Enhancing Driver Behavior Learning: Study on Feature Grouping and Combination

Delin Mu, †Charles Zhang, †Josh Sun, †Mark Crowley

University of Waterloo
Canada

August 5th, 2023

1 Project Background and Motivation

Autonomous driving is one of the most promising industry in travel and transportation. Environmental perception modules are playing crucial roles in automatic driving system and driving behaviour prediction. The modules use various type of sensors to construct the perception of real-time road environment. The popular sensors used includes ultrasonic radar systems, LiDAR, cameras, and internal vehicle SCADA (Supervisory Control And Data Acquisition) system. The ultrasonic sensors emit short ultrasonic impulses that bounce off obstacles [1]. LiDAR(Light Detection and Ranging) uses laser light to measure distance and create environment maps [2].Camera is considered to be a low-cost application using external light to illuminate photosensitive cells, generating an electrical charge through a photoelectric effect. The row selection unit then transmits image signals to the analog signal processing and digital-to-analog conversion units for conversion into digital image signals and output [3].

As a result, sensory systems are directly liable for the quality of collected data which could affect the performance of a driving behaviour prediction system. This raises 3 interesting topics:

- Multi-sensor Model & Sensor Malfunction Impact: assuming a driver learning behavior prediction model is developed based on large multi-model, time-series dataset from various sensors. If one single sensor and/or various sensors malfunction (i.e. unable to collect data), how would it affect the performances of the prediction model?
- Inform Engineering Design: we focus on identifying the key data features crucial for the driving learning prediction model, irrespective of where or how they are collected. Later, we can trace back and determine which sensors were utilized to capture these essential data features. This can lead more topics in engineering design of a multiple function sensor system solely for the purpose of driving behavior learning.
- Data Feature Grouping and Combination Study: we refrain from grouping data features based on sensors (data sources). Instead, we apply transportation engineering practices and road engineering design concepts to categorize the data features. This enables us to assess the significance of each group of data features on the driver learning behavior prediction model's success, with a stronger emphasis on engineering considerations rather than being constrained by the limitations or capabilities of specific sensors.

Acknowledgments to Professor Mark Crowley and his Driver Behaviour Learning (DBL) research team for providing me with access to collected & processed data and valuable research results and insights. Their contributions have enabled me to explore the topics mentioned above.

1.1 Project Goals

The project is divided into 2 major goals listed below:

- **Deep Learning Fundamental**: Develop strong understanding of the architectural of deep learning time series prediction models including recurrent neural network (RNN) and Long short-term memory (LSTM), develop an autograd engine, MLP, RNN and LSTM models from scratch
- **DBL Study**: Develop steps, strategies, and results to conduct a preliminary investigation of Data Feature Study and Data Study as mentioned in **Section Project Background and Motivation**

1.2 Project Schedule

Major tasks for the project as well as the their duration and highlighted milestones are included in the Fig. 1.

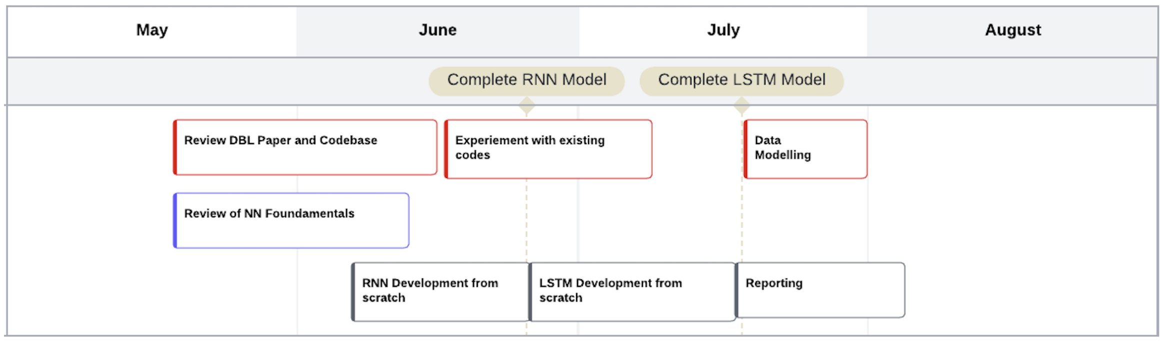


Figure 1: Project Schedule

2 Project Tasks

Project tasks as well as associated works are included in the section below:

- **Task 0 - Review of DBL Research Paper and Codebase** (Mid-May - Mid-June): the goal of the task is to develop understanding of the scope of the project "A Study of Driver Behavior Learning" completed by the University of Waterloo DBL Research Team led by Prof. Mark Crowley and Prof. Sebastian Fischmeister [4]. More specifically, the task focus on how Behavior 6 learning (i.e. discover the techniques employed by the driver as they navigate the subject vehicle along a curved road, taking into account the designated driving speeds and road conditions) was carried out in the original DBL paper [4].
- **Task 1 - Deep Learning Fundamental and Time Series Prediction Model Development** (Mid-May - Mid-June): the goal of the task is to development a strong understanding on neural network fundamentals including concepts of neurons, forward propagation, backprogatation, as well as activation functions. In order to solidifying understandings, a scalar-valued autograd engine that implements backpropagation is created as well as a multilayer perception (MLP) were created from scratch. With the engine, LSTM model is developed from scratch [5].

- **Task 2 - DBL Studying** (July - Early August): The objective of the task is to employ various of LSTM models on the original B6 dataset as well as newly created dataset for predicting the ego car's average velocity ("Vehicle Speed Average Driven") and turning angle ("Steering Wheel Angle"). We start by creating baseline models and then conduct a data feature grouping study using transportation engineering principles. Subsequently, we train models using different data feature groups and compare their results against the baseline models.

