CS230

Identifying Driver Intention From Vehicle Data: A Deep Learning Project

Elliot Weiss
Stanford University
elliotdw@stanford.edu

Yijun Hou
Stanford University
yijunhou@stanford.edu

Vasudevan Iyengar
Stanford University
vasuiyen@stanford.edu

Abstract

Increasingly automobiles are becoming mini computers on wheels capable of generating various types of data from sensors and actuators that can be used by deep learning models to better understand human driving behavior. The goal of this project is to extract useful information about the high-level intention of the driver and to predict the driver's future controls to the vehicle using low-level vehicle state and control input data. These learned driver models can be applied to the design of novel driver assistance systems that understand the intentions of the human operator and also as a guidance interface for drivers based on individual driving history. In this project, we proposed two LSTM RNN models which achieve high accuracy in maneuver classification and low error in predictions of the driver's future control inputs.

1 Introduction

In this project, we have designed and trained deep learning models that can understand a driver's intentions from data collected by vehicle sensors. To train our models, we collected tabular vehicle state and control input data while driving in a fixed-based virtual reality (VR) driving simulator. Since we are using time series data, we have focused on the use of recurrent neural networks (RNNs), which are capable of storing information through temporally-updated internal states, to make classifications and predictions. We have implemented RNNs with long short-term memory (LSTM) layers because of their ability to solve the problem of vanishing or exploding gradients found in a "vanilla" RNN implementation. In this report, we demonstrate our approach to two different problems: one is the classification of the driver's high-level maneuver from a set of discrete options, and the other is the prediction of the driver's future sequence of control inputs to the vehicle.

2 Related work

There are various studies and papers available that address the general problem of modeling driver behavior with methods from deep learning. "Human motion trajectory" [1] and "Driver Lane Intention" [2] provide two surveys of driver intention prediction; the first paper provides a broad overview of relevant research, and the latter focuses on highway lane change scenarios. "Human-robot interaction" [3] demonstrates the learning process for human-robot interaction based on hand-collected driving simulator data for traffic weaving maneuvers. "Path prediction" [4] uses RNNs to understand the intentions of drivers with a focus on cases at intersections. "Driver behaviors in dynamic environments" [5] builds different data-centric models for classifying driving intent based on labeled driving simulator data using traditional machine learning methods including Support Vector Machines, Random Forests and Logistic Regression. This last source provides a useful baseline to understand the effectiveness of our approaches. The approaches that we have taken in this project can be extended to various use cases, beyond those previously studied, that are relevant for the design of partially and fully autonomous vehicle technology.

,

3 Dataset

3.1 Data Collection

We collected time series vehicle state and control input data in a VR driving simulator. Two separate datasets were collected in different driving conditions, which are each used for one of the two different problems in this project.

3.1.1 Two-Lane Overtaking

For the maneuver classification problem, we collected a dataset that contains 99 driving trials that were evenly split into 3 classes: aggressive overtaking in front of an oncoming vehicle, not overtaking, and waiting for an oncoming vehicle to pass before overtaking. Each trial in the simulator lasted approximately 30 seconds, and data was collected at a rate of 50 Hz. State data was collected from the three relevant vehicles: the ego vehicle (controlled by the human driver), the lead vehicle, and the opposing vehicle. The state data comprises two-dimensional position, velocity, and acceleration; yaw angle; and yaw rate. For the ego vehicle, we additionally collected the steering wheel angle, throttle pedal position, and brake pedal position control input data.

3.1.2 Highway Driving

For the control input prediction problem, we immersed the driver in a more complex highway driving scenario where the ego vehicle interacted with multiple vehicles creating traffic on the road. On the virtual highway, we collected 2.5 hours of continuous driving data, which was split into 53 trails of 3 minutes each. The data was once again collected at a 50 Hz rate. The same state data as described for the Two-Lane Overtaking dataset was collected, although in this scenario we collected it for the ego vehicle and the two closest other vehicles on the highway at any given point in time. The same ego vehicle control input data was also collected. Lastly, we collected road curvature at the ego vehicle's current location, as this strongly influences the driver's steering commands in a curving highway environment.

