Received: 12 April 2025 | Revised: 11 July 2025 | Accepted: 5 August 2025 | Published online: 16 September 2025

RESEARCH ARTICLE

Journal of Computational and Cognitive Engineering
2025, Vol. 00(00) 1-8
DOI: 10.47852/bonviewJCCE52025911

BON VIEW PUBLISHING

Spatio-temporal Learning for Trajectory Prediction in Full Self-Driving Cars

Sreenivasa Chakravarthi Sangapu[1,*], Amirthavarshini Venkadesh[1], Jayant Toleti[1], and Sountharrajan Sehar[1]

[1] *Department of Computer Science and Engineering, Amrita Vishwa Vidyapeetham Chennai, India*

Abstract: Autonomous driving technology is advancing quickly. Full self-driving systems aim to navigate safely in complex and busy traffic conditions. A key challenge for these systems is predicting how nearby vehicles will move. Accurate trajectory prediction helps the vehicle make better decisions. It improves motion planning and reduces the risk of collisions. This study introduces a deep learning model that improves trajectory forecasting. The model uses convolutional neural networks (CNNs) to learn spatial features from the traffic scene. It also uses long short-term memory (LSTM) networks to understand how vehicle movement changes over time. Together, these methods help the system predict future paths more effectively. The model was tested in various real-world traffic scenarios. It performed well across different conditions and showed strong accuracy in its predictions. These results show that combining CNN and LSTM networks can help autonomous vehicles better understand their surroundings. This approach supports safer and smarter decision-making on the road.

Keywords: autonomous driving systems, trajectory prediction, short-term prediction, long short-term memory (LSTM), convolutional neural network (CNN)

1. Introduction

Recent years have witnessed transformative advancements in AI and full self-driving (FSD) systems, particularly in the transportation sector [1, 2]. Modern FSD technology empowers vehicles to perceive their environment, make critical decisions autonomously, and navigate complex real-world scenarios with minimal human intervention. Breakthroughs in computer vision, reinforcement learning, and deep learning have significantly enhanced these systems' capabilities, enabling safer operation in dynamic conditions [3, 4].

However, deploying autonomous systems in real-world environments presents numerous challenges. Among these, trajectory prediction emerges as a pivotal capability—the system's ability to anticipate future movements of surrounding vehicles, pedestrians, cyclists, and other agents [5, 6]. This function is indispensable for safe navigation, directly affecting collision avoidance, smooth interaction with road users, and precise maneuver execution. Beyond safety, accurate trajectory prediction improves traffic flow, reduces energy consumption, and facilitates cooperative behaviors among autonomous systems [7]. Its applications span self-driving cars, service robots in crowded spaces, and intelligent surveillance, making robust real-time prediction crucial across domains [8].

To address this challenge, researchers have developed diverse modeling approaches. Although traditional physics-based and rule-driven models offer interpretability, they often fail to capture the complexity of human behaviors, which are inherently nonlinear and socially influenced. In contrast, data-driven deep learning methods—including convolutional neural networks (CNNs), RNNs, long short-term memory (LSTM), and Transformer-based architectures—demonstrate superior ability to learn intricate motion patterns directly from data. These models adapt effectively to varying contexts, delivering enhanced predictive performance in highly dynamic environments.

***Corresponding author:** Sreenivasa Chakravarthi Sangapu, Department of Computer Science and Engineering, Amrita Vishwa Vidyapeetham Chennai, India. Email: ss_chakravarthi@ch.amrita.edu

Despite their advancements, current trajectory prediction models still face critical limitations. Real-world driving scenarios are inherently unpredictable, demanding systems that can generalize across diverse environments and reason effectively under uncertainty. Accurate forecasting relies not only on historical motion patterns but also on contextual understanding—including road semantics, traffic regulations, and the inferred intentions of surrounding agents. To address these challenges, recent research has focused on integrating spatio-temporal modeling with attention mechanisms and multimodal data fusion, enhancing models' contextual awareness. These hybrid approaches represent a crucial evolution toward more robust and scalable autonomous navigation systems capable of handling real-world complexity.

This paper introduces a light but expressive deep learning design to address some of the critical limitations of the existing trajectory prediction models for autonomous vehicles. Although previous work has attempted CNN–LSTM hybrids, the model presented here develops specific architectural novelties that enhance both predictive performance and real-time feasibility. In particular, a two-stage approach is taken, wherein 1D CNNs are employed to learn local spatial motion features from temporal input sequences, followed by stacked LSTM layers that capture temporal relationships. One novelty is the application of axis-specific linear decoders for both X and Y coordinates, allowing the model to learn directionally unique motion behaviors, a highly applicable property in dynamic urban scenarios where lateral and longitudinal movements have disparate trends.

In addition, uncertainty-aware evaluation through the application of Brier scores and common trajectory metrics, including average displacement error (ADE), final displacement error (FDE), and miss rate (MR), was employed to provide a more safety-focused evaluation of model performance. Implementation was developed based on NuScenes data, utilizing real-world signals such as velocity, acceleration, and heading to enrich inputs. A well-crafted preprocessing pipeline alleviates the effects of noise and outlier artifacts typical in dynamic traffic data. As opposed to computationally costly Transformer-based or heavy networks with attention, the design is edge-optimized, balancing performance and efficiency most ideally suited for real-time autonomous driving scenarios where low-latency inference is crucial.

2. Literature Review

Over the past few years, there has been a significant advancement in the field of autonomous systems such as advanced decision-making and perception that help in navigating complex environments safely and efficiently. However, one of their main challenges lies in generating trajectories that are optimal in terms of safety, length of path, and time taken and are adaptable to the dynamically changing environment, which can have moving obstacles, varying traffic conditions, and unpredictable human behavior. Traditional approaches [9] rely on preplanned trajectories, which is effective only in static environments and is inefficient for complex real-world scenarios. Thus, there is a need for dynamic, sequential models that can adapt to changing conditions. Karaman et al. introduced PRM* and RRT*, ensuring asymptotic optimality in path planning [10].