3 Task 0: Review of DBL Research Paper and Codebase

After reviewing the DBL research paper, key information related to B6 is summarized below:

- B6 Task: the primary goal of B6 was to determine driver maneuvers on a curved road. Due to the facts that it was hard to define a curve road as well as challenges in establishing threshold, the team decided to predict entirety of drives instead of defining a curved road [4].
- Model Inputs and Outputs: model inputs were 200 frames (5 seconds) of history as input to predict next 40 frames (1 second). Comparing with B2, a new feature was added as a time-shifted version of the steering wheel angle. The following targets were predicted: **Vehicle Speed Average Driven** and **Steering Wheel Angle** . The **Steering Wheel Angle** was normalized to [0, 1] for simplicity and could be used to extract the travel direction of the ego vehicle [4].
- Training Model: 2 layers of LSTM, each with 20 units. The outputs of LSTM then fed into 2 parallel dense layers, where one layer predicted speed and the other predicted angle. The cumulative loss value was obtained by summing the individual losses of the steering wheel angle and the velocity. [4]
- Important notes: when using full drives data including highway and non-highway data, it could cause numerical instability due to exploding gradients, therefore, only highway data was used for B6 prediction, and the same data was used for this project. Additionally, it was found that when using non-normalized training data, the model had difficulty to converge when predicting steering angle - the plateauing of predictions around $30°$ off the correct steering-angle mark on the validation set. On the other hand, when using raw and non-normalized dataset, the model was able to provide stable behavior with good convergence. [4]

4 Task 1 - Deep Learning Fundamental - MLP, RNN, and LSTM

4.1 Review of Scripts

A MLP was developed from scratch to grasp the the fundamental of a neural network. The implementation of MLP from scratch was with a aim to show the complexity of backpropagation even with a smaller and simpler deep learning model. The detailed description for each developed script is included below:

- **autograd.py**: the script introduces a class named "Number." This class embodies a scalar value and facilitates fundamental mathematical operations, including addition, subtraction, multiplication, division, and power operations. It keeps track of all preceding operations internally and provides support for backpropagation through its dedicated *backward* method.

- **model.py**: the script includes 3 classes, namely "Neuron," "Layer," and "MLP," which collectively form a Multi-Layer Perceptron (MLP). Each of these classes inherits from the "Module" class, enabling the storage of gradient values throughout the backpropagation procedure. This hierarchical design ensures efficient gradient handling and facilitates seamless communication between the components of the MLP architecture. When gradient update is completed, the stored value is then cleared to be 0.
- **digraph_util.py**: the script contains two methods namely "trace" and "draw_dot". The functions work together to trace and visualize a computation graph, representing the flow of the Class Number (from autograd.py) and their gradients through the graph. The resulting visualization helps in understanding the structure and information flow within the graph.

4.2 MLP Implementation

Fig. 2 shows the sample MLP implemented using the scripts developed from the scratch. The MLP contains 4 layers in total includes 1 input layer, 2 hidden layers, and 1 output layer. The input layer contains 3 inputs and a bias unit (not shown in figure), the hidden layers have a dimension of 4 and 3, respectively. The output layer contains 1 output unit. The activation function used for the MLP is RELU.

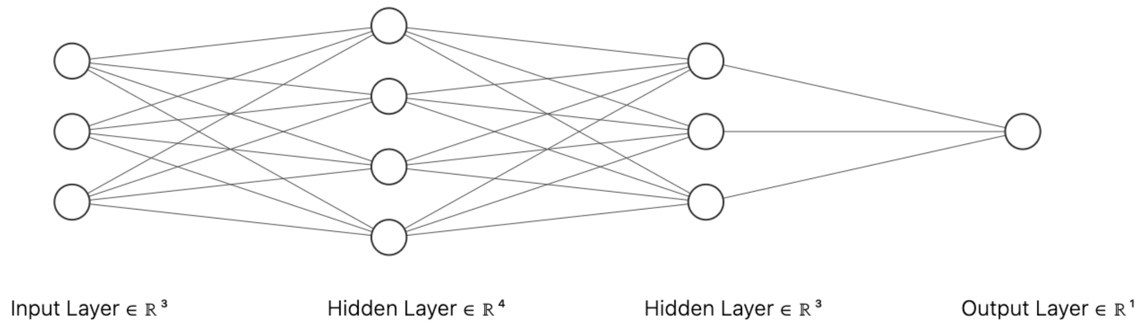

Input Layer $\in \mathbb{R}^3$
Hidden Layer $\in \mathbb{R}^4$
Hidden Layer $\in \mathbb{R}^3$
Output Layer $\in \mathbb{R}^1$

Figure 2: MLP Implementation Example

To leverage the backward function within the Class "Number," all training data and target values were generated using it. This approach enabled us to effectively utilize the functionality provided by the backward function. Using the plotting function developed in script, we can observe how backpropagation procedure is done on this simple MLP architecture in Fig. 3.

