smart cities

MDPI

Article

Intelligence Collision Detection Using a Combination of Tuning Base Methods and Convolutional Long Short Term Memory Models

Mohammed Hilfi * and Lubna Alazzawi *

Electrical and Computer Engineering, Wayne State University, 3126 Engineering Detroit, Detroit, MI 48202, USA
* Correspondence: mhilfi@wayne.edu (M.H.); alazzawi@wayne.edu (L.A.)

Highlights

What are the main findings?

- The research proposes a new, optimized version of customized combinations of convolutional layers with Long Short-Term Memory (LSTM) using the grid search strategy to avoid possible collisions in the current, next five-time steps, and the next ten-time steps.
- Comparing the proposed model with the state-of-the-art models like Graph Neural Network (GNN), Temporal Fusion Transformers (TFTs), Transformer, bidirectional LSTM, and demonstrating the superiority of the proposed model over the other evaluated models.

What are the implications of the main findings?

- The proposed model has been tested in two different scenarios: jaywalking and vehicle-to-motorcyclist. The proposed model has shown 99.76% accuracy, 99.77% precision, 99.76% recall, and a 97.29% F1-score for jaywalking collision detection. The model is evaluated for the detection of possible collisions between motorcycles and vehicles, demonstrating 99.58% accuracy, 97.29% precision, 97.29% recall, and a 99.76% F1-score. The proposed model has shown 91.93% accuracy, 91.90% precision, 91.90% recall, and 91.90% F1-score for the next 0.5-s collision prediction for jaywalking.
- The proposed model has shown 90.10% accuracy, 90.12% precision, 90.10% recall, and 90.10% F1-score for the next 0.5-s collision prediction between motorcycles and vehicles. The proposed model has shown 89.50% accuracy, 89.52% precision, 89.50% recall, and 89.50% F1-score for the next 1-s collision prediction for jaywalking. The proposed combination of a convolutional layer with an LSTM layer has demonstrated superior performance in all conditions. Other models, like GNN, have shown 99.41% accuracy, 96.83% precision, 96.41% recall, and 96.12 F1-score for jaywalking for the current time step collision prediction. TFT has shown 33.45% accuracy, 33.46% precision, 33.45% recall, and 33.45 F1-score for the jaywalking scenario as well. Compared to other models, bidirectional LSTM has demonstrated better results in both the jaywalking and the motorcycle with vehicles scenario, with 99.53% accuracy, 99.53% precision, 99.53%recall, and a 99.52 F1-score for the jaywalking scenario. For the second scenario bidirectional LSTM has shown 99.58% accuracy, 97.29% precision, 97.29% recall, and a 97.15% F1-score. In general, the proposed model has outperformed other models with 1 to 2% improvements over all the criteria.

Abstract

Effective traffic control using Artificial Intelligence (AI) is essential to ensure safe passage for all road users. AI-based collision detection systems offer advanced mechanisms to prevent accidents and improve highway safety. This research investigates two distinct collision scenarios: vehicle–pedestrian and vehicle–motorcyclist interactions. The proposed

Check for updates

Academic Editor: Pierluigi Siano

Received: 1 November 2025
Revised: 8 February 2026
Accepted: 23 March 2026
Published: 31 March 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

method in this research involves the bidirectional Long Short Term Memory (LSTM), Convolutional Neural Network with LSTM (CNN–LSTM), and transformer models. The model is furthermore tuned using random or grid search. For the pedestrian–vehicle scenario, the CNN–LSTM model achieved 99.76% accuracy, 99.77% precision, and 99.76% recall, highlighting its strong classification performance. In the vehicle–motorcyclist scenario, the bidirectional LSTM reached 99.73% accuracy with precision and recall of 99.15%, demonstrating its effectiveness in detecting imminent crashes. The optimized CNN-LSTM by random search has focused on decreasing the false-positive rate and increasing the positive rate. It has achieved superior results compared to previous research. These results suggest that the system could be effectively implemented as an early collision warning solution on edge devices.

Keywords: artificial intelligence; vehicle to everything; road safety; convolutional neural network; long short-term memory; CNN–LSTM; transformer model

1. Introduction

Autonomous driving is a crucial aspect of automation and an integral part of the future of life, encompassing various scenarios such as collisions between vehicles and motorbikes, pedestrians, cyclists, and other objects. Safe and proper communication, as well as Artificial Intelligence (AI) [1], are two pillars of autonomous driving. Vehicle-to-Everything (V2X) is part of the communication strategy between the vehicles and other objects on the road. Other Vulnerable Road Users (VRUs) should be considered while providing proper traffic automation systems [2]. Road traffic crashes remain a major global public health concern, causing approximately 1.19 million deaths annually. VRUs, including pedestrians, cyclists, and motorcyclists, account for more than half of these fatalities, highlighting their disproportionate risk exposure [3].

There are different communication tools, such as Basic Safety Message (BSM) [4]. To process and determine the data, two different procedures, such as edge computation or cloud computing systems, are used. By increasing the number of vehicles and VRUs on the road, the required computational capability increases exponentially. Thus, as much as the AI agent becomes heavier and deeper, the computational complexity increases [5]. There are two main subsections of AI, namely Deep Learning (DL) and Machine Learning (ML). DL uses layered neural networks to learn directly from raw data, while ML often relies on manual feature extraction and simpler models [6]. Data produced by V2X systems is episodic in nature, resulting in a time-series format that makes DL models especially well-suited for analyzing such datasets. Gated Recurrent Unit (GRU) [7] and attention [8] are sublime for extracting proper features from the time series dataset as well. GRU and attention are lightweight DL layers for extracting features from the time series data. In this research, we make the model lightweight and proper for edge computing. The evaluated models in this research are bidirectional LSTM, Convolutional Neural Network (CNN) with bidirectional LSTM [6], and the Transformer [9]. The proposed architecture is trained using BSM data, which includes geolocation, velocity, and acceleration information. In addition, the BSM signal encodes the selected trajectory for each vehicle and motorcycle.

The study employs the Simulation of Urban Mobility (SUMO) [10] together with Vehicles in Network Simulation (VEINS) [10] to generate the required dataset. The scenario for collecting the dataset involves the process of a possible collision between a pedestrian and a moving object. The second scenario is the possible collision between the motorcyclist and the vehicles.

Recent studies have primarily concentrated on simulating collision scenarios involving vehicles and VRUs, with a predominant emphasis on DL models for collision detection [11,12]. While both DL and ML approaches have been explored, previous researchers have largely overlooked critical aspects such as hyperparameter optimization and the discriminative contribution of individual features in distinguishing collision events from normal instances [13]. Furthermore, the impact of class imbalance, specifically the disproportionate representation of normal versus collision samples, has not been adequately addressed during model training, potentially affecting detection performance and generalizability [14]. This research aims to address the identified limitations and enhance the accuracy of collision detection.

The proposed methodology processes the input data and classifies samples into collision and non-collision scenarios. The model is further tuned to improve the true positive rate while reducing false positives. The remainder of the paper is organized as follows: Related work on collision detection is examined, with particular attention to research gaps and how they are addressed. The materials and methods describe the simulation process used to generate the dataset and present the architecture of the proposed collision detection system, including hyperparameter tuning. Experimental results report the outcomes obtained for each scenario and highlight key performance metrics such as accuracy and recall. The discussion provides a comparative analysis of these results against similar studies, emphasizing the advantages of the proposed approach, its potential for real-time applications, and the limitations that remain. The conclusion summarizes the methods and findings and outlines directions for future work.

The proposed method can be integrated within the smart city framework by linking traffic signals to urban traffic management instruments, thereby enabling a data-driven decision-support system for autonomous vehicles. Beyond addressing the technical challenges associated with collision detection, the approach is consistent with the broader objectives of smart cities, including safeguarding vulnerable populations, enhancing traffic efficiency, and promoting resilient urban environments.

