International Journal of Research in Science and Technology
Volume 9, Issue 2: April - June, 2022
ISSN 2394 - 9554

# VEHICLE TRAJECTORY PREDICTION USING OPTIMIZED STA-LSTM FOR AUTONOMOUS DRIVING

**Kezia M, Anusuya K.V and Nihilesh K B**
Department of ECE, PSG College of Technology, Coimbatore

**ABSTRACT**
*The trajectory prediction is crucial for autonomous vehicles to avoid collisions. This paper optimizes trajectory prediction and lane-changing decisions for autonomous driving using spatial and temporal weights of vehicles using the LSTM model. Additionally, safety distance weight is added for spatial weights of LSTM. The gradient descent technique is applied with hyperparameter tuning to yield good accuracy. Also, the autonomous steering system based on a raspberry pi microcontroller and servo motor is designed to control the angle of steering by combining trajectory and lane prediction for various cases.*

*Keywords— Autonomous driving, Deep learning, Lane detection, Spatial and Temporal Attention Weights, LSTM, Trajectory prediction.*

## I. INTRODUCTION
The increased vehicles on the roads results in an increased possibility of collisions and accidents[1]. This situation is more prevalent on highways than in urban scenarios. The number of accidents and collisions can be controlled to some extent by individual accident prevention features such as proximity sensors etc. But this number can be further controlled by group communication and real-time analysis of vehicle trajectory prediction[2]. By enhancing the trajectory prediction algorithm, both the driver assistance system and vehicular communication can be improved and can effectively reduce the possibility of accidents and collisions in highway scenarios. Vehicle trajectory prediction has gained much attention from researchers as it improves the performance of autonomous driving significantly [3].

Accurate and real-time prediction of vehicular trajectory helps vehicles adjust their manoeuvres intelligently as per the running status of the neighbour vehicles. For autonomous vehicles, estimating the trajectory of neighbour vehicles is quite challenging in lane changing and highly dynamic traffic scenarios [4] [5]. As there is no default number of interactive vehicles with a target vehicle, the prediction accuracy should be scalable with respect to the number of neighbour vehicles [6].

## II. LITERATURE REVIEW
Paper [7] discusses prediction of trajectory based on local navigation map using spatial and temporal attentions in a highway scenario. Authors have considered time dimension and intelligent decision-making in dynamic circumstances. As a result, creating a spatial-temporal navigation map by linking the time and space dimensions through prediction helps in easy path planning in such environments. A Long Shot-Term Memory (LSTM) based framework based on the data is constructed to anticipate probable trajectories of many neighbouring vehicles within a specified range of the target vehicle using NGSIM dataset. Thus, the problem of dynamic disturbance may be overcome by combining dynamic targets and static impediments into a single domain or map.

The work in [8] predicts lane-changing decisions using bidirectional LSTM (Bi-LSTM) for autonomous driving on Highways. Autonomous vehicles' data acquired by scanners, and sensors were used to train and evaluate the proposed decision-making system. The output characteristics are set up to calculate the possibility of three manoeuvres: lane change on left and right sides, and lane-keeping. It has provided better accuracy in a case study than the earlier approaches.

Paper [9] comprehends the complicated dynamics of vehicle movements using LSTM over occupancy grid. The LSTM is trained with the coordinates and velocities of nearby vehicles derived from sensor readings and the LSTM predicts the future coordinate based on the previous trajectory input. The LSTM is intended to provide the probability of occupancy for the surrounding cars on the occupancy grid map in order to handle uncertainty in generating predictions.

Deo and Trivedi have suggested an LSTM model that leverages convolutional social pooling using NGSIM datasets[10]. An encoder, decoder, and CSP layers make up the model. The LSTM encoder learns the vehicular dynamics, while social pooling captures the interdependencies of vehicle motion in the scene. The LSTM decoder outputs the lateral and longitudinal manoeuvre probabilities for future trajectory predictions through two SoftMax layers.

Spatial-temporal attention-based Long Short-Term Memory (STA-LSTM) in [11] predicts the trajectory of the target vehicle for given spatial and temporal weights of neighbour vehicles. The space is discretized into a 3x13 grid. Vehicles around the target vehicle are placed into one of the grid cells based on the front bumper position. The history of movement of the vehicles along with other information such as speed, position, lane number is fed as input to the STA- LSTM algorithm.

Inspired from the previous works, this paper

- Predicts the trajectory of the vehicle in a highway scenario using an optimized STA LSTM algorithm (deep learning algorithm) and improves the corresponding prediction accuracy.
- Implements an autonomous driving system by combining the optimized STA-LSTM trajectory prediction algorithm with the lane detection algorithm in highway scenarios.

