Harry Zarcadoolas

Homework 1 Assignment: Report

September 21, 2024

Overview and Key Points

In this analysis, I used SUMO traffic simulator data at an intersection in Gainesville over a timespan of a few hours to cluster vehicle trajectories based on both spatial and temporal features. This was to see if patterns could be identified in vehicle movement or speed to discern vehicle path. The SUMO data contained vehicle movement and trajectory data, including coordinates, maneuver types, as well as timestamps to go along with the movement. I then compared results to ground truth values to see which methods were effective. I performed spatial clustering twice on different measures, the first being entry point and maneuver and the second being based on trajectory data using x and y coordinates. For the temporal data, I based clusters on the total time it took each vehicle to cross the intersection.

For the actual clustering algorithms, I used mainly **K-Means** algorithms with cluster amounts mirroring the ground truth amount, 11. Additionally, I implemented a **DBSCAN** clustering algorithm for spatial clustering with coordinates to observe possible variations with a density-based scan that considered possible noise. These methods were from the 'scikit-learn' python package.

Approach

Data Loading and Exploration

I began by loading the SUMO simulation dataset and exploring the dataset columns. The key columns were cur_time, vehicle_id, x_cord, y_cord, entry, maneuver, and cluster. The remaining less relevant columns were vehicle_or_pedestrian and signal. I also gained information on the data types and completeness of the columns, which turned out to be a complete data set. I also observed basic statistics on the row data from each column to see what expected values were for each attribute. The initial exploration helped gain an understanding of the characteristics of each data attribute and how to handle them for future clustering.

Spatial Clustering

Simple Clustering with Entry and Maneuver

I performed the initial clustering based on the combination of **entry** and **maneuver** attributes, which were based on vehicle lane at the entry of the intersection and turning behavior going into the intersection. I first decided to group and count the number of unique combinations of these

features to determine a proper k value, (confirming my inferred value of 3^2=9 combinations). After, I ran the K-Means algorithm with the features being entry and maneuver for each vehicle. The resulting cluster IDs were then visualized using the x and y coordinate data from the vehicles in the original data set (merged with the new cluster IDs). A basic scatterplot implemented from the 'seaborn' and 'matplotlib' packages were used.

K-Means on Vehicle Trajectory Features

Next, I used the K-Means algorithm to cluster vehicles based on their trajectory data features. These were essentially the **x** and **y coordinates,** including **start** and **finish**, as well as **total distance in each direction** (from end coordinates subtracted by start coordinates). I considered the following features: start and end x and y coordinates and the range of movement in both x and y directions via basic subtraction of end minus start. Eleven clusters were ultimately chosen to be generated, matching the number of ground truth clusters. Before I performed a comparison to the ground-truth cluster values, I performed an internal index strategy of using a silhouette score to measure how well the clusters were. Then I did a basic plotting of the trajectory clusters using a scatterplot with x-coordinates and y-coordinates on their respective axis and the hue being the cluster ID. After this, I observed that the generated cluster IDs did not directly match the ground truth values, and I made the realization that the numbers would likely have to be mapped first. So, I **mapped** the K-Means clusters to the most frequent ground-truth clusters using cross-tabulation and considering the trajectory cluster ID that corresponded most to each ground truth cluster ID. This allowed for a direct comparison between the two. The clustering in this case was **highly effective**, around 95%.

DBSCAN

To experiment, I used an alternative clustering method, **DBSCAN**, and applied it to the same trajectory features. DBSCAN in general can be useful for identifying clusters with varying densities and ignoring outliers. I was curious to use it on this data set because I assumed the noise would already be very minimal, but the densities could be applicable. I filtered out possible noise points (clustered as -1 in DBSCAN) and calculated the silhouette score for the remaining clusters to do a basic evaluation of the clusters – the score was sufficient for decent clusters.

Temporal Clustering

Feature Creation

To identify temporal features, I calculated the **total time** taken by each vehicle to cross the intersection by subtracting the vehicle's entry time from its exit time.

K-Means on Total Time

I normalized the total time values using **MinMaxScaler** and followed this by applying K-Means to group vehicles based on their normalized crossing times. This purpose of the temporal clustering allowed the identification of paths vehicles took based on their varying speed patterns. I then used another scatterplot with unique vehicle ID on the x-axis and normalized total time (to cross the intersection) on the y-axis, with time clustering as the coloring (for data points). Similarly to spatial clustering, I mapped the resulting temporal clusters to the ground-truth clusters. After, I calculated the match rate.

Results and Performance

Spatial Clustering Performance

The spatial clustering was highly effective. Each silhouette score was in an acceptable range for promising clusters. Additionally, the match rate for trajectory clusters compared to the ground truth values was nearly 90%, showing a strong alignment. The DBSCAN performed a little more poorly, but that was somewhat expected as the cluster counts were already predefined, and outliers were slim.

Temporal Clustering Performance

The temporal clustering was significantly less effective than spatial clustering, but the effectiveness was still notable and exceeded my expectations. The match rate between temporal clusters and the ground truth cluster values was nearly 20%, showing that there might not be a direct correlation but still a loose connection to certain clusters. Therefore, further data analysis can prove that difference in vehicle speed patterns could indeed affect paths taken at the intersection.

Conclusions and Insight

Spatial clustering had well-defined trajectories and movement features which led to predictable cluster identification with high certainty. Considering that the path of vehicles is spatial by nature, it is transparent that spatial clustering would have such a high effectiveness. On the other hand, a different metric was considered, being speed via total time for a vehicle to go through the intersection. This formed the basis for temporal clustering features, which showed a loose alignment with the ground truth table values. This suggests that speed may have influence on the vehicle's path but remains an unviable option to determine specifically where the car travels. Instead, this temporal clustering could be better suited to determine a metric like traffic flow or throughput. Additionally, there could be overall further investigation of the use of other clustering algorithms like Hierarchical Clustering to better capture the relationship between speed and trajectory and lead to effective clustering methods.