2. Related Work

The safety of both the vehicle and pedestrians is a crucial aspect of autonomous driving. Different objects on the road need to be considered for road safety. Vehicles, pedestrians, buses, bicycles, and motorcycles are examples of objects on the road. Communication between these objects can be facilitated through safety BSM, Wi-Fi, or radio signals. Mobile Edge Computing (MEC) devices enable rapid response times for analyzing the aforementioned data and deploying AI models on it [15]. This study reviews prior work that has employed comparable combinations of traffic signals, vehicle trajectories, and physical attributes such as velocity and acceleration for collision detection. A summary of the reviewed research is provided below.

Parada et al. [11] applied the VEINS framework, which combines the SUMO traffic mobility model with the ns-3 network simulator. This setup generated controlled datasets of motorcyclist and vehicle trajectories under realistic V2X communication conditions. The inputs were time-series data, including position, velocity, and heading, sampled at regular intervals. These features were processed using a stacked unidirectional LSTM network [16]. The model was tuned to capture temporal dependencies in vehicle–VRU interactions. A sliding-window approach was used to predict collision risks. Performance was evaluated in two collision scenarios. Metrics included detection rate, Average Prediction Time (APT), Correct Decision Percentage (CDP), and false-positive count. Results showed detection accuracies of 96% (APT = 4.53 s, CDP = 41%, 78 false positives) and 95% (APT = 4.44 s, CDP = 43%, 68 false positives) for Scenarios A and B. These findings demonstrate that the

LSTM can provide early warnings while meeting V2X latency requirements. However, the study has limitations. It relies only on simulated motorcycle VRU data and lacks real-world validation. False-positive rates remain high. The evaluation is restricted to two specific collision patterns, which limits confidence in the model's generalizability to broader urban environments and diverse VRU types.

Zhang et al. [17] utilized a dataset of real-world vehicular accident combined with V2X communication logs from a beyond-5G experimental environment. A Random Forest (RF) classifier [18] was developed to identify key contributory factors and classify accident severity levels. The model achieved a classification accuracy of about 80%. Despite this result, the study does not specify the dataset's scale or diversity, lacks comparisons with other AI approaches such as deep neural networks, and does not assess latency or resilience under varying network conditions. These omissions highlight important directions for future research.

Sharma et al. [19] examined urban vehicular trajectories collected in a 6G network communication setting. The approach employs a deep deterministic policy gradient agent [20] that simultaneously analyzes speed, distance, direction, and time to detect anomalous motion patterns. Model performance was assessed using classification accuracy, achieving about 97% on the test set. Despite this strong result, the study does not clarify the dataset's scale or representativeness, lacks comparisons with other anomaly detection or reinforcement learning methods, and does not evaluate latency or robustness under different network and traffic conditions. These gaps highlight areas for further investigation.

Oliveira et al. [12]. worked on the publicly available highD dataset, which comprises 10 Hz vehicle trajectories recorded from a drone over multiple German highway sections—to train and evaluate their proposed Temporal Convolutional Network Attention (TCN-Attn) [21] model. Their model first applies a stack of TCN blocks to encode short-term motion patterns, and uses a multi-head self-attention layer to reweight those temporal features; then, it finally decodes future positions via fully connected layers. The model's performance is measured using mean displacement error and final displacement error. The TCN-Attn model lowers both errors by about 12% compared to LSTM and Social-LSTM. The authors also tested how well the model detects driving maneuvers. They reported 89% accuracy in telling the difference between lane changes and lane keeping. However, the paper didn't evaluate generalization to other road types or mixed traffic (e.g., urban streets, pedestrians), omits uncertainty quantification under realistic V2X latency and packet-loss conditions, and lacks ablation studies on the attention module's hyperparameters—gaps that future work should address.

Prathiba et al. [22] worked with a custom simulation dataset. The dataset included expert maneuvers, overtaking, and lane changes. It was generated within a 6G-V2X testbed. This dataset was used to train and validate their cooperative collision avoidance scheme for autonomous vehicles. Their model integrates inverse RL [23] augmented with Gaussian process regression to infer reward functions from limited expert demonstrations and to mimic human decision-making in overtaking and lane-change scenarios. Performance is reported in terms of classification accuracy, collision avoidance rate, and decision latency, with the proposed model achieving a classification accuracy of exactly 92.5%. However, the study's dependence on simulated data precludes assessment under real-world traffic heterogeneity; the dataset's scope and statistical properties are not fully detailed, no comparisons with alternative inverse RL or DL architectures are provided, and the impact of varying V2X network latency on system robustness remains unexplored.

Fu et al. [24] conducted a survey rather than collecting new data. They reviewed major public trajectory datasets, including Next-Generation Simulation (NGSIM) [25], highD [26], and V2X communication logs from experimental testbeds, to benchmark

collision-avoidance research in intelligent transportation systems (ITSs). Instead of proposing a single architecture, the study systematically compared a range of AI approaches. These included convolutional encoders, LSTM/recurrent neural network predictors [27], graph-neural models, and reinforcement-learning controllers, all evaluated against established safety scenarios. Performance was measured using standard collision-detection and motion-forecasting metrics. The highest accuracy reported in the surveyed literature was 96.7% on the NGSIM dataset using a hybrid CNN-LSTM model. Despite these findings, the review highlights several gaps: the absence of a unified benchmarking framework or standardized metrics, limited real-world validation under V2X latency and packet-loss conditions, and insufficient exploration of multi-agent coordination in complex urban environments.

Kandhro et al. [28] employed a customized 6G-V2X testbed that integrated IoT sensors and networking logs from autonomous vehicle trials as the source dataset. They introduced an anomaly detection framework that combines multi-agent reinforcement learning [29] with maximum entropy inverse reinforcement learning to identify and isolate rogue vehicles in real-time. Model performance was benchmarked against an IoT-V2X baseline, with the proposed approach achieving an exact 8.01% improvement in classification accuracy over existing methods. Despite this gain, the study does not specify the dataset's scale or diversity, lacks comparisons with alternative anomaly-detection architectures, and does not evaluate robustness under varying 6G-V2X latency or packet-loss conditions. These omissions highlight important directions for future research.

Ribeiro et al. [30] drew upon a V2X communications dataset generated with the VEINS cosimulation framework integrating SUMO for urban mobility and ns-3 for network behavior to capture time series streams of motorcycle and vehicle states at intersections. Their proposed system employed stacked unidirectional LSTM networks that ingest sequences of positional, velocity, and heading data to forecast imminent VRU (motorcyclist) collisions several seconds before impact. Evaluation metrics include collision prediction accuracy, APT, CDP, and false-positive count; the model achieves exactly 96% classification accuracy (Scenario A: APT = 4.53 s, CDP = 41%). Nonetheless, the study is limited by its exclusive reliance on simulated data without real-world validation, its focus on a single VRU category and intersection topology, and a relatively high false-positive rate that currently precludes automated safety interventions—gaps.

Zeng et al. [31] utilized both public traffic-flow benchmarks and proprietary V2V communication logs containing synchronized trajectories, speeds, accelerations, and relative positions to train and validate a collision-risk prediction framework. Their approach involved constructing a dynamic interaction graph of vehicles, applying a graph attention network [32] to capture spatiotemporal inter-vehicular features, and integrating deep reinforcement learning to optimize driving strategies. Model performance was assessed using true warning and false-positive rates, with the system achieving 80% true warning accuracy. Despite this result, the study does not specify which public datasets were employed or describe their statistical properties, lacks comparisons with other graph-based or sequence-model architectures, and does not evaluate robustness under varying V2V latencies, packet-loss conditions, or heterogeneous traffic densities. These limitations highlight important directions for future research.