As illustrated in Fig. 1.

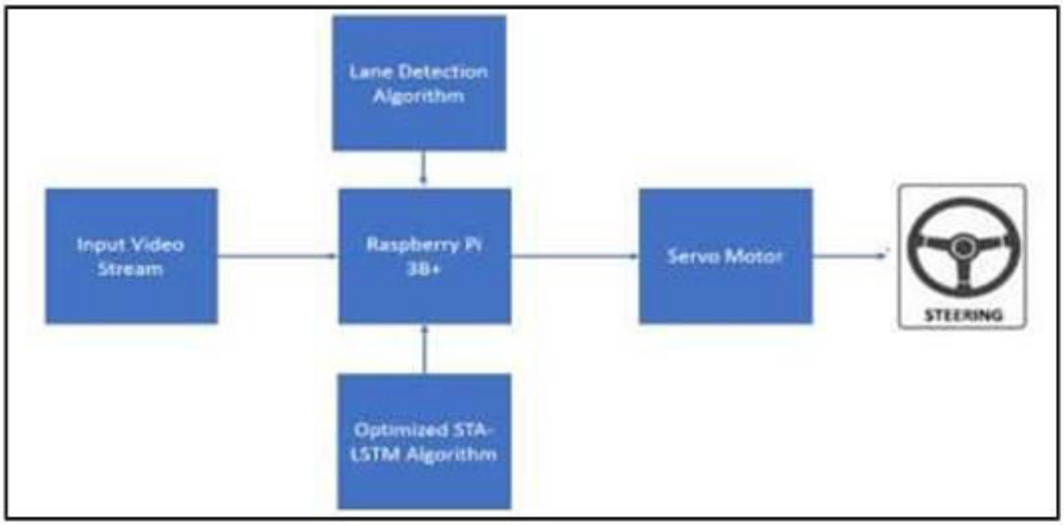



Fig 1: Proposed System block diagram

## III. PROPOSED METHOD

The proposed system works by feeding the past trajectory information to the optimized STA-LSTM algorithm, which suggests the best path of travel by considering neighboring vehicle influences and other factors such as lane position, speed of travel etc. A lane detection algorithm works in parallel to get the real time lane position, lane curvature etc. from the road. These two inputs are combined to get the steering wheel angle (turn prediction). The working of this system is illustrated by the following block diagram.

### A. Lane detection algorithm

Lane detection is an important aid for driver to get an in-depth traffic knowledge and further prevents collisions. Lane identification is a fundamental functional module in the realm of vehicle safety and intelligent vehicle navigation. Lane detection includes the following steps as per [12]:

1) Read and decode video files into frames
2) Grayscale conversion of image
3) Reduce noise by applying a filter
4) Canny edge detection
5) Mask the canny image
6) Hough Line Transform - transform used to detect straight lines.

### B. Optimized STA-LSTM

An LSTM model predicts target vehicle trajectory using spatial and temporal attention weights. This process uses crucial historical trajectories for predicting the future trajectory of the ego vehicle at the temporal level and takes the influence of surrounding cars using the safety distance between the cars at the spatial level. The target vehicle is placed in the center row of a 3x13 grid and each cell in the grid is 4m in width. A grid cell is assigned to each vehicle depending on the front position of the bumper. The proposed model learns

 ISSN 2394 - 9554

Spatial and temporal weights from t-step trajectories of vehicles and predicts the trajectory of a target vehicle for h-steps. The flow of the proposed model is shown in Fig.2.

The $\{Xt^V\}$ traces are utilized to generate hidden states $ht^V$, that are used to determine the attention weights in temporal level for each vehicle, represented by $At^V$.

$A^v = soft\,m(tanh(\mathcal{W}_\alpha S^v)), A^v \in R^{1\times T}, \mathcal{W}_\alpha \in R^{1\times d}$ (1)

$t$ $t$ $t$

Wa – learnable weights

St – Hidden states

The temporal level attention weights are combined with the hidden states to fill the tensor value $Ht^V$ (has a vehicle) or 0 (no vehicle).

5 5 5 5 5-1

5 5 5 5

$5_5 = 5_5 (5_5) = \sum_{5=5-5+1} 5_5 h_5, 5_5 \in 5$ (2)

Using the tensor values, spatial-level attention weights Bt are calculated, and trajectory of the objective vehicle is calculated.

