University of Stuttgart
Institute of Industrial Automation
and Software Engineering

Deep Learning-based Monitoring of Urban Traffic using MOBATSim

Presented by: Nikita Jhawar

Supervised by: Sheng Ding

Examined by: Jun.-Prof. Dr.-Ing. Andrey Morozov

AIM OF THESIS

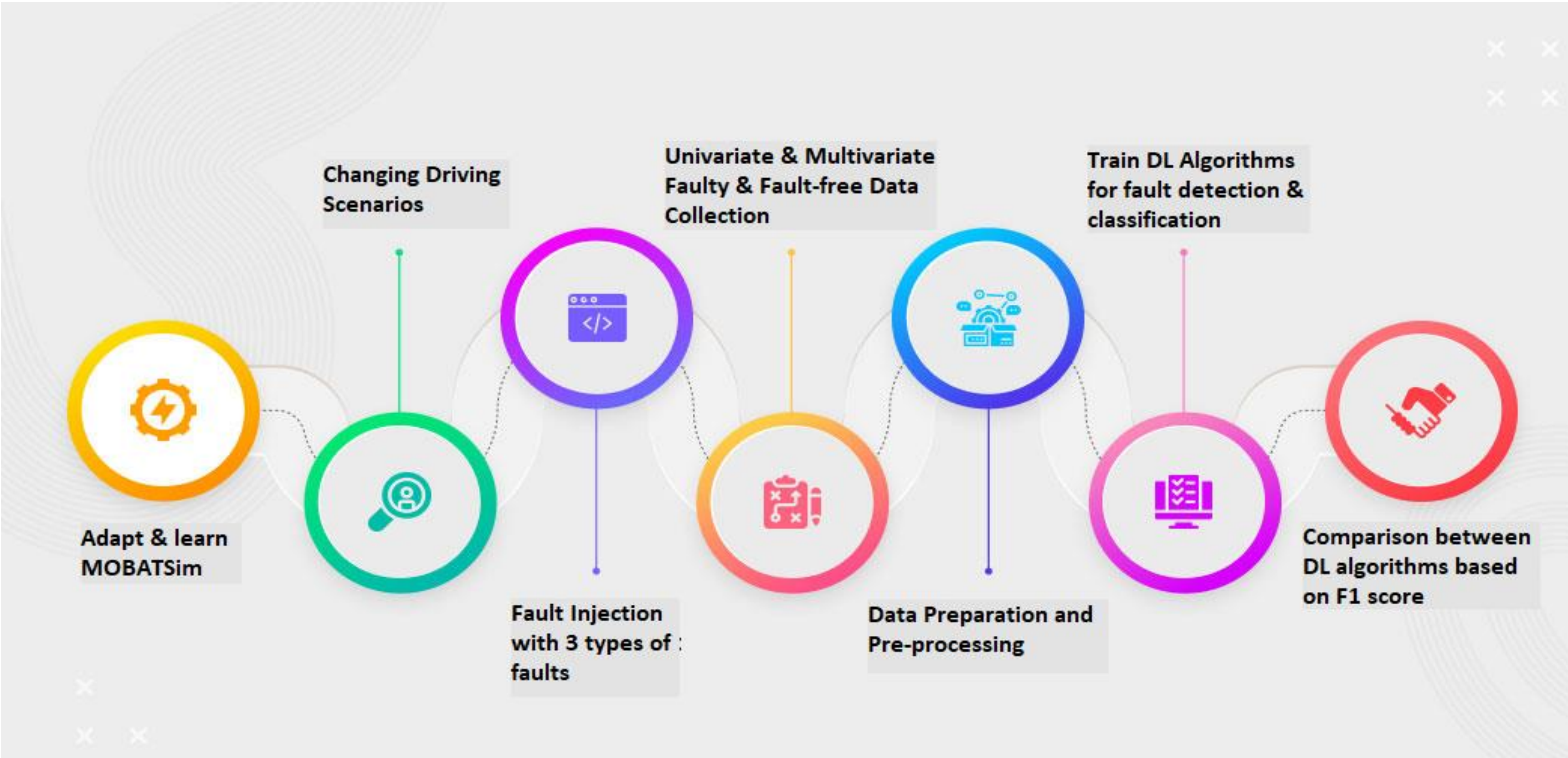


TABLE OF CONTENTS

MOBATSim

MOTIVATION

1. The automotive sector needs new tools and techniques to assess and analyze the safety of the vehicles as well as the traffic.
2. Automated Traffic system helps in optimizing transport system and make it safer.
3. MOBATSim allows simulation-based fault injection in order to evaluate the safety of autonomous driving systems [1].
4. Deep Learning-based anomaly detection in Cyber-Physical Systems have proved to be very effective [2].

Related Work

The use of MOBATSim for fault error failure chain analysis has been presented using an illustrative case study showing the effects of fault types of two different sensors on the overall traffic safety as an example of the capabilities of MOBATSim [3] by injecting fault in one vehicle in the traffic model.

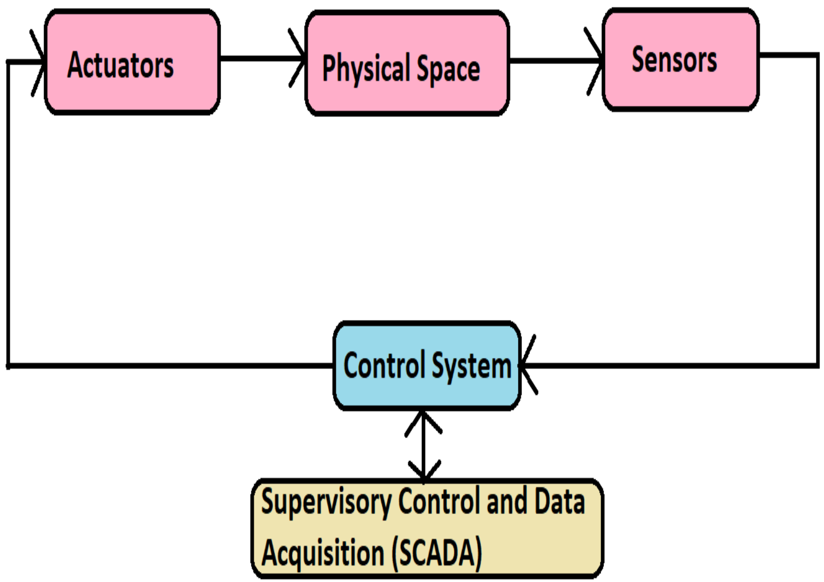
Thesis work

In this thesis, we introduce injecting faults in more than one vehicle and also consider different driving scenarios for analyzing the behavior of faulty vehicles.

INTRODUCTION

Cyber-Physical System

- Cyber physical systems (CPS) is defined as systems of collaborating computational entities that are in thorough connection with the neighboring physical world as well as its ongoing processes, thus enabling data processing services [4].
- MOBATSim is a type of CPS, allowing the user to develop automated driving algorithms



MOBATSim: Model-based Autonomous Traffic Simulation Framework

.The MOBATSim traffic Model has 10 vehicles sharing their speed, rotation, and translation data after simulation