Based on the evaluated research, the most frequently applied models for collision prediction are LSTM and CNN-LSTM. The dominant scenarios considered involve either vehicle-to-vehicle or vehicle-to-pedestrian interactions. A key limitation of the reviewed studies is that the proposed models are often evaluated in only a single scenario, which prevents validation of their robustness across diverse traffic conditions. Another gap is the reliance on the simplest form of time-dependent models, such as basic LSTM architectures,

without exploring more advanced variants. In addition, many studies fail to address the imbalance in data distribution by employing proper sampling methods, which undermines the reliability of the reported results. Furthermore, the parameters and architectures of the models are not systematically optimized through hyperparameter tuning frameworks, leaving performance improvements unexplored. To address these issues, the following solutions are proposed:

- Evaluating the proposed model for different VRUs, such as pedestrians and motorcyclists. Different collision scenarios in conjunction with three ways are investigated in this research.
- Evaluating different DL models, such as bidirectional LSTM, CNN-LSTM, and the Transformer model, for detecting the collision scenarios.
- Proposing a hyperparameter tuning strategy to tune the previous models based on decreasing the false-positive responses and increasing true positive responses.
- Proposing a new collision detection system with the ability to store information and update the weights of the models as an online learning strategy.

3. Materials and Methods

3.1. Dataset

Considering the gaps identified in prior studies, as discussed in Section 2, employing appropriate datasets across diverse scenarios is essential for developing effective collision-avoidance solutions. To address these scenarios, we focus on the case of jaywalking, which involves collisions between vehicles and pedestrians in undesignated crossing areas.

The second scenario is based on using the available datasets for collisions between motorcycles and vehicles [30]. The following is the explanation of each scenario for the collision detection.

3.2. Scenario A (Jaywalking)

The growing intricacy of urban traffic patterns and the rapid advancement of intelligent transportation networks have driven a pronounced need for high-fidelity, scalable traffic simulation frameworks. These virtual platforms offer a cost-effective, risk-free setting for evaluating traffic management strategies, refining autonomous driving algorithms and V2X communication protocols, and conducting rigorous safety assessments across a wide range of operational scenarios.

To simulate these scenarios with high realism and control, we employed a combination of well-established tools: Simulation of Urban Mobility (SUMO) [10] for microscopic traffic simulation, Objective Modular Network Testbed in C++ (OMNeT++) [33] as a discrete event simulation platform, and VEINS, which acts as a bridge between traffic dynamics and wireless communication models. This integrated setup allowed us to simulate and analyze real-world traffic behaviors involving various agent types, including vehicles, motorcyclists, and pedestrians. The primary objective of this study is to design and implement distinct traffic conflict scenarios, extract key behavioral features of agents (e.g., speed, acceleration, position, heading), and analyze their dynamics through data collected using the Traffic Control Interface (TraCI) API [34]. This scenario was carefully constructed to reflect common yet critical traffic situations that may involve safety risks or potential conflicts. VEINS is another simulation tool that was used in this research. VEINS is an open-source framework designed to enable realistic simulation of vehicular communication systems by integrating traffic mobility from SUMO with wireless communication models in OMNeT++. It provides a bidirectional, real-time coupling of the two simulators through the TraCI protocol, ensuring synchronized behavior between vehicle movement and communication logic.

In the context of this research, VEINS acts as the middleware that bridges mobility and networking. While SUMO controls the physical dynamics of vehicles (e.g., position, speed, lane changes), OMNeT++ models the communication stack, decision-making processes, and interactions between nodes such as vehicles and Road Side Units (RSUs). VEINS ensures that each mobile node in OMNeT++ accurately reflects its corresponding vehicle in SUMO throughout the simulation.

The first scenario focuses on jaywalking between two parked cars. Two vehicles are stationary on the side of a one-lane road, while a pedestrian emerges from between them and attempts to cross to the opposite sidewalk. Meanwhile, other vehicles move normally along the road. This scenario is designed to model visibility constraints and potential conflicts with oncoming traffic, as illustrated in Figure 1.

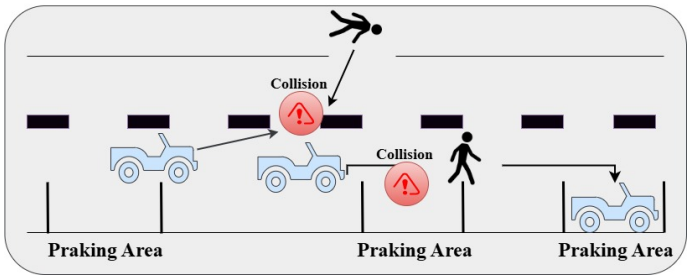


Figure 1. Pedestrian walking path between two parked vehicles and the trajectory of a vehicle attempting to park, illustrating areas of limited visibility.

The figure shows the normal paths of vehicles and the pedestrian's path between the parked cars, highlighting two potential intersections where collisions could occur. The implementation details are as follows:

- The two parked vehicles were modeled using <vehicle> tags with long-duration <stop> commands.
- The pedestrian was modeled using <personFlow> and walks from edge sideL1 to sideR2.
- Continuous traffic flow was created with a vehicle <flow> running on the same edge.
- All agents were defined in the .rou.xml file and linked via the .sumocfg.

The simulation process is based on accepting communication through the VRUs, assuming that every vehicle with a driver is equipped with a board unit. The pedestrian is equipped with a board unit. Both devices, mounted on vehicles and on pedestrians, must broadcast their data through the standard vehicular messages. The process of detecting a collision involves sharing information between VRUs, predicting the possibility of collision, and sending warning signals back to the vehicle, while also working on the modification path.

Simulation Specifications for Scenario A

The simulation length is set to 18 h. The process of communication between objects is based on basic safety messages. In these messages, the following data is transferred:

- Time (milliseconds);
- Vehicle ID;
- Longitude (m);
- Latitude (m);
- Altitude (m);
- Velocity (m/s);
- Acceleration (m/s^2);
- Angle (degrees);

- Lane ID;
- Vehicle Length (m);
- Vehicle Width (m);
- Object Types (Vehicle or Person);

The simulation specification involves a frequency of 10 Hz (1/s), a maximum transmission power of 20 dBm, a nominal range for successful delivery of 300 m, a receiver noise floor of 9 dB, a bit rate of 6 Mbps, and a payload rate of 18 Mbps. The communication strategy utilizes IEEE 802.11p technology.

To analyze the behavior of the objects in the dataset, acceleration and velocity are examined in relation to longitude and to each other, as presented below.

Figure 2 illustrates the range of velocity and acceleration, highlighting their influence on collision occurrence. The results show that most collisions arise under conditions of negative acceleration or near-zero speed.