$B_t = soft\max(\tanh(\mathcal{W}_\beta G_t)), \mathcal{W}_\beta \in R^{1\times d}$ (3)

By adding an additional safe distance checking function [13], we ensure that the leading and the following vehicle do not get closer than the safe distance permitted.

$D_{is} = 5_5 5 + \frac{(5_5)^2 - (5_5)^2}{25_{55555}} + L$ (4)

Where, $V_f$ – Longitudinal velocity of the following vehicle

$V_l$ - Longitudinal velocity of the leading vehicle

$\rho$ – Mean response time of the driver

$a_{brake}$ – Brake Deceleration of two vehicles

$L$ – Average length of two vehicles

Also written as,

$D_{is} = 5v'\ \Delta v + 5$ (5)

Where, $5 = \frac{1}{5_{55555}}$

Average velocity, $v' = \frac{5_5 + 5_5}{2}$

Relative velocity, $\Delta v = V_f - V_l$

$\beta = V_f\,\rho + L$

The learnable weight is directly proportional to the safe distance and inversely proportional to longitudinal distance between Vi and Vs.

$W_5 \propto \frac{5_{55}}{\Delta\, 5_{55}}$ (6)

- Finally, Vt is then fed into a feedforward network to predict objective vehicle's trajectory for H-steps.

$5_5 = 5_5(5_5)^5 = \sum_{5=1}^{5} 5^5 5^5_{55}$ (7)

 ISSN 2394 - 9554

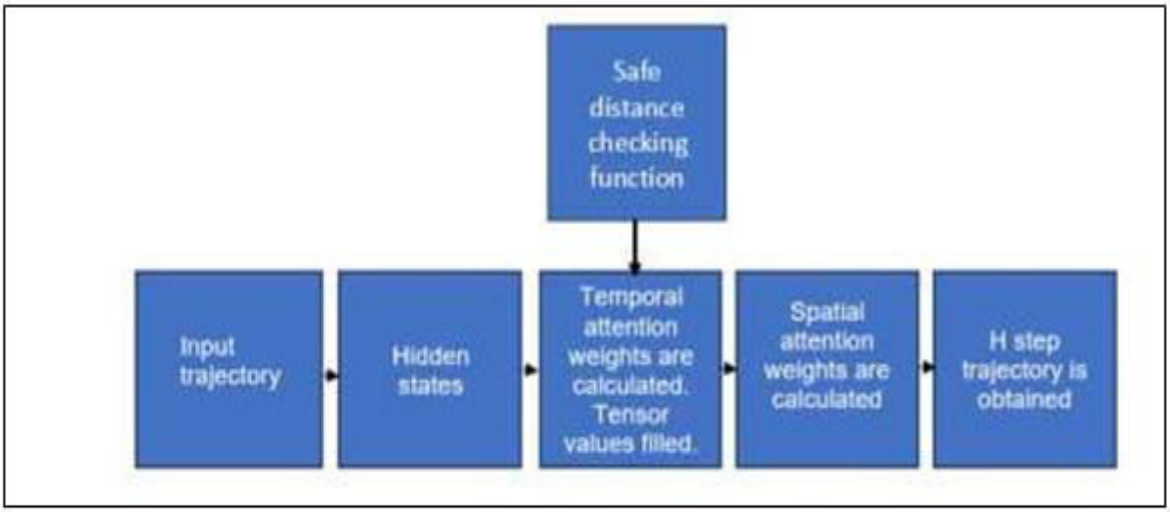



Fig. 2: Optimized STA-LSTM algorithm

**C. Hardware setup**

It includes a servo motor, raspberry pi 3b+ as shown in Fig. 3. The video stream is fed as input and using grayscale format noise is reduced in these frames using OpenCV. The lane detection algorithm and the STA-LSTM with gradient descent optimization (proposed algorithm) are fed to the microcontroller (Raspberry Pi 3b+) to estimate the trajectory of the vehicle with respect to its current position. The input frames of the video are processed and are classified into directions of movement (right, left, straight). The position of the vehicle is calculated with respect to lanes at any given point. The central point of the lane is calculated by using the position of both the left lane and the right lane.