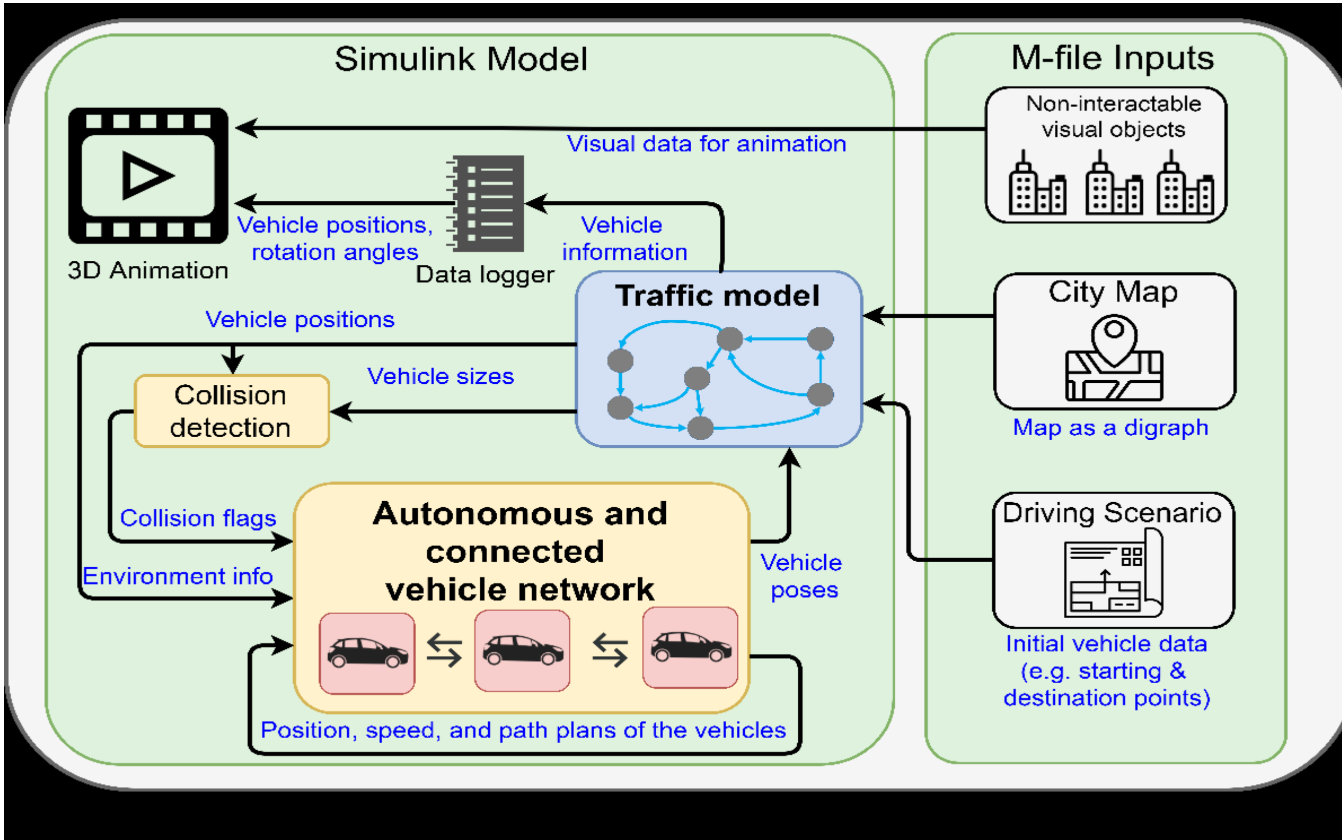
Out of these 10 vehicles, FI block is used to inject fault in 2 vehicles at speed and distance sensor.

Three types of sensor faults are injected namely Noise, StuckAt and Offset

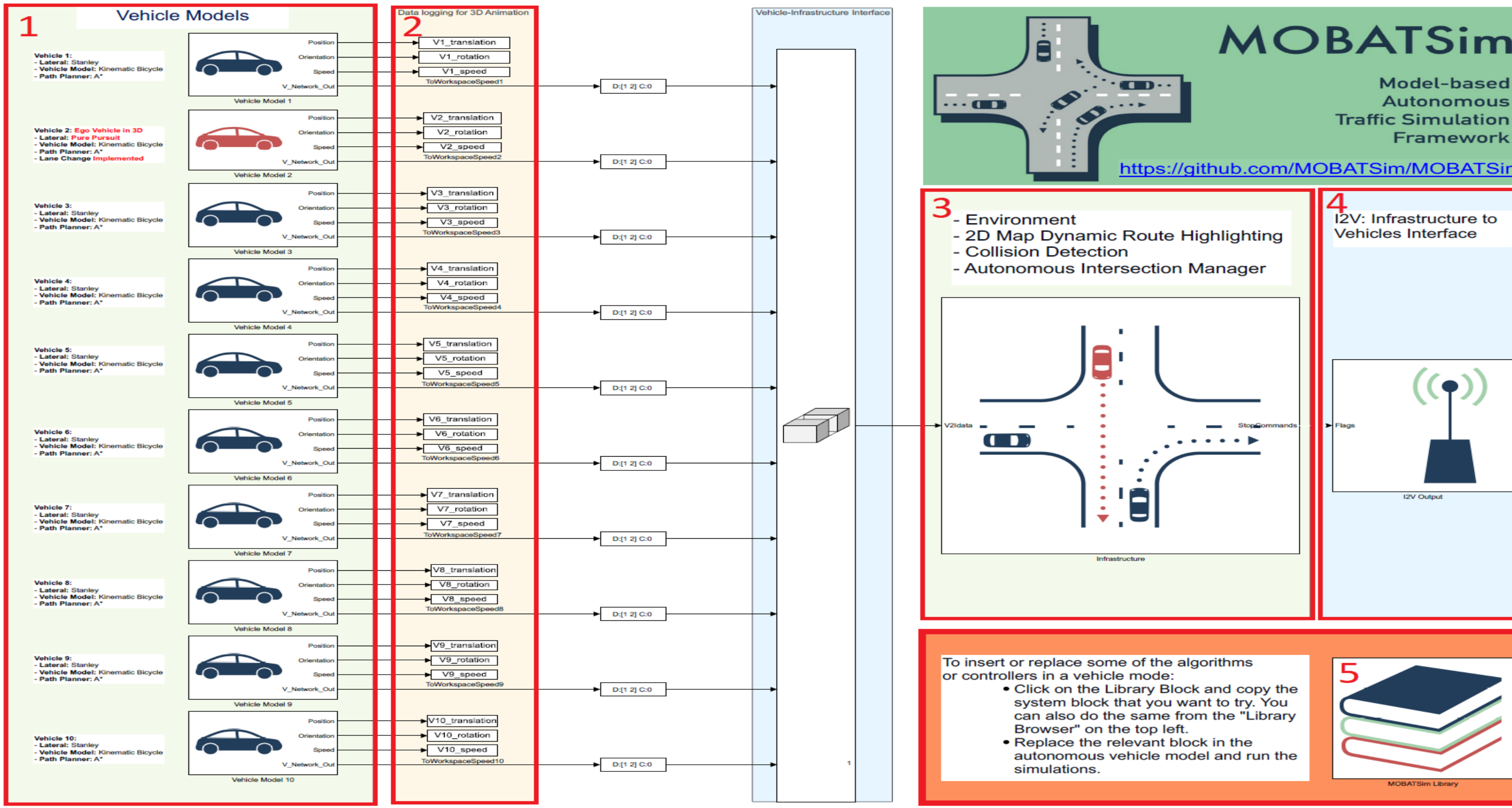
Simulation is run for 3 different driving scenarios with different fault values.

