A LSTM Approach to Detection of Autonomous Vehicle Hijacking

Naman Singh Negi[1], Ons Jelassi[1], Stephan Clemencon[1] and Sebastian Fischmeister[2]
[1]*LTCI, Telecom ParisTech, Paris, France*
[2]*Dept. of Elec. and Computer Eng., University of Waterloo, Waterloo, Canada*

Keywords: Anomaly, Anomaly Detection, Recurrent Neural Networks, Long Short-term Memory.

Abstract: In the recent decades, automotive research has been focused on creating a driverless future. Autonomous vehicles are expected to take over tasks which are dull, dirty and dangerous for humans (3Ds of robotization). However, augmented autonomy increases reliance on the robustness of the system. Autonomous vehicle systems are heavily focused on data acquisition in order to perceive the driving environment accurately. In the future, a typical autonomous vehicle data ecosystem will include data from internal sensors, infrastructure, communication with nearby vehicles, and other sources. Physical faults, malicious attacks or a misbehaving vehicle can result in the incorrect perception of the environment, which can in turn lead to task failure or accidents. Anomaly detection is hence expected to play a critical role improving the security and efficiency of autonomous and connected vehicles. Anomaly detection can be defined as a way of identifying unusual or unexpected events and/or measurements. In this paper, we focus on the specific case of malicious attack/hijacking of the system which results in unpredictable evolution of the autonomous vehicle. We use a Long Short-Term Memory (LSTM) network for anomaly/fault detection. It is, first, trained on non-abnormal data to understand the system's baseline performance and behaviour, monitored through three vehicle control parameters namely velocity, acceleration and jerk. Then, the model is used to predict over a number of future time steps and an alarm is raised as soon as the observed behaviour of the autonomous car significantly deviates from the prediction. The relevance of this approach is supported by numerical experiments based on data produced by an autonomous car simulator, capable of generating attacks on the system.

1 INTRODUCTION

The past few decades have seen the automotive industry invest significant amount of resources in the development and deployment of autonomous and connected vehicles. Autonomous vehicles are capable of sensing their environment and navigating under different driving conditions without any human intervention. Connected vehicles are capable of using various communication technologies to connect and communicate with a network i.e. with the driver, other cars on the road (vehicle-to-vehicle [V2V]), roadside infrastructure (vehicle-to-infrastructure [V2I]), and the "Cloud" [V2C]. With the advancement in sensor technology, information exchange networks, wireless technology such as Bluetooth, Wi-Fi and 4G and the ease of processing data, autonomous systems are becoming exceedingly capable and efficient at performing different driving tasks. It is expected that automation in vehicles will help improve vehicle safety, decrease traffic congestion and commute times, increase fuel efficiency and provide in-Car infotainment. *'Autonomous Vehicle Sales Forecast 2018'* by *IHS Markit* suggests Autonomous vehicle sales to surpass "33 million annually in 2040. It is predicted that there will be a quarter of a billion connected vehicles on the road by 2020 (Velosa et al., 2014). Hence, with time, autonomous and connected vehicles will find increasing use in real-world applications.

As the whole autonomous environment is data driven, data acquisition and data reliability become important aspects for smooth and efficient working of the system. In the future a typical autonomous vehicle data ecosystem will include data from internal sensors, infrastructure, communication with nearby vehicles, and other sources. A data based environment is a delicate structure and is vulnerable to error and hacking, which makes the autonomous and connected vehicles highly susceptible to malicious attacks and information tampering, along with system failures.

