Two-Stage Trip Destination Prediction

Rohin Mahesh, Subramanian Subramanian
Harman International

Abstract- In this paper, we propose a two-stage approach for predicting a trips destination by clustering geocoordinates and leveraging incremental learning and contextual rule-based methods for the prediction of end destinations. The proposed approach dynamically adapts to users change in behavior and is resilient against concept drift. Using this solution, 76.39% of the predictions were within 100 meters of the actual end destination and 75% of the predictions at most 79.88 meters from the actual end destination.

I. INTRODUCTION

Harman launched the Experience Per Mile initiative in 2019 to shift the focus from connected vehicle technology to consumer-centric mobility experiences and personalization. As part of this initiative, Harman engineers have been working on features to provide hyper-individualized experiences to solve real consumer needs and maximize the time a user spends in their car. One of the features focused on enriching the experience is trip destination prediction.

In this feature, a trip is defined as the movement of the vehicle from a start location to the end location. The trip is qualified by the ignition on/off event, distance traveled, duration of the journey and the run-to-idle time ratio. When the user starts a new trip in their car, the navigation unit in the car will present one or more possible destinations to select and set one of them in the navigation unit. The selection of possible destinations is based on past trip patterns of the user. Using past trip data, the Harman team has developed a predictive solution to suggest possible destinations where the user might be going to in the current trip. Our approach to this problem is to break the prediction into two stages, first predicting the "region" the user will travel to using an incremental learning model, and then make predictions to suggest possible destinations where the user might be going to in the current trip using contextual rule-based methods.

II. TWO-STAGE APPROACH

The proposed process for predicting vehicle's destinations is a two-stage process, combining an incremental learning model with a rule-based frequency method. This process starts with clustering of the users start and end geocoordinates using a Density-Based Spatial Clustering of Application with Noise model (DBSCAN). DBSCAN relies on a density-based notion of clusters which is designed to discover clusters of arbitrary shape [3]. DBSCAN also removes the need to predetermine the number of clusters to create, as it creates clusters using the epsilon and minimum samples hyperparameters.

When working with large spatial data, well known clustering algorithms often suffer from severe drawbacks when applied to large spatial databases. Hierarchical clustering algorithms, for example, face issues in deriving appropriate parameters for the termination condition when applied to large spatial databases. Other popular unsupervised learning methods may not be ideal when working with spatial data. Another example would be the K-means algorithm, which has issues with geodetic distances, as it minimizes the variance. DBSCAN on the other hand works better with arbitrary distances and focuses on compressing spatial data into a set of representative features [3]. In order to reduce the dataset size, we identify coordinates of one point from each cluster that is assigned to serve as a centroid using the great-circle distance. This allows us to represent each cluster with a single point [2].

Next, we map the geocoordinates to addresses as spatially redundant or approximately redundant points often map back to a single feature, rather than many distinct spatial features [2]. Thus, a spatial location can be represented with a single address, in comparison to multiple geocoordinates. With many geocoordinates mapping to the same address, the spatial dimensionality is reduced, and distinct locations are represented concisely.

When a GPS system is used, some outlier points exist and therefore datasets must be filtered to handle these outliers. Some researchers address this by filtering data using a rule that the minimum distance between two consecutive points must be at least 30 meters [1]. Another researcher proposes setting the DBSCAN minimum cluster size allowed to 1, essentially making this a single-link hierarchical clustering process. Resulting from this, every data point gets assigned to either a cluster or forms its own cluster of size 1 [2]. We utilize a custom rule-based system to address outliers to evaluate whether the anomaly should be represented as its own cluster, merged into an existing cluster, or removed from the dataset.

A. Stage 1: Incremental Learning

After utilizing DBSCAN, we can associate a cluster with vehicles start and end location of a trip and we generate a table containing this information. We define the vehicles start cluster as the centroid with the closest proximity to the vehicle's current location. Stage 1 predicts the cluster or "region" that the destination will end in. We found that over a period of time, on average, users tend to end trips at the certain frequent locations but exhibit varying driving behavior to different regions.

When learning in batch, the predictive model would not be able to capture this varying driving behavior and learn from the new trips dynamically. The model's inability to capture this change can lead to it being more susceptible to concept drift, where the statistical properties of the data change over time. Using an incremental learning approach, we train a predictive model to learn continuously from a stream of data. With this approach, the model can capture the change in behavior dynamically, learning one example at a time and updating itself in real-time [4].

Using this table generated from DBSCAN, we train a Self Adjusting Memory model for k Nearest Neighbors (SAM-kNN) to learn incrementally from the vehicle's trips. The combination of Self Adjusting Memory and the k Nearest Neighbor provides the ability to cope with heterogeneous concept drift [6]. This model takes in information at the start of the vehicle's trip and makes a prediction of the cluster or general "region" that the vehicle will end their destination. We utilize river, a Python library for online machine learning to train the SAM-kNN [4].