Two types of data i.e. Univariate and Multivariate data is collected

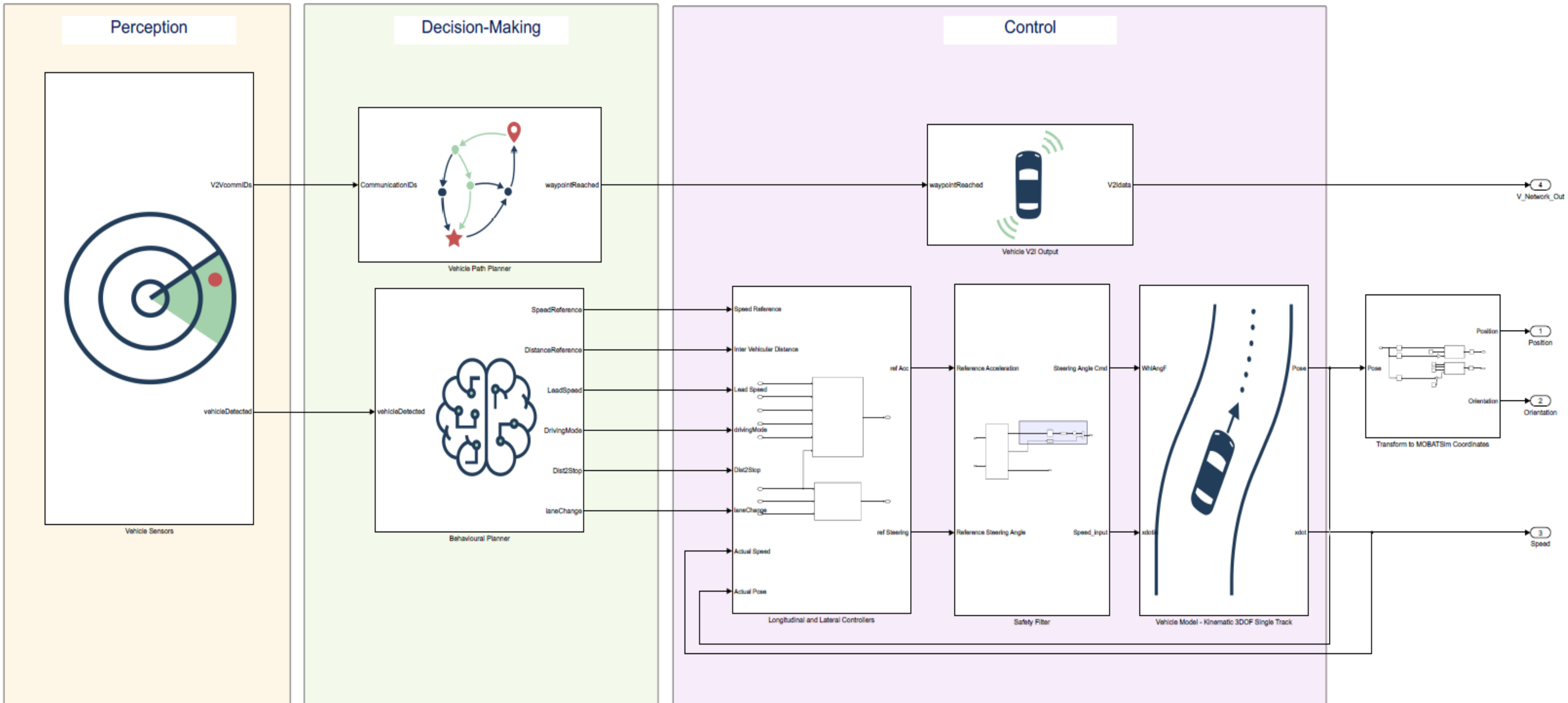
Workflow Diagram



Main Simulink Model

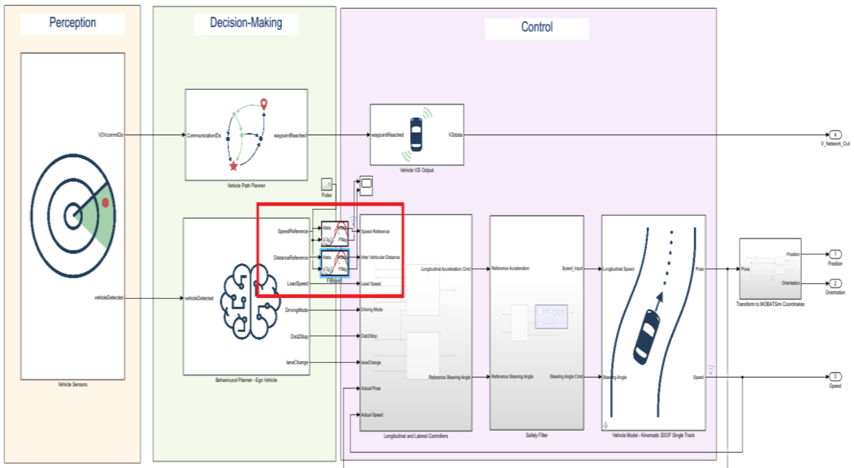


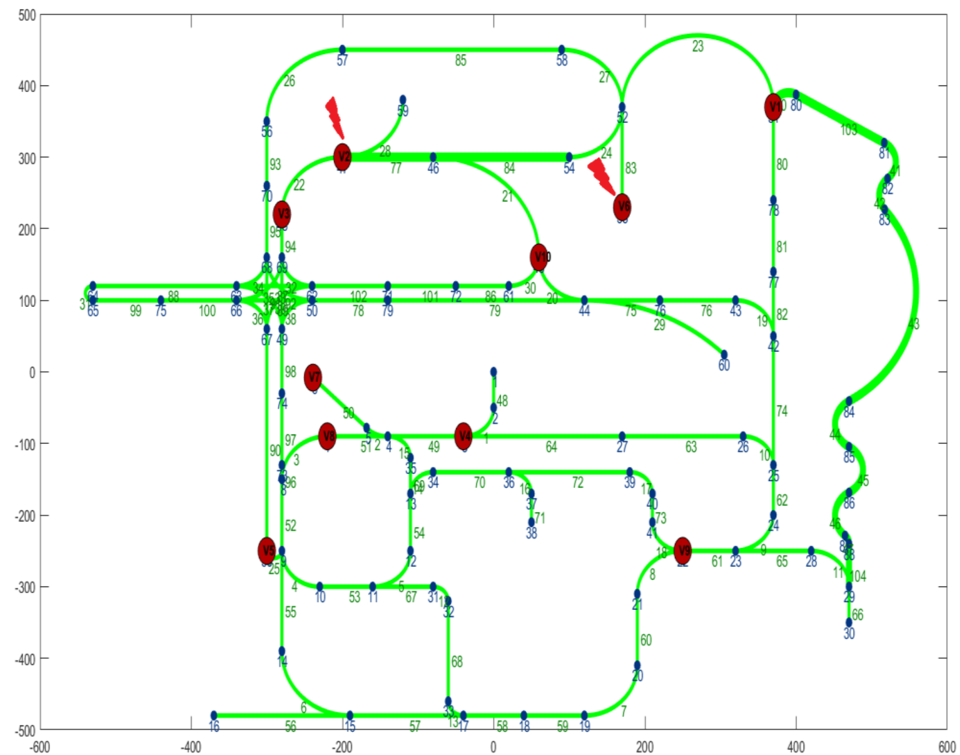
Structure of Vehicle Model



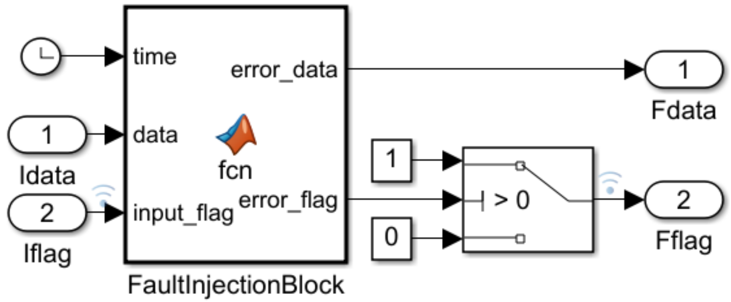
FAULT INJECTION

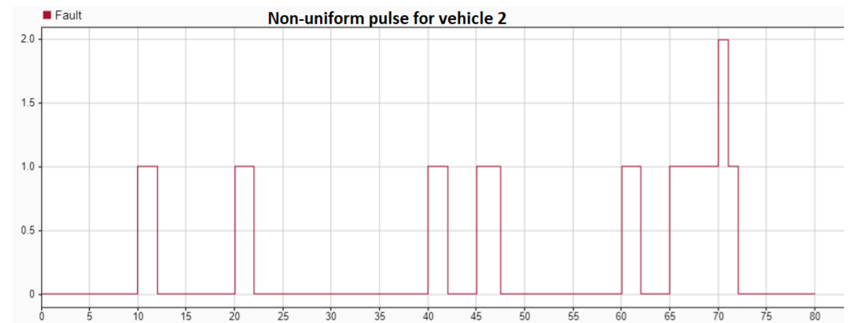
Vehicle Model After Fault Injection

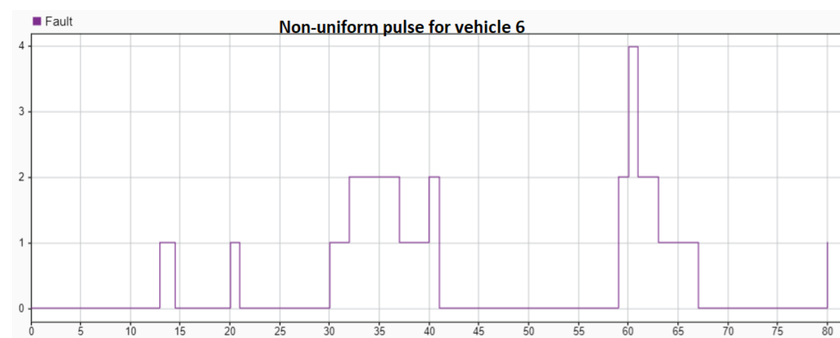




FI Block and Non-uniform Pulse Generator

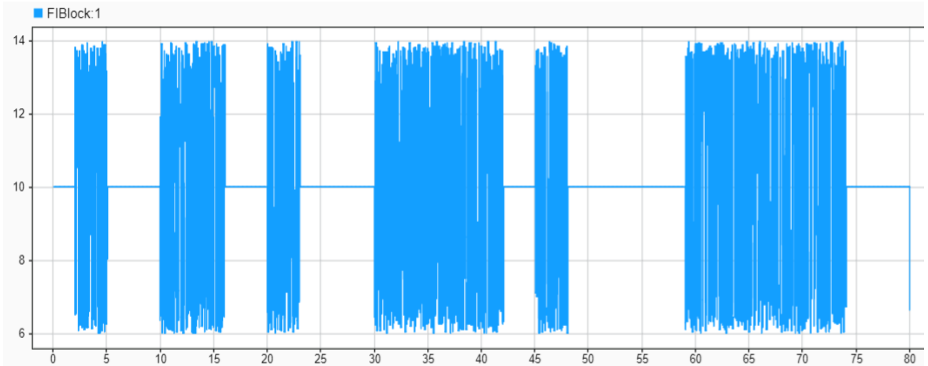




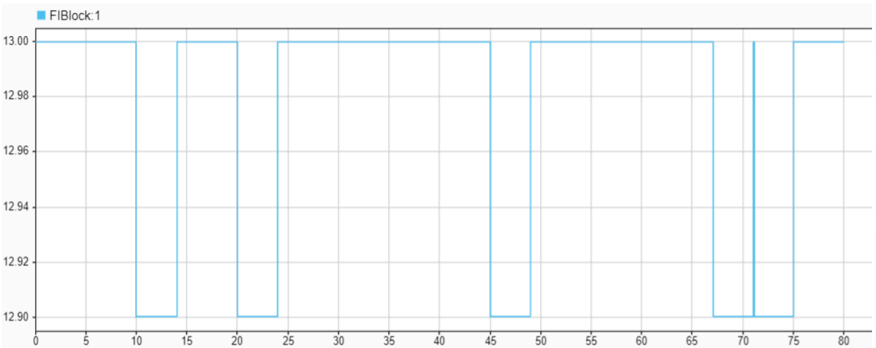


Fault Category	Fault Value	Fault Delay	Fault Duration
Noise	20%, 30%, 40%, 50%	1, 2, 3, 4, 5	1, 2, 3, 4, 5
Stuck-at	Stuck to last value	1, 3, 5, 7, 9, 11, 13, 15	2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12