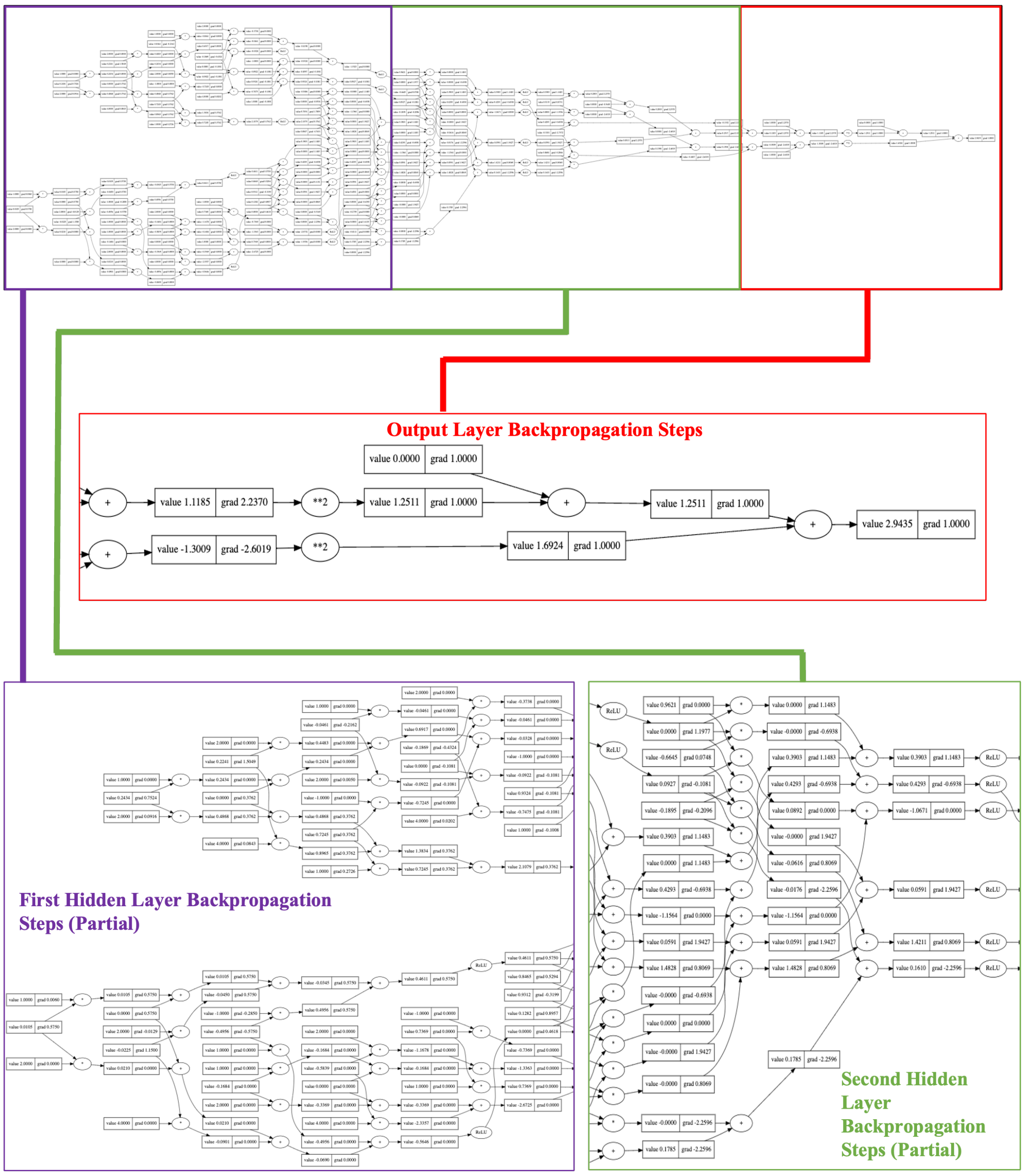

Output Layer Backpropagation Steps
First Hidden Layer Backpropagation Steps (Partial)
Second Hidden Layer Backpropagation Steps (Partial)

Figure 3: MLP Backpropagation Implementation Graph

4.3 RNN & LSTM Implementation

As mentioned in **Section 3: Review of DBL Research Paper and Codebase**, the data for the B6 learning can be classified as sequential data (i.e. inputs are 200 frames or 5 second of history, and targets are 40 frames or 1 second in future). Comparing with the implemented MLP model, Recurrent Neural Network (RNN) has better performance in modelling and predicting results from sequential data. [6] RNN operates in a cyclic manner, as shown in Fig. 4, applying the same calculation to each element of a given sequence. It allows RNN to retain information from previous computations and create a "memory" that influences future predictions. The memory-like capability allows the RNN to capture temporal patterns and dependencies within sequential data, making it well-suited for tasks involving time series, natural language processing, and other sequential information.

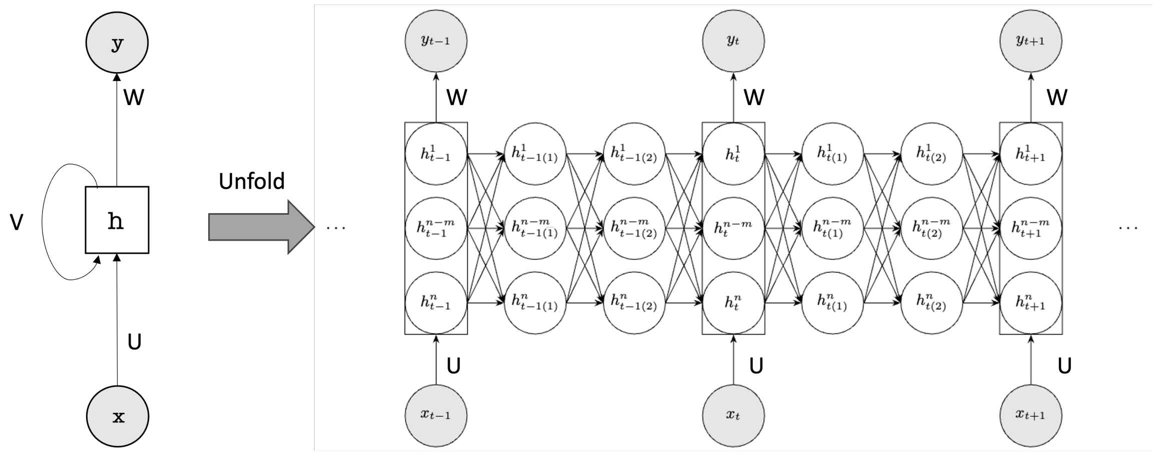


Figure 4: Vanilla RNN Implementation

Fig. 4 shows the key elements within a RNN:

- x: input sequence of data points;
- y: model output of every timestep;
- t: a timestep, we shown 3 timesteps: t-1, t, and t+1 as examples;
- U: weight matrix for input sequence of data points;
- W: weight matrix for computing output for every timestep;
- h: hidden state (memory of network) for a given time step; and
- V: weight matrix for recurrent computation (to pass along the sequence)

While RNNs do possess memory-like capabilities using the hidden state, the vanishing gradients problem presents challenges for standard RNNs when dealing with longer sequences, leading to memory retention issues. To address this limitation, gated hidden units were introduced, giving rise to the Long Short-Term Memory (LSTM) cell and the Gated Recurrent Unit (GRU), which are prominent examples [7], [8]. Both LSTM and GRU have demonstrated improved performance in preserving and reusing memory across subsequent timesteps. In our case, B6 in DBL utilizes an LSTM architecture for prediction. Our focus for the task was to develop a strong understanding of LSTM by implementing it from scratch using the functions and classes we created for MLP (related development saved in LSTM.py). Fig. 5 illustrates the architecture of LSTM, featuring a chain-like structure similar to RNNs but incorporating a more complex design.