Reda et al. [11] categorized traditional trajectory planning techniques into four types, namely, graph, gradient, sampling, and optimization-based methods. Methods based on graphs use graph theory and algorithms such as Dijkstra's and A* search to find the shortest possible path in the graphical representation depicting the vehicle's environment [12]. Some advanced graph-based methods use bio-inspired algorithms such as the improved seagull optimization algorithm for high-quality path computation [13]. However, graph-based methods fail to perform well in computationally intensive scenarios when the graph density increases.

Rapidly exploring random trees (RRT) and probabilistic roadmaps (PRM) are examples of sampling methods. These are effective for high-dimensional spaces and generate feasible paths. However, they rely on static environments [14]. Gradient-based methods use techniques such as gradient descent and optimization-based methods use techniques such as model predictive control (MPC) to calculate the trajectory based on a cost function. These methods are computationally intensive and do not adapt quickly to a dynamic environment [15]. The limitation of preplanned paths in dynamic environments, as mentioned by Machavaram [16], is largely due to obstacles, adverse traffic conditions, and unpredictability in other agent's behaviors [17].

Some models such as time-optimal trajectory planning and tracking use a layered structure for control. This is usually used with an offline trajectory optimization module and an online NMPC module, which accurately tracks the trajectories in reference [18]. Zhao et al. [18] proposed a quintic polynomial-based method to generate smooth trajectories. These help in generating trajectories and tracking these trajectories in real-time but face challenges in adaptability.

These limitations highlight the need for dynamic, sequential-based models that can continuously update the trajectory based on real-time sensor data and environmental feedback [19]. Deep reinforcement learning (DRL) [20] has introduced more data-driven and adaptive approaches for trajectory planning. It combines data-dependent applications with control systems to enhance trajectory planning and dynamic control [21]. DRL uses techniques such as a deep Q-network (DQN) to train an agent to engage the environment through a hit-and-miss-based approach to learn optimal driving policies [22, 23].

DQN guides the agent to opt for an optimal strategy to act by observing the expected cumulative reward of state-action pairs. Other DRL-based methods such as proximal policy optimization [24] aide decision-making by improving and optimizing the policy instead of approximating the value function using objective functions to ensure stability and policy improvements [25]. Hoel et al. demonstrated that DRL is used for decision-making tasks such as speed and lane changes through a DQN agent [26]. Beyond traditional DRL, game-theoretic approaches [27] have shown promise in UAV networks, optimizing trajectories through AoI minimization for latency-sensitive decisions in dynamic environments.

Al-Kamil et al. suggested motion capture technology to enhance real-time data for optimal path planning and control in mobile robot systems. This technology uses sensors such as camera and LiDAR to measure and monitor the position and movement of the robot. However, it falters when the robot is put in a situation with multiple vehicles [28].

Heuristic algorithms such as genetic algorithms, particle swarm optimization, gray wolf algorithm, ant colony algorithm, and differential evolution algorithm have also been studied for path planning optimization but are not very effective for dynamic environments. Such methods provide suboptimal solutions but lack adaptability, and they are complex computationally [29, 30].

Large language models (LLMs) have enhanced the trajectory prediction capabilities for autonomous systems [31]. Various traditional ML and DL models such as decision trees, random forest, SVMs, RNNs, CNNs, and LSTMs are being used to learn the behaviors and predict traffic patterns for better path planning [32, 33].

Li et al. [34] introduced a large language driving assistant to interpret and adapt to local traffic rules and driving policies. It uses a pretrained LLM, GPT-4V, to generate an executable policy by extracting relevant traffic rules using a traffic rule extractor. Other methods used by the models to learn traffic rules are in the form of metric temporal logic, linear temporal logic, and signal temporal logic. Although they are good for training the model, these methods face scalability issues when trained on a vast number of rules.

The research for the application of vision-language models (VLMs) alongside LLMs for enhanced scene understanding and path planning in autonomous systems has been expanded recently. As demonstrated by Tian et al. [35], their DriveVLM framework utilizes a chain-of-thought reasoning process to systematically analyze critical scene elements and generate hierarchical plans through comprehensive scene description and analysis. Complementing this work, Pan et al. [36] developed a vision-language-planning module that incorporates bird's eye view feature maps to strengthen semantic representation and reasoning, specifically designed to align path planning with both driving objectives and real-time vehicle status. For security-critical scenarios, RSSI-aware RL methods [37] demonstrate how autonomous systems can dynamically adapt trajectories to avoid malicious interference, highlighting the importance of real-time environmental sensing.

For trajectory prediction, LSTM networks remain particularly effective due to their established performance in processing sequential data. Sun et al. [38] proposed an integrated LSTM–MPC approach, where LSTM networks predict surrounding vehicle trajectories that subsequently inform the MPC's path planning decisions for the ego vehicle. However, current implementations of LLM-based models exhibit several limitations in practical applications, including suboptimal real-time performance, limited adaptability compared to conventional algorithms, excessive dependence on heuristic functions, and significant scalability constraints.

Although traditional trajectory planning methods have formed the foundation of autonomous navigation systems, their effectiveness diminishes in highly dynamic environments. This limitation emphasizes the critical need for more adaptable, sequence-based modeling approaches. The integration of heuristic methods with modern deep learning techniques presents a promising direction for developing robust trajectory planning systems capable of handling the complexities of real-world driving scenarios.

