Supervised Learning to Predict Human Driver Merging Behavior

Derek Phillips, Alexander Lin
{djp42, alin719}@stanford.edu

June 7, 2016

Abstract

This paper uses the supervised learning techniques of linear regression and support vector machines in an attempt to predict the merging behavior of drivers on U.S. Highway 101 based on current traffic patterns. Using highway data from the Federal Highway Administration, the paper approaches the problem from numerous angles, eventually concluding that using a two-step approach achieves the best result. The first step is to cluster the drivers based on driver history, and train separate models for each of the clusters. This results in the best results in every case.

1 Introduction

Understanding human driver behavior is a critical component in the development of autonomous vehicles. In previous research, lane-shifting and regular highway driving has been well studied, but the analysis of merging behavior is sparse. We are developing a model for predicting merging behavior on highways given positional and trajectory information of the traffic. The model predicts vertical position of a merging vehicle given the current traffic and time elapsed.

2 Data and Processing

Our data is from the NGSIM database and includes positional data for vehicles over a stretch of highway 101 and I80. We transformed the data from the NGSIM database to include directional velocity, and ultimately arranged it in grid format to allow for a constant feature size to be input into our models. For each grid entry, we keep an average of the features of the vehicles in that grid area. The grid has low resolution to maintain a small feature set size, and we further cut down the feature set size by limiting the window of interest to a subset of the highway from when the merge begins to when it ends. For each frame the vehicle is in, the input is the current grid of traffic (with the merging vehicle removed) and the time elapsed since the vehicle first entered the area. The output will be either the X or the Y position of the merging vehicle.

In its raw form, we only have position and X-velocity and X-acceleration. We decided to augment our data to include velocity and acceleration on the Y-axis. Since we are studying the merging and positional behavior of vehicles, this data is extremely valuable.

To streamline the runtime of our prediction algorithms, we precompute each of the data representations that we plan on using. We have created several models as described below, and plan

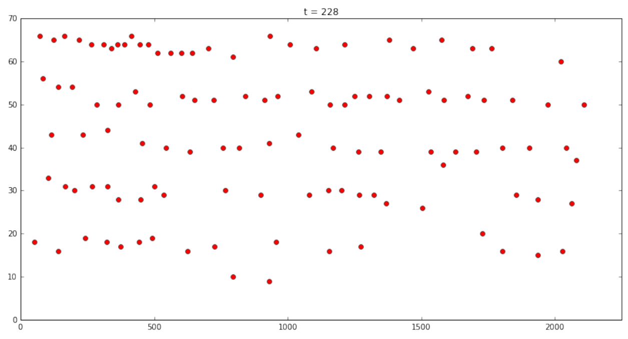

Figure 1: Visualization of Highway Vehicles

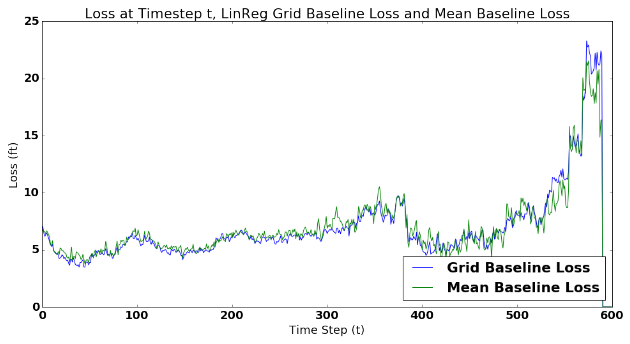

Figure 2: LinReg Baseline Losses

on testing more to identify one with which we can get successful results.

Additionally, we built several visualization tools to help us gain intuition about our dataset as well as sanity check our work. A frame from our visualization tool is shown - you can clearly see congestion building up, as well as cars merging on and off of the roadway.

3 Preliminary Results and Baseline

We generated our baseline results using two naive feature implementations. The first consisted simply of the mean velocity of vehicles on the road at the time. Our second baseline consisted of a simple grid, with no other information besides the number of vehicles within each grid square. We ran both SVM and linear regression on this dataset, and the results are shown below. Although the overall scores / losses for these baselines were relatively similar, the distribution of what contributed to the losses were different.

Generally, it appears that the performance of the predictor based off of just the mean roadway velocity is better at earlier timesteps, whereas the predictor based off of vehicle location is able to better fit the data at later timesteps. As we enhanced our data, our results would more closely match the grid baseline.