Negi, N., Jelassi, O., Clemencon, S. and Fischmeister, S.
A LSTM Approach to Detection of Autonomous Vehicle Hijacking.
DOI: 10.5220/0007726004750482
In *Proceedings of the 5th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2019)*, pages 475-482
ISBN: 978-989-758-374-2

(Koscher et al., 2010) demonstrated that hacking into the car's internal network can give the attacker access to wide range of vehicle control functions like disabling the brakes, manipulating the speed, stopping the engine, etc. In 2016, Troy Hunt managed to hack into a Nissan Leaf using an unsecured API in the HVAC component shipped by an OEM vendor. (Petit and Shladover, 2015) researched potential threats and cyber- attacks on automation and cooperation automated vehicles. Other studies have also shown that various vehicle systems can be hacked to unable a remote takeover of the vehicle (Checkoway et al., 2011). Chrysler recalled 1.4 million vehicles after (Miller and Valasek, 2015) showed how a Jeep Cherokee was remotely hacked and stopped on a highway.

Moreover increase in the connectivity of the vehicles further tends to increase its vulnerability to malicious attacks. Problems in one car can affect other connected cars and networks in the connected environment. This means that not only device-level security must be addressed, but also data security during transmission and storage. Protecting connected vehicles hence becomes a challenging task due to complexity, connectivity (large attack surface) and legacy (unsafe and outdated technologies).

Hence autonomous systems must be complemented by anomaly-detection systems, in particular to answer the question : Can the data received be trusted? Anomaly detection is a technique used to identify unusual patterns that do not conform to expected behavior. For such a system to be effective, it has to be computationally light, and detect faults with high degree of both precision and recall. A too-high rate of false positives will lead operators ignoring the system whereas a too-low rate will make it ineffective. In addition, the faults must be detected quickly after their occurrence, so that they can be dealt with before they become catastrophic.

1.1 Related Work

Anomaly detection for real-time data has been researched in the past especially in the field of robotics and automation. (Goel et al., 2000), (Sundvall and Jensfelt, 2006), (Cork and Walker, 2007) use Kalman Filter (KF) as a tool for detecting anomalies by comparing predicted values with the observed values. KF used alone leads to a large number of false positives and therefore is combined with other computational techniques to provide robust detection. (Goel et al., 2000) combines KF with neural networks whereas (Cork and Walker, 2007) uses a non linear model associated to KF. (Chakravarty, 2013), (Hong, 2014) use rule-based techniques to extract and identify abnormal driving behaviors. Supervised learning-based classification techniques (see (Chen, 2015)) have also been used to identify anomalous patterns in data. (Pokrajac, 2007) uses K-Nearest Neighbor (KNN) to identify local outliers in data streams. (Brotherton and Mackey, 2001) uses the Mahalanobis distance to differentiate between nominal and abnormal behavior of aircraft signals. (Laurikkala et al., 2000), and (Lin et al., 2010) use the Mahalanobis Distance for detecting anomalies in multivariate data. Other techniques such as threshold-based detection and Bayesian assumptions of prior distributions(Rajasegarar et al., 2008), local messaging based distributed detection (Chen, 2015), (Branch et al., 2013), support vector machines (SVM) based detection (Zhang et al., 2013), have been studied in past research.

In this paper, we develop a LSTM approach to online hijacking detection for autonomous vehicles in two steps. It is assumed that, in the absence of an attack on the system, the behavior of a self-driving car is smooth and highly predictable at a short term horizon. Using this assumption, the first step consists of training the LSTM network to understand the system's baseline performance and behaviour. The trained model is then used to predict the vehicle parameters over a number of future time steps and an alarm is raised as soon as the observed behaviour of the autonomous car significantly deviates from the prediction. We show that such a LSTM model learnt using only the normal sequences can be used for detecting anomalies in various vehicle inputs time-series data namely: speed, acceleration, and jerk. The dataset used in this study was generated from experiments performed on a treadmill based autonomous car simulator at University of Waterloo, Canada(Simulator, 2018)

The rest of this paper is organized as follows: Section 2 presents the LSTM approach promoted and related works. Section 3 describes the Treadmill Demonstrator we used to generate the dataset and the parameters of the LSTM model. In section 4, the performance of our approach is investigated and some concluding remarks are collected in section 5.