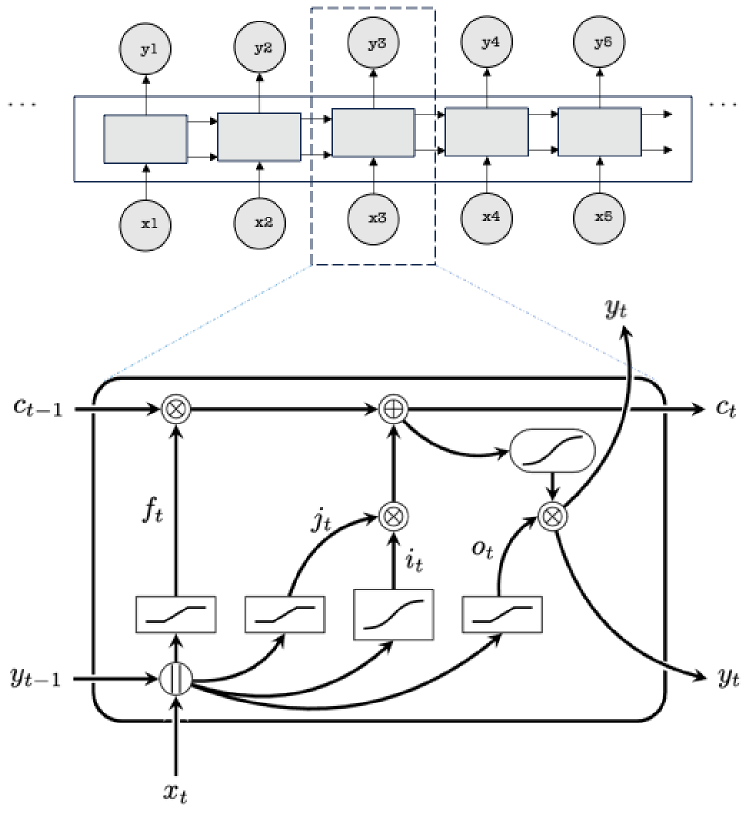


Figure 5: LSTM Implementation

There are four components in a LSTM unit: a cell (c), forget gate (f), input gate (i), and an output gate(y) [9]. The fully implementation of the LSTM network is included in Model.py. The equations for each components are summarized below [9]:

$$
\begin{aligned}
f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
\tilde{c}_t &= \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\
h_t &= o_t \odot \sigma_h(c_t)
\end{aligned}
$$

4.4 Task 1 - Conclusion

Implementation of a MLP from scratch and observation of the steps in MLP's backpropagation allows users to understand complexities even in the simplest deep learning network models. LSTM's implementation also demonstrated algorithm complexity in deep learning modelling. Based on the development experience, we can conclude the following points:

- **Computational Complexity**:the sample MLP model only contains 2 hidden layers and a total of 7 neurons, we observe that backpropagation progress starts to get complicated. As shown in Class "Number", backpropagation involves computing gradients for each parameter in the model based on the chain rule of calculus. As the MLP becomes larger with more layers and neurons, the computational complexity of backpropagation can increase. Same applies to RNN and LSTM models. [10].

- **Memory Requirement**: from Fig. 3, we observe that backpropagation requires storing intermediate activations and gradients during the forward and backward passes. As the model gets larger, the memory requirements for storing these values increase. Memory limitations may become a challenge, especially when training complex MLPs on resource-constrained systems [10] [11]. For B6 prediction, we need to leverage better computing resources for backpropagation and temporary gradient value storage.
- **Model Interpretability**: from Fig. 3 Because there are lots of parameters associated with the deep learning model, it is hard to interpret the hidden layer's works of our simple model. When MLPs or any deep learning networks become larger and more complex, the interpretability of the models' "inner" workings may decrease. The numerous layers and neurons make it challenging to understand how models arrive at their predictions. In our project, for feature reduction process for B6 prediction, explaining the learned representations or attributing decisions to specific features can be more difficult [12].

5 Driver Behaviour Learning

5.1 Task Summary

We intend to investigate how different features may affect the prediction results from the original B6 experiment. Three major tasks completed to achieve this goal:

- Baseline Results Development
- Feature Study and Category Grouping
- Model Training
- Result Discussion

5.2 Baseline Results Development

5.2.1 Data Selection and Pre-processing

The original raw data were directly from DBL B6 training dataset: highway portions of dataset with a total of 123 features. B6 used 50 drivers' highway-only data, this project utilized only the highway-only data from 3 drivers (P8, P9, P10). With the data, 4 datasets were prepared to produce baseline results. The prediction targets were "Vehicle Speed Average Driven" and "Steering Wheel Angle". 5 seconds of history (200 frames) in the dataset were used as input, the model aimed to predict next 1 second in history (40 frames). A summary of each dataset for baseline result development was include in the Table 1.

