher}|destination_{lower}) \times P(destination_{lower}) \end{aligned}}{P(x_{i-1} \rightarrow x_i, source, destination_{higher})}$$

However, $P(x_{i-1} \rightarrow x_i, source, destination_{higher})$ is a normalization term and not required for the purpose of comparing different probabilities. In addition, this joint probability distribution is computationally expensive to calculate. Therefore, we simplify the above expression and compare based on proportionality. Thus, we look for the destination cluster which maximises:

$$\begin{aligned} P(destination_{lower}|x_{i-1} \rightarrow x_i, source, destination_{higher}) \propto \\ P(x_{i-1} \rightarrow x_i|source, destination_{higher}, destination_{lower})\times \\ P(x_{i-1}|source, destination_{higher}, destination_{lower})\times \\ P(source|destination_{higher}, destination_{lower})\times \\ P(destination_{higher}|destination_{lower}) \times P(destination_{lower}) \end{aligned}$$

For example assume a trip starts in cluster 70 and in the current time step is at cluster 206 having previously been in cluster 115. Assume further that the high level model has predicted a destination region 5. Then we have:

$$P(destination_{lower}|x_{i-1} = 115 \rightarrow x_i = 206, source = 70, destination_{higher} = 5)$$

The experiments were run using different variations of the hierarchical model described. These are as follows:

Baseline The baseline model always predicts the centroid coordinates of the current cluster the taxi is in as the destination.

Model 1 Reports the centroid coordinates of the most likely lower level destination cluster within the high level region (a Maximum A Posteriori estimate).

Model 2 Reports the weighted average coordinates of all lower level destination cluster centroids within the high level region. The centroid locations of all lower clusters are weighted by the probability of each cluster and summed to give the expected destination coordinates.

Model 3 Reports the weighted average coordinates of the top 5 most likely lower level destination cluster centroids within the high level region. The centroid locations of the top 5 lower clusters are weighted by the probability of each cluster and summed to give the expected location. (The probabilities of the top 5 clusters were rescaled before weighting was carried out.)

Model 1 simply picks the most likely destination cluster at the lower level and reports the cluster centroid as the predicted location. However, we have seen that there is an expected error of at least $0.707r$ (where r is the average radius of a cluster) because of the difference between the actual drop off location inside the cluster and the cluster centroid which the algorithm reports (a discretization error arising from the clustering scheme). Models 2 and 3 are motivated as a means to try and address this. In models 2 and 3, the algorithm reports an expected location by multiplying the lower cluster centroids with their probability and summing. This means that if a particular dropoff is within cluster A but very close to or bordering cluster B, then the probability of cluster B will be higher than normal therefore pulling the reported location somewhere between clusters A and B which is the desired behaviour.

4.3 Data Sparsity

A problem encountered with the above method is the high dimensionality of the search space. The number of parameters mean that an exceedingly large number of combinations of those parameters are possible. Assuming a clustering scheme of 259 clusters (which comprises source and visit clusters) and the high level clustering of 8 clusters (the destination regions) then there are 259 sources by 259 previous locations by 259 current locations by 8 destination regions giving 138991832 possibilities at the higher level alone. This is not including the lower level destination prediction which must be made once a destination region is decided.

Some of these possibilities are ruled out by geographic features of the map itself. There are some clusters which cannot be reached from others in one transition and hence cannot form a (previous location, current location) pair. Even excluding these impossible transitions however, we found that the dataset did not contain information on every feasible transition. In absence of any observed data, the model predicts a zero probability for these transitions, which is clearly not the case. A common technique to counter this is to assign a non-zero probability to every possible destination (a uniform prior).

We propose a reasonable approximation of what the missing probabilities might be as a smoothing technique. The idea is given a certain current location and candidate destination, assign higher probabilities to next locations which are in the general direction of that candidate destination.

In figure 7, the vehicle has travelled to its current location (marked C) from the source S. From there the vehicle has some choices around it for the next cluster to visit (marked green). Given the source and candidate destination D_2, the model assigns higher probabilities to next locations in the general direction of D_2. The model achieves this by calculating the angle $\Delta\theta$ which is the difference in heading between the possible next location and candidate destination. This angle is calculated for all possible next locations. The algorithm then ranks the next locations according to increasing $\Delta\theta$. Probability mass is applied to entries in this ordered list by a negative exponential function. When making a prediction, the algorithm first checks if there is data available in the data itself.

If there is none available, it then uses the estimated probability from the model described above.