3. Working Methodology

3.1. Dataset preprocessing

NuScenes is an image-based large-scale autonomous driving benchmark dataset that supports research in perception and planning for autonomous cars [39]. It provides multimodal sensor data for over 1000 diverse scenes collected in urban environments in Boston and Singapore. Each scene spans 20 s and contains multiple annotated objects, including vehicles, pedestrians, and cyclists. This work is primarily focused on vehicle trajectory prediction, extracting approximately 11,000 vehicle trajectories. Each trajectory was segmented into overlapping windows comprising 8 historical frames (past 0.8 s) and 12 future prediction steps (next 1.2 s). After preprocessing and filtering out inconsistent or noisy samples using statistical techniques such as Z-score-based outlier detection, the dataset was split into training (70%), validation (15%), and testing (15%) subsets, ensuring a representative distribution of different scene types such as intersections, straight roads, and turning maneuvers. The dataset chosen for this study consists of multidimensional time-series data representing object trajectories. Each trajectory comprises spatial coordinates (x, y), along with motion-related features such as speed, acceleration, and heading rate. The reason for choosing this over other datasets is that NuScenes provides a more accurate localization that cannot be found in similar FSD datasets such as KITTI [40]. Such localization systems are vulnerable to GPS outages. More importantly, to facilitate this study, the NuScenes dataset provides raw CAN bus data such as, velocities, accelerations, torque, steering angles, wheel speeds.

All annotations of the dataset can be represented as a schema displayed in Figure 1. Out of all of these, only some parts of the dataset are required. The focus is on the *translation*, *instance_token*, and *attribute_tokens*. The translation attribute will give bounding box locations in meters as *center_x*, *center_y*, and *center_z*. The *instance_token* and the *attribute_token* help in fetching the velocity, acceleration, and heading.

The dataset extraction flowchart is shown in Figure 2. Given the temporal nature of the data, they are structured in sequential windows, allowing the model to identify spatio-temporal relations effectively. The data were thoroughly cleaned and formatted to accommodate sudden jumps, or unrealistic movements (e.g., instantaneous large displacements) were detected using statistical thresholding (e.g., Z-score method) and removed to avoid skewing the model's learning.

Figure 1
NuScenes dataset schema

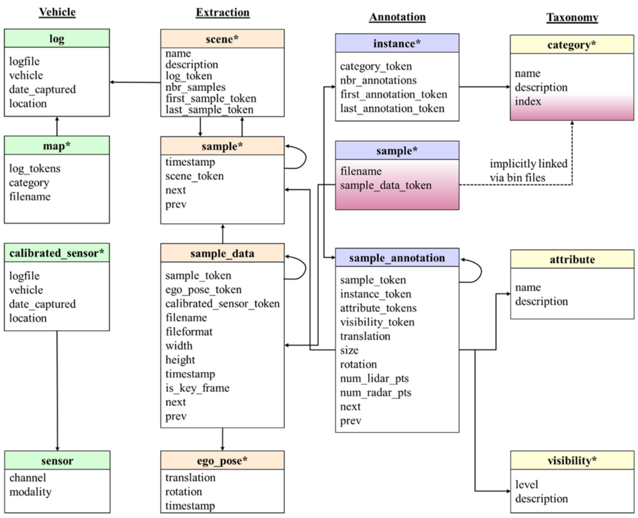
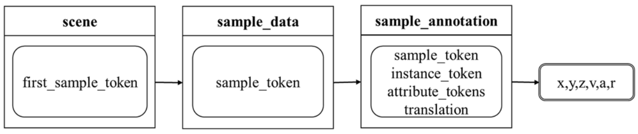


Figure 2
Dataset extraction flow



To prepare the trajectory data for training, a sliding window mechanism was applied to have overlapped time windows. This assists with the extraction of temporal dependencies [41].

3.2. Model architecture

A fundamental challenge in deep learning involves balancing effective feature extraction with temporal sequence modeling. Although CNNs demonstrate exceptional capability in detecting local spatial patterns, they inherently lack mechanisms to capture long-term temporal dependencies [42–44]. Conversely, LSTM networks excel at modeling sequential relationships but are inefficient at learning spatial hierarchies [45]. To bridge this gap, researchers have proposed a CNN–LSTM hybrid architecture that synergistically combines convolutional layers for spatial feature extraction with LSTM networks for temporal sequence learning [46].

As shown in Figure 3, the proposed architecture follows a two-stage spatio-temporal framework designed to balance predictive accuracy with real-time performance. The first stage involves spatial feature extraction using two 1D convolutional (Conv1D) layers, each followed by ReLU activation and max pooling. The first Conv1D layer uses 64 filters with a kernel size of 3, and the second expands to 128 filters to capture higher-level spatial abstractions. These layers operate on input sequences comprising motion features such as velocity, acceleration, and heading, derived from the NuScenes CAN bus data. Max pooling serves to reduce noise and dimensionality, making the subsequent learning more efficient. The resulting feature maps are flattened and passed into the temporal modeling stage.

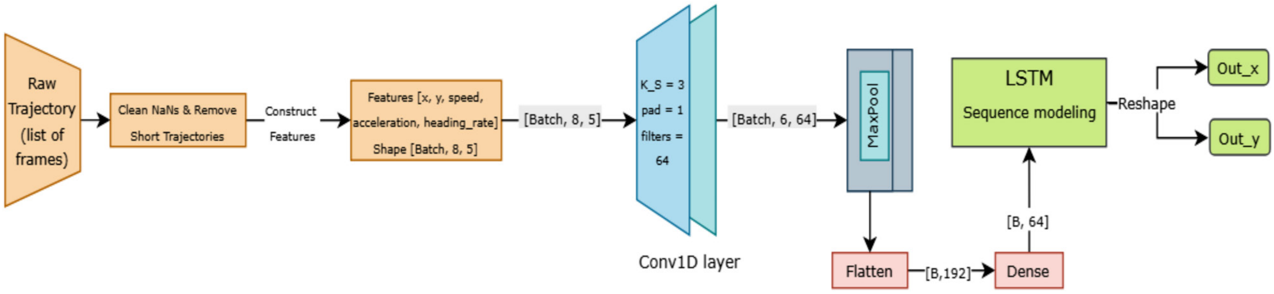
The second stage uses a stacked LSTM architecture with two layers: the first with 128 hidden units and the second with 64 units. This setup enables the model to learn sequential dependencies and encode temporal patterns in vehicle behavior. To improve generalization, dropout regularization (p = 0.3) is applied between layers. The final LSTM output is fed into two parallel fully connected layers: one for the X-coordinate and one for the Y-coordinate prediction. This axis-specific decoding allows for more precise modeling of lateral and longitudinal motion—an essential feature for trajectory prediction in dynamic urban environments. The model is trained using mean squared error (MSE) loss and optimized with the Adam optimizer. In addition, model confidence is evaluated using Brier scores, offering a safety-aware perspective often missing in traditional CNN–LSTM approaches. This lightweight yet effective architecture is well suited for real-time FSD systems that operate under latency and computational constraints.