Bias/Offset	-1, -2, -3	1, 2, 3, 4, 5	2, 3, 4, 5, 6, 7

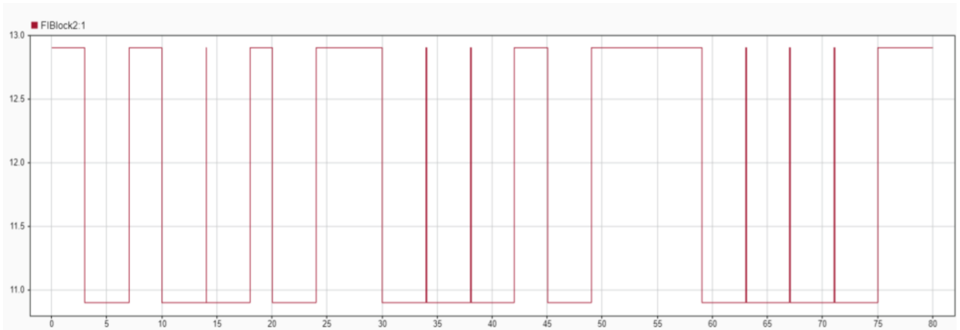
Three Categories of Fault



Noise Fault : Sampling Rate 0.02sec



Stuck-at Fault : Sampling Rate 0.02sec



Offset Fault : Sampling Rate 0.02sec

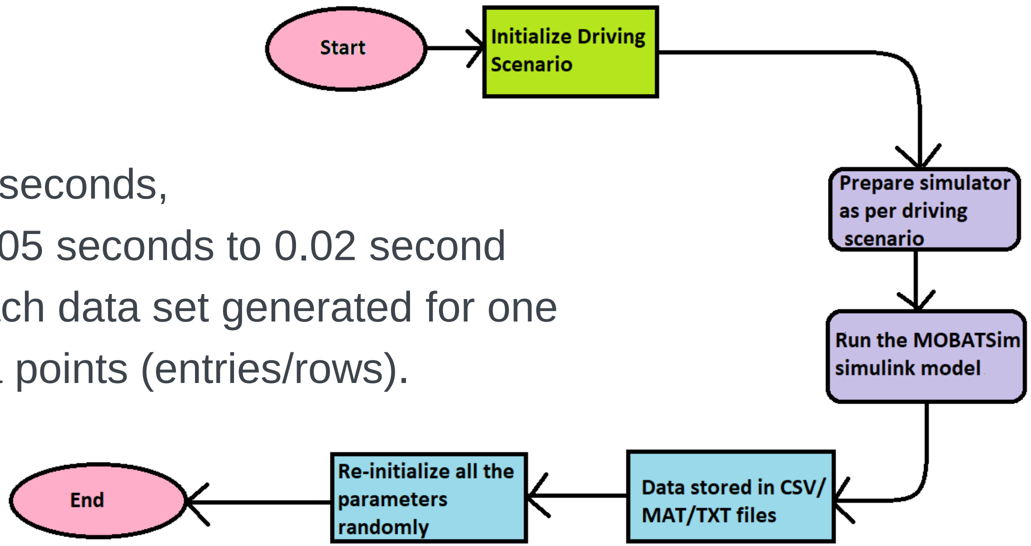
DATA COLLECTION

Fault-Free Data Collection

The process is repeated for three driving scenarios :

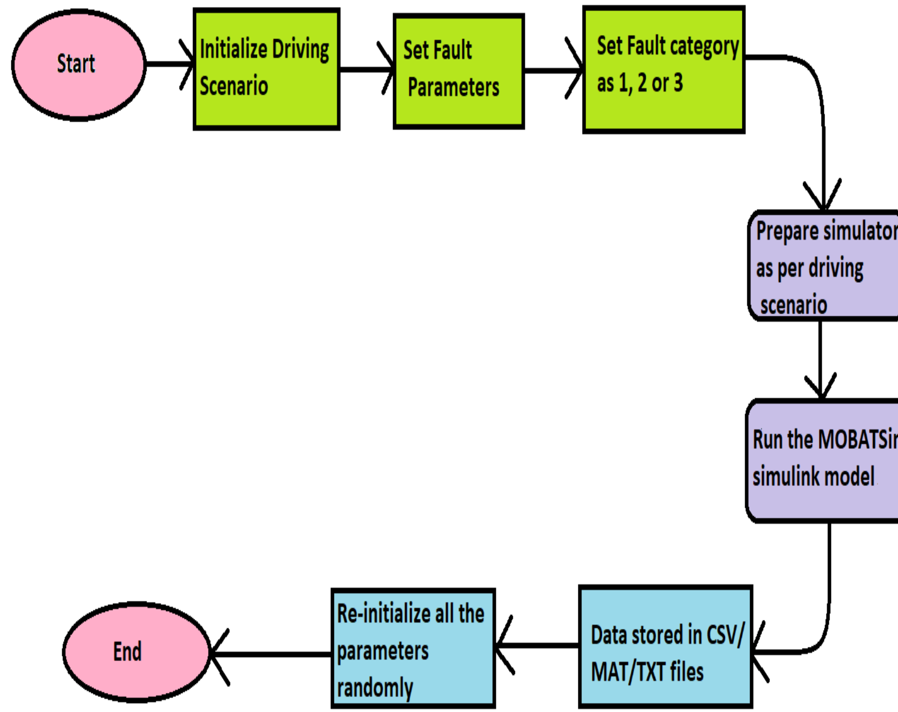
- Platoon Control
- Road Merge Collision
- Urban City Traffic

The duration of the simulation is 80 seconds,
and the data is re-sampled from 0.005 seconds to 0.02 second
for data compression. Therefore, each data set generated for one
simulation of 80 secs has 4000 data points (entries/rows).