2 LSTM BASED ANOMALY DETECTION

This section presents the rationale behind our approach. We start by briefly describing LSTM network models. Then, we describe how LSTM is used to model the dynamic behaviour of the system (autonomous vehicle in our case) in order to gather

knowledge about the baseline performance (model training stage). The model is then used to detect changes in the system as well as outliers using root mean square error metrics (prediction stage).

2.1 Long Short-Term Memory (LSTM) Networks

The persistence of information in our brain helps us in understanding any situation based on the memory of the past events. The human brain does not erase everything each time a new situation occurs and start from scratch. Recurrent neural networks use the same logic and in essence are neural networks with loops in them which allows information to persist. A loop allows information to be passed from one step of the network to the next.

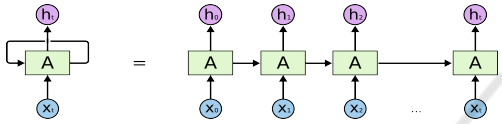

Figure 1: Recurrent neural networks(RNN).

Thus, RNNs use past information to understand the present situation. One major drawback of RNN is how far in the past should we search. Sometimes, the recent past can provide enough information to execute the present task, but there are also times when we have to look further back in the memory to extract the required and relevant information. It's entirely possible that for certain applications or in certain scenarios this gap between the relevant information and the point where it is needed becomes very large. Performance of RNNs deteriorates as this gap grows. Deep neural networks are already used in testing autonomous cars like in (Tian et al., 2018).

Long Short Term Memory networks are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter and Schmidhuber in (Hochreiter and Schmidhuber, 1997) and are explicitly designed to avoid the long-term dependency problem. LSTMs also have this chain like structure, but the repeating module has a different structure. Unlike RNNs that have a single neural network layer, LSTM includes four layers interacting in a special way. The LSTM has the ability to remove old information or add new information at any point, which is regulated by structures called gates. Gates are composed of a sigmoid neural net layer and a pointwise multiplication operation and are a way to exchange informations. A LSTM has three of these gates, to protect and control the information.

- Forget Gate: to decide what information we're going to throw away from the block.
- Input Gate: to decide what new information we're going to update/store in the block
- Output Gate: to decide what to output based on the input and on the memory of the block.

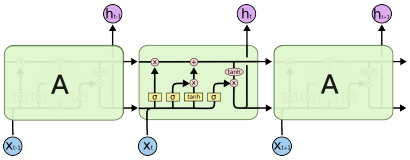

Figure 2: LSTM.

3 DATA ACQUISITION

3.1 Experimental Setup

The autonomous car simulator at the University of Waterloo, Canada was used to collect data under different driving scenarios. This demonstrator is a laboratory platform used for research and validation of results on real-time safety-critical systems in the context of assisted and autonomous driving algorithms. The platform consists of a treadmill which mimics the movement of a vehicle on a straight road. The position control places the vehicle on the treadmill without it drifting away. The car model is capable of emulating various driving scenarios like free fun, slalom, platooning and collision avoidance. For the present study, the following data was collected for different driving scenarios with and without injection of attacks/anomalies

- Position Data (Infrared Sensor)
- Vehicle Orientation (Infrared Sensor)
- Vehicle Commands (Steer/Throttle)
- Anomaly Information

The position data acquired from the different tests was used to calculate the

- Velocity: Rate of change of position
- Acceleration: Rate of change of Velocity
- Jerk : Rate of change of Acceleration

3.1.1 Dataset 1: Non Anomalous Data

The purpose of this dataset was to gather information about the normal (non-anomalous) driving behavior. This dataset is used to train the LSTM predictive model. The data acquired was from a free run driving scenario which represents the vehicle moving at constant velocity in a straight line. Under this scenario, control inputs (which represent driver inputs)

were given to change the position of the vehicle in the longitudinal and lateral direction at different intervals (Figure 3).