Table 1: Original Dataset Summary

Dataset	No. of Drivers Included	Min-Max Normalization	No. of Features
Dataset 1	1	N/a	123
Dataset 2	1	Yes	123
Dataset 3	3	N/a	123
Dataset 4	3	Yes	123

Knowing that our goal of the project is to investigate how features affect prediction results, the rational of using only 1 drivers' and 3 drivers' data are as follows:

- Additional data may lead to a decrease in overall validation loss, but since drivers exhibit distinct driving behaviors, we opt to use data from a single driver for developing the initial two baseline results. This decision ignores the impact of driving behavior variations among drivers. For future joint studies, it is recommended to utilize data from drivers with similar driving behaviors as a cohesive input for model training. More information related to improvements are included in the **Section Next Steps**.
- We utilized data from three drivers to explore the potential benefits of having more data. However, as described in the following section, we discovered that using data from just one driver yielded better results. This observation could be attributed to the fact that different drivers exhibit distinct driving behaviors, and combining data from all drivers may have a detrimental impact on our model.

5.2.2 Model

The baseline architectural is the same as the one used in B6 architecture, as mention in **Section Review of DBL Research Paper and Codebase** [4]. The architectural uses a 2-layer LSTM following with 2 fully connected layers with 40 perceptrons, each predict a target (i.e. Speed or Angle). Each LSTM layer contains 20 LSTM units. An example of a LSTM unit is shown in Fig. 5. As mentioned in Driving Behavior Learning Report, the model built for B6 is extended with experience gained from B1 and B2 studies [4].

Besides using B6 original architecture, we increased the complexity of the model to do more data experiment: 2-layer LSTM (each layer with **64** LSTM units) following with 2 fully connected layers of 40 perceptrons.

5.3 Feature Grouping and Combination of Groupings

5.3.1 Early Attempt - Feature Grouping by Sensors

The data features were collected from internal and external sources, including Delphi sensors, Lidar, GPS antenna, and VLT camera. Initially, the project aimed to examine the model's performance when one or more sensors malfunction. The baseline model, which utilized data from all sensors, served as the benchmark for evaluating each scenario. However, identifying the sources of data features proved challenging, making it difficult to achieve the initial objective.

5.3.2 Quick Note on PCA

PCA is a dimensionality reduction technique used to transform variables into a smaller set while retaining crucial information. In the original DBL project, PCA was conducted, but the reduced dimensionality dataset from PCA was not used for training. Instead, the 123 features for the baseline model were selected based on PCA and other analyses, enabling direct interpretation of features from the original dataset and facilitating feature grouping activities specifically for our project.

5.3.3 Feature Category Development

After the early attempt, we disregard the data sources and concentrated on categorizing them into similar functional and distinct categories. We categorized the 123 features in B6 into 7 categories: Ego Vehicle Features, Preceding Vehicle Features, Road Objects Features, Road Design Features, Traffic Control System Features, Ego Vehicle Visual Monitoring System Features, Driving Condition Features, and miscellaneous Features. Description of each categories are show below.

- **Ego Vehicle Features**: Characteristics and measurements related to the ego vehicle, such as speed, acceleration, and steering angle.
- **Preceding Vehicle Features**: Information regarding the preceding vehicle's signal observation such as observations on turn signals and break lights. It was separated from the "Road Objects Features", because we wanted to investigate if the behaviour of the preceding vehicle may have a bigger impact on the behavior of ego vehicle comparing with other objects on road.
- **Road Objects Features**: Data pertaining to objects on the road, such as obstacles and pedestrians, other even cars other than preceding vehicle. In the dataset, it was hard to interpret what objects were detected. The feature group included a board group of any objects on road around ego vehicle except for the preceding vehicle. We separated preceding vehicle features from the road objects features for the reason mentioned in "Preceding Vehicle Features" above.
- **Road Design Features**: This feature category pertains to road design characteristics, specifically focusing on lane markings information. Different municipalities have their own design standards for lane width dimensions. For example, the Region of Waterloo recommends a preferred road line width of 3.65 meters, with a lower bound of 3.35 meters and an upper limit of 3.65 meters [13]. Narrower road lines imply shorter distances for drivers to cross a line, while wider lines mean longer distances. The design of the road can influence driver behavior, and we developed this feature category to investigate its impact on driver learning behaviour.
- **Traffic Control System Features**: This feature category includes data about traffic signals, signs, and other control systems like cameras. Traffic signal design is a separate engineering branch related to road design. However, since our dataset is limited to highway-only data in this study, excluding the Traffic Control System Features may not significantly impact our model's prediction results.
- **Ego Vehicle Visual Monitoring System Features**: Features includes the status of the visual monitor system: 'Blurred Image', 'Some Camera Blockage', and 'Full Camera Blockage'. Those may be a indicator of the quality of collected data.
- **Driving Condition Features**: Data reflecting driving conditions, two were captured: "Fog Indicated" and "Ice On Wind Shield". While there were no changes in driving conditions, it was

important to have them included in one feature category, as the weather can also affect one's driving behavior. This can be further studied in feature research endeavor.

- **Miscellaneous Features**: Other relevant features not falling into the previous categories.

Table 2 below summarizes each feature category.

Table 2: Feature Categories Summary

Feature Categories	No. of Features	Feature Example
Ego Vehicle Features	9	actual vehicle acceleration, heading angle, accelerator effective position
Preceding Vehicle Features	10	turn signal on preceding vehicles, brake lights on preceding vehicles
Road Objects Features	40	objective movement directions, object polar distances, object time to collision inverse, object velocities (relative, lateral)
Road Design Features	4	crossing lane marking, location of lane markings, predict lane marking
Traffic Control System Features	40	Horizontal Distance to Traffic Signs, Lateral Distance to Traffic Signs
Ego Vehicle Visual Monitoring System Features	3	Blurred Image, Some Camera Blockage, and Full Camera Blockage
Driving Condition Features	2	Fog Indicated, Ice On Wind Shield
Miscellaneous Features	15	Lighting types, ImageGlare

5.3.4 Grouping Combinations - Date Feature Selection for Training

Based on the eight (8) developed data feature categories, we proceeded to combine different combinations of these categories to create new datasets with reduced dimensions. We generated a total of seven combinations for the purpose of our study, as outlined in Table 3 below.