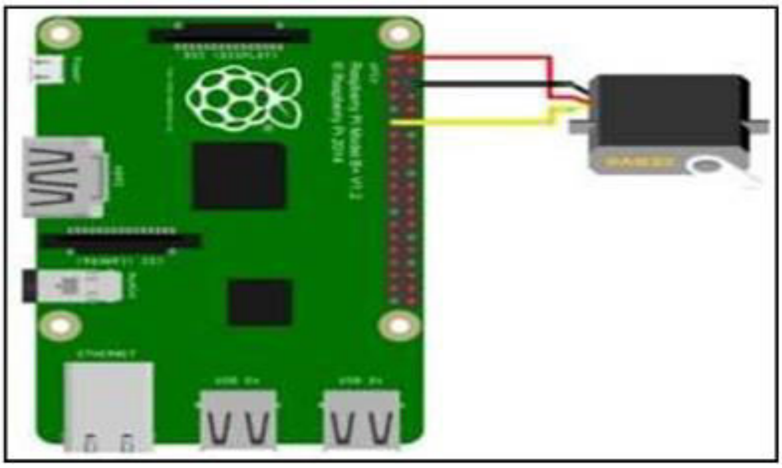


Fig 3: Hardware setup for Autonomous driving

The difference between the lane center and the image center is calculated and

- If it is positive, it is predicted as a right turn and
- If it is negative, it is predicted as a left turn.

The autonomous driving system or the driver assistance systems have the same accuracy as the trajectory prediction with only a slight computational overhead of processing the video stream, applying lane detection, and calculating the steering wheel offset.

**IV. IMPLEMENTATION AND RESULTS**

***A.* Dataset**

The proposed method is trained and tested using vehicle trajectories in Next Generation Simulation (NGSIM) dataset collected in USA highways namely US-101 and I-80 in busy hours for 15 minutes duration[14]. The dataset contains vehicle position, respective lane, time stamp, intersection, traffic details, crossings, etc. Fig. 4 displays the snapshot of the NGSIM dataset.

 ISSN 2394 - 9554

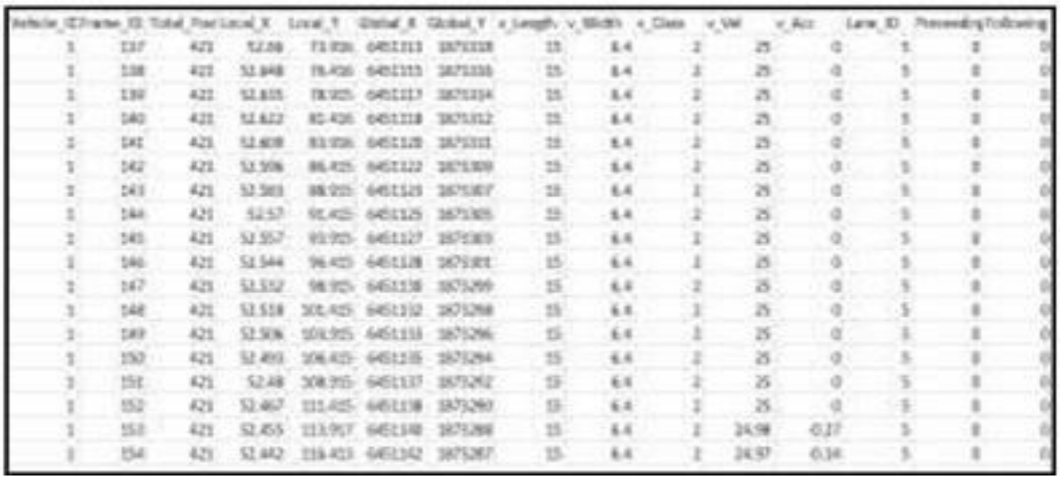


Fig. 4. NGSIM Dataset

### *B.* Hyperparameter tuning for Optimizing STA-LSTM

Further hyperparameters are tuned to improve the learning and output accuracy. The hyperparameters used are:

*Epoch:* It indicates the number of times the algorithm passed through the training samples.

*Batch size:* The total number of training samples used in a iteration.

*Adam Optimizer*: Adaptive Moment Estimation algorithm combines Momentum & Root Mean Square Propagation (RMSP) in optimization. It efficiently works on larger data with less memory.

*Validation Split:* To estimate the accuracy of a model when explicit dataset is unavailable.

By continuously tuning the hyperparameters, the values that give the maximum prediction accuracy are depicted n in Table I.

**TABLE I. HYPERPARAMETER TUNINIG VALUES**

| Hyperparameter | STA-LSTM | Optimized STA- LSTM |
|---|---|---|
| **Epoch** | 100 | 200 |
| **Batch Size** | 100 | 240 |
| **Optimizer** | Adam | Adam |
| **LSTM layer** | 10 | 25 |

### *C.* Optimization techniques

Gradient-based optimization is well suited for updating spatial and temporal weights for LSTM models since it uses past values. There are three major gradient optimization techniques [15], [16]:

*Gradient Descent* – Also known as epoch training, updates the model only after all training samples are evaluated. It is computationally efficient and gives stable error gradient.