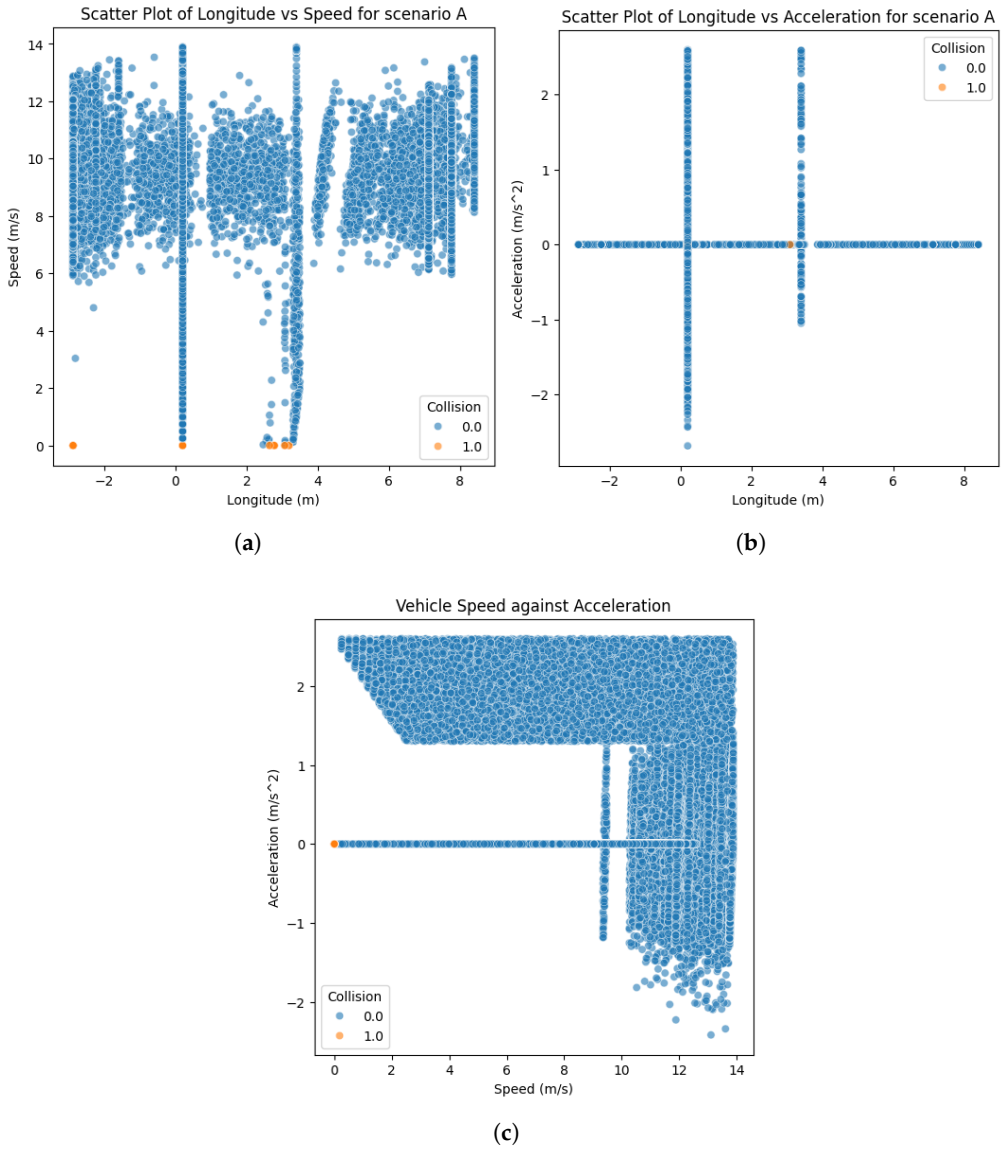


Figure 2. Representative examples from Scenario A include: (**a**) Speed plotted against Longitude, (**b**) Acceleration plotted against Longitude, and (**c**) Acceleration plotted against Speed.

3.3. Scenario B (Vehicle to Motorcyclist)

The second scenario is based on the dataset referenced in [30]. The data collection process follows the same approach as in Scenario A, utilizing the IEEE 802.11p [35] technology

with a transmission power of 20 dBm, a noise floor of −98 dBm, a minimum power level of −110 dBm, and a center frequency of 5.89 GHz. Similar features to those used in Scenario A are also applied here. Scenario B examines a potential vehicle–motorcyclist collision at a road intersection, as illustrated in Figure 3. In Scenario B1, the vehicle moves from left to right while the motorcyclist crosses along the main direction, presenting a possible point of conflict. In Scenario B2, the vehicle enters the intersection from the first entrance, while the motorcyclist crosses from the second entrance, cutting across the vehicle's path.

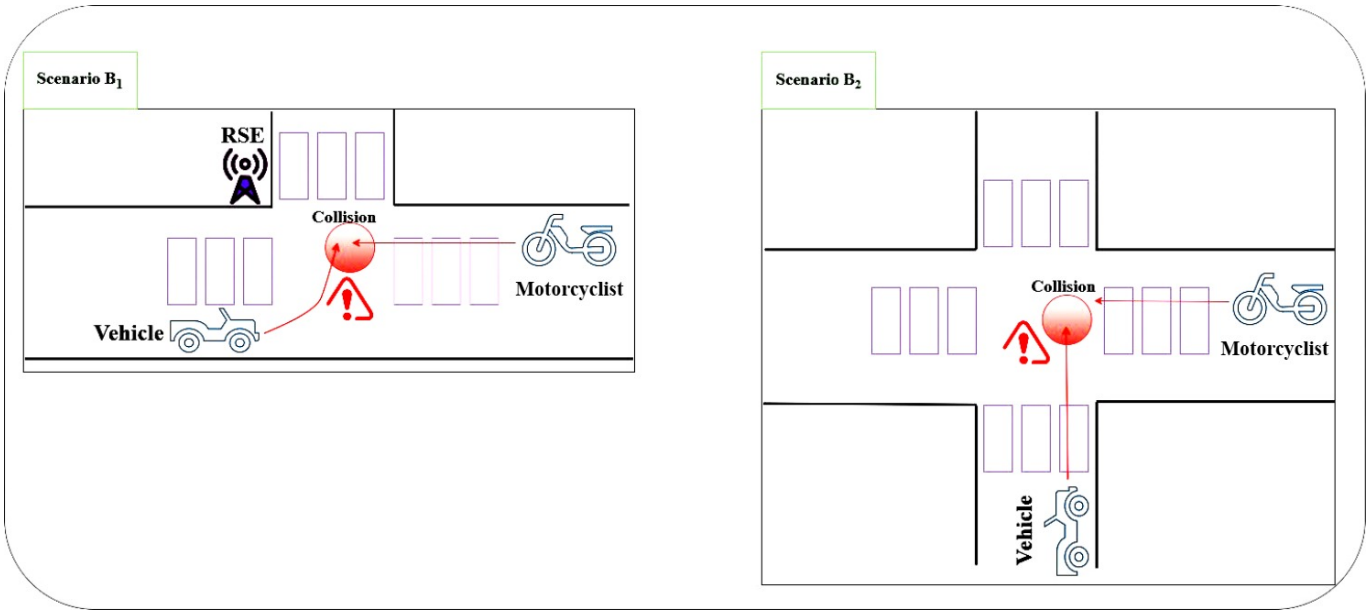
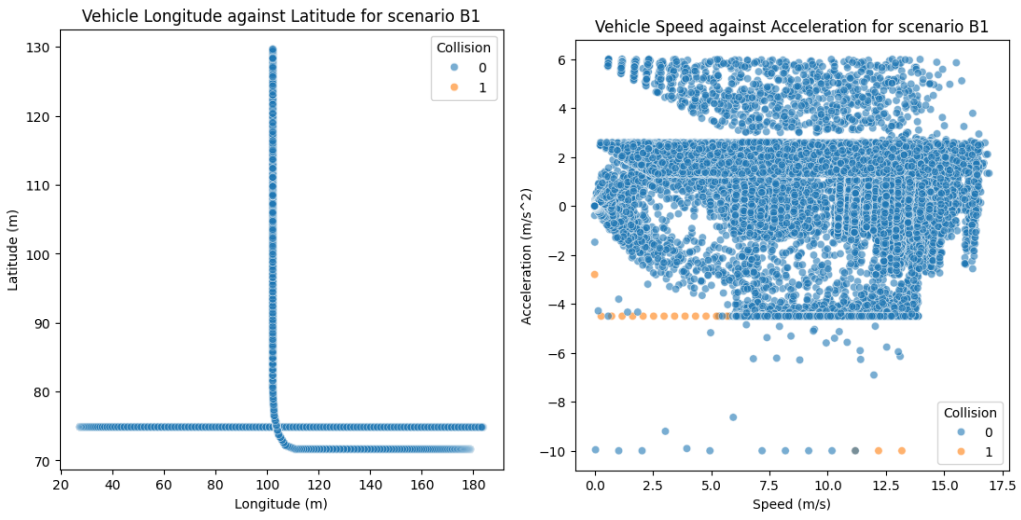


Figure 3. Illustration of Scenarios B1 and B2 showing potential collision paths between a vehicle and a motorcycle at a road intersection.