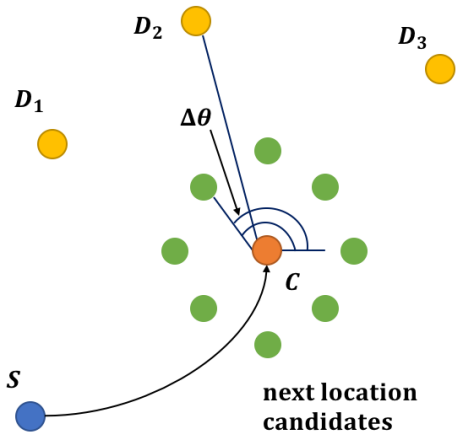


Fig. 7. Estimation of Missing Probabilites

5 Experiments

To conduct the experiments, for each problem, the data was split into a training and test set. Approximately 80% of the trips were randomly sampled for the training set with the remaining 20% being allocated for testing. The evaluation metric used is the Mean Haversine Distance between our predicted drop off coordinates and actual drop off coordinates.

The results from the different model variants (as described in the previous section) in the case of the smaller bounding box are shown in figure 8 and for the larger bounding box in figure 9. In the figures, the x axis indicates the percentage of trip travelled. The y axis is the average haversine distance error over all the test trips at each point in the trip. In the smaller bounding box, the best performance is an error of about 0.6 km in the smaller map and 1.6 km dipping below 1 km as the trip progresses in the larger map. This performance is seen with Models 2 and 3 which show the best performance in both smaller and larger area maps. On the graphs these lines coincide with minimal difference between them. This indicates that picking only the top 5 lower clusters doesn't alter the prediction performance too much as compared to weighting over all lower level destination clusters. The peak at the end of the trip is because at this stage the taxi is actually at its destination but the model still attempts to make a prediction which increases the error. It is noted that within the city centre region, there are patterns to learn and leverage to make a fairly accurate prediction. Increasing the area covered captures the longer trips which was found to have exhibit different and more variable behaviour. This resulted in a somewhat lower

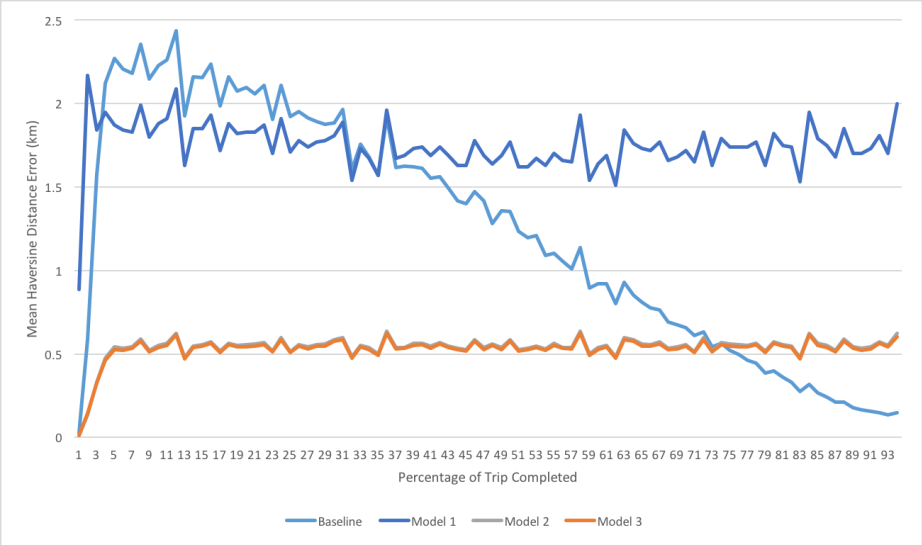


Fig. 8. Experiment Results for smaller bounding box. Models 2 and 3 coincide.



Fig. 9. Experiment Results for larger bounding box. Models 2 and 3 coincide.

prediction accuracy particularly earlier on in the trip. Furthermore, as there is a larger number of destination clusters to choose from, the best result from the larger bounding box is still less accurate than those from the smaller bounding box. Finally the clusters themselves are larger which in turn amplifies any penalty from picking the wrong cluster.

It is noteworthy that in the smaller bounding box, the model reaches peak prediction accuracy very early on (from around 5% into the trip). We always report the centroid of the most likely cluster in Model 1. Our analysis showed that even if we predicted the correct cluster 100% of the time, the discretization would still give us a theoretical minimum error of 0.22 km. To address this problem, in Models 2 & 3, we report the weighted mean location of the low level cluster centroids (weighted by the probability of the cluster). The weighting also helps us in cases where our high-level cluster prediction is wrong. We believe the majority of these cases are where the true destination coordinates are close to the boundary between high level clusters so we end up picking an adjacent high level cluster instead. The low-level prediction is still able to predict the neighbouring low level clusters giving a final predicted location closer to the actual destination.