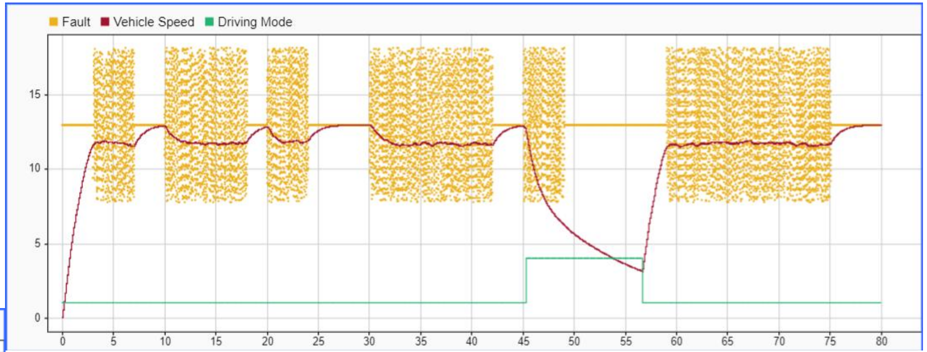


Faulty Data Collection

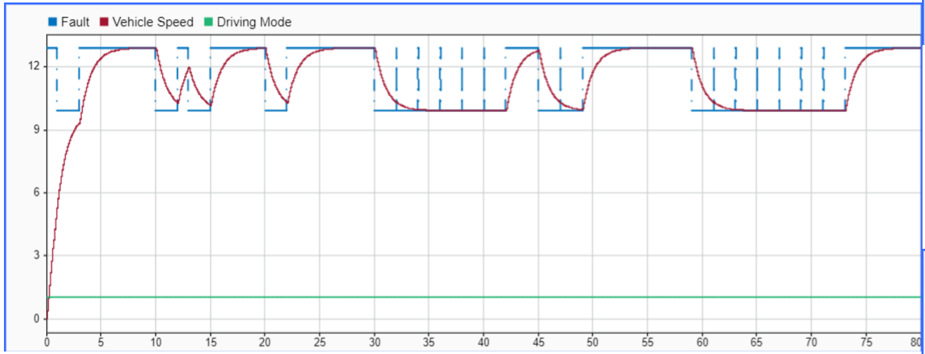
- For each value of fault duration from the Table, data is collected for every scenario considering each fault value consecutively.
- At respective fault values, data is collected for random fault delays and fault duration chosen from the table for Noise, Stuck-at and Offset/Bias for both vehicle 2 and vehicle 6 for all three scenarios.
- Data is generated for each uni-variate and multivariate data. Around more than 200 faulty data files are collected with each data set generated for one simulation of 80 secs has 4000 data points



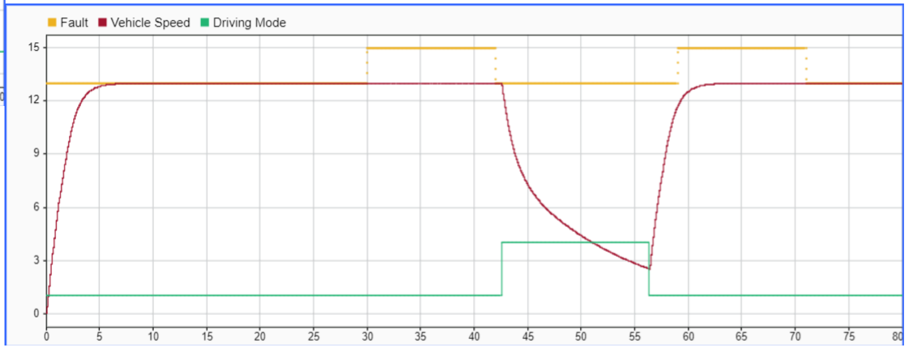
Sample Faulty Data for Vehicle 2



Noise fault injection

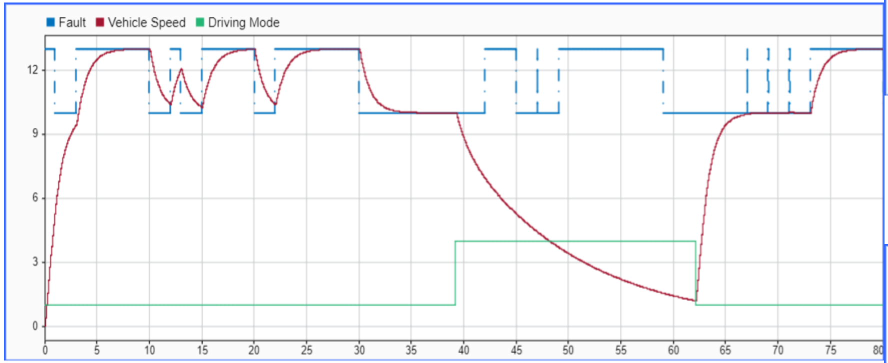


Offset fault injection

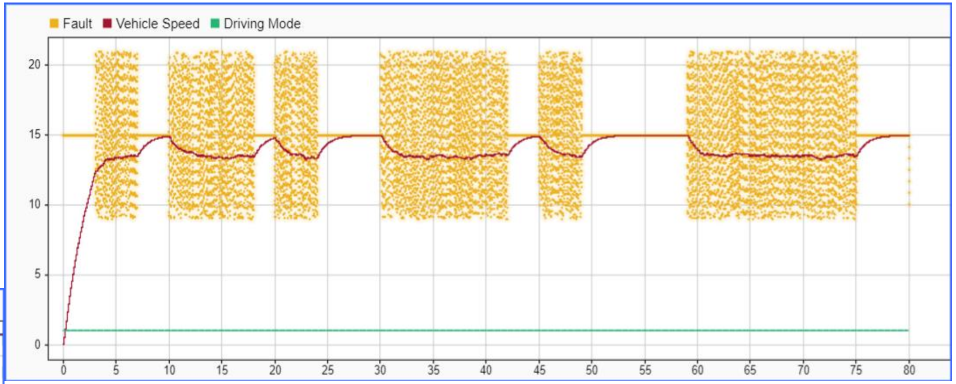


Stuck-At fault injection

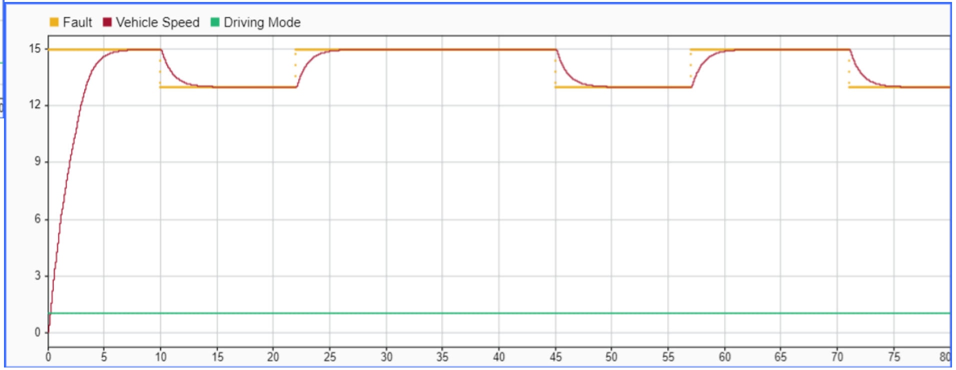
Sample Faulty Data for Vehicle 6



Offset fault injection



Noise fault injection



StuckAt fault injection

DATA PRE-PROCESSING

Ground Truth Labeling

New column Label is introduced consisting of values 0, 1, 2 and 3 where 0 denote fault-free scenario, 1 for noise fault, 2 for stuck-at and 3 for offset..

Data Shuffling

The final data set is made up by concatenating all the generated simulation files in a sequential manner. In order to build a better DL model, we shuffle the data set for our further use.