3.2 Data Processing

The two datasets were processed in a similar manner. To create model inputs $X^{(i)}$, we broke each trial into sliding windows of 50 time steps each, resulting in 1 second chunks of data. For the model output, the classification problem used the known maneuver label from each trial encoded as a three element one-hot vector, while the prediction problem used the next 250 time steps (representing the next 5 seconds of driving) of ego vehicle control inputs for prediction. Table 1 shows the total number of data points ($X^{(i)}$, $Y^{(i)}$) for each problem.

	Classification	Prediction
Number of ($X^{(i)}$, $Y^{(i)}$)	138,303	261,047

Table 1: Data Size

3.3 Training/Test Splits

Once the ($X^{(i)}$, $Y^{(i)}$) pairs were collected and transformed into time series inputs as mentioned above, they were randomly shuffled and divided into training and test sets, with 80% for training and 20% for testing.

4 Methods

4.1 Classification Problem

A three layer RNN architecture was developed to model the relationship between the 1 second history of vehicle data and maneuver classifications as shown in Figure 1. The data first goes through an LSTM layer with 100 hidden units to capture dependencies over the entire time span of the input

data. The second layer is a fully connected layer with 100 hidden units with a rectified linear unit (ReLU) activation function. The third and final layer has 3 hidden units that produce a softmax output for multiclass predictions. Table 2 tabulates other information about our model including the loss function, optimizer, and number of parameters.

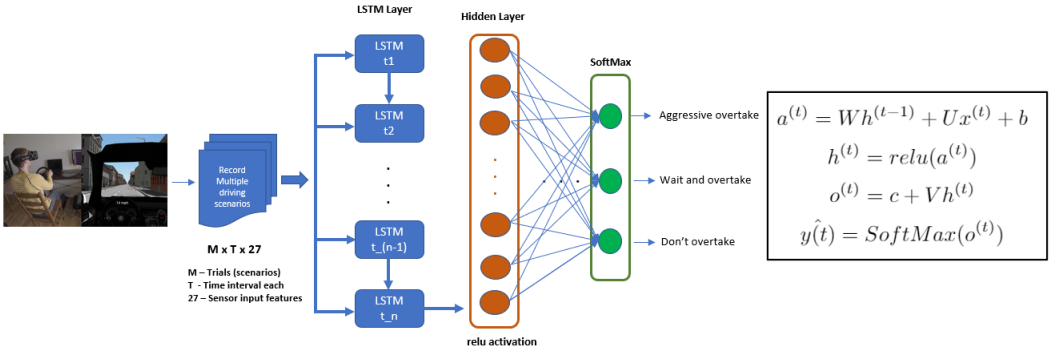

Figure 1: LSTM network architecture for the classification problem.

4.2 Prediction Problem

As shown in Figure 2, we developed a five-layer RNN to model the input-output relationship for the prediction problem. Since both the input and output are time-series data, we used an encoder-decoder sequence to sequence model. The data first goes through an LSTM layer with 100 hidden units to capture the entire input time span. The second layer is a 500 unit dense layer with rectified linear unit (ReLU) activation function, followed by a Repeat-Vector layer to add an extra dimension required for the output. This is then followed by another LSTM layer with 100 hidden units and then finally a time distributed layer that outputs the sequence of steering, throttle, and brake controls. Table 2 contains other information about this model.

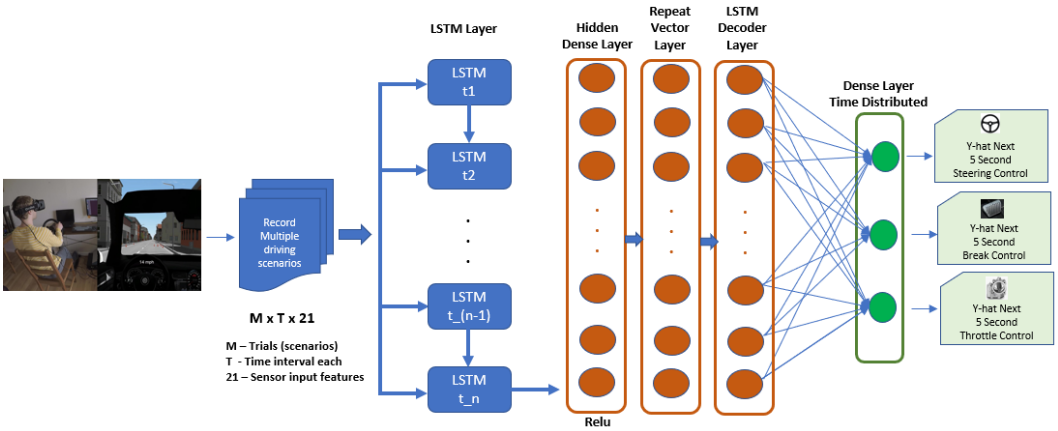

Figure 2: LSTM network architecture for the prediction problem.