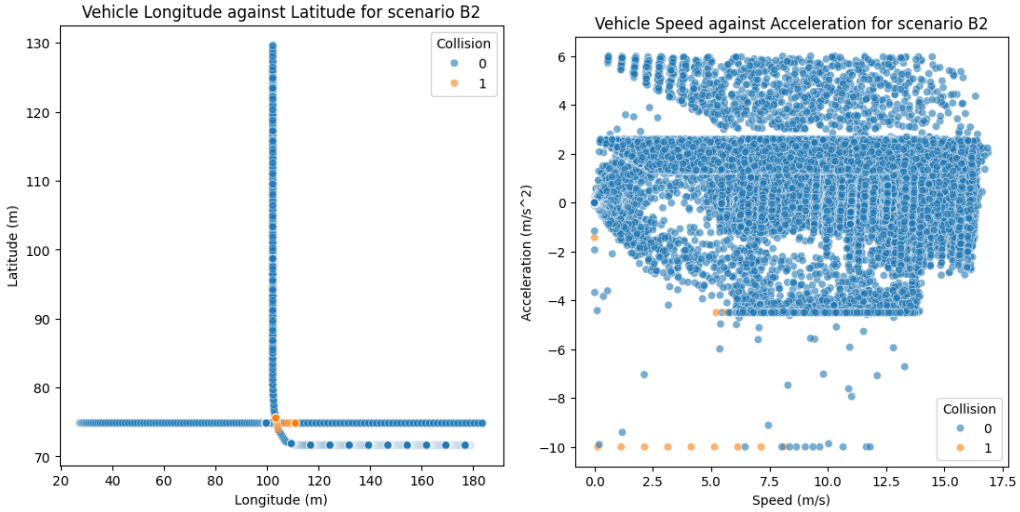
Similar to Scenario A, the behaviors of vehicles are illustrated in terms of velocity, acceleration, and their movements across longitude and latitude in the Figure 4.

Figure 4 presents similar patterns for the second scenario, where collisions are concentrated around low velocity and negative acceleration. Additionally, Figure 4 indicates that the majority of accidents occur in the middle of the road, corresponding to junction locations.



(**a**) (**b**)

Figure 4. *Cont.*



(**c**) (**d**)

Figure 4. Illustrates representative examples for Scenario B, including (**a**) latitude versus longitude for Scenario B1; (**b**) acceleration versus speed for Scenario B1; (**c**) latitude versus longitude for Scenario B2; and (**d**) acceleration versus speed for Scenario B2.

Simulation Specifications for Scenario B

The same communication settings as Scenario A are applied for Scenario B. The features are extracted from the SUMO using TraCI. Thus, the number of input features in both scenarios is the same. An equal number of features leads to the design of models that can be trained on both datasets, regardless of the number of samples for training them.

3.4. Preprocessing

The first step of the preprocessing is converting all the features into numerical values. Object type is a string feature, and we have used a label encoder to convert it into numerical values [36].

The proposed methodology is developed on a time-series dataset. During preprocessing, a min–max scaling technique is applied. A primary concern across both scenarios remains the potential for collisions.

The possibility of collision is rare, and the dataset is completely imbalanced. The balance ratio of Scenario A is shown in Table 1.

Table 1. Imbalance ratio for both scenarios based on the possibility of an accident.

Scenario Name	Number of Collisions	Number of Normal	Imbalance Ratio
A	4658	812,370	0.0057
B1	51,134	51,499,001	0.0010
B2	51,524	51,518,438	0.0010

To overcome this problem, the sampling methods are evaluated. The Synthetic Minority Over-Sampling Technique (SMOTE) [37] is used as the sampling method. Traditional SMOTE generates synthetic samples by interpolating between minority class instances in feature space, disregarding the time ordering. This can disrupt temporal dependencies and lead to unrealistic samples. To address this, adaptations like T-SMOTE have been proposed, which generate synthetic sequences while preserving temporal patterns and continuity [38]. In T-SMOTE, the algorithm first identifies minority class sequences and their temporal

neighbors, and then synthesizes new sequences by interpolating not just feature values but also their positions in time. It often focuses on generating samples near the decision boundary to improve classifier performance. -SMOTE generates collision samples within normal sequences by leveraging vehicle movements and pattern similarities to collision events. This technique preserves collision-related patterns while modifying long-term patterns among normal samples. To ensure unbiased evaluation, the validation and test sets were left unchanged.

3.5. Methodology

To tackle the process of collision detection, three models are developed based on the time-series behavior of the input dataset. The relationships between the features in both scenarios are illustrated in Figure 5.