4 Methods and Techniques

4.1 Linear Regression vs. SVM

As we developed our model, it quickly became clear that SVM regression was the superior of the two techniques for our application.

4.2 Tuning SVM

One of the main characteristics of an SVM is the ability to tune the hyper parameters – the penalty weight and the epsilon value. These alter the bias and variance of the model, so choosing the best parameters is crucial to producing accurate results. Using a subset of the data, various values were tested, including penalties centered at the default value of 1, maxing out at 10,000, and bottoming out at 0.01. The epsilons were tested between values of 1e-6 and 100. All pairs of possible combinations performed worse than the default (penalties=1, epsilon=0.1), except for the pair with a penalty of 0.1 and epsilon of 10, which performed better in our scoring metrics. However, these results were too invariant to the different possible car trajectories. Thus, the default parameters were chosen for the rest of the project, as they provided the best balance of average accuracy and producing trajectories that matched the shape of the test vehicles.

4.3 Mean-Centering Data

Seeing as how absolute data is not always as useful as relative data, one attempted strategy was to mean center the grids before creating the model. This did not have any significant change to accuracy of the models (linear regression and SVM), and it took a fair amount of time longer, so the rest of the project uses non-mean centered data.

4.4 Tuning Features

The next logical step to examine variation in the model such that the best possible model can be produced is variation of the features. Initially, the features were the grid of current traffic; the time elapsed since the merging vehicle entered the area. The grid itself is what was first examined for improvement. The grid first contained the number of vehicles in the grid, and the average instantaneous velocity and acceleration of those vehicles in the x direction, but there were many other features that could have been chosen.

4.4.1 Time and Space Headway

The first of these features to be tested was headway. The motivation behind headway was that not all of the vehicles are the same size, so it might be more useful for the model to know how much space is between cars than other features. The results were disappointing; with the SVM model doing worse than without headway, and linear regression doing marginally better.

4.4.2 Y-Velocity, Y-Acceleration

Similar results arose from including the average instantaneous velocity and acceleration in the y direction. The hypothesized cause for these failures is that adding more features made it harder for the SVM to find the correct weights.

4.4.3 Frame History

Another examined variation on the feature set was giving the models 2 frame grids per training example, the frame from 10 time steps ago as well as the current frame. The goal of this approach was to give the model more information about the history of the traffic as supposed to only the current snapshot. Doubling the feature size unsurprisingly resulted in a significantly longer runtime, with limited to no improvements to show for it.

4.4.4 Vehicle Initial Position

One feature that increased the accuracy of both models was including the initial position of the merging vehicle along with the current traffic and time elapsed. This helped because, not only did it only add 1 feature to the training examples, but it gave perspective to the model, as it is possible for different merging vehicles to begin in very different locations of the merge ramp.

Table 1: Results of Feature Tweaking

Characteristics	SVM Score	Linreg Score
SVM-1-0.1	0.24	0.16
SVM-10000-0.1	-.4	0.16
SVM-10-1e-06	0.24	0.16
Mean-Centered	0.24	0.16
Headway	-0.007	0.36
VyAy	-0.004	-0.05
Frame History	0.24	0.06
Inital Position	0.29	0.26

Table 2: Here, the default grid features in a particular are #vehicles, instantaneous velocity and acceleration in x direction. The 'Characteristics' are additional features as explained in the previous sections. (score ≤ 1 and using sklearn score for each model)

4.5 K-Means Clustering to Classifying Driver Behavior

Up until this point the models have been predicting the trajectory given only information about the current situation. Additionally, the models tend to perform well on merging vehicles that only merge one lane, but poorly on vehicles that move multiple lanes over immediately after merging. So the problem becomes: how can the model better between cars that are likely to merge numerous lanes and cars that are not (without cheating it and telling it the future)? The answer is with behavioral data for the driver, based on the driver's historical information. For example, a historically aggressive driver is more likely to merge into a small gap, or to speed up to overtake a car in the lane that it is trying to get to, along with being more inclined to traverse multiple lanes immediately after merging. However, NGSIM does not historical data for the drivers, and how can the behavior be calculated in such a way as to be helpful for the learning models?