*Stochastic Gradient Descent* – It checks the error for each training example and revises the parameters one at a time. The frequent updates give more detail and speed at the cost of computational loss and noisy gradients. But it helps in finding global minimum by skipping local one.

*Mini Batch Gradient Descent* – It splits the training samples into mini-batches using both the above methods. It revises each batch to strike a computational efficiency of both the above methods.

The gradient descent technique has higher accuracy compared to other techniques when combined with the STA-LSTM algorithm with hyperparameter tuning as depicted in Table II.

**TABLE II. OPTIMIZATION METHODS AND HYPERPARAMETER TUNING VALUES**

| | Hyperparameters | | | | | Accuracy |
|---|---|---|---|---|---|---|
| | **Epoch** | **Batch size** | **Optimizer** | **Train: Validation: Test split ratio** | **LSTM layers** | |
| **Gradient descent** | 200 | 96.94 % | Adam | 0.7:0.2:0.1 | 25 | 96.94 % |
| **Stochastic Gradient descent** | 100 | 95.16 % | Adam | 0.7:0.2:0.1 | 39 | 95.16 % |
| **Mini batch** | 150 | 94.92 % | Adam | 0.7:0.2:0.1 | 47 | 94.92 % |

### *D.* Performance Analysis of Trajectory Prediction

The optimized STA-LSTM algorithm is compared over a physics-based model, the basic LSTM model, and the STA-LSTM model. The results are evaluated using confusion matrix as detailed below.

*a) Confusion Matrix:* It summarizes the functioning of the learning algorithms in terms of true and false positives and negatives viceversa. An example of a confusion matrix is shown in Fig 5.

True Positive: The total real positives which are identical to the positives predicted.

False Positive: The quantity of erroneously predicted negatives as positives.

True Negative: The total real negative values are identical to negatives predicted.

*e) Specificity:* Quantity of real negatives that have been projected as negative.

Sensitivity = TN/(TN+FP)

**TABLE III. STA-LSTM VS OPTIMIZED LSTM**

| METRICS | STA-LSTM | PROPOSED WORK | | |
|---|---|---|---|---|
| | | Optimization in Spatial weight mechanism (A) | Hyperparameters tuning + Gradient Descent Optimization (B) | Optimized model (C=A+B) |
| Accuracy | 94.767295360565 | 96.641935372 | 96.941861212253 | 96.952935372 |
| RMSE | 0.6505923946246 | 0.5540134226 | 0.5530044111710 | 0.5342134226 |
| Sensitivity | 0.3043478260869 | 0.6026723 | 0.625 | 0.6279053 |
| Specificity | 0.7173913043478 | 0.734570642 | 0.7586206896 | 0.7587203524 |

**TABLE IV. COMPARISON OF PERFORMANCE METRICS FOR DIFFERENT ALGORITHMS**

| Metric | Physics-Based Model | STA-LSTM | Optimized STA-LSTM |
|---|---|---|---|
| Accuracy (%) | 91.2435 | 94.7673 | 96.9419 |
| RMSE | 1.0888 | 0.6506 | 0.5530 |
| Sensitivity | 0.1267 | 0.3043 | 0.6250 |
| Specificity | 0.3278 | 0.7174 | 0.7586 |

The optimized STA-LSTM algorithm's accuracy has been improved by 2.25% compared to the STA-LSTM algorithm as depicted in Table III. The RMSE, sensitivity and specificity has been improved with respect to the STA-LSTM algorithm by 15.4%, 51.36%, 5.4% respectively. The proposed model performs better than the other models, with its graph being the closest one to replicating the actual trajectory as shown in Fig.6.

The comparison of the physics-based model, STA-LSTM, and the optimized STA-LSTM algorithm in terms of the validation metrics are shown in Table IV. The physics-based model is an implementation based on only linear mathematical equations. The STA-LSTM algorithm has been implemented by following the paper [10]. The optimized STA-LSTM algorithm is developed using gradient descent optimization and hyperparameter tuning. Clearly, the accuracy, sensitivity, and specificity values are improved implying greater learning and response to changes in inputs. The decrease in RMSE indicates the increase in the correct number of predictions and better training.