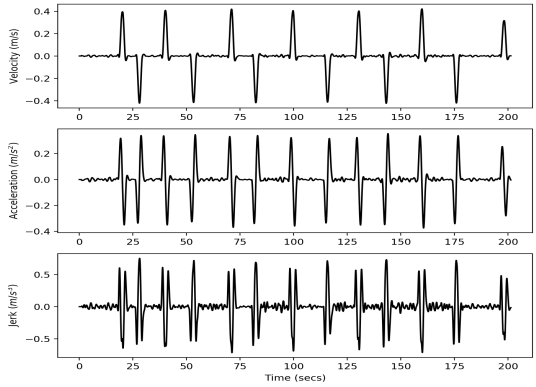

Figure 3: Non Anomalous Training Data.

3.1.2 Dataset 2: Anomalous Data

The purpose of this dataset was to gather information about the anomalous driving behavior which represents possible attack to the system. This dataset was used to test the prediction efficiency of the trained LSTM model. The injected anomaly is called compound injection and it simulates a scenario where a malicious attacker manages to gain access to the car's transmission control using a wirelessl network, by causing the throttle value to be multiplied by the specified positive factor. The dataset consists of a mix of normal driving inputs along with injected anomalies

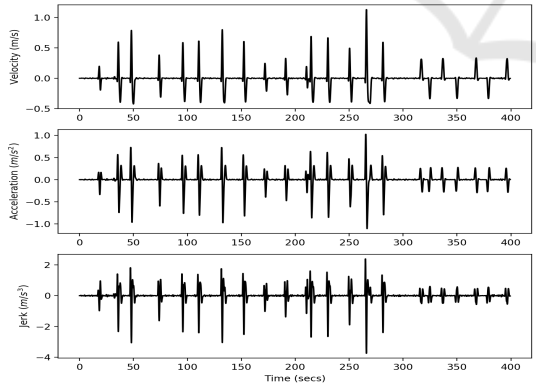

Figure 4: Anomalous Testing Data.

3.2 Performance Metrics

To calculate the performance of the LSTM anomaly detection algorithm, a set of thresholds were used based on the range of values of the prediction error. These thresholds were used to classify data into normal and abnormal behaviour. The performance metrics used in this paper are

- True Positive Rate (TPR) or Sensitivity or Detection Rate measures the proportion of correctly classified actual (true) positives (in this case the anomalies)

$$TPR = \frac{TP}{TP+FN} \tag{1}$$

- Specificity (or true negative rate) measures the proportion of correctly classified actual (true) negatives.

$$Specificity = \frac{TN}{TN+FP} \tag{2}$$

- False Positive Rate (FPR) or False Alarm Rate measures the proportion of wrongly classified actual (true) negatives

$$FPR = \frac{FP}{FP+TN} \tag{3}$$

- False Discovery Rate (FDR) measures the proportion of positive test results that are incorrectly identified

$$FDR = \frac{FP}{FP+TP} \tag{4}$$

- F_β score is the weighted mean of precision and recall.

$$F_\beta = \frac{(1+\beta^2)*P*R}{\beta^2*P+R} \tag{5}$$

Area Under the Curve (AUC) calculated for the Sensitivity- Specificity curve and Decision Rate-False Alarm Rate was also used to compare the performance.

3.3 Evaluating LSTM Algorithm

In (Malhotra et al., 2015), LSTM is used to model time series data and proved to be efficient for detecting anomalies. In this paper, we use a similar approach for on-line detection of malicious attacks on autonomous vehicles. In the training stage, a LSTM model adapts its weights to mimic the training data. In our case, we train the model using normal data as we would like the model to learn and understand a normal driving behaviour. This model is next used for prediction of future values. A significant deviation from the predicted behavior tends to indicate the occurrence of an attack on the system. The intuition behind this approach is that the LSTM model is only shown normal instances during training and hence it learns to reconstruct the normal behavior of the system. When the trained model is given an anomalous dataset, it will not be able to reconstruct the signal well, and hence would lead to higher prediction errors compared to the prediction errors for the normal

data. When evaluating the performance of the modelling algorithm, the following two parameters were optimized:

- Training Window Size: Number of previous time steps used as input to predict the next step
- Predict Ahead: Number of future time steps that are predicted
  The range of values used for the two tuning parameters can be seen in Table 1.