3.3. Training, testing, and validation

Training of the CNN–LSTM model involved the MSE as the loss function, with optimization carried out using the Adam optimizer. It was trained with a learning rate starting at 0.001. To prevent overfitting and adapt the learning process dynamically, a ReduceLROnPlateau learning rate scheduler was used in the training process, reducing the learning rate by a factor of 0.5 when the validation loss plateaued for two consecutive epochs.

The hybrid model was trained for 12 epochs, processing sequential input windows of 8-time steps (SEQ_LENGTH) and predicting future values over a 12-step horizon (PREDICTED_LENGTH). The training

Figure 3
CNN + LSTM architecture diagram



process iteratively minimized the loss using batch-wise optimization, where performance metrics such as training loss and validation loss were logged after each epoch.

To evaluate the model's generalization capabilities, comprehensive testing using a separate validation dataset containing previously unseen trajectories was conducted. The evaluation process involved preprocessing input sequences through min–max scaling before feeding them to our trained CNN–LSTM model. The model's predictions were then converted back to their original scale using inverse transformation, enabling accurate comparison with ground-truth trajectory data. For qualitative assessment, a visualization method that plots the model's predicted trajectories alongside actual paths was implemented. This provided clear insight into its ability to capture and replicate real-world motion patterns. This approach allows the thorough examination of the model's performance in learning and predicting complex movement dynamics.

4. Result Analysis

4.1. Metrics

Assessing trajectory prediction models requires comprehensive metrics that quantify accuracy, reliability, and predictive uncertainty [47]. In addition, to evaluate the acceptability and diversity of the trajectories, the following measures are necessary as they define compliance at many levels, including road and kinematic compliancy [48–52].

MR: Calculates the percentage of test cases where none of the predicted trajectories fall within a 2-m threshold of the actual final position. This reveals how often the model completely misses the target destination.

Minimum FDE (minFDE): Measures the straight-line distance (L2 norm) between the ground truth endpoint and the closest predicted endpoint among all forecasted trajectories. This identifies the model's best-case positional accuracy.

Minimum ADE (minADE): The minADE computes the mean L2 distance between the points in ground truth and the overall best trajectory (i.e., the one with the lowest endpoint error) across all time steps.

Brier Scores for Minimum FDE (brier-minFDE): The brier-minFDE is a variation of minFDE, incorporating uncertainty by penalizing low-probability predictions.

Brier Scores for Minimum ADE (brier-minADE): The brier-minADE extends minADE by incorporating uncertainty penalties similar to brier-minFDE [53].

The chosen evaluation metrics provide a balanced assessment of both spatial accuracy and predictive reliability. Metrics such as minADE and minFDE capture the model's ability to produce accurate short- and long-term trajectory estimates, and MR highlights failure cases that are critical in safety-sensitive applications such as autonomous driving. To go beyond deterministic accuracy, Brier scores were incorporated to evaluate the model's confidence calibration, offering a probabilistic lens on performance. This combination ensures that the evaluation reflects not only how close predictions are to ground truth but also how trustworthy and robust they are under uncertainty.

4.2. Results

To visualize the results of the model, the prediction results are extracted and mapped as x–y coordinates on a plane. This approach allows us to compare the predicted trajectory against the actual movement of the vehicle, providing insights into the model's accuracy and effectiveness. In the visualization, different colored paths represent distinct aspects of the vehicle's trajectory:

Green Path: Represents the ego vehicle's historical movement before prediction begins. This provides essential context for both the model's input and the viewer's interpretation.

Blue Path: Shows the actual future trajectory (ground truth) after the prediction point, serving as the benchmark to evaluate the model's accuracy against the predicted path.

Red Path: Displays the model's predicted trajectory, forecasting the next 12 positional points based on learned motion patterns and behavioral cues.

As illustrated in Figure 4, the model's predicted trajectory (red path) can be directly compared against the ground truth (blue path) to evaluate its performance. An effective model will generate predictions that closely follow the actual trajectory, demonstrating its ability to minimize positional deviation and accurately capture the vehicle's natural movement patterns. Significant divergences between the paths, however, may reveal specific challenges, such as difficulty predicting sharp turns, abrupt stops, or responses to environmental factors not adequately represented in the training data. This comparative visualization provides valuable insights into the model's strengths and limitations. The visual feedback becomes an essential tool for iteratively refining both the model's design and its training methodology.

From the results shown in Table 1 and Table 2, it can be inferred that the CNN + LSTM with LinearX, LinearY model outperforms all other models across all metrics. This suggests that combining CNN and LSTM is effective in capturing both spatial and temporal dependencies. Using separate LinearX and LinearY layers further improves performance, likely because it allows for better independent modeling of x and y coordinates.

The LSTM-only models perform worse, suggesting that LSTMs alone are not as effective in learning spatial features. Their predictions are less accurate (higher ADE and FDE) and more uncertain (higher Brier scores). Overall, the CNN layer does a good job of capturing spatial patterns, which helps with better feature extraction. In addition, using separate linear layers for X and Y seems to help in modeling their trajectories more independently and accurately.