	Loss Function	Optimizer	Total Number of trainable parameters
Classification	Categorical cross-entropy	Adam	61,603
Prediction	Mean squared error	Adam	339,603

Table 2: Model Summary

5 Results and Discussion

5.1 Model Training

To train and evaluate our models, we used the `TensorFlow 2` framework run in Google Colab environments with GPU as the run time environment. The average training time for both of the models is about 25 minutes with 20 epochs and batch-size of 256.

5.2 Results - Classification Problem

The model achieves 81.31% accuracy on the training set, and 81.86% on the testing set, indicating that we have not overfit to the training data. Figure 3 shows the model's maneuver predictions over time on data for three different trials, each trial demonstrating one of the maneuvers. As shown in the plots, the RNN is able to correctly identify the maneuver as the maneuver unfolds (ie the driver begins accelerating to cut in front of the opposing vehicle for an aggressive overtaking maneuver). Furthermore, the aggressive overtake maneuver is the most quickly distinguished, as it is most immediately obvious when the driver is speeding up to aggressively overtake the lead vehicle.

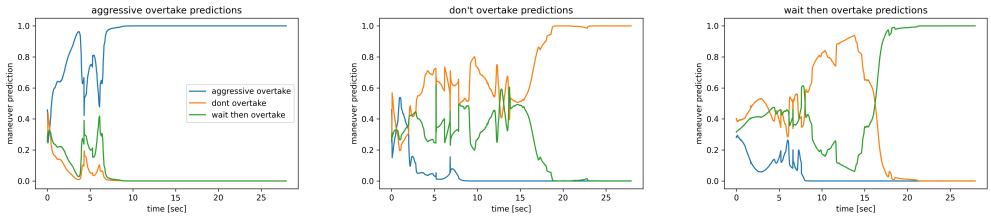

Figure 3: RNN predictions for vehicle data over one trial representing each of the three maneuvers.

We also compared our RNN to two simpler models that don't take into account the time dependent nature of the input data. These baselines are a random forest model and a logistic regression model from the python `sklearn` package with default parameters. Table 3 shows the accuracy and performance of each model. Figure 4 shows the classification accuracy over time averaged across all trials of each maneuver type for these three different models. Interestingly, the random forest model almost immediately jumps to the right prediction and stays with that prediction for the duration of the trials, a result that we are still working to understand.

	RNN	Random Forest	Logistic Regression
Accuracy	81.86%	98.90%	76.27%

Table 3: Accuracy of three models

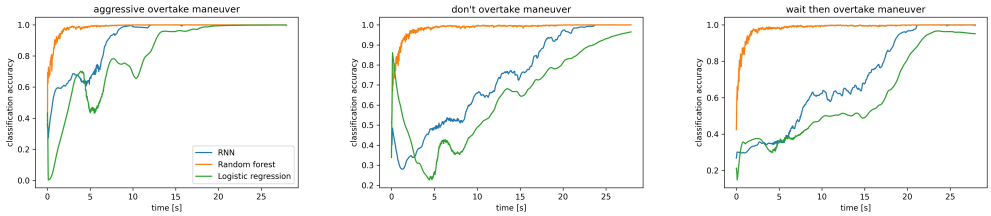

Figure 4: Average classification accuracy for each maneuver across all three models.

These experiments gave us a good understanding of how the RNN works with time series vehicle data and provided us with an appropriate structure for the prediction problem. Ultimately, this is a relatively straightforward problem that can be solved with reasonable accuracy without much model tuning, while the control input prediction problem proved more challenging.