Table 1: Model Tuning Parameters.

Window Size	3, 6, 9, 12, 15, 18
Predict Ahead	1, 2, 3, 4, 5, 6, 7, 8, 9

4 RESULTS AND dISCUSSIONS

This section discusses results of LSTM modelling on the control parameters which have different levels of difficulty as far as anomaly detection is concerned.

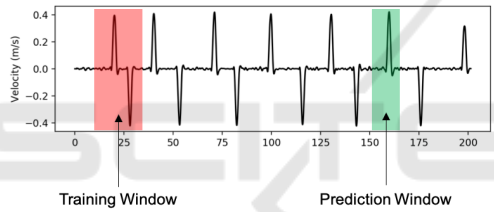

Figure 5: Tuning Parameters.

Training window size represents the size of the input sequence used to predict the future values (Figure 5). For example window size of 9 means that 9 previous data points were used as input to model and predict one or more future values. Similarly prediction window size represents the number of future values that were predicted using the trained model. Hence a prediction window size of 3 means that 3 at any time 't',the model was used to predict values at time t+1,t+2 and t+3.

4.1 Model Parameters

LSTM Model trained on the Acceleration data was used to optimize the parameters discussed in Section 3.3. Root mean square error (RMSE) of the predicted data values was used to compare the performance.

4.1.1 Training Window Size

Figure 6 below shows the prediction result for different window sizes. Predict ahead value of 1 was used for all training window sizes. Hence for a window size of 'n', input sequence data from t=0 to t=n was used to model and predict the value for t=n+1. Similarly input sequence data from t=1 to t=n+1 was used to model and predict the value for t=n+2 s and so on.

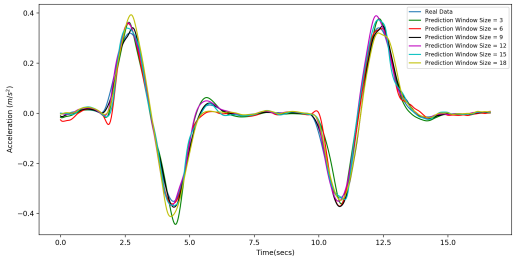

Figure 6: Window Size Tuning.

Figure 7 below shows the RSME error for different window sizes. It can be seen that least error and hence maximum performance was found using window size 9 i.e. 9 previous values used for prediction.

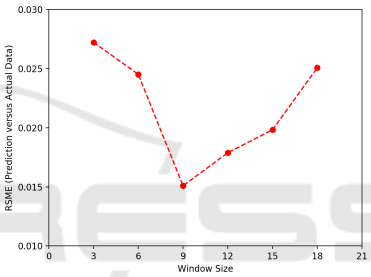

Figure 7: RSME for different Window Sizes.

4.1.2 Predict Ahead

Figure 8 below shows the prediction result for different predict ahead values. Training window size of 9 was used for all predictions.

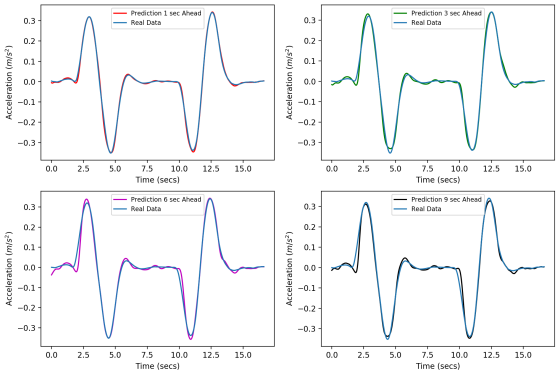

Figure 8: Predict Ahead Parameter Tuning.

For every time step the values were predicted 'n' times i.e. a data value at 't+n' was predicted 'n' times. To find a consolidated prediction value a weighted

mean of all the predicted values was calculated every time step

$$X_{t+n} = \frac{\sum_{t}^{t+n} w_t * X_t}{\sum_{t}^{t+n} w_t} \quad (6)$$

Figure 9 below shows the RSME error for different predict ahead values. It can be seen that error and hence performance deteriorates as the model tries to predict further ahead.

