**Research Notes**

**Why Useful?**

**These technologies have great potential to revolutionize a number of industries especially with the rise of augmented reality. Examples include the ride-sharing industry, effective advertising by location and other location based services.**

Device ID cannot be tracked with multiple days as it is reset. You can't find a pattern for each device inherently.

Velocity could be important as people tend to travel slower in cities and faster when they are outside city centre. But also if they are rushing for work in the morning vs coming home. Think about velocity contextually.

Look into adding a fourth trajectory to those who have three.

Also, filling in blanks where the tracker was not recording.

Get map of Georgia City in USA.

Look into general cleaning of GPS data and See IBM and NTT Examples.

Think of outliers or possible errors associated with how it was collected using the trackers.

**IBM Blue Taxi Paper Method: (Similar Trips, KNN and Haversine Distance)**

Two similar trips will likely end up in the same place.

So if two people follow the same path and have similar coordinates they might be going to the same place.

- Find (example 10) trips that are very similar to "Trip A" in final destination coordinates.

- For each of pair of trips A and B compute mean Haversine distance.

- Kernel Regression is a smooth version of a K-NN regression. Use this to calculate destination predictions to be used as features. KR requires a bandwidth parameter, set to 0.005, 0.05, 0.5 for varying results.

- But some trips have very different start points but end up in the same place. They attempted to include these by incorporating predictions based on the later trajectories without the first ones.

- Some GPS data can be erroneous and noisy. If people are stuck in traffic, etc. These can affect the features, we can use a RDP algorithm to simplify trips with e = 1x10-6, 5x10-6, 5x10-5. Then repeat the KR regression to extract features.

- Create new features; such as Distance (from city centre to first and to last coordinates points. If former is larger they are considered to be moving towards.)

- To make predictions you should handle latitude and longitude independently. Any regression model works, (try RandomForest because of robust nature and ability to assess each features' contribution), Support Vector Regression, etc). Feature selection using rfcv function in randomforest.

- The evaluation metric for destination prediction is Mean Haversine Distance or find an equivalent for 2-D plane.

**NTT Lab Method: (Transition Probability using RNN encoder-decoder framework)**

- Represent data as a discrete grid-space and let the RNN learn the transitions from one cell to another. The RNN is key to avoid a data sparsity problem. This method finds a probability of reaching multiple destinations not one exact point.

- RNN will take an input of a one-hot representation of location g at timestamp t. Vector gt is then embedded in low dimensional space to obtain semantic representations. These embedded features are fed into hidden layers of LSTM units [5,3]. A Softmax output layer will output the transition probability for each grid cell.

- You can also use the raw coordinates as input to this model.

- An error with the current model is not having any constraint on the distance of a single transition within the grid. Inaccuracies occur as cells jump suddenly to other distant cells. You can update the transition probabilities to account for this (see paper).

- Problem of distinguishing arrival states from moving states. The goal grid cells are connected with other common grid cells and indicate whether the sample arrives at a destination or moves through it.

**Notes on KNN Methods (Try SVM & other Alternatives):**

# Use KNN or SVM for prediction

# The choice of K and the metric (distance) to be used are critical (see what IBM used). KNN is also very sensitive to bad features (attributes) so feature selection is also important. KNN is also sensitive to outliers and removing them before using KNN tends to improve results.

# SVM is good with outliers as it will only use the most relevant points to find a linear separation (support vectors). SVM needs to be tuned, the cost "C" and the use of a kernel and its parameters are critical hyper-parameters to the algorithm.

# Use SVM for higher dimensional input in general.

# Finally you asked about unpredictable situations, in that case I think either a SVM with a RBF kernel or a Random Forest are your best choices.