The approach taken was to create clusters based on the headway, acceleration, and velocity of the cars. The motivation behind this was that an aggressive driver would keep less headway between them and the car in front of them, and would have shaper acceleration and a higher speed. To ensure that no unfair advantages were used, the clustering data was taken from the portion of the highway after the merge, which still provided about a quarter mile's worth of data, and positional data was not used. Additionally, before the clustering began, the data was adjusted to be relative to the current traffic. What this means is that there will be no bias of the data where cars that travel in heavy congestion will are more inclined to be aggressive because there is less headway in front of them. All vehicles were classified to give a broader data set than just using the merging vehicles. After this mean-centering was performed, we ran 3-Clustering on the data to output different classifications of drivers. We separated our models based on these clusters, and tested performance.

The results from this were fairly interesting. The clusters were not evenly distributed, with the vast majority of vehicles belonging to one cluster. However, the performance on that cluster was much better than any other models, and the performance on the other clusters was only slightly worse than normal. However, linear regression performed terribly.

Table 3: Clustering Scores

Model	cluster 0	cluster 1	cluster 2
SVM	0.19	No examples	0.48
Linreg	-3.92	No examples	-0.01

4.6 Reducing Grid Size

Another method in which we attempted to improve our results was through varying the size of our feature grid. We recognized that it was likely that some grid positions (i.e. the leftmost lanes furthest from the merging vehicle) would have little effect on predicting the merging vehicle's behavior. Additionally, these increased features could have led to overfitting of our model. Reducing the size of our grid from 60x30 indices to 60x10, and ignoring the top lanes decreased our loss across all examples.

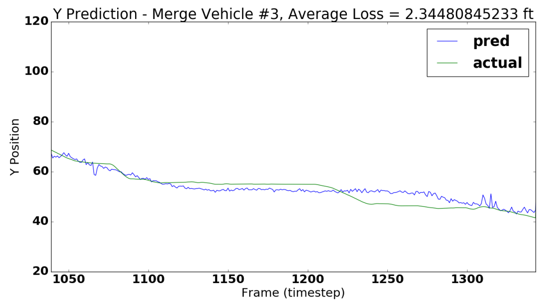

Figure 3: A well-behaved, single lane merge

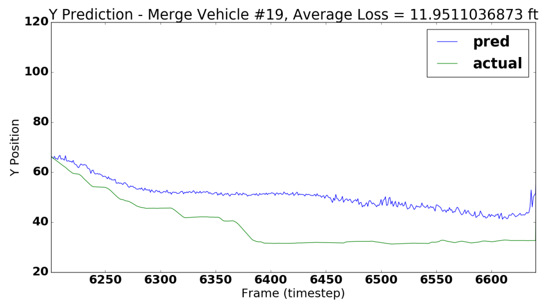

Figure 4: Example of multi-lane merge

5 Final Results and Analysis

Our most successful models came from using clustered models trained on the smaller grid size (60 x 10), with each grid position having features for number of vehicles, x-velocity and x-acceleration. It is important to note that our predictions were trained and scored over the entire time that a vehicle was present in the grid frame. However, the grid frame for some merging vehicles extended over 600 time frames, so we are tracking in total one minute of vehicle position data. As one would expect, the losses at earlier points

Additionally, it seemed that throughout our experiments with feature modification, the overall shape of our predictions were relatively similar between runs. For the vehicles that remained within the first lane, this allowed us to achieve incredibly high accuracy, with some examples having average losses of below 1.5 feet across the entire dataset. However, this resulted in worse performance over vehicles that merged across several lanes right away.

We hypothesize that the result described above can be attributed to several factors. The first, simple explanation is that we could be overfitting our data. However, we believe there may be a deeper issue. In our training data, we have approximately 130 vehicles per 15 minute time period that merge onto the road. Although we break these down into hundreds of training examples each, we were unable to get around the fact that out of these approximately 130 vehicles, very few of them changed into multiple lanes. With the dataset used, we found it difficult to identify features that directly correllated to whether or not a driver would merge across multiple lanes consistently, even with k-clustering of driver behavior.

6 Future Work

We have identified several viable methods which we hope can produce better results in the future. Running more tests on feature selection and identifying the features that most strongly correllate to multiple-lane merges would likely greatly improve results. Much of the observed losses were a result of our model being unable to accurately predict the trajectory of a car merging multiple lanes, so a good first step would be to classify based on historical driving data (of the specific driver) if the driver will merge multiple lanes, and if so use a different model than for merging only one lane.