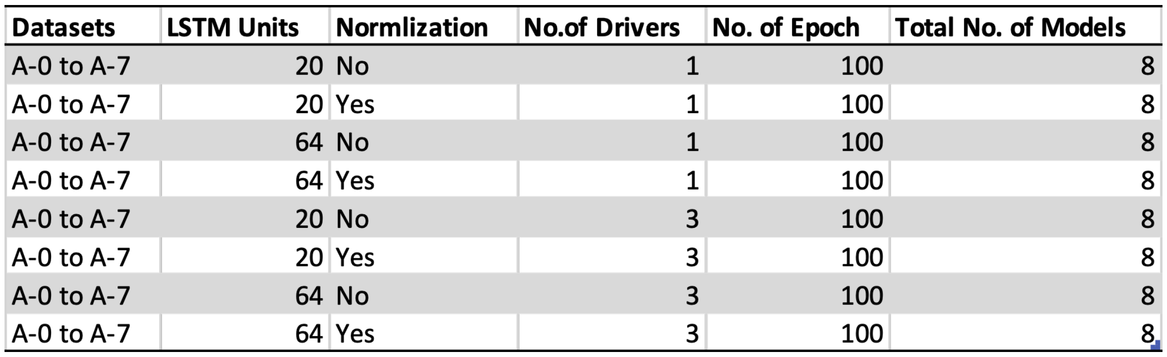
Table 3: Dataset Group Combinations Summary

Dataset Group Code	Included Feature Categories	Excluded Feature Categories
A-0 (Base Dataset)	All	N/a
A-1	Ego Vehicle + Preceding Vehicle + Road Objects + Ego Vehicle Visual Monitoring System + Driving Condition + Miscellaneous Features	Road Design
A-2	Ego Vehicle + Ego Vehicle Visual Monitoring System + Driving Condition + Miscellaneous Features	Road Design + Preceding Vehicle + Road Objects
A-3	Ego Vehicle + Preceding Vehicle + Ego Vehicle Visual Monitoring System + Driving Condition + Miscellaneous Features	Road Design + Road Objects
A-4	Ego Vehicle + Road Objects + Ego Vehicle Visual Monitoring System + Driving Condition + Miscellaneous Features	Road Design + Preceding Vehicle
A-5	Preceding Vehicle + Road Objects + Ego Vehicle Visual Monitoring System + Driving Condition + Miscellaneous Features	Road Design + Ego Vehicle
A-6	Preceding Vehicle + Ego Vehicle Visual Monitoring System + Driving Condition + Miscellaneous Features	Road Design + Ego Vehicle + Road Objects
A-7	Road Objects + Ego Vehicle Visual Monitoring System + Driving Condition + Miscellaneous Features	Road Design + Ego Vehicle + Preceding Vehicle

5.4 Model Training

A total of 72 of model training was conducted for the project. All training was conducted on a MSI GeForce RTX 2070 (8 GB) GPU with a total training time of approximately 27 hours. Table 4 below summarized all the training conducted for the project.

Table 4: Dataset Group Combinations Summary



Datasets	LSTM Units	Normlization	No.of Drivers	No. of Epoch	Total No. of Models
A-0 to A-7	20	No	1	100	8
A-0 to A-7	20	Yes	1	100	8
A-0 to A-7	64	No	1	100	8
A-0 to A-7	64	Yes	1	100	8
A-0 to A-7	20	No	3	100	8
A-0 to A-7	20	Yes	3	100	8
A-0 to A-7	64	No	3	100	8
A-0 to A-7	64	Yes	3	100	8

5.5 Model Results

5.5.1 Base Model

Non-normalized results from the base model (A-0) were chosen for reporting due to their direct interpretability. It's important to note that in the original experiment (Driving Learning Behavior Report), steering angle prediction with 24 participants' data encountered convergence difficulties [4]. In our project, involving 1 or 3 participants' data, all models struggled to generate meaningful predictions for steering angle (no convergence, high loss of approximately 36° deviation). While investigating the root cause was intriguing, our project focused on discussing the impact of different category groups on velocity prediction results.

The selected results for the base models are summarized in Fig. 6. The results from the following models are included in the figure:

- 1 Driver - 20 LSTM & 1 Driver - 64 LSTM: the models includes 1 participant/driver's data. Both models uses 2 layer LSTM, one using 20 LSTM units in each layer, and other other using 64 LSTM units in each layer.
- 3 Drivers - 20 LSTM & 3 Driver - 64 LSTM: the models includes 3 participant/driver's data. Both models uses 2 layer LSTM, one using 20 LSTM units in each layer, and other other using 64 LSTM units in each layer.

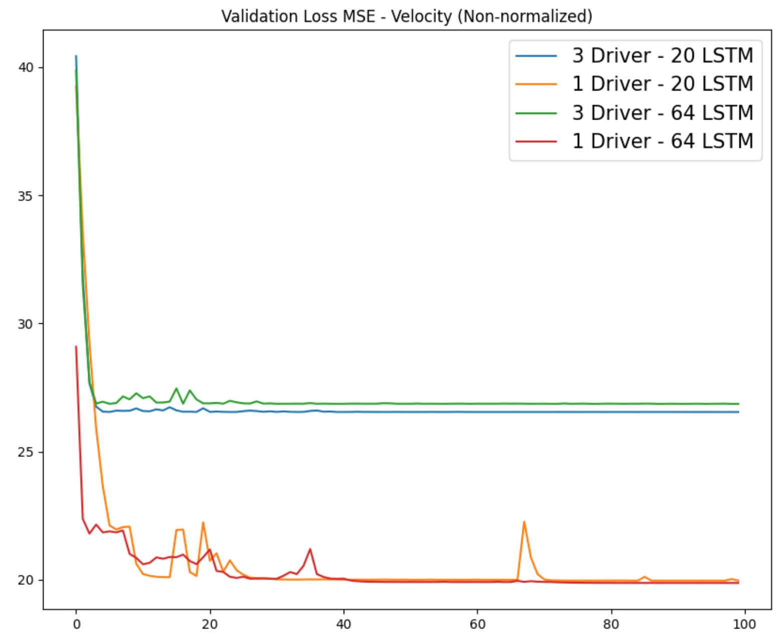


Figure 6: Velocity Validation Loss (MSE) Results using non-normalized highway-only data