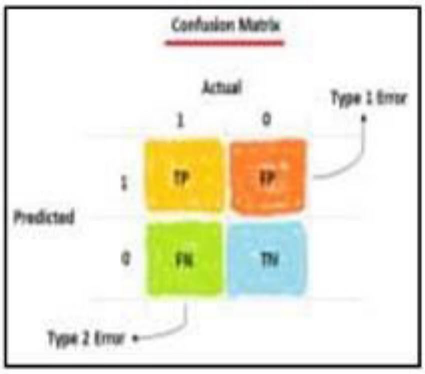



Fig 5: Confusion matrix

False Negative: The quantity of mistakenly predicted negatives as positives.

*b) Accuracy:* It gives the relation between the right predictions to the entire predictions.

ISSN 2394 - 9554

Accuracy = (TP+TN)/(TP+TN+FP+FN)

*c) RMSE:* Standard deviation of the errors which occur when a prediction is made on a dataset.

*d) Sensitivity:* Quantity of real positives that have been projected as positive.

Sensitivity = TP/(TP+FN)

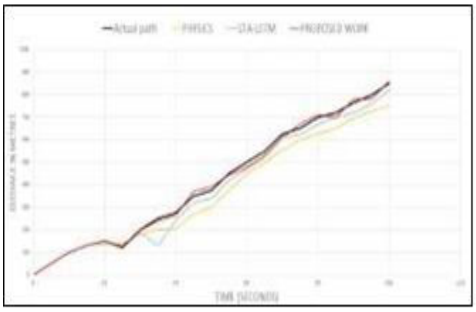


Fig 6 Trajectory Comparison

*f) Loss VS Epoch:* The cost or loss function plays a prominent role in refining the features to a single digit. Epoch indicates the total passes of the algorithm over the training data. Loss indicates the mean squared error over projected and the real trajectory of the vehicle.

From the graph in Fig 7, it is clear to infer that as the epoch increases loss decreases. The loss value is minimized to 0.378 as the epochs is set to 200. These graphs can show if the model has over- learned, under-learned, or is well-fit to the training data. Here, the model is perfectly tuned to get the best possible accuracy and learning.

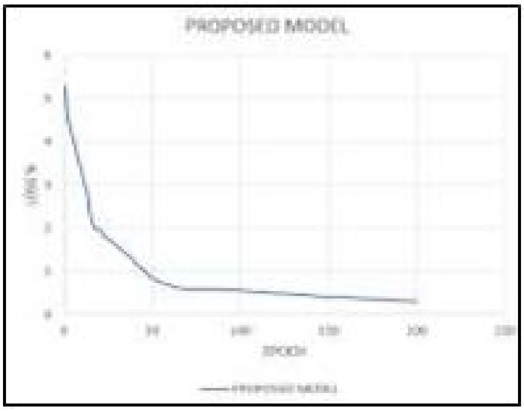
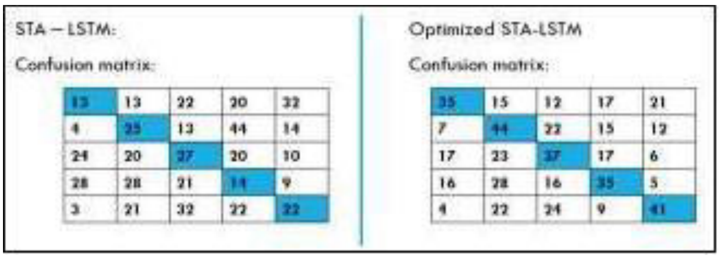



Fig 7. Loss Vs Epoch of Optimized STA-LSTM Algorithm

*g) Confusion Matrix comparison:* Fig 8 shows the comparison of the confusion matrix of the STA-LSTM algorithm and the optimized STA-LSTM algorithm. The diagonal value of the confusion matrix indicates the correct of predictions out of the total test cases. The higher the value of the test cases in the diagonal, the higher the performance accuracy of the corresponding algorithm. This is clearly shown in the confusion matrix below.



STA – LSTM:

Confusion matrix:

| | | | | |
|---|---|---|---|---|
| 13 | 13 | 22 | 20 | 32 |
| 4 | 25 | 13 | 44 | 14 |
| 24 | 20 | 37 | 20 | 10 |
| 28 | 28 | 21 | 14 | 9 |
| 3 | 21 | 32 | 22 | 22 |

Optimized STA-LSTM

Confusion matrix:

| | | | | |
|---|---|---|---|---|
| 35 | 15 | 12 | 17 | 21 |
| 7 | 44 | 22 | 15 | 12 |
| 17 | 23 | 37 | 17 | 6 |
| 16 | 28 | 16 | 35 | 5 |
| 4 | 22 | 24 | 9 | 41 |

Fig 8. Confusion matrix comparison

*h) Output labels for Lane Changing:*

The output of the optimized STA-LSTM trajectory prediction algorithm is a series of five values from 0 to 4. Each of these values indicates the recommended trajectory of the vehicle considering the neighboring influences, speed, braking, lane position, etc. The label for each of these values is given in Table V.