Train-Test Data Split

The data is then split randomly with 0.33 part being test data and 0.67 being the train data with a random state of 42. The test data is also further split into half for assigning value to validation data.

Categorical Conversion

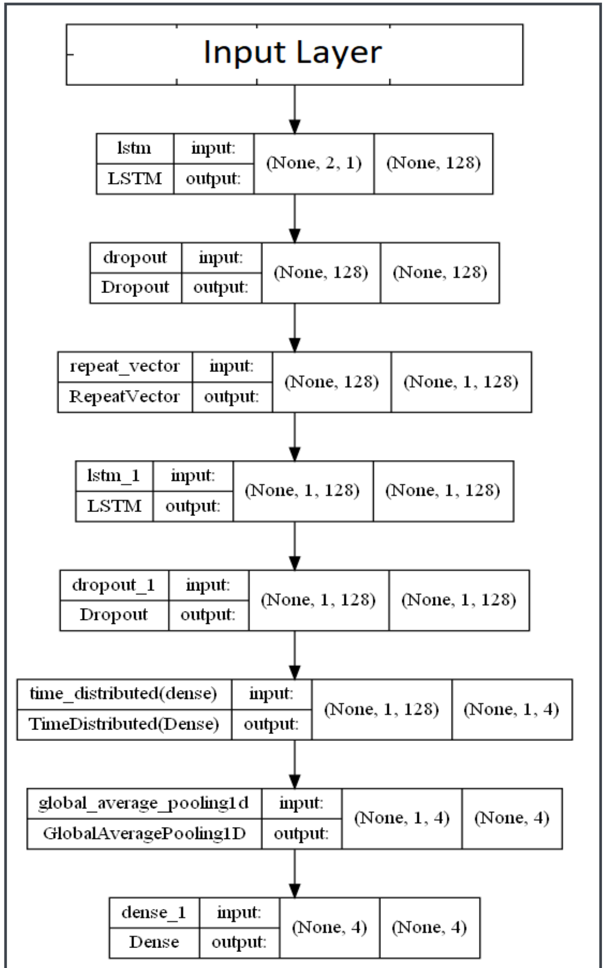
Since multiple fault categories are considered, the conversion of class vector(integers) to binary class matrix for the train data set in order to use with categorical cross-entropy.

FAULT DETECTION USING DL MODELS

Deep Learning Models Used

- Recurrent Neural Network [5] consists of three types i.e., Simple RNN, Gated Recurrent Unit (GRU) and Long Short-Term Memory(LSTM).
- Autoencoder is a type of neural network that is referred to learn a compressed representation of raw data as it is composed of an encoder and a decoder sub-models.
- Transformer models apply an evolving set of mathematical techniques, called attention or self-attention, to detect subtle ways even distant data elements in a series influence and depend on each other.
- In this thesis, we train and test our data with 6 different DL models i.e. Simple RNN, GRU-Simple RNN, BiGRU-BiSimple RNN [6], Autoencoders [9], Transformers [8], and hybrid LSTM-Autoencoders [7].

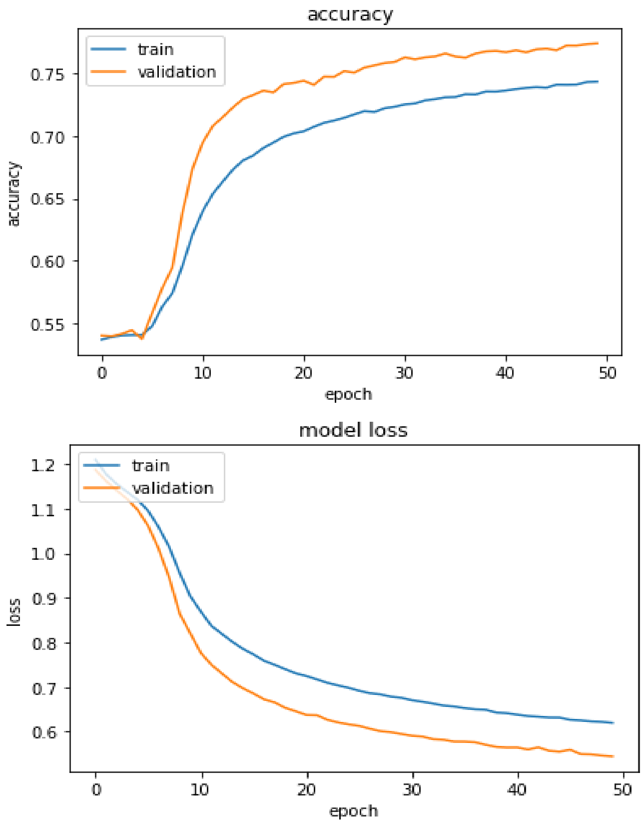
LSTM-Autoencoder Architecture



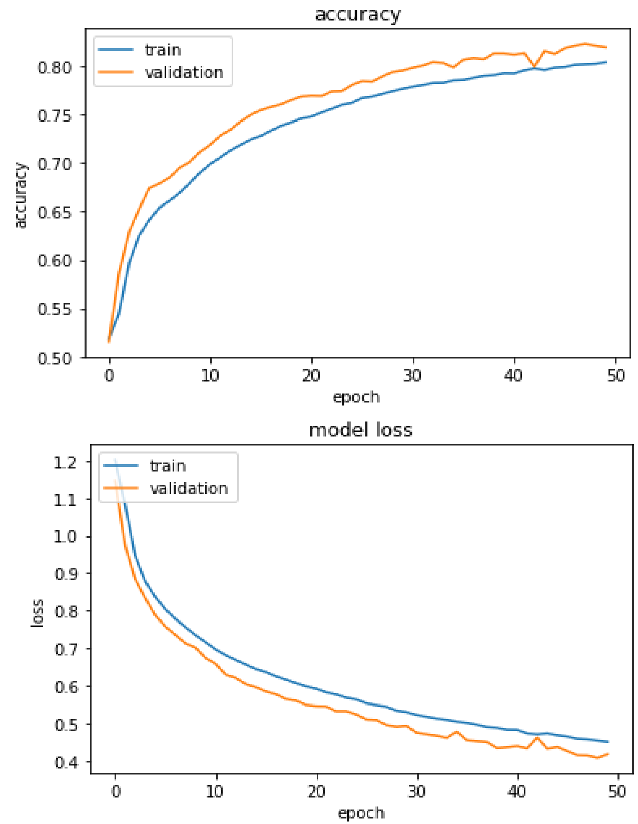
Hyper-Parameters	Value
Optimizer	Adam Optimizer
Learning Rate	0.001
Type	Sequential
Loss	Categorical Crossentropy
Batch Size	2048
No. of Epochs	50