Fig. 6 illustrates that when considering data from just 1 participant, models (with either 20 or 64 LSTM units per layer) outperformed those using data from 3 participants using non-normalized data. The validation set's velocity loss for 1 participant was around 19 km/h, while for 3 participants, it was approximately 32 km/h. This provided support to our earlier hypothesis that more participant data doesn't necessarily enhance model prediction results. It's noteworthy that DBL reported a mean square error of about 4 km/h using data from 24 participants, significantly lower than our loss [4]. While more data can reduce loss, for optimal results in velocity prediction for driving behavior studies, it's advisable to group participants with similar driving behaviors for model training. In this particular project, we selected to use only 1 driver's data to forward with next stage of our experiment due to the observation from the base model. By doing this, we can eliminate the effect of different driver's behavior on our model prediction results. Additionally, we observed that, for the base model, increasing model complexity didn't significantly enhance performance in both 1 participant and 3 participants models when the data was not normalized. However, it's premature to conclude that more complex models are unhelpful, as further participant data is needed to validate this finding.

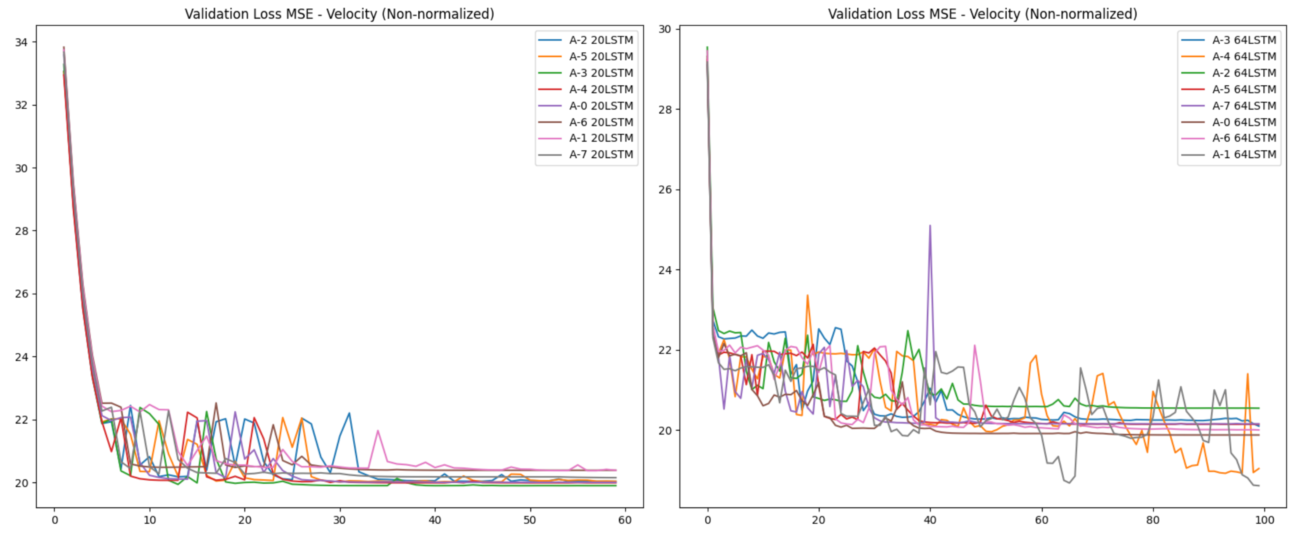
5.5.2 Group Combinations Summary

Fig.7 shows the results for non-normalized data in 8 different group combinations using two different model architectures (i.e. 2-layer LSTM with either 20 or 64 LSTM units per layer). From Fig.7(a), we observed that when the model was not complex, the prediction results across 8 different group combinations were similar with validation loss ranging from 20 to 21 km/h after roughly 35 epochs. Even though it appeared that A-1 (i.e excluding "Road Design" Feature) performed the worst, and A-3 (excluding both "Road Design" and "Road Objects" Features) performed the best among all combinations, the lost difference between them were only around 0.4 km/h.

When employing a more complex model (64 LSTM units per layer), noticeable enhancements were observed for specific combinations: A-1 and A-4. On average, they exhibited a 1.2 km/h lower loss than the base model. A-1 excludes the "Road Design" feature category, while A-4 excludes "Road

Design" and "Preceding Vehicle" feature categories. Both models that omit road design features exhibit improved overall performance. An initial interpretation was that "Road Design features" encompass road design characteristics, primarily focusing on lane markings information. For the particular participant dataset on the highway, all data associated with lane markings or crossings were zero (totaling 34546 data points). This could stem from the data source or the infrequency of lane changes when car velocity remains relatively stable during highway driving.

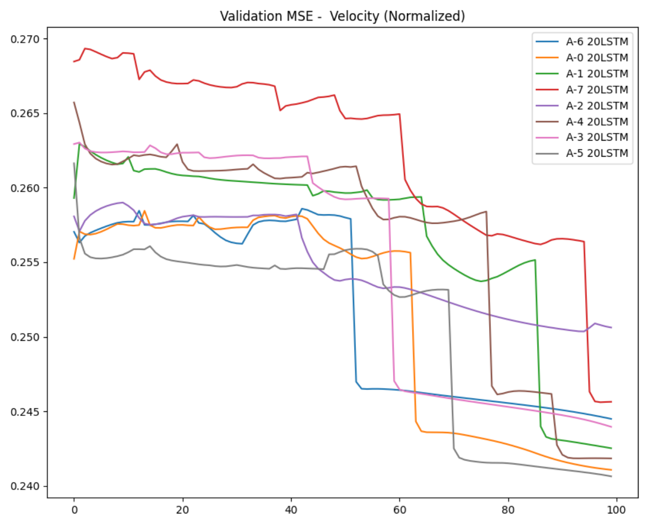
On the other hand, A-2 exhibited the poorest overall performance, averaging 1.0 km/h higher loss compared to the base model and approximately 1.95 km/h higher loss than the best-performing model A-1 and A-4. Similar to A-4, A-2 excluded an additional category of data: the "Road Objects" feature category. The "Road Object" category includes 40 features related to objects like obstacles, pedestrians, and other cars. This suggests that incorporating information about objects around the vehicle, in addition to preceding car information, might be crucial for achieving improved model performance.