The Self Adjusting Memory model consists of partitioning of knowledge into two portions: Short-Term memory (STM) which containing data relating to the current concept, and Long-Term memory (LTM) maintaining the compressed knowledge of past concepts in a way that does not contradict those of the STM. The Combined Memory (CM) is created as the union of the STM and the LTM. Three separate distance weighted kNN classifiers are trained using the STM, LTM and CM and the prediction of our complete model relies on the sub-model with the highest weight [6]. Utilizing the Self Adjusting Memory model enables the predictive model to dynamically adapt to the users change in behavior, by updating the LTM.

B. Stage 2: Frequency

Once the SAM-kNN model takes in information at the start of the vehicles trip and makes a prediction of the cluster or general "region" that the vehicles will end their destination, we build off of our analysis that on average, vehicles tend to end trips at the certain frequent locations. The vehicles end geocoordinates for every cluster is first converted to addresses. Next, we calculate the relative frequency for addresses for every end cluster. The resulting vehicle id, end cluster and relative frequency is stored in a Frequency table. This table is continually updated in order to reflect the change in driving behavior. Once a prediction of the end cluster is made in Stage 1, the Frequency table is queried (using the predicted end cluster and vehicle id) and up to three addresses with the highest relative frequency are returned to the head unit of the vehicle.

These addresses serve as our most confident prediction as to where the vehicle will end its trip based on previously observed driving behavior. A factor that is detrimental to the destination prediction models performance would be one-off trips (situations where vehicles make distinct trips to locations). Our two-stage approach addresses this issue as the most confident predictions are the end destinations with the highest relative frequency. These one-off trips will have the lowest relative frequency and the two-stage process can dynamically adapt to the change in driving behavior, including the change in relative frequency for one-off end destinations that are being visited.

III. RESULTS

We evaluated our two-stage approach over three metrics over all 95 vehicles. The first metric is the Stage 1 Accuracy, which we averaged over all 95 vehicles. Our Stage 1 incremental learning model yielded an average accuracy of 54.52%. Next, we measured the meters between our best suggested address (out of the three provided by Stage 2) and the vehicles end destination. On average, for our two-stage approach: 25% of our predictions were at most 1.7 meters from the actual end destination, 50% of our predictions were at most 6.48 meters from the actual end destination, and 75% of our predictions were at most 79.88 meters from the actual end destination.

The second metric is the 100 Meter Precision. This is defined per vehicle as the number of predictions that are within 100 meters from the actual end destination divided by the total number of trips a vehicle has taken. We found that across the 95 vehicles: 6.32% of the vehicles had a 100 Meter Precision less than 0.61, 89.47% of the vehicles had a 100 Meter Precision between 0.61 and 0.90, and 4.21% of the vehicles had a 100 Meter Precision greater than 0.90. The average 100 Meter Precision across all 95 vehicles was 76.39%.

$$100\ Meter\ Precision = \frac{Number\ of\ Predictions\ within\ 100\ Meters}{Number\ of\ Trips}$$

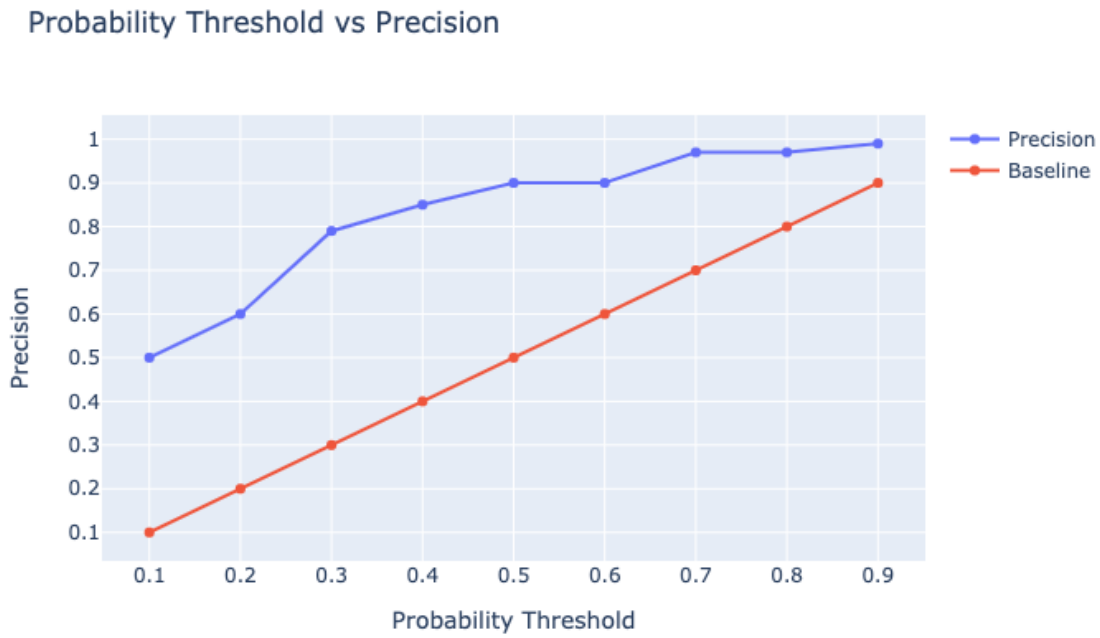
1st Quartile	Median	3rd Quartile
1.70 meters	6.48 meters	79.88 meters