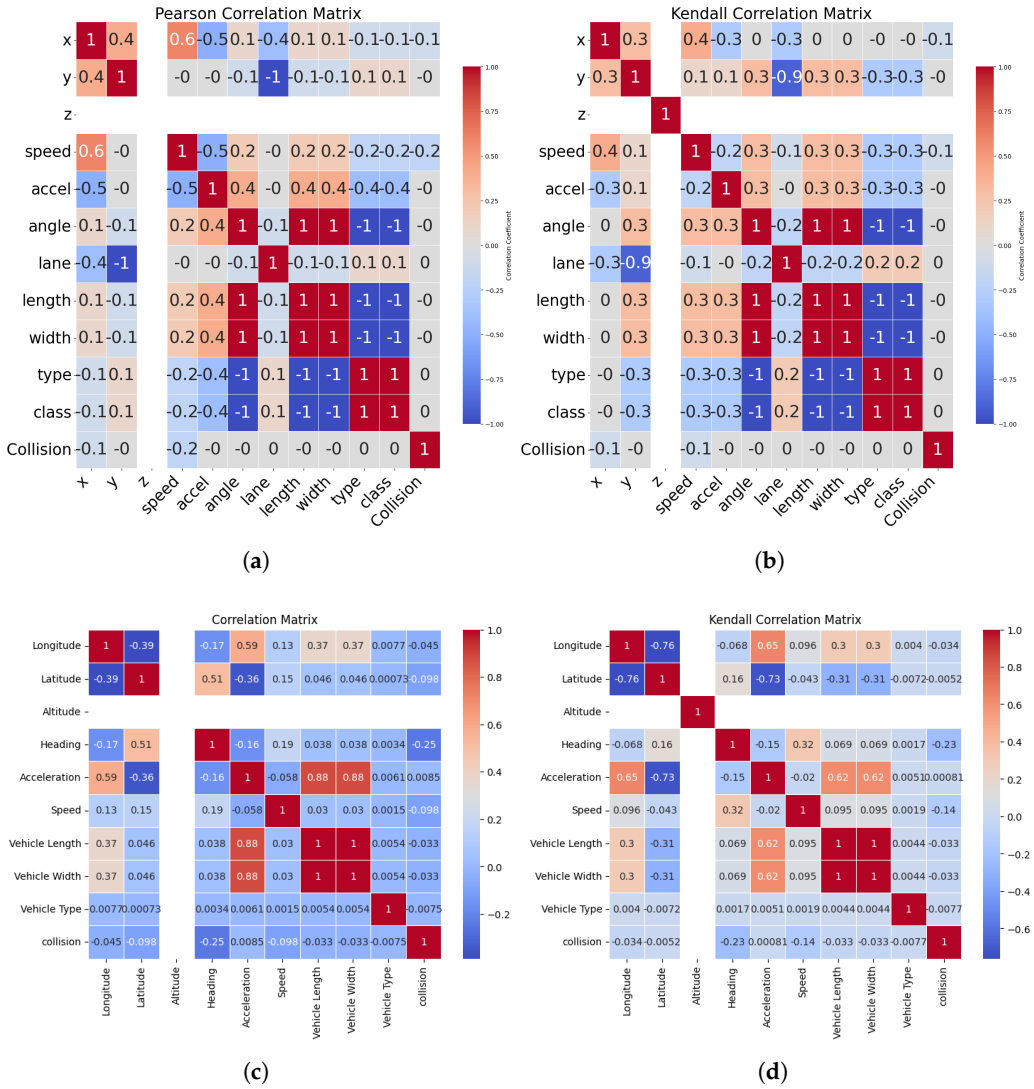


Figure 5. Correlation matrices between features for different scenarios: (**a**) Pearson correlation for jaywalking (Scenario A), (**b**) Kendall correlation for jaywalking (Scenario A); (**c**) Pearson correlation for Scenario B1; (**d**) Kendall correlation for Scenario B1.

As shown in Figure 5a,b for Scenario A, the strongest linear and non-linear correlations with collision outcomes are associated with heading and speed. However, overall correlations between the features and the target variable remain below 1%, while the weakest negative correlations fall within the range of 15% to 23%. These weak relationships create complex challenges for accurate prediction. In Figure 5c,d,, all features exhibit negative

correlations with the collision variable, with longitude and target velocity showing the strongest negative associations. Given the poor linear and non-linear correlations between features and the target, regression-based machine learning models are unlikely to yield reliable results [39]. Similarly, tree-based models struggle to identify meaningful split points under such conditions [40].

In this research, a time series-based model is employed to detect collisions in both scenarios. As highlighted in previous research, models such as LSTM, CNN-LSTM, and attention-based CNNs are commonly used for final classification tasks. Accordingly, this work uses three models: bidirectional LSTM, Transformer [41], and CNN-LSTM for classification. To optimize model performance, hyperparameter tuning was performed using random search and grid search methods [42].

3.5.1. Bidirectional LSTM

LSTM units function as specialized memory modules that selectively retain pertinent temporal information while discarding irrelevant signals. As an enhancement over standard recurrent neural networks, LSTMs are explicitly engineered to capture long-term dependencies within sequential data [6]. Their internal architecture is organized around three gating mechanisms—input, forget, and output gates—that regulate the flow of information into, through, and out of the cell. This structured gating effectively mitigates the vanishing gradient problem, enabling LSTMs to learn and represent complex temporal patterns. The architecture of the LSTM model is shown in Figure 6.

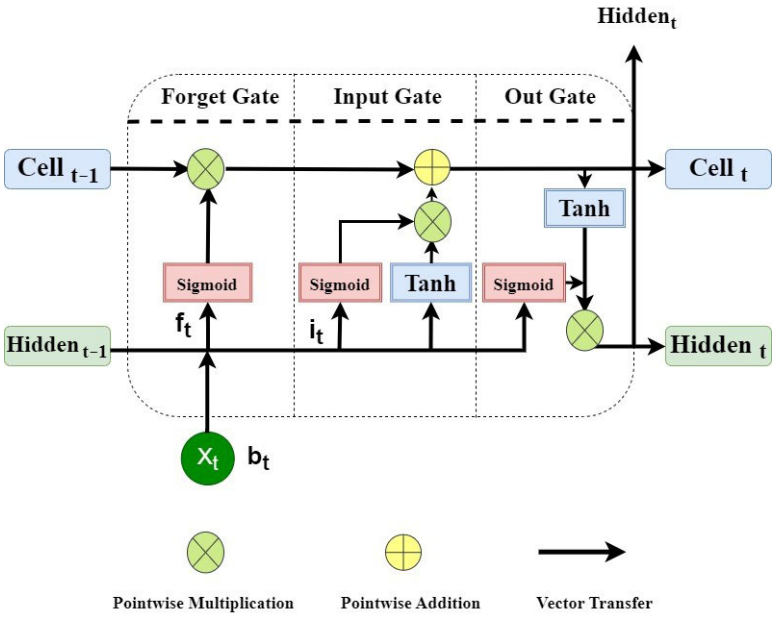


Figure 6. The architecture of the LSTM model in the vanilla format.

$$Sigmoid(X) = \frac{1}{1 + e^{-X_i}} \tag{1}$$

$$Tanh(X) = \frac{e^{Z_i} - e^{-X_i}}{e^{X_i} + e^{-X_i}} \tag{2}$$

$$Z_t = Sigmoid\left(X_i^{\Re\ m} * W_z^{\Re\ m*1} + H_{t-1}^{\Re\ m} * W_z^{\Re\ m*1} + b_z^{\Re\ 1}\right) \tag{3}$$

$$R = Sigmoid\left(X_t^{\Re^m} * W_r^{\Re^{m*1}} + H_{t-1}^{\Re^m} * W_r^{\Re^{m*1}} + b_r^{\Re^1}\right) \tag{4}$$

$$H_t^1 = Tanh\left(X_t^{\Re^m} * W_h^{\Re^{m*1}} + U_h * (R * H_{t-1}) + b_h^{\Re^1}\right) \tag{5}$$

$$U = Sigmoid\left(X_t^{\Re^m} * W_u^{\Re^{m*1}} + H_{t-1}^{\Re^m} * W_u^{\Re^{m*1}} + b_u^{\Re^1}\right) \quad (6)$$

$$H_t^2 = (1-U) * \left(H_{t-1}^{\Re^m}\right) + U * H_t^1 \quad (7)$$

where X_t is the input to the LSTM layer. W_{fis} is the weight for the forget weight for the output calculation, W_i is the ignore weight for the model, and H_{t-1} is the hidden state of the LSTM for the next stage. Equations (1) and (2) define the mathematical expressions employed for computing the sigmoid and hyperbolic tangent functions, respectively. The first equation encapsulates the transformation inherent in the sigmoid computation, while the second provides the formulation for the hyperbolic tangent. Equations (3) and (4) detail the methodology used to derive the weights associated with the mechanisms of forgetting and ignoring. These derivations form the crux of the gating processes that regulate information flow within the LSTM architecture. Lastly, Equation (7) succinctly characterizes the final output produced by the LSTM cells after integrating these computed factors. A bidirectional LSTM architecture extends the standard LSTM by incorporating two separate LSTM layers that process the input sequence in opposite directions [43]. One LSTM handles the sequence in the forward direction (from the beginning to the end), while the other processes it in the reverse direction (from the end back to the beginning). Their outputs are typically combined, often by concatenation, thus providing context from both the past and the future for each time step. In contrast, a traditional LSTM processes data unidirectionally, relying solely on past context, which may limit performance in tasks where future context significantly contributes to understanding the sequence. In this research, just one layer of bidirectional LSTM with one dense layer is used for the classification. The hyperparameters for the model are tuned using grid search.