TABLE V. OUTPUT LABELS OF LANE CHANGING ALGORITHM

| Output Labels | Response |
|---|---|
| 0 | Follow |
| 1 | Change to left lane |
| 2 | Overtake leading vehicle by switching to left lane |
| 3 | Change to right lane |
| 4 | Overtake leading vehicle by switching to right lane |

The decisions taken by the optimized STA-LSTM algorithm for a series of input data is shown in Fig 9. The Y-axis indicates the labels shown above and X-axis represents the time steps synchronized to the timestamps in the dataset.

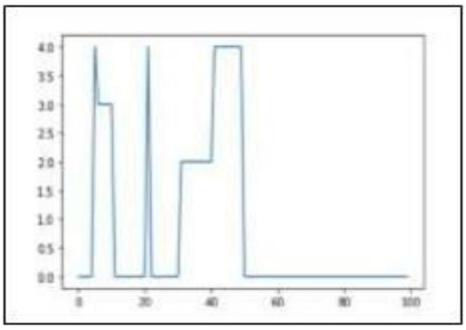



Fig 9. Output decisions by STA-LSTM algorithm

*E. Results of autonomous driving systems*

Autonomous driving requires trajectory prediction and lane detection as two main components.

*a) Hardware Implementation:*The following figure shows the connection diagram of the Raspberry Pi microcontroller to the servo motor as shown in Fig. 10. The servo motor is attached with the steering wheel to simulate the steering wheel of a car. The steering wheel turns automatically when the offset is calculated i.e., when the trajectory is predicted.

Fig 10. Hardware used for autonomous driving

*b) Lane detection:* Lane detection algorithm based on image processing is implemented using OpenCV and python. Cannny edge detection and houghman transform combinely used to detect the edge in the image. The detected lane is shown in Fig 11, where the lane lines have been identified and highlighted in blue colour.

Fig 11. Lane detection

 ISSN 2394 - 9554

*c) Turn Prediction:* For predicting the vehicle's heading, the lane's middle point is calculated by combining the positions of the left and right lanes. Then, to anticipate the lane turn, we utilize the location of the vehicle's center, i.e., the image's center. We compute the difference between the lane center and the image center, and if it is positive, we can properly forecast the lane's right turn; if it is negative, we can detect the lane's left turn as shown in Fig 12 and 13.

This technique can adapt to a wider range of lane configurations than previous lane recognition models, and its robustness comes from the fact that accurate road marking information is retrieved in a wide range of road circumstances while remaining robust and effective.


Fig 12. The curve direction is identified as left and the highlighted section indicates the lane that the target vehicle should follow

Fig 13. The curve direction is identified as right, and the highlighted section indicates the lane that the target vehicle should follow

*c) Steering wheel response:* Fig 14, 15, and 16 show the response of the steering wheel in accordance with the predicted trajectory and lane curvature as straight, right, and left turns respectively. The steering wheel offset is calculated, and the output is fed to the servo motor by the raspberry pi controller to turn the steering wheel of the vehicle in the appropriate direction.


Fig 14. The straight path is identified and the steering wheel responds correspondingly

Fig 15. The left curve is being identified and the steering wheel responds correspondingly (towards left).

Fig 16. The right curve is being identified and the steering wheel responds correspondingly (towards the right).

## V. CONCLUSION

The proposed optimization for STA-LSTM based on gradient descent gives best prediction accuracy when combined with Adam optimizer for epoch value of 200, batch size of 240, and 25 LSTM layers. The gradient descent technique when applied separately yields an accuracy of 96.642%, and when combined with the hyperparameter tuning gives an accuracy of 96.953%. The optimized algorithm minimized the loss function to 0.378 when epoch is set to

200. The autonomous steering system based on raspberry pi microcontroller and servo motor controls the angle of steering. By combining trajectory and lane prediction for various cases.

## REFERENCES

[1] C. S. Evangeline, V. B. Kumaravelu and A. Joshi, "Safety and Driver Assistance in VANETs: An Experimental Approach for V2V," 2019 International Conference on Communication and Electronics Systems (ICCES), 2019, pp. 397-402, doi: 10.1109/ICCES45898.2019.9002562.