Figure 4
Mapping the prediction on a plane revealing the accuracy of the model and potential for improvement

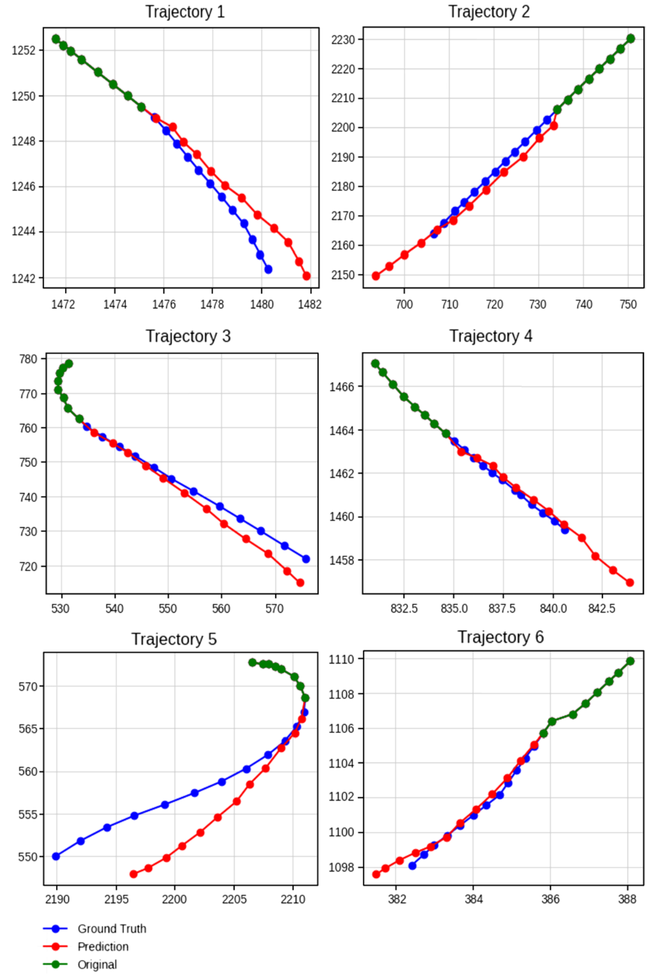


Table 1
Metrics results of the LSTM model

Metrics	LSTM (LinearXY)	LSTM (LinearX, LinearY)
avg_ade	3.221	3.210
avg_fde	3.985	3.772
avg_mr	2.766	2.246
avg_brier_minADE	3.294	3.197
avg_brier_minFDE	3.873	3.687

To isolate the impact of each architectural component, an ablation study across four model configurations was conducted.

1) LSTM (LinearXY) – Baseline with shared output layer.
2) LSTM (LinearX, LinearY) – Axis-specific outputs.
3) CNN + LSTM (LinearXY) – Hybrid model with shared output.
4) CNN + LSTM (LinearX, LinearY) – Proposed full model.

This ablation analysis highlights the distinct advantages offered by each architectural enhancement in our proposed model. The introduction of CNN layers significantly improves the model's ability to extract localized spatial features from vehicle-centric data, such as sudden positional shifts, lane changes, or turning patterns. Traditional

Table 2
Metrics results of the CNN + LSTM model

Metrics	CNN + LSTM (LinearXY)	CNN + LSTM (LinearX, LinearY)
avg_ade	1.934	1.097
avg_fde	3.132	2.060
avg_mr	1.845	0.765
avg_brier_minADE	1.659	1.077
avg_brier_minFDE	2.617	2.051

LSTM-only models, although competent at modeling temporal sequences, are inherently limited in capturing spatial nuances across consecutive frames. By applying 1D convolutions in the initial stages, our model learns localized motion cues that reflect not only vehicle velocity and heading but also trajectory curvature and relative motion trends. These features provide a more robust representation to the LSTM layers, enabling them to learn temporal dynamics from enriched spatial embeddings.

Moreover, decoupling the output layers into separate linear projections for the X- and Y-axes offers additional flexibility in capturing motion trajectories with directional specificity. This design choice acknowledges that horizontal and vertical displacements in a 2D driving plane may follow different dynamics—for example, lateral maneuvers such as lane shifts often involve sharp, brief changes, whereas longitudinal motion such as acceleration or deceleration is smoother and more sustained. By learning independent mappings for each axis, the model avoids blending these distinct behaviors into a shared latent space, thereby reducing prediction variance and improving final displacement accuracy. The resulting CNN + LSTM configuration with split outputs consistently outperformed all other variants across both accuracy and uncertainty metrics, validating our hypothesis that architectural disentanglement enhances the model's spatio-temporal learning capacity in trajectory forecasting tasks.

In addition to evaluating predictive performance, a benchmarking analysis to assess the real-time viability of the proposed model was conducted. The full CNN–LSTM architecture was tested on an NVIDIA RTX 3060 GPU, and the average inference time per input sequence was measured to be approximately 8.3 ms. This low latency indicates that the model is suitable for onboard edge deployment in FSD systems where rapid decision-making is critical. The model's lightweight structure, which avoids attention mechanisms and transformer layers, contributes significantly to this efficiency. Further performance gains can be achieved through model optimization techniques such as quantization, which reduces numerical precision to speed up inference, and pruning, which removes redundant weights to lower computational overhead. These enhancements are particularly useful for deployment on embedded systems or edge devices with constrained resources, without significantly compromising predictive accuracy.

5. Conclusion

This study proposed a lightweight CNN–LSTM hybrid model for short-term trajectory prediction in FSD systems, addressing the need for real-time, accurate path forecasting in complex urban environments. By integrating 1D convolutional layers for spatial feature extraction with stacked LSTM layers for temporal modeling, the architecture effectively learns dynamic motion patterns from vehicle-centric sensor data. Furthermore, the use of separate linear output layers for X and Y coordinates improves the granularity of the predictions, enabling more accurate modeling of maneuver variations such as sharp turns or lane changes.

Through comprehensive evaluation on the NuScenes dataset, our model demonstrated competitive performance across multiple metrics, including ADE, FDE, and Brier scores. An ablation study confirmed the individual benefits of the CNN and LSTM components, and the contribution of axis-specific decoding. Importantly, the architecture achieves these results while maintaining low computational overhead—making it a viable candidate for deployment on resource-constrained autonomous platforms.