3.5.2. CNN-LSTM

CNN is one of the most important models in AI for extracting spatial features from pictures or time-dependent data from the input features. In this research, a one-dimensional convolutional layer is employed to extract features. The formula for the convolutional layer is shown in Equation (8).

$$(F * G)(a) = \int_{t=-\infty}^{t=\infty} F(a)G(a-t)dt \quad (8)$$

where x is the input, g is the convolutional kernel, and f is the input function. The convolutional function used in the CNN originated from the concept of sparse connection [44]. The formula for the one-dimensional convolution layer is shown in Equation (9).

$$y_{c_{\text{out}}}(n) = b_{c_{\text{out}}} + \sum_{c_{\text{in}}=1}^{C_{\text{in}}} \sum_{k=0}^{K-1} W_{c_{\text{out}},c_{\text{in}}}[k]\, x_{c_{\text{in}}}(n-k) \quad (9)$$

where $X_{c_{in}}$ is the input at channel c_{in} and position n, $W_{c_{out},c_{in}}$ represents the learnable weights (kernel size), $b_{c_{out}}$ is the bias for output channel, C_{in} is the number of input channels, and c_{out} is the number of output channels. The evaluated model in this research is a combination of CNN and LSTM.

The CNN model extracts time-dependent features and passes them to the subsequent LSTM layer, which then learns and identifies the relationships between short-term and long-term information.

Figure 7 demonstrates the proposed model. The connection between the bidirectional LSTM and the convolutional layer is continuous during training. The model's hyperparameters are selected using hyperparameter tuning techniques, such as grid search.

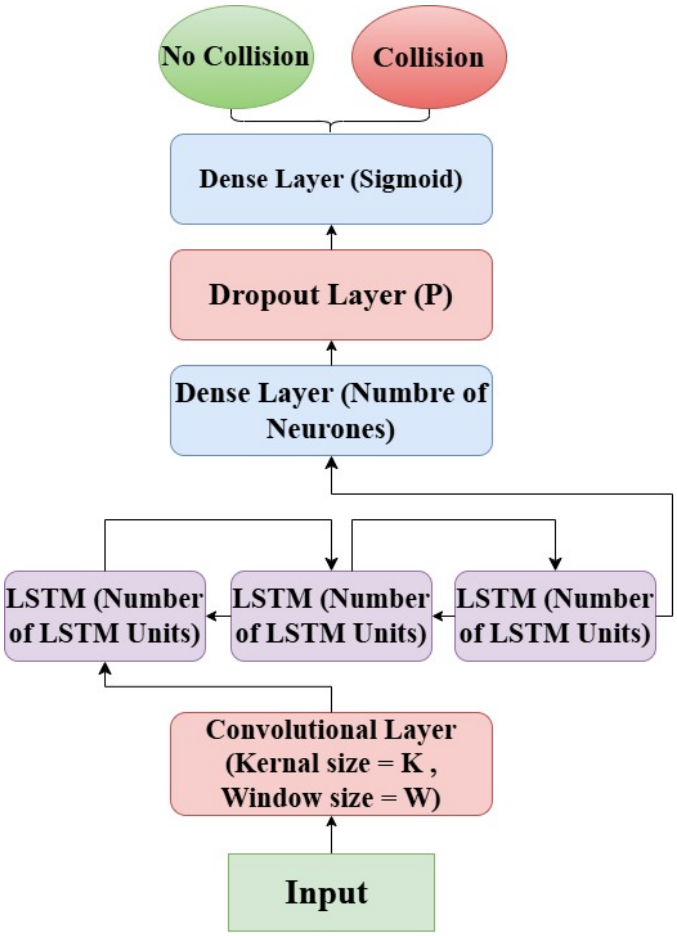


Figure 7. The architecture of the proposed CNN and LSTM layer for the collision detection.

3.5.3. Transformer Model

Up to this point, we have examined models designed to capture both short-term and long-term correlations within the dataset samples. The Transformer architecture consists of two primary components, beginning with an encoder that generates embedding vectors from the input features [45]. The Transformer model is enriched with positional encoding to extract temporal features, and multi-head attention layers were used in encoding to emphasize the important long-term dependent time series data. The attention mechanism enables the model to weigh distant time steps adaptively, which is especially powerful in modeling seasonal trends, periodic patterns, and structural shifts [8]. In this study, sparse attention was incorporated into the encoding blocks, enhancing both efficiency and performance for long-horizon forecasting [46].

After generating the encoder part, the decoder part focuses on using the embedding vectors from the encoder section and employs the multi-head attention, norm, and dense layers to produce the final classification output. The last layer for the Transformer model is similar to the CNN LSTM model, and generates the probability of collisions for each sample. In general, a Transformer model uses the produced embedding vectors and the current input at time t to predict the collision in the next time step $t + 1$. These are the basic blocks of the Transformer models, and in this research, different blocks of the Transformer model are used for generating the whole model.

Figure 8 illustrates the proposed model architecture used for classification. The model hyperparameters for predicting collision probability are determined using grid search.

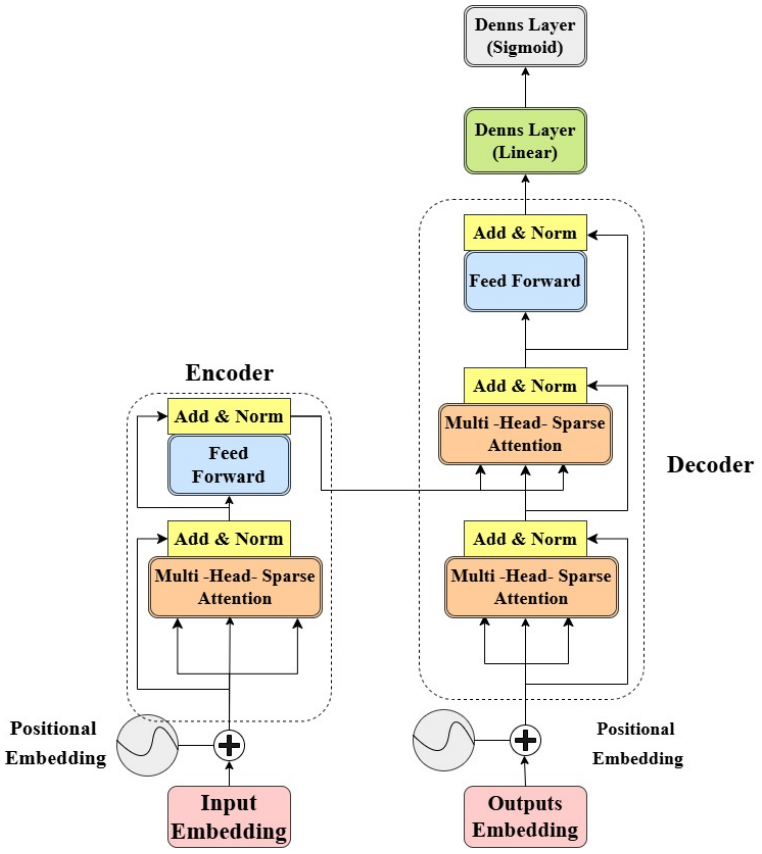


Figure 8. The architecture of the evaluated Transformer model.

3.6. Hyperparameter Tuning

The proposed method in this research examined all three models for the collision classification. The architecture of the proposed model is set, but the hyperparameters of the model are chosen by hyperparameter tuning. There are different options for each hyperparameter; to check and evaluate them, many combinations of these hyperparameters need to be evaluated. Grid search creates a grid out of all the possible scenarios between the defined ranges of the hyperparameters. The evaluated range for the hyperparameters is shown in Table 2.