5.3 Results - Prediction Problem

The plots shown in Figure 5 show the LSTM model's predicted control inputs against the actual values over a representative 20 second segment of driving data. The steering and braking predictions somewhat follow the actual values, while the throttle predictions show a much closer match. This result is likely due to the strong presence of varied throttle commands throughout the dataset to keep the vehicle traveling at highway speeds, while nonzero steering and braking commands are less prevalent in the collected data. We have additionally demonstrated the real time performance of this RNN by visualizing the future states expected from the first 1 second of predicted control inputs in the VR simulator – this visualization is shown in our project video along with a similar real time visualization of the classification RNN.

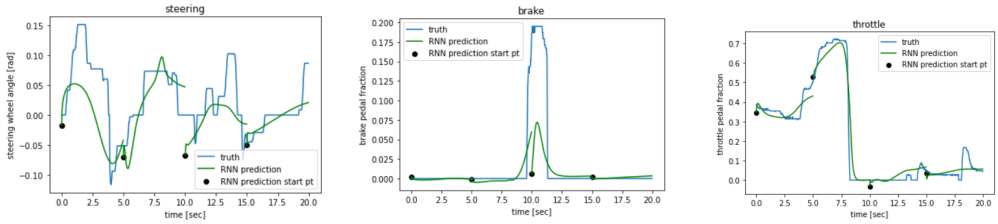

Figure 5: Control input predictions over the next 5 seconds vs ground truth.

5.4 Hyperparameters

The following are some of the hyperparameters that we have explored to improve results for the control input prediction RNN. (1) Even though the loss values are generally low for both train and test data, higher epochs showed better convergence. (2) Adding additional dense layers provided a better fit solution for the prediction model without overfitting the training data. (3) Experimenting with different numbers of time steps for future predictions illustrated how RNN performance varies as we look different distances into the future. Using mean squared error as our evaluation metric, we have tabulated the results of these RNN optimizations in Table 4. The time steps parameter shown in the table represent the number of seconds that is being predicted, also any increase in layers shown represent an additional hidden dense layer with 500 units.

No. of epochs	No. of Layers	Time Steps (in sec)	Dropout (0.8)	MSE Value (Training loss)
10	5	1	No	0.0024
10	6	1	No	0.0022
10	5	5	Yes	0.0036
10	5	5	No	0.0052
20	5	5	No	0.0034

Table 4: MSE (loss) Summary

6 Conclusion

In this work we have developed two different RNNs for understanding driver intent from vehicle data. We achieve good performance at both the classification and prediction problems considering the relatively small amount of collected data for this project. Future work consists of expanding the dataset to consider more complex driving scenario and environments, including situations involving bikers, pedestrians, and intersection management. We are further interested in exploring how these maneuver classification and control input prediction models could be used to design more intent-aware driver assistance systems.

7 Contributions

All three members of the team contributed significantly to the success of this project. All team members collaborated on the literature review, proposal, milestone, and final report. Elliot collected the data; helped with data processing, model tuning, and results analysis; created real time visualizations; and put together the final video. Yijun helped on model prototype and implementation, and final report. Vasudevan helped with model design and testing, optimization and code framework.

GitHub Repo $\Longrightarrow$ https://github.com/vasuiyen/CS230_Project.git

References

[1] Andrey Rudenko, Luigi Palmieri, Michael Herman, Kris Kitani, Dariu Gavrila, and Kai Arras. Human motion trajectory prediction: A survey. *arXiv*, May 2019.

[2] Yang Xing, Chen Lv, Huaji Wang, Hong Wang, Yunfeng Ai, Dongpu Cao, Efstathios Velenis, and Fei-Yue Wang. Driver lane change intention inference for intelligent vehicles: framework, survey, and challenges. *IEEE Transactions on Vehicular Technology*, 68(5):4377–4390, 2019.

[3] E. Schmerling, K. Leung, W. Vollprecht, and M. Pavone. Multimodal probabilistic model-based planning for human-robot interaction. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3399–3406, 2018.

[4] A. Zyner, S. Worrall, and E. Nebot. Naturalistic driver intention and path prediction using recurrent neural networks. *IEEE Transactions on Intelligent Transportation Systems*, 21(4):1584–1594, 2020.

[5] K. Driggs-Campbell and R. Bajcsy. Identifying modes of intent from driver behaviors in dynamic environments. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 739–744, 2015.