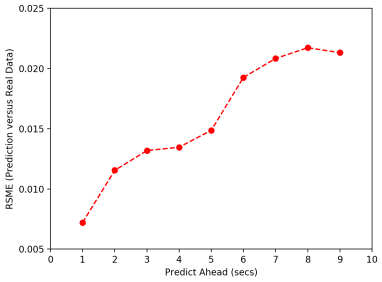

Figure 9: RSME for Predict Ahead Parameter Tuning.

4.1.3 Final LSTM Model

3 Uni-variate and 3 Multi-variate LSTM models are used in this paper. The Multi-Variate models uses all three time series (Velocity, Acceleration, and Jerk) together as input to predict the three parameters individually. A single layer LSTM model was used in order to keep the model simple and computationally light. Table 2 shows summary of the different models.

Table 2: LSTM Models Summary.

Model 1	Univariate	Velocity	Velocity
Model 2	Univariate	Acceleration	Acceleration
Model 3	Univariate	Jerk	Jerk
Model 4	Multivariate	Velocity Acceleration Jerk	Velocity
Model 5	Multivariate	Velocity Acceleration Jerk	Acceleration
Model 6	Multivariate	Velocity Acceleration Jerk	Jerk

4.1.4 Training and Prediction

Window size of 9 was used to train the model for 500 epochs. Each input sequence was an array of size (Number of Parameters X Window Size). Hence in this case array of size (1 X 9) for univariate models and (3 X 9) for multivariate models. The length of the entire dataset for training was 1000 seconds (at 30Hz sampling rate). Average time to train the model was 0.03 s/sequence/epoch for univariate models and 0.09 s/sequence/epoch for multivariate models. The trained model was used to predict values one time step ahead. The length of the testing dataset was 2220 seconds (30 Hz sampling rate). The average prediction time was 0.15 milliseconds/datapoint.

4.2 Model Performance

The efficiency of the different models is calculated by comparing the testing set and training set errors, and it was found that a certain threshold value can be used to detect attacks. The latter may vary with the efficiency of the prediction model but the trend remains the same with prediction error being higher in the event of an anomalous driving manoeuvre.

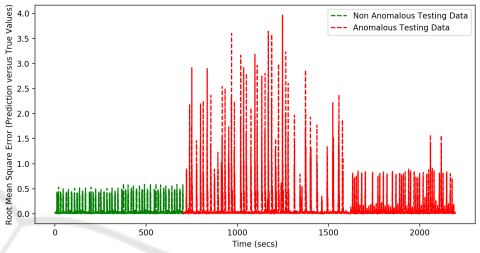

Figure 10: Model Performance: Prediction vs True values.

Figure 10 below shows the RMSE between the values predicted by the LSTM model and the true values captured by the vehicle sensors (Jerk Values). As can be seen from the figure RMSE is smaller for the non anomalous data (green) which shows that the model performed well in replicating normal behaviour. On the other hand the model was not very efficient in reproducing the anomalous behaviour which is evident from higher values of RMSE for anomalous testing data.

Figure 11 shows the Sensitivity versus Specificity curve for different models and Table 3 shows the AUC value for the respective curves.

Based on the AUC values it can be seen that LSTM velocity models (Univariate and Multivariate) are the least efficient and the models based on Jerk shows the best performance.

	AUC
Model 1	0.42
Model 2	0.75
Model 3	0.97
Model 4	0.56
Model 5	0.87
Model 6	0.97

F_β score was used to find the optimum value of threshold for the prediction error to detect anomalies. Figure 12 shows the value of F_β score for different threshold values.