In the smaller map, while the prediction accuracy reaches its peak 5% into the trip, the prediction doesn't improve from there as in the larger map. Analysing the predictive behaviour of the model, we note the model soon starts to pick either the actual destination cluster or an adjacent cluster and this behaviour continues until the end of the trip. The Markov property may have an effect here as the model makes a prediction based only on the current information at each time step. In the current case, the best models outperform the baseline until about 75% into the trip in the smaller map (considering the average error across all test trips at the 75% point). This indicates that the prediction model is delivering an insight of value for about three quarters into each trip on average.

The Predestination [2] algorithm reports an error of 2.8 km at the start of the trip and dropping to 1 km at the end (on data from a different city). In other papers where this data from Kaggle has been utilized, the authors note that the test set provided is very small for reliable model comparison and therefore test on their own local validation sets as we have done. [3] & [1] report errors on the Kaggle public leaderboard of 2.27 km and 2.39 km respectively. One caveat is that these results may have come from models trained on data which has been less aggressively cleaned than our approach. Our model on the small Kaggle test set of 320 trips scored 2.76 km on the public leaderboard. We believe the local validation is more significant as the results are calculated over a much larger number of trips.

6 Conclusion

In this paper, we presented a hierarchical model for predicting the destination of a moving vehicle. The model made an initial prediction at a high level using Discrete Time Markov Chains for each possible source and destination pair. It then zoomed down to refine the prediction within that higher level cluster using Bayesian Inference. We showed that in earlier phases of the trip, the model

correctly predicts the destination to within 0.5 km on average in our smaller map and around 1.6 km on the larger map (this figure improves as trip progresses on the larger map).

We will improve on these results by considering extra pieces of information. For instance, we will factor trips by time buckets and attempt to learn time based patterns. This captures the idea that the patterns of pickups and dropoffs will vary a lot depending on time of day. E.g.: Dropoffs at restaurants and entertainment venues are more likely in the evening. A second avenue to explore is to relax the Markov assumption and consider greater number of locations (such as last 5) in making our prediction. Finally we will attempt to learn driver preferences (such as favouring a particular section of the city for drop offs and working during a specific time of the day) to develop more flexible models.

Acknowledgement. This research was supported by Science Foundation Ireland under Grant No. 12/RC/2289.

References

1. De Brébisson, A., Simon, E., Auvolat, A., Vincent, P., Bengio, Y.: Artificial neural networks applied to taxi destination prediction. In: Proceedings of the 2015th International Conference on ECML PKDD Discovery Challenge - Volume 1526. pp. 40–51. ECMLPKDDDC'15, CEUR-WS.org, Aachen, Germany, Germany (2015), http://dl.acm.org/citation.cfm?id=3056172.3056178
2. Krumm, J., Horvitz, E.: Predestination: Inferring destinations from partial trajectories. In: Eighth International Conference on Ubiquitous Computing (UbiComp 2006). Springer (September 2006), https://www.microsoft.com/en-us/research/publication/predestination-inferring-destinations-from-partial-trajectories/
3. Lam, H.T., Diaz-Aviles, E., Pascale, A., Gkoufas, Y., Chen, B.: (blue) taxi destination and trip time prediction from partial trajectories. In: Proceedings of the 2015th International Conference on ECML PKDD Discovery Challenge - Volume 1526. pp. 63–74. ECMLPKDDDC'15, CEUR-WS.org, Aachen, Germany, Germany (2015), http://dl.acm.org/citation.cfm?id=3056172.3056180
4. Rousseeuw, P.J.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics 20(Supplement C), 53 – 65 (1987), http://www.sciencedirect.com/science/article/pii/0377042787901257
5. Xue, A.Y., Zhang, R., Zheng, Y., Xie, X., Huang, J., Xu, Z.: Destination prediction by sub-trajectory synthesis and privacy protection against such prediction. ICDE 2013 (April 2013), https://www.microsoft.com/en-us/research/publication/destination-prediction-by-sub-trajectory-synthesis-and-privacy-protection-against-such-prediction/
6. Yuan, N.J., Zheng, Y., Xie, X., Sun, G.: T-drive: Enhancing driving directions with taxi drivers' intelligence. IEEE Transactions on Knowledge and Data Engineering (January 2013), https://www.microsoft.com/en-us/research/publication/t-drive-enhancing-driving-directions-with-taxi-drivers-intelligence/
7. Ziebart, B.D., Maas, A.L., Dey, A.K., Bagnell, J.A.: Navigate like a cabbie: Probabilistic reasoning from observed context-aware behavior. In: Proceedings of the 10th International Conference on Ubiquitous Computing. pp. 322–331. UbiComp '08, ACM, New York, NY, USA (2008), http://doi.acm.org/10.1145/1409635.1409678