However, the model currently focuses solely on vehicle kinematics and does not incorporate higher-level contextual cues such as road semantics, traffic rules, or interactions with pedestrians and other agents [54, 55]. In addition, the evaluation focuses on average-case performance and lacks a detailed assessment of edge-case or safety-critical scenarios—areas that are vital for real-world autonomous deployment. The model also does not estimate predictive uncertainty beyond Brier scores, which may limit its utility in high-risk decision-making contexts.

To address these gaps, future research will explore the integration of multimodal data sources, including LiDAR, radar, and high-definition map, to improve scene understanding [56, 57]. Future work includes incorporating intent prediction and interaction modeling, potentially through the use of attention mechanisms or graph-based relational encoders. Further, techniques such as Bayesian deep learning and Monte Carlo dropout could be employed to quantify prediction uncertainty. Real-time deployment considerations such as quantization, model pruning, and hardware-specific optimization will be investigated to enhance inference speed without compromising safety [58].

In conclusion, although our CNN–LSTM approach may be seen as an incremental advancement, its modular design, axis-specific decoding, and real-time readiness position it as a strong baseline for practical trajectory prediction in FSD applications. By building upon this foundation with richer context modeling and safety-aware evaluation, the aim is to push toward more robust and trustworthy autonomous navigation systems.

Acknowledgement

The authors are grateful to Amrita School of Computing, Amrita Vishwa Vidyapeetham, Chennai Campus, for providing the infrastructural assistance and perform data preprocessing and analysis in the NEGCES Lab.

Ethical Statement

This study does not contain any studies with human or animal subjects performed by any of the authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

Data Availability Statement

Data are available on request from the corresponding author upon reasonable request.

Author Contribution Statement

Sreenivasa Chakravarthi Sangapu: Conceptualization, Formal analysis, Investigation, Writing – review & editing, Visualization, Supervision, Project administration. **Amirthavarshini Venkadesh:** Methodology, Software, Validation, Data curation, Writing – original draft. **Jayant Toleti:** Methodology, Software, Validation, Formal analysis, Investigation, Resources, Writing – original draft, Visualization. **Sountharrajan Sehar:** Resources, Data curation, Writing – review & editing.

References

[1] Sonko, S., Etukudoh, E. A., Ibekwe, K. I., Ilojianya, V. I., & Daudu, C. D. (2024). A comprehensive review of embedded systems in autonomous vehicles: Trends, challenges, and future directions. *World Journal of Advanced Research and Reviews*, *21*(1), 2009–2020. https://doi.org/10.30574/wjarr.2024.21.1.0258

[2] Garikapati, D., & Shetiya, S. S. (2024). Autonomous vehicles: Evolution of artificial intelligence and the current industry landscape. *Big Data and Cognitive Computing*, *8*(4), 42. https://doi.org/10.3390/bdcc8040042

[3] Fawole, O. A., & Rawat, D. B. (2024). Recent advances in 3D object detection for self-driving vehicles: A survey. *AI*, *5*(3). https://doi.org/10.14569/ijacsa.2024.0150759

[4] Wu, J., Huang, C., Huang, H., Lv, C., Wang, Y., & Wang, F. Y. (2024). Recent advances in reinforcement learning-based autonomous driving behavior planning: A survey. *Transportation Research Part C: Emerging Technologies*, *164*, 104654. https://doi.org/10.1016/j.trc.2024.104654

[5] Wang, B., Lei, H., Shui, Z., Chen, Z., & Yang, P. (2024). Current state of autonomous driving applications based on distributed perception and decision-making.

[6] Xie, J., Zhou, X., & Cheng, L. (2024). Edge computing for real-time decision making in autonomous driving: Review of challenges, solutions, and future trends. *International Journal of Advanced Computer Science and Applications*, *15*(7), 598-607. http://dx.doi.org/10.14569/IJACSA.2024.0150759

[7] Bharilya, V., & Kumar, N. (2024). Machine learning for autonomous vehicle's trajectory prediction: A comprehensive survey, challenges, and future research directions. *Vehicular Communications*, *46*, 100733. https://doi.org/10.1016/j.vehcom.2024.100733

[8] Xing, H., Liu, W., Ning, Z., Zhao, Q., Cheng, S., & Hu, J. (2024). Deep learning based trajectory prediction in autonomous driving tasks: A survey. In *2024 16th International Conference on Computer and Automation Engineering*, 556–561. https://doi.org/10.1109/ICCAE59995.2024.10569771

[9] LaValle, S. M. (2006). *Planning algorithms*. UK: Cambridge University Press.

[10] Karaman, S., & Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, *30*(7), 846–894. https://doi.org/10.1177/0278364911406761

[11] Reda, M., Onsy, A., Haikal, A. Y., & Ghanbari, A. (2024). Path planning algorithms in the autonomous driving system: A comprehensive review. *Robotics and Autonomous Systems*, *174*, 104630. https://doi.org/10.1016/j.robot.2024.104630

[12] Bajwa, A. (2025). AI-based emergency response systems: A systematic literature review on smart infrastructure safety. *American Journal of Advanced Technology and Engineering Solutions*, *1*(1), 20-40. https://doi.org/10.63125/xcxwpv34

[13] Pathak, V. K., Gangwar, S., & Dikshit, M. K. (2025). A comprehensive survey on seagull optimization algorithm and its variants. *Archives of Computational Methods in Engineering*, 1–35. https://doi.org/10.1007/s11831-025-10249-0

[14] De Oliveira, C. S., Toledo, R. D. S., Tulux, V. H., & Von Wangenheim, A. (2025). Trajectory planning for autonomous cars in low-structured and unstructured environments: A systematic review. *IEEE Access*. https://doi.org/10.1109/ACCESS.2025.3551453

[15] Yang, K., Li, S., Wang, M., & Tang, X. (2025). Interactive decision-making integrating graph neural networks and model predictive control for autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*. https://doi.org/10.1109/TITS.2025.3532936

[16] Machavaram, R. (2025). Intelligent path planning for autonomous ground vehicles in dynamic environments utilizing adaptive Neuro-Fuzzy control. *Engineering Applications of Artificial Intelligence*, *144*, 110119. https://doi.org/10.1016/j.engappai.2025.110119