[2] Baek, Minjin, Donggi Jeong, Dongho Choi, and Sangsun Lee. 2020. "Vehicle Trajectory Prediction and Collision Warning via Fusion of Multisensors and Wireless Vehicular Communications" Sensors 20, no. 1: 288. https://doi.org/10.3390/s20010288.

[3] K. Messaoud, N. Deo, M. M. Trivedi and F. Nashashibi, "Trajectory Prediction for Autonomous Driving based on Multi-Head Attention with Joint Agent-Map Representation," 2021 IEEE Intelligent Vehicles Symposium (IV), 2021, pp. 165-170, doi: 10.1109/IV48863.2021.9576054.

[4] H. Cheng, W. Liao, X. Tang, M. Y. Yang, M. Sester and B. Rosenhahn, "Exploring Dynamic Context for Multi-path Trajectory Prediction," 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021, pp. 12795-12801, doi: 10.1109/ICRA48506.2021.9562034.

[5] Chen, Y., Yu, H., Zhang, J. et al. “Lane-Exchanging Driving Strategy for Autonomous Vehicle via Trajectory Prediction and Model Predictive Control”. Chin. J. Mech. Eng. 35, 71 (2022). https://doi.org/10.1186/s10033-022-00748-7

[6] P. Rathore, D. Kumar, S. Rajasegarar, M. Palaniswami and J. C. Bezdek, "A Scalable Framework for Trajectory Prediction," in IEEE Transactions on Intelligent Transportation Systems, vol. 20, no. 10, pp. 3860-3874, Oct. 2019, doi: 10.1109/TITS.2019.2899179.

[7] M. Fu, T. Zhang, W. Song, Y. Yang and M. Wang, "Trajectory Prediction-Based Local Spatio-Temporal Navigation Map for Autonomous Driving in Dynamic Highway Environments," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 7, pp. 6418-6429, July 2022, doi: 10.1109/TITS.2021.3057110.

[8] J. Gao, J. Yi and Y. L. Murphey, "A Lane-Changing Detection Model Using Span-based Transformer," 2021 33rd Chinese Control and Decision Conference (CCDC), 2021, pp. 2733-2738, doi: 10.1109/CCDC52312.2021.9601457.

[9] Y. Jeong, "Predictive Lane Change Decision Making Using Bidirectional Long Shot-Term Memory for Autonomous Driving on Highways," in IEEE Access, vol. 9, pp. 144985-144998, 2021, doi: 10.1109/ACCESS.2021.3122869.

ISSN 2394 - 9554

[10] Deo, Nachiket and Mohan Manubhai Trivedi. "Convolutional Social Pooling for Vehicle Trajectory Prediction." 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2018): 1549-15498.

[11] L. Lin, W. Li, H. Bi and L. Qin, "Vehicle Trajectory Prediction Using LSTMs With Spatial–Temporal Attention Mechanisms," in IEEE Intelligent Transportation Systems Magazine, vol. 14, no. 2, pp. 197-208, March-April 2022, doi: 10.1109/MITS.2021.3049404.

[12] S. A. Alamsyah, D. Purwanto and M. Attamimi, "Lane Detection Using Edge Detection and Spatio-Temporal Incremental Clustering," 2021 International Seminar on Intelligent Technology and Its Applications (ISITIA), 2021, pp. 364-369, doi: 10.1109/ISITIA52817.2021.9502232.

[13] Danjue Chen, Anupam Srivastava, Soyoung Ahn, Tienan Li, "Traffic dynamics under speed disturbance in mixed traffic with automated and non-automated vehicles," Transportation Research Procedia, Volume 38, 2019, 709-729, ISSN 2352-1465.https://doi.org/10.1016/j.trpro.2019.05.037.

[14] NGSIM Dataset (website): https://www.opendatanetwork.com/dataset/datahub.transportation.gov/8ect-6jqj.

[15] L. S. Riza, I. F. Nasrulloh, E. Junaeti, R. Zain and A. B. D. Nandiyanto, "gradDescentR: An R package implementing gradient descent and its variants for regression tasks," 2016 1st International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE), 2016, pp. 125-129, doi: 10.1109/ICITISEE.2016.7803060.

[16] X. Peng, L. Li and F. -Y. Wang, "Accelerating Minibatch Stochastic Gradient Descent Using Typicality Sampling," in IEEE Transactions on Neural Networks and Learning Systems, vol. 31, no. 11, pp. 4649-4659, Nov. 2020, doi: 10.1109/TNNLS.2019.2957003.