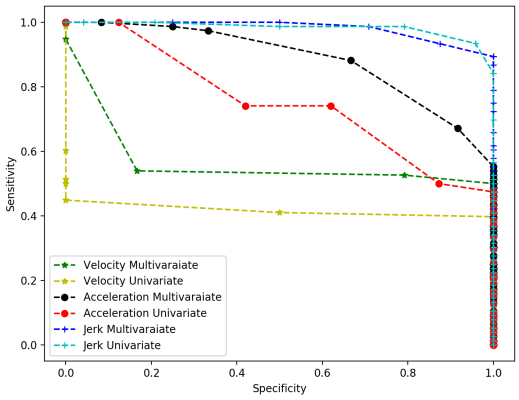

Figure 11: Model Comparison.

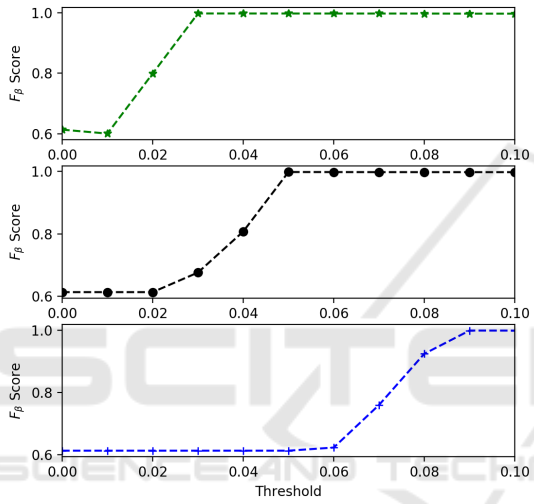

Figure 12: ROC Curve.

In the results shown in Table 4 and Table 5, $F_\beta = 0.8$ and 0.9 is used to find the threshold for different parameters. Using these threshold values the different performance metrics were calculated.

It can be seen that Univariate Velocity Model (Model 1) shows the worst performance (low TPR and high FPR). The Multivariate Velocity Model (Model 4) shows lowest FPR but also has the lowest TPR whereas on the other hand Multivariate Jerk model (Model 6) has the highest TPR but also shows high FPR.

5 CONCLUSION

In this paper, we have proposed an anomaly detection model for hijacking detection based on Long Short-Term Memory Recurrent Neural Network. We have provided empirical evidence that LSTM networks are relevant to predict the normal behaviour of a self-driving vehicle at a short term horizon, and can be next used to detect possible attacks on the system. We showed that even a basic LSTM Model approach yielded promising results on three different datasets.

Table 3: $F_\beta = 0.8$

Model	TPR	FPR	FDR
Model 1	0.43	0.58	0.30
Model 2	0.74	0.26	0.19
Model 3	0.98	0.5	0.24
Model 4	0.53	0.21	0.2
Model 5	0.88	0.33	0.19
Model 6	1.0	0.5	0.24

Table 4: $F_\beta = 0.9$

Model	TPR	FPR	FDR
Model 1	0.42	0.5	0.27
Model 2	0.5	0.08	0.07
Model 3	0.93	0.04	0.03
Model 4	0.5	0.0	0.0
Model 5	0.67	0.08	0.07
Model 6	0.97	0.13	0.08

The results show that the multivariate models show better performance than their counterpart univariate models in terms of TPR, FPR and FDR values. The LSTM models based on the Jerk parameter showed the best performance based on the AUC values for the Sensitivity-Specificity curve and also showed highest True Positive Rate. But the model also showed high False Positive rate. A possible explanation for this result is insufficient tuning of the model leading to relatively high prediction errors even for the normal behavior in the testing stage. Further tuning the model by using more data for training, increasing the complexity of the model and/or running the model over higher number of epochs might help reduce the False Positive rate.