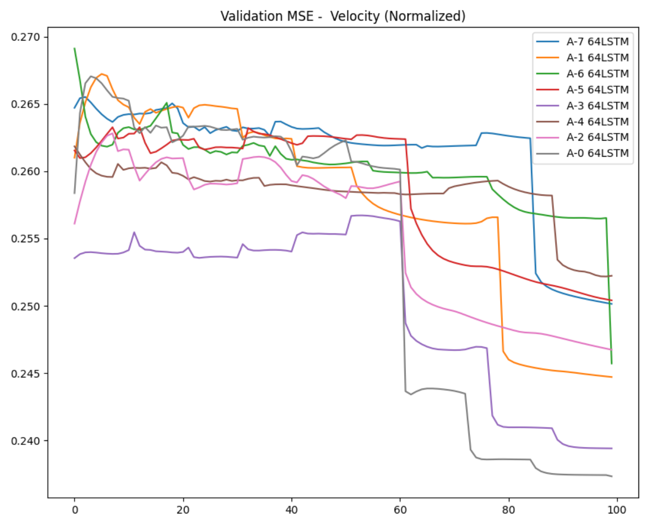
(a) 2-layer LSTM with 20 LSTM Units (b) 2-layer LSTM with 64 LSTM Units

Figure 7: 1 Driver Group Combination Results with Non-normalized Data

Fig. 8 presents normalized data results for 8 group combinations, using two model architectures (2-layer LSTM with 20 or 64 LSTM units per layer). Similar to the 2-layer LSTM with 20 LSTM units in Fig. 8(a), simpler models show validation loss across group combinations ranging from 0.240 to 0.255. Fig. 8(b) displays results of normalized data using a 2-layer LSTM with 64 LSTM units. These outcomes diverged from those of non-normalized data. Normalization aligned data to a common scale (-1 to 1), enhancing consistency and convergence speed during training but possibly altering data patterns. Normalized values showed the base model excelling, with a low loss of 0.237. The worst-performing normalized model (A-4) previously excelled with non-normalized data. A-4 now had a roughly 0.15 higher loss than the best model (A-0). Conclusions on prediction enhancement across models proved challenging due to small-scale loss differences. Further investigation is required to comprehend how grouping combinations impact model results on normalized data.



(a) 2-layer LSTM with 20 LSTM Units



(b) 2-layer LSTM with 64 LSTM Units

Figure 8: 1 Driver Group Combination Results with Normalized Data

6 Next Steps

The project initiated the driving behavior learning investigation through two primary steps: establishing a solid understanding of deep learning and diverse architectures, and then undertaking a study involving grouping and combining data features from the driving behavior learning dataset. The aim was to identify improved yet more concise feature combinations for enhancing prediction model performance. This project serves as an initial exploration into studying DBL features. To enhance the project further, the following steps can be considered to further improvement the project:

- **Data**: The project highlights that increased data doesn't necessarily enhance model predictions if participants exhibit diverse driving behaviors. To advance, it's suggested to include more participants showing similar driving behaviors for improved model predictions.
- **Data Feature Grouping**: Data feature categorization in the project was done using the B6 dataset. Categorizing data directly from the original dataset could capture more essential information, enhancing the feature grouping strategy.
- **Model**: The project indicates that complex models yield improved outcomes. Exploring models with additional LSTM units can help uncover variations in results.

7 Reference

References

[1] J. Park, Y. Je, H. Lee, and W. Moon, "Design of an ultrasonic sensor for measuring distance and detecting obstacles," *Ultrasonics*, vol. 50, no. 3, pp. 340–346, 2010.

[2] D. Lv, X. Ying, Y. Cui, J. Song, K. Qian, and M. Li, “Research on the technology of lidar data processing,” in *2017 First International Conference on Electronics Instrumentation Information Systems (EIIS)*, pp. 1–5, 2017.

[3] A. Cartenì, “The acceptability value of autonomous vehicles: A quantitative analysis of the willingness to pay for shared autonomous vehicles (savs) mobility services,” *Transportation Research Interdisciplinary Perspectives*, vol. 8, p. 100224, 2020.

[4] M. Crowley, S. Fischmeister, and W. D. R. Team, “A study of driver behavior learning,” 2022.

[5] A. Karpathy, “Makemore: An autoregressive character-level language model for making more things,” 2022.

[6] D. E. Rumelhart and J. L. McClelland, *Learning Internal Representations by Error Propagation*, pp. 318–362. 1987.

[7] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder–decoder for statistical machine translation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1724–1734, Association for Computational Linguistics, Oct. 2014.

[8] A. Sherstinsky, “Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network,” *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.

[9] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to Forget: Continual Prediction with LSTM,” *Neural Computation*, vol. 12, pp. 2451–2471, 10 2000.

[10] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, M. Hasan, B. C. Van Essen, A. A. S. Awwal, and V. K. Asari, “A state-of-the-art survey on deep learning theory and architectures,” *Electronics*, vol. 8, no. 3, 2019.

[11] Y. Bengio. 2009.

[12] M. Repetto, “Multicriteria interpretability driven deep learning,” *Annals of Operations Research*, 2022.

[13] B. M. P. . U. D. I. . P. A. with AECOM, “Context sensitive regional transportation corridor design guidelines,” p. 134, 2013.