[17] Paden, B., Čáp, M., Yong, S. Z., Yershov, D., & Frazzoli, E. (2016). A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles*, *1*(1), 33-55. https://doi.org/10.1109/TIV.2016.2578706

[18] Zhao, J., Zhao, W., Deng, B., Wang, Z., Zhang, F., Zheng, W., …, & Burke, A. F. (2024). Autonomous driving system: A comprehensive survey. *Expert Systems with Applications*, *242*, 122836. https://doi.org/10.1016/j.eswa.2023.122836

[19] Wang, Z., Yan, H., Wei, C., Wang, J., Bo, S., & Xiao, M. (2024). Research on autonomous driving decision-making strategies based deep reinforcement learning. In *Proceedings of the 2024 4th International Conference on Internet of Things and Machine Learning*, 211–215. https://doi.org/10.1145/3697467.3697643

[20] Zhang, Y., Zhao, W., Wang, J., & Yuan, Y. (2024). Recent progress, challenges and future prospects of applied deep reinforcement learning: A practical perspective in path planning. *Neurocomputing*, *608*, 128423. https://doi.org/10.1016/j.neucom.2024.128423

[21] Li, Z., Yu, Z., Lan, S., Li, J., Kautz, J., Lu, T., & Alvarez, J. M. (2024). Is ego status all you need for open-loop end-to-end autonomous driving? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14864–14873. https://doi.ieeecomputersociety.org/10.1109/CVPR52733.2024.01408

[22] Yuan, K., Huang, Y., Yang, S., Zhou, Z., Wang, Y., Cao, D., & Chen, H. (2024). Evolutionary decision-making and planning for autonomous driving based on safe and rational exploration and exploitation. *Engineering*, *33*, 108–120. https://doi.org/10.1016/j.eng.2023.03.018

[23] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., …, & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533. https://doi.org/10.1038/nature14236

[24] Sumiea, E. H., Abdulkadir, S. J., Alhussian, H. S., Al-Selwi, S. M., Alqushaibi, A., Ragab, M. G., & Fati, S. M. (2024). Deep deterministic policy gradient algorithm: A systematic review. *Heliyon*, *10*(9), e30697. https://doi.org/10.1016/j.heliyon.2024.e30697

[25] Teng, S., Hu, X., Deng, P., Li, B., Li, Y., Ai, Y., …, & Chen, L. (2023). Motion planning for autonomous driving: The state of the art and future perspectives. *IEEE Transactions on Intelligent Vehicles*, *8*(6), 3692–3711. https://doi.org/10.1109/TIV.2023.3274536

[26] Hoel, C. J., Wolff, K., & Laine, L. (2018). Automated speed and lane change decision making using deep reinforcement learning. In *2018 21st International Conference on Intelligent Transportation Systems*, 2148–2155. https://doi.org/10.1109/ITSC.2018.8569568

[27] Han, Z., Yang, Y., Wang, W., Zhou, L., Nguyen, T. N., & Su, C. (2022). Age efficient optimization in UAV-aided VEC network: A game theory viewpoint. *IEEE Transactions on Intelligent Transportation Systems*, *23*(12), 25287–25296. https://doi.org/10.1109/TITS.2022.3180928B

[28] Al-Kamil, S. J., & Szabolcsi, R. (2024). Optimizing path planning in mobile robot systems using motion capture technology. *Results in Engineering*, *22*, 102043. https://doi.org/10.1016/j.rineng.2024.102043

[29] Liu, J., Chen, Z., Zhang, Y., & Li, W. (2020). Path planning of mobile robots based on improved genetic algorithm. In *Proceedings of the 2020 2nd International Conference on Robotics, Intelligent Control and Artificial Intelligence*, 49–53. https://doi.org/10.1145/3438872.3439054

[30] Yuan, Q., Sun, R., & Du, X. (2022). Path planning of mobile robots based on an improved particle swarm optimization algorithm. *Processes*, *11*(1), 26. https://doi.org/10.3390/pr11010026

[31] Kong, X., Zhang, W., Hong, J., & Braunl, T. (2024). Embodied AI in mobile robots: Coverage path planning with large language models. *arXiv Preprint: 2407.02220*. https://doi.org/10.48550/arXiv.2407.02220

[32] Singh, R., Ren, J., & Lin, X. (2023). A review of deep reinforcement learning algorithms for mobile robot path planning. *Vehicles*, *5*(4), 1423–1451. https://doi.org/10.3390/vehicles5040078

[33] Chen, S., Hu, X., Zhao, J., Wang, R., & Qiao, M. (2024). A review of decision-making and planning for autonomous vehicles in intersection environments. *World Electric Vehicle Journal*, *15*(3), 99. https://doi.org/10.3390/wevj15030099

[34] Li, B., Wang, Y., Mao, J., Ivanovic, B., Veer, S., Leung, K., & Pavone, M. (2024). Driving everywhere with large language model policy adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14948–14957. https://doi.ieeecomputersociety.org/10.1109/CVPR52733.2024.01416

[35] Tian, X., Gu, J., Li, B., Liu, Y., Wang, Y., Zhao, Z., …, & Zhao, H. (2025). Drivevlm: The convergence of autonomous driving and large vision-language models. *In Proceedings of The 8th Conference on Robot Learning*, *270*, 4698-4726.