The multivariate LSTM model for Velocity showed the best performance in terms of False Positive rate but performed poorly in terms of True Positive rate and AUC values for Sensitivity-Specificity curve. This may be explained by the high number of anomalies with a very small displacement in the longitudinal or lateral direction resulting in velocity values not very different from the normal behavior. Thus this model can be useful for anomalies with displacement similar or greater than normal behaviour. In future work, we will focus on:

1. LSTM model parameter tuning to improve the robustness
2. Improve the anomaly detection efficiency of the model
3. Model extension to be able to discriminate between different types of anomalies

REFERENCES

Branch, J. W., Giannella, C., Szymanski, B., Wolff, R., and Kargupta, H. (2013). In-network outlier detection in wireless sensor networks. Knowledge and information systems, vol. 34, no. 1.

Brotherton, T. and Mackey, R. (2001). Anomaly detector fusion processing for advanced military aircraft. pages 3125–3137. IEEE Proceedings on Aerospace Conference.

Chakravarty, T., G. A. B. C. C. A. (2013). Mo-bi drive score a system for mobile sensor based driving analysis: A risk assessment model for improving one's driving. pages 338–344. 7th International Conference on Sensing Technology (ICST),IEEE.

Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., Koscher, K., Czeskis, A., Roesner, F., and Kohno, T. (2011). Comprehensive experimental analyses of automotive attack surfaces. San Francisco, USA. USENIX Security Symposium.

Chen, Z., Y. J. Z. Y. C. Y. L. M. . (2015). Abnormal driving behaviors detection and identification using smartphone sensors. pages 524–532. 12th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), IEEE.

Cork, L. and Walker, R. (2007). Sensor fault detection for uavs using a nonlinear dynamic model and the imm-ukf algorithm. pages 230–235. IDC.

Goel, P., Dedeoglu, G., Roumeliotis, S. I., and Sukhatme, G. S. (2000). Fault-detection and identification in a mobile robot using multiple model estimation and neural network. ICRA-00.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hong, J. H., M. B. D. A. K. (2014). A smartphone-based sensing platform to model aggressive driving behaviors. pages 4047–4056. 32nd Annual ACM conference on Human Factors in Computing Systems.

Koscher, K., Czeskis, A., Roesner, F., Patel, S., Kohno, T., Checkoway, S., McCoy, D., Kantor, B., Anderson, D., and Shacham, H. (2010). Experimental security analysis of a modern auto- mobile. in security and privacy (sp). page 447–462, Stamford, CT, USA. IEEE Symposium.

Laurikkala, J., Juhola, M., and Kentala, E. (2000). Informal identification of outliers in medical data. Fifth International Workshop on Intelligent Data Analysis in Medicine and Pharmacology.

Lin, R., Khalastchi, E., and Kaminka, G. A. (2010). Detecting anomalies in unmanned vehicles using the mahalanobis distance. pages 3038–3044. ICRA.

Malhotra, P., Vig, L., Shroff, G., and Agarwal, P. (2015). Long short term memory networks for anomaly detection in time series. 23rd Euro- pean Symposium on Artificial Neural Networks.

Miller, C. and Valasek, C. (2015). Remote exploitation of an unaltered passenger vehicle.

Petit, J. and Shladover, S. E. (2015). Potential cyberattacks on automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):546–556.

Pokrajac, D. (2007). Incremental local outlier detection for data streams. IEEE Symposium on Computational Intelligence and Data Mining.

Rajasegarar, S., Leckie, C., and Palaniswami, M. (2008). Anomaly detection in wireless sensor networks. IEEE Wireless Communications, vol. 15, no. 4.

Simulator, A. C. (2018). University of waterloo.

Sundvall, P. and Jensfelt, P. (2006). Fault detection for mobile robots using redundant positioning systems. ICRA-06.

Tian, Y., Pei, K., Jana, S., and Ray, B. (2018). Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th International Conference on Software Engineering*, ICSE '18, pages 303–314, New York, NY, USA. ACM.

Velosa, A., Hines, J., LeHong, H., and Perkins, E. (2014). *Predicts 2015: The Internet of Things*. Gartner, Stamford, CT, USA.

Zhang, Y., Meratnia, N., and Havinga, P. J. (2013). Distributed online outlier detection in wireless sensor networks using ellipsoidal support vector machine. Ad hoc networks, vol. 11, no. 3.