LSTM-Autoencoder Performance

Accuracy and Loss for Uni-variate Data



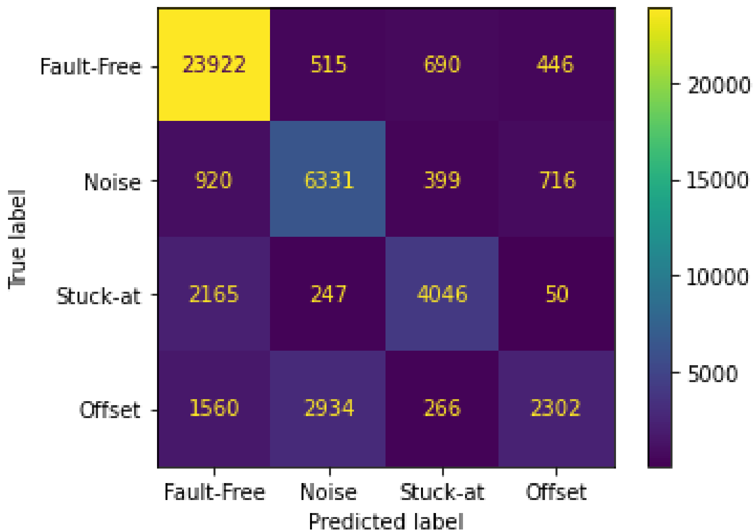
Accuracy and Loss for Multivariate Data



RESULTS

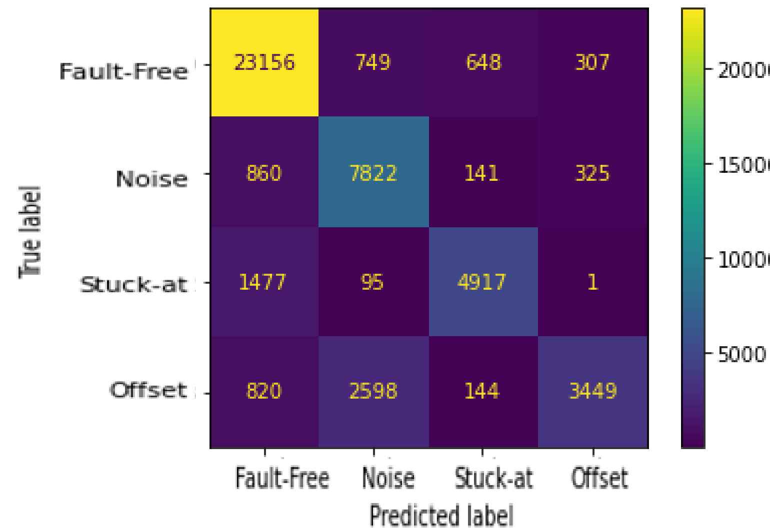
Fault Classification of LSTM-Autoencoders on Test Data

Univariate Data



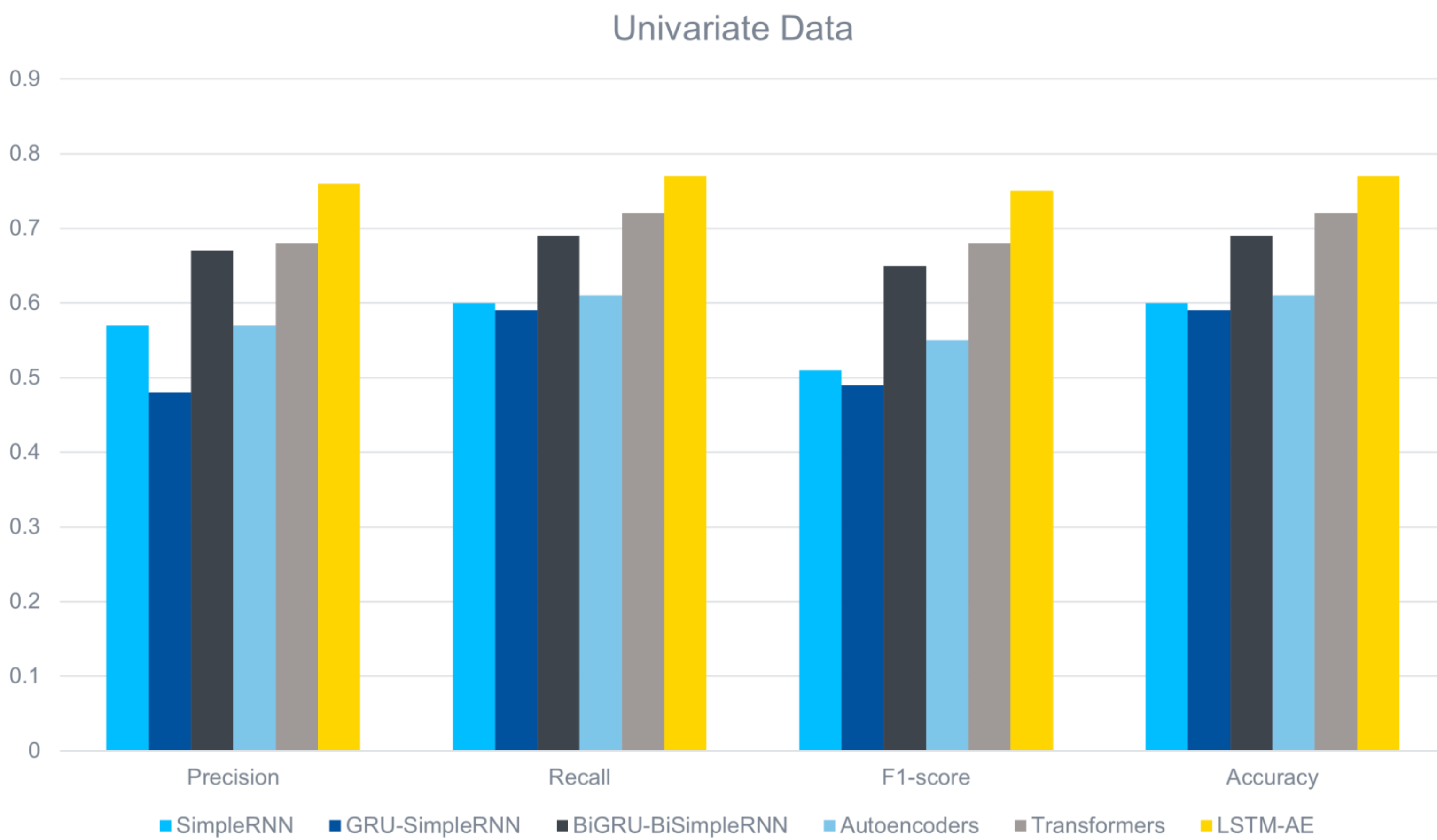
	precision	recall	f1-score	support
Fault-Free	0.84	0.94	0.88	25573
Noise	0.63	0.76	0.69	8366
Stuck-at	0.75	0.62	0.68	6508
Offset	0.66	0.33	0.44	7062
accuracy			0.77	47509
macro avg	0.72	0.66	0.67	47509
weighted avg	0.76	0.77	0.75	47509