[36] Pan, C., Yaman, B., Nesti, T., Mallik, A., Allievi, A. G., Velipasalar, S., & Ren, L. (2024). VLP: Vision language planning for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14760–14769. https://doi.org/10.1109/CVPR52733.2024.01398

[37] Han, Z., Yang, Y., Wang, W., Zhou, L., Gadekallu, T. R., Alazab, M., …, & Su, C. (2022). RSSI map-based trajectory design for UGV against malicious radio source: A reinforcement learning approach. *IEEE Transactions on Intelligent Transportation Systems*, *24*(4), 4641–4650. https://doi.org/10.1109/TITS.2022.3208245

[38] Sun, W., Pan, L., Xu, J., Wan, W., & Wang, Y. (2024). Automatic driving lane change safety prediction model based on LSTM. In *2024 7th International Conference on Advanced Algorithms and Control Engineering*, 1138–1142. https://doi.org/10.1109/ICAACE61206.2024.10549175

[39] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., …, & Beijbom, O. (2020). NuScenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11621–11631. https://doi.org/10.1109/CVPR42600.2020.01164

[40] Geiger, A., Lenz, P., Stiller, C., & Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, *32*(11), 1231–1237. https://doi.org/10.1177/0278364913491297

[41] Zhao, R., Luo, C., Gao, F., Gao, Z., Li, L., Zhang, D., & Yang, W. (2024). Application-layer anomaly detection leveraging time-series physical semantics in CAN-FD vehicle networks. *Electronics*, *13*(2), 377. https://doi.org/10.3390/electronics13020377

[42] Tzoumpas, K., Estrada, A., Miraglio, P., & Zambelli, P. (2024). A data filling methodology for time series based on CNN and (Bi)

LSTM neural networks. *IEEE Access*, *12*, 31443–31460. https://doi.org/10.1109/ACCESS.2024.3369891

[43] Murugan, G., Moyal, V., Nandankar, P., Pandithurai, O., & Pimo, E. S. J. (2023). A novel CNN method for the accurate spatial data recovery from digital images. *Materials Today: Proceedings*, *80*, 1706–1712. https://doi.org/10.1016/j.matpr.2021.05.351

[44] Lee, J. S., & Park, T. H. (2020). Fast road detection by CNN-based camera–LiDAR fusion and spherical coordinate transformation. *IEEE Transactions on Intelligent Transportation Systems*, *22*(9), 5802–5810. https://doi.org/10.1109/TITS.2020.2988302

[45] Fukuoka, R., Shigei, N., Miyajima, H., Nakamura, Y., & Miyajima, H. (2021). Self-driving model car acquiring three-point turn motion by using improved LSTM model. *Artificial Life and Robotics*, *26*, 423–431. https://doi.org/10.1007/s10015-021-00697-9

[46] Karim, F., Majumdar, S., Darabi, H., & Chen, S. (2017). LSTM fully convolutional networks for time series classification. *IEEE Access*, *6*, 1662–1669. https://doi.org/10.1109/ACCESS.2017.2779939

[47] Westny, T., Olofsson, B., & Frisk, E. (2025). Toward unified practices in trajectory prediction research on Bird'View Datasets. In *2025 IEEE Intelligent Vehicles Symposium (IV)*, 83-89. https://doi.org/10.1109/IV64158.2025.11097573

[48] Chen, C., Pourkeshavarz, M., & Rasouli, A. (2024). Criteria: A new benchmarking paradigm for evaluating trajectory prediction models for autonomous driving. In *2024 IEEE International Conference on Robotics and Automation*, 8265–8271. https://doi.org/10.1109/ICRA57147.2024.10610911

[49] Teresa, R., Thomas, B., Mohammad, B., & Stefanie, M. (2024). On the way to reliable trajectory prediction in urban traffic. In *Emerging Cutting-Edge Developments in Intelligent Traffic and Transportation Systems*, 111–127. https://doi.org/10.3233/ATDE240027

[50] Schmidt, J., Monninger, T., Jordan, J., & Dietmayer, K. (2023). LMR: Lane distance-based metric for trajectory prediction. In *2023 IEEE Intelligent Vehicles Symposium*, 1–6. https://doi.org/10.1109/IV55152.2023.10186555

[51] Wu, J., Qiao, S., Li, H., Sun, B., Gao, F., Hu, H., & Zhao, R. (2024). Goal-guided graph attention network with interactive state refinement for multi-agent trajectory prediction. *Sensors*, *24*(7), 2065. https://doi.org/10.3390/s24072065

[52] Prutsch, A., Bischof, H., & Possegger, H. (2024). Efficient motion prediction: A lightweight & accurate trajectory prediction model with fast training and inference speed. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 9411–9417. https://doi.org/10.1109/IROS58592.2024.10802425

[53] Guo, X., Adl, M., Abdi, B., & Emadi, A. (2025). Intersection-specific trajectory prediction for road users: A review. *IEEE Access*, *13*, 40054–40075. https://doi.org/10.1109/ACCESS.2025.3546325

[54] Thakur, N., & Han, C. Y. (2021). An ambient intelligence-based human behavior monitoring framework for ubiquitous environments. *Information*, *12*(2), 81. https://doi.org/10.3390/info12020081

[55] Ngankam, H. K., Pigot, H., & Giroux, S. (2022). OntoDomus: A semantic model for ambient assisted living system based on smart homes. *Electronics*, *11*(7), 1143. https://doi.org/10.3390/electronics11071143

[56] Abdulmaksoud, A., & Ahmed, R. (2025). Transformer-based sensor fusion for autonomous vehicles: A comprehensive review. *IEEE Access*, *13*, 41822–41838. https://doi.org/10.1109/ACCESS.2025.3545032

[57] Ogunsina, M., Efunniyi, C. P., Osundare, O. S., Folorunsho, S. O., & Akwawa, L. A. (2024). Advanced sensor fusion and localization techniques for autonomous systems: A review and new approaches. *International Journal of Frontline Research in Engineering and Technology*, *2*(1), 51–60. https://doi.org/10.56355/ijfret.2024.2.1.0022

[58] Khaleel, A., & Ballagi, Á. (2024). Exploration techniques in reinforcement learning for autonomous vehicles. *Engineering Proceedings*, *79*(1), 24. https://doi.org/10.3390/engproc2024079024

How to Cite: Sangapu, S. C., Venkadesh, A., Toleti, J., & Sehar, S. (2025). Spatio-temporal Learning for Trajectory Prediction in Full Self-Driving Cars. *Journal of Computational and Cognitive Engineering*. https://doi.org/10.47852/bonviewJCCE52025911