Stage 1 Accuracy	100 Meter Precision
54.52 %	76.39 %

Percentage of Predictions within 100 Meters	Number of Vehicles
40 – 50	1
51 – 60	5
61 – 70	21
71 – 80	36
81 – 90	28
91 – 100	4

The third metric that we are tracking is the Precision. The Precision is similar to the 100 Meter Precision, but this metric evaluates the number of predictions that are within 100 Meters with the total number of predictions made. We compare the Precision across every probability threshold as well as with the baseline. The probability threshold is the relative frequency at particular cutoff value. Starting at a probability threshold of 0.1, we first found the predictions where the best prediction (of the candidate addresses, the address closest the actual end destination) probability is less than or equal to 0.1. Based on those trips, we calculate the Precision. The baseline serves as a benchmark that we set to be above at every probability threshold. If we are ever below a baseline for any probability threshold, we do not make and return vehicle destination predictions. We found that the Precision was above the baseline across all probability thresholds.

$$Precision = \frac{Number\ of\ Predictions\ within\ 100\ Meters}{Total\ Number\ of\ Predictions}$$



IV. AREAS FOR IMPROVEMENT

Many of the state-of-the art approaches to this problem utilizes external information from the geographical environment [1]. The idea is that the combination of trajectory data with personalized data leads to more accurate destination predictions [5]. Some researchers merge open source POI and land use data with the GPS dataset for the destination prediction task, allowing for the prediction of an engineered feature called the "trip purpose". Given this trip purpose, a 5-tier architecture for destination prediction is implemented, yielding an accuracy of 52.7%. Other researchers link locations to roadways and infers previous trips' routes to determine location [5]. Examples of this include the generation of street maps, map-matching, identification of crossroads, and engineering of features based on this external information [1]. We also propose utilizing various external information available at a head unit level of the vehicle to establish a Destination Suggestion Service (DSS).

V. IGNITE DESTINATION SUGGESTION SERVICE

Users take trips for a variety of reasons such as going to work, to keep up an appointment, dining, shopping, filling up gas, charging an EV, visiting park/beach, movie theater, etc. Along with displaying the destination and the route to the destination, user experience can be enhanced by showing additional information based on user's current context, past trips or cohort group trips.

Current research and implementation considers trip patterns from the past to predict where the user will be going in the current trip. We envision an Ignite Destination Suggestion Service (DSS) that is based not only on past trip patterns, but also on different destination sources and context. These suggestions can also be classified based on how the suggestions are made, namely - inference, prediction and recommendation. These can be computed independently, in parallel, or in sequence and combined before sending it back to the service caller. DSS can also be called with options to indicate what sources it should consider for destination suggestion, giving the caller complete flexibility.