Multivariate Data

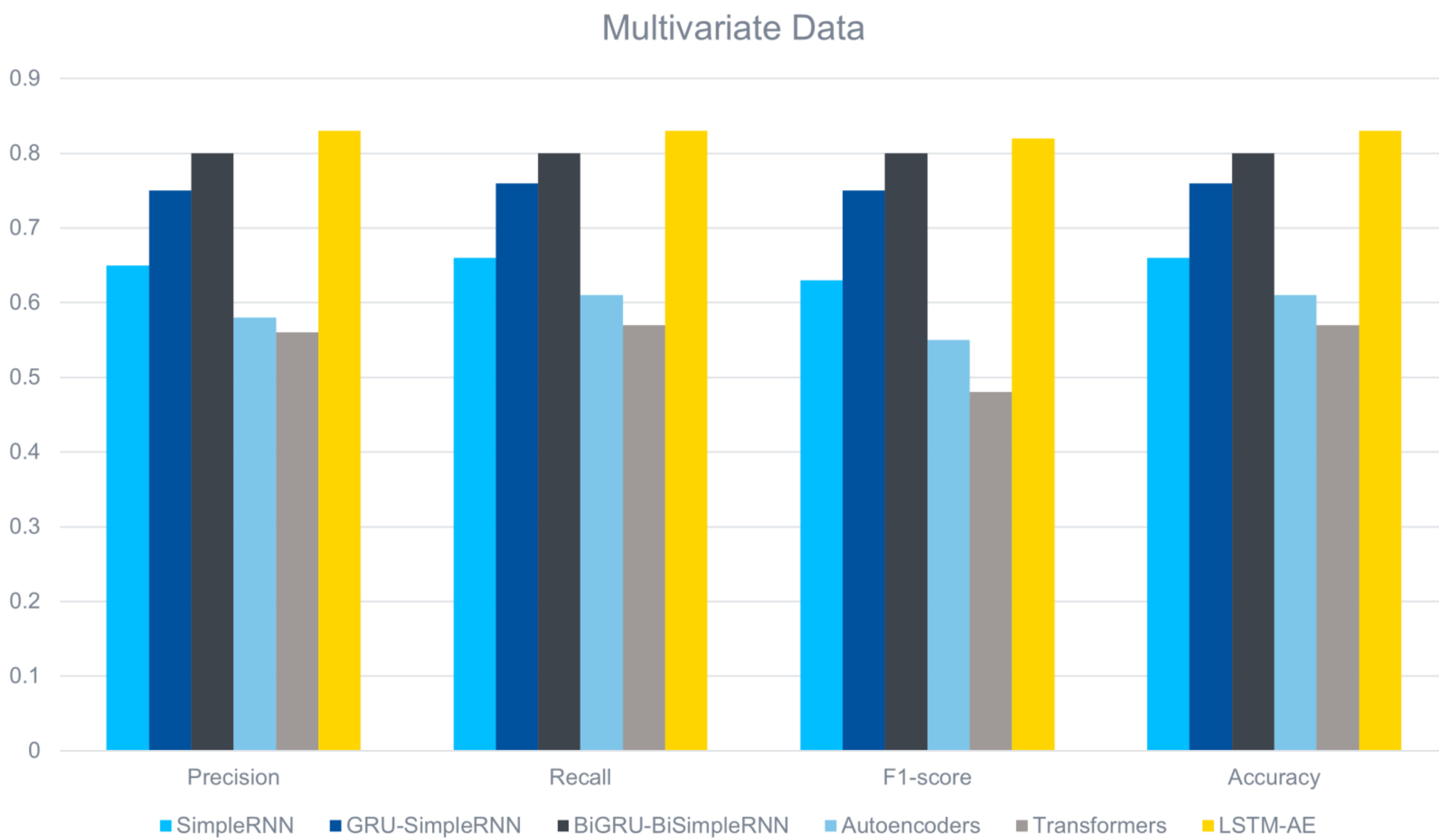


	precision	recall	f1-score	support
Fault-Free	0.88	0.93	0.91	24860
Noise	0.69	0.86	0.77	9148
Stuck-at	0.84	0.76	0.80	6490
Offset	0.84	0.49	0.62	7011
accuracy			0.83	47509
macro avg	0.81	0.76	0.77	47509
weighted avg	0.83	0.83	0.82	47509

Comparison between DL Models for Univariate Data



Comparison between DL Models for Multivariate Data



CONCLUSION & FUTURE SCOPE

Conclusion

01 The Traffic Model of MOBATSim consisting of 10 vehicles with 3 different scenarios is adapted. Sensor Faults are injected using FI block at speed and distance sensor of Vehicle 2 and vehicle 6 .

02 Faulty and Fault-Free (univariate and multivariate) data is generated and eventually, labeled and pre-processed

03 Simple RNN, GRU-Simple RNN, Bidirectional GRU-Bidirectional Simple RNN, Autoencoders, Transformer-based classifier and LSTM-Autoencoder architectures are used for training both types of data.

04 LSTM-Autoencoders perform the best out of all the DL models used for both types of data giving an accuracy of 77% with univariate data and 83% with multivariate data.

Future Scope

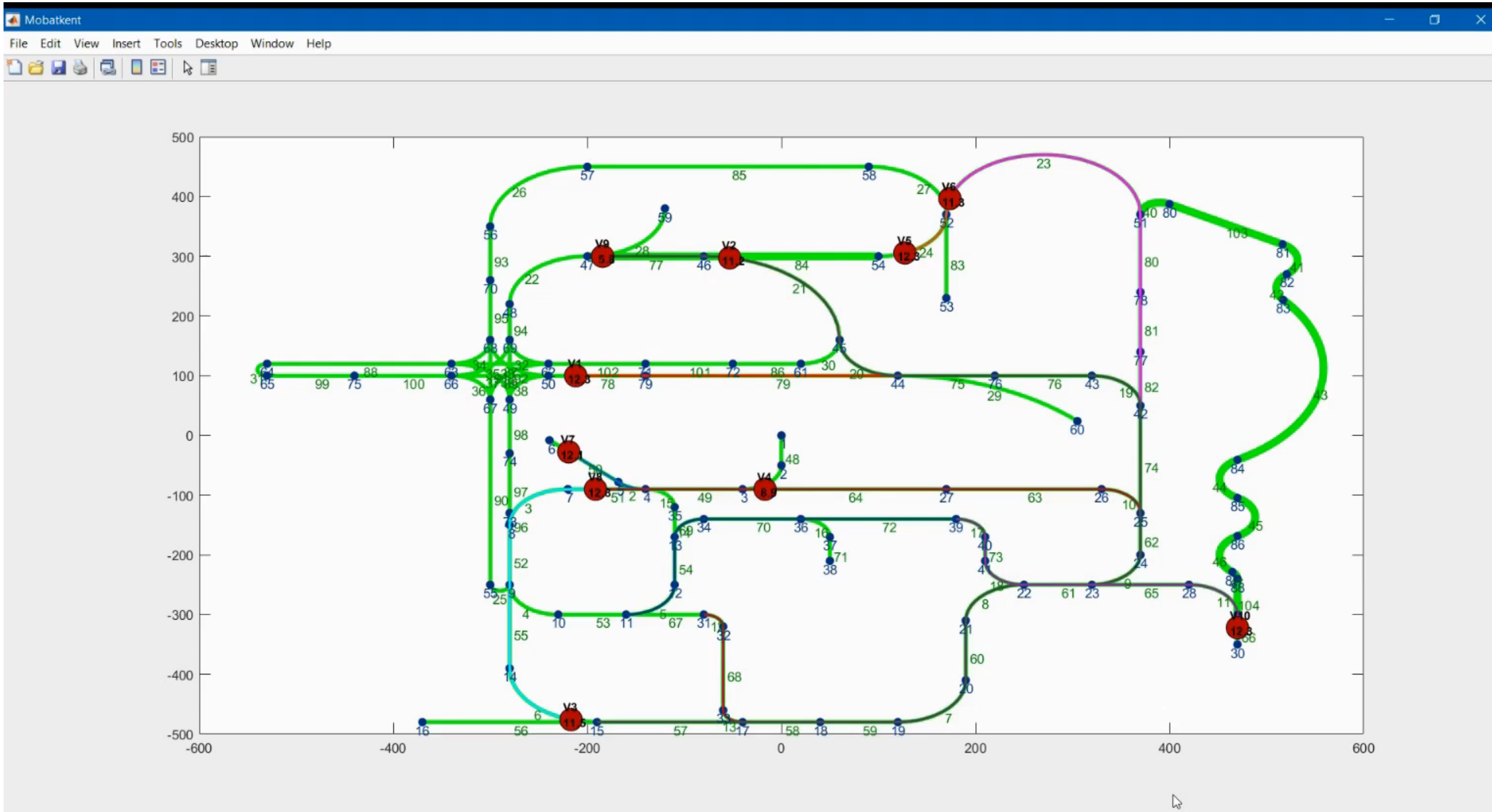
Data Collection with different changes in the Vehicle Model for MOBATSim

Other fault types like hardware and network faults can be implemented.

Other DL and ML models can be used for fault classification.

Different pre-processing methods can be used for other architectures

Demonstration Video



REFERENCES

[1] https://mobatsim.com/

[2] **Deep Learning-Based Anomaly Detection in Cyber-Physical Systems: Progress and Opportunities,** Yuan Luo, Ya Xiao, Long Cheng, Guojun Peng, Danfeng Daphne Yao, arXiv:2003.13213v2 **[cs.CR]**

[3] http://www.sciencedirect.com/science/article/pii/S2405896319304100“

[4] Monostori L. (2018) Cyber-Physical Systems. In: The International Academy for Production, Chatti S., Tolio T. (eds) CIRP Encyclopedia of Production Engineering. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-35950-7_16790-1

[5] A. Nanduri and L. Sherry, "Anomaly detection in aircraft data using Recurrent Neural Networks (RNN)," *2016 Integrated Communications Navigation and Surveillance (ICNS)*, 2016, pp. 5C2-1-5C2-8, doi: 10.1109/ICNSURV.2016.7486356.

[6] https://medium.com/@felixs_76053/bidirectional-gru-for-text-classification-by-relevance-to-sdg-3-indicators-2e5fd99cc341

[7] “LSTM-Autoencoder based Anomaly Detection for Indoor Air Quality Time Series Data”, Yuanyuan Wei, Julian Jang-Jaccard, Wen Xu, Fariza Sabrina, Seyit Camtepe, Mikael Boulic, arXiv:2204.06701v1 **[cs.LG]**

[8] “TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data” , Shreshth Tuli, Giuliano Casale, Nicholas R. Jennings, arXiv:2201.07284v6 **[cs.LG]**

[9] https://towardsdatascience.com/anomaly-detection-using-autoencoders-5b032178a1ea

THANK YOU