A. Inference

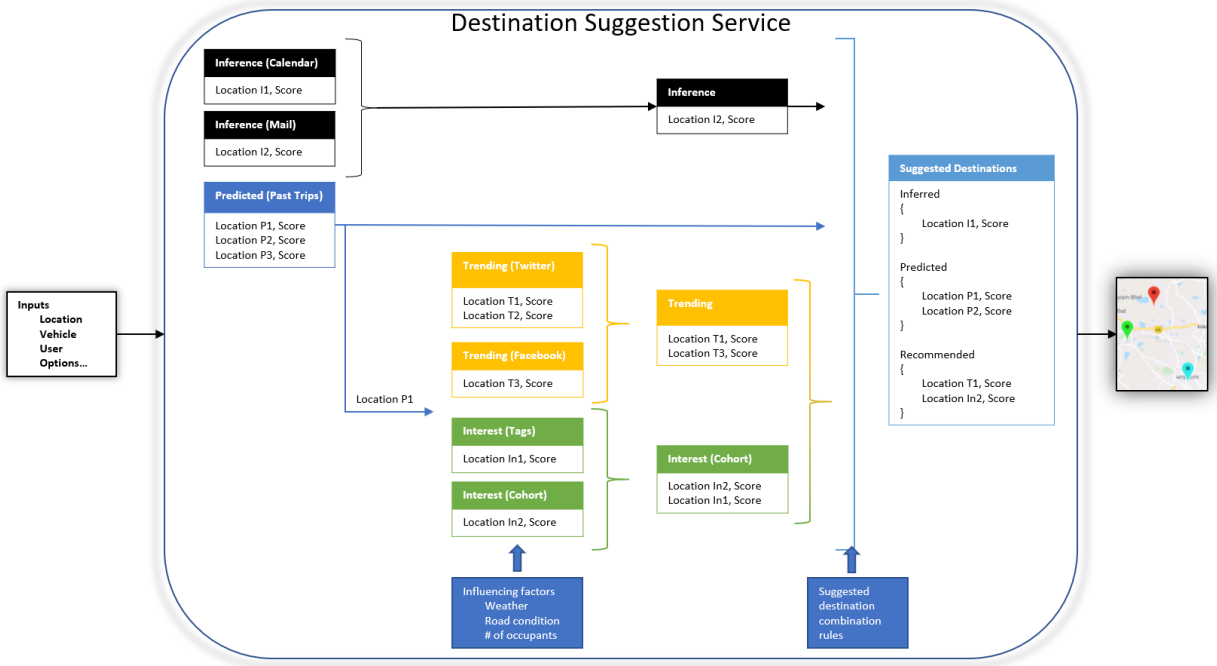
Inference-based suggestions are destinations suggested based on the data source context, such as appointments from the user's calendar. Here we look at contexts such as appointments, meetings, or reminders in the user's calendar to infer where the user might be going. For example, from users' emails, we can infer that user has to pick up a package from Amazon Locker and add the locker location as a destination suggestion. These inference-based suggestions can be used as rules and yield the highest weightage in the process and can flexibly override predictions and recommendations.

B. Prediction and Recommendation

The Prediction component refers to destination locations predicted based on past trips. The proposed two-stage destination prediction approach is used to predict 1 to 3 possible destinations along with a confidence score. Predicted destination will also include via-points on the way to the destination. Recommendations are destinations that can be suggested to users using various contextual information based on what users may be interested in. These recommendations are derived from Trending Locations and Personal Interest Locations.

Trending Locations are locations where some events (e.g. food festival, game, store sales) are happening or trending, as understood by mining data from social feeds like Facebook, Twitter, etc. A crawler is first built to scrape data from popular social media websites such as Twitter while filtering on geocoordinates, which can be taken either from inferences or predictions. Next, Named Entity Recognition (NER) is performed on the corpus using transformers, yielding entities such as organizations, locations, and people. Finally, the corpus is filtered to include locations and organizations, which can be used in conjunction to derive locations as recommendations.

Personal Interest Locations are locations based on user specified interest tags e.g. music, food, hiking, etc. These tags are derived from the users end destinations by utilizing API's like Yelp. Another approach for personal interest location includes building cohort groups of users in the system. To build the cohort groups, the destinations of all the trips in the system are categorized and then the users are grouped based on variables representing the categories of the destinations they have visited. This data is fed into a recommender system to make recommendations for possible tags that users may be interested. These tags can be used to search and filter for locations in the proximity of users end destination (derived from either inference and/or prediction component) to be used as recommendations.



VI. CONCLUSION

The proposed two-stage destination method approaches the problem in two stages: first training an incremental learning model to predict the cluster or "region" of the end destination, then generating up to three predictions of the end destination using a rule-based frequency approach. This method yields precise estimations of the end destination of vehicles, as 75% of the predictions were at most 79.88 meters from the actual end destination. Due to varying driving behavior across vehicles, there are no universal hyperparameters that are optimal for all users. We strongly recommend using further contextual external information to aid in making more precise predictions using the two-stage approach.

REFERENCES

[1] Alvarez-Garcia, J.A., Ortega, J.A., Gonzalez-Abril, L., and Velasco, F., Trip destination prediction based on past GPS log using a Hidden Markov Model, Expert Systems with Applications (2010), doi: 10.1016/j.eswa.2010.05.070.

[2] Boeing, G. (2018) Clustering to Reduce Spatial Data Set Size.

[3] Ester, M., Hans-Peter, K., Sander, J., and Xu, X. (1996) A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.

[4] Halford, Max and Bolmier, Geoffrey and Sourty, Raphael and Vaysse, Robin and Zouitine, Adil., river: Online machine learning in Python.

[5] Krause and Zhang. (2019). Short-term travel behavior prediction with GPS, land use, and point of interest data.

[6] Losing, V., Hammer, B., and Wersing, H. (2017) KNN Classifier with Self Adjusting Memory for Heterogeneous Concept Drift.