Table 2 presents the evaluated options for selecting optimal hyperparameters. For Bidirectional LSTM training alone, the model was tested across 1024 distinct configurations. During training, we aimed to use the lightest possible model to facilitate real-time collision prediction. Therefore, the maximum range allowed in hyperparameter tuning for all models was limited to 4. As illustrated, exhaustively exploring all possible configurations is computationally expensive. To mitigate this, a random search strategy was employed, evaluating only a subset of the available options while applying accuracy thresholds to reduce the need for exhaustive exploration [42]. Using these techniques, the number of training subsets has decreased to 248 training cycles for CNN LSTM and just 100 for the Transform models. The optimal hyperparameter settings for each proposed model are summarized in Table 2. The criterion for choosing the best parameters is based on adding the results of accuracy, recall, and precision, and the highest value corresponds to the best hyperparameters. This approach prioritizes automation in model development, focusing on maximizing true positives and true negatives while reducing false positives and false negatives. Minimizing false positives reduces unnecessary warnings, while minimizing false negatives is critical to preventing potential accidents.

Table 2. Hyperparameter ranges and selected values for LSTM, CNN-LSTM, and Transformer models.

Bidirectional LSTM			
Hyperparameter	Lower	Upper	Chosen
Number of Dense layers	1	4	2
Hidden Dimension	32	256	128
Dropout Probability	0.1	0.5	0.3
Memory Unit	1	8	1
CNN-LSTM			
Hyperparameter	Lower	Upper	Chosen
Convolutional Filter	3	32	8
Convolutional Window	3	9	3
Number of Dense layers	1	4	2
Hidden Dimension	32	256	128
Dropout Probability	0.1	0.5	0.3
Memory Unit	1	8	1
Transformer Model			
Hyperparameter	Lower	Upper	Chosen
Hidden Dimension	32	256	128
Dropout Probability	0.1	0.5	0.3
Number of Transformer Layers	1	4	1

4. Experimental Results

This section examines the training process and the hyperparameters employed in model development. The training procedure includes selecting the batch size, optimizer, and loss function, and applying early stopping. The specific model configurations are detailed in the subsequent sections.

4.1. Training Settings

The batch size used for training varied between scenarios. For Scenario A, the batch size was set to 256, while Scenario B, having a larger number of samples, used a batch size of 2048. All models were trained for 100 epochs. To prevent unnecessary training, early stopping was applied: the model's accuracy was monitored every 20 epochs, and if no improvement was observed within this threshold, training was terminated. This approach also helped streamline hyperparameter tuning by reducing training time.

The choice for the optimizer is Adaptive Moment Estimation with decoupled Weight decay (AdamW). AdamW extends the Adam optimizer by decoupling L2 weight decay from the moment-based gradient update, applying regularization directly to parameters rather than folding it into the adaptive learning-rate term by maintaining per-parameter first and second moment estimates of gradients alongside this decoupled decay [47].The chosen option for the loss function is binary cross-entropy with logits loss, combining a sigmoid activation and the standard binary cross-entropy into one numerically stable operation. We allocate 80% of the data for training, with the remaining 10% each reserved for validation and testing. Applying the sampling method to the training set resulted in a distribution of 71% normal and 29% collision samples for Scenario A, and 83% normal to 17% collision samples for both Scenario B1 and Scenario B2. The distribution in both the validation and test sets mirrored the ratios observed in the original dataset.

To train the model, we used an NVIDIA A40 GPU that ensures high throughput and efficient memory handling, especially with large batch sizes like 1024. The A40's ample Virtual Randomized Accessible Memory (VRAM) and tensor core acceleration allow

for mixed-precision training to speed up computation and reduce memory usage. The allocated VRAM for the training is 124 GB. The designated CPU for training is Intel's 11th-Gen Core i7 with 20 cores.

4.2. Evaluation Parameters

The aim of the proposed model is classification; thus, the criterion for training is based on the classification matrix. Metrics such as accuracy, recall, precision, F1-score, true-positive, and false-positive predictions are reported in this article. The formula for calculating these criteria is mentioned as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{10}$$

$$Precision = \frac{TP}{TP + FP} \tag{11}$$

$$Sensitivit = \frac{TP}{TP + FN} \tag{12}$$

$$F1_score = \frac{2 * TP}{2 * TP + FP + FN} \tag{13}$$

$$Correct\ Decision\ Percentage\ (CDP) = \frac{Predicted_{Collision}}{False_{Positive} + Total_{Collision}} \tag{14}$$

$$Collision\ Ratio\ (CR) = \frac{Total_{Numberof Collision}}{Total_{Numberof Samples}} \tag{15}$$

where TP, TN, FP, and FN refer to the true-positive, true-negative, false-positive, and false-negative predictions. To check the number of true predictions, we have checked the total number of collisions to check the ratio of the true predictions compared to the total number of collision accidents.

4.3. Scenario A

Three different models evaluate the proposed methodology in this research. The CR for each scenario is different. For the jaywalking, the CR after sampling is 66.7%. The CR for scenario B is 44%. The results achieved for all the metrics are shown in Table 3.

Table 3. Model performance for jaywalking.

Model	Scenario	Accuracy (%)	Precision (%)	Recall (%)	False Positives	F1-Score (%)
Bidirectional LSTM	Jaywalking	99.53	99.53	99.53	22	99.52
CNN LSTM	Jaywalking	99.76	99.77	99.76	1	99.76
Transformer model	Jaywalking	99.51	0.00	0.00	448	0.00

The time steps in the future prediction are two. Table 3 displays the performance outcomes across all scenarios. Although results vary, the CNN–LSTM hybrid consistently outperforms the other models. The bidirectional LSTM ranks second, and the Transformer model comes in last. Transformer models struggle with time series forecasting primarily because their self-attention mechanism treats inputs more like sets than ordered sequences. At the same time, full self-attention scales quadratically with sequence length, forcing practitioners either to trim look-back windows, losing vital temporal context, or to down-sample aggressively, which blurs out fine-grained fluctuations that models like CNN–LSTM naturally capture [48]. On top of these architectural gaps, vanilla Transformer models are massively overparameterized for most real-world time series datasets [9,46]. They require enormous amounts of training data to avoid overfitting, while typical forecasting tasks

have modest history and noisy measurements. In this research, the simplest form of the Transformer model was used. Convolutional components in the CNN–LSTM framework extract varied feature channels from the time series, which the LSTM units then employ to model and retain temporal dynamics. The bidirectional LSTM acts as a memory mechanism that evaluates time-dependent relationships in the data from both past and future contexts. The results of the confusion matrix for the investigated scenario are shown in Table 4.

Table 4. The results of the confusion matrix for collision classification on the jaywalking scenario using CNN-LSTM.

	Predicted +	**Predicted —**
Actual +	99.77%	0.24%
Actual —	0.23%	99.76%

4.4. Scenario B

The second scenario is based on the dataset referenced in [30]. The samples for Scenario B are divided into two phases, as illustrated in Figure 3.

The first phase, B1, involves a vehicle moving toward the top right and crossing the path of a motorcyclist. The second phase, B2, consists of a car moving in a straight line while the motorcyclist moves to the left. The results for both phases are summarized in Table 5.

Table 5. Performance metrics of different models across scenarios B1 and B2.

Model	**Scenario**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**CDP (%)**	**False Positive**
Bidirectional LSTM	B1	98.56	94.39	94.39	84	25
CNN LSTM	B1	99.58	97.29	97.29	88	7
Transformer model	B1	91.12	90.79	90.79	72	30
Bidirectional LSTM	B2	99.73	97.15	97.15	87	4
CNN LSTM	B2	99.11	96.43	96.43	86	6
Transformer model	B2	91.20	90.81	90.81	73	29

Table 5 presents the performance of the proposed methods. Similarly to Table 3, the performances of the bidirectional LSTM and CNN LSTM are higher than that of the Transformer model. For scenario B1, the model is CNN LSTM, and for scenario B2, the best model is bidirectional LSTM. One FP and CDP inform us that even with a lower CR, the proposed model can detect accidents.

To understand why the Transformer will not perform well compared to others, we have gone deeper into the architecture of the Transformer and visualized the attention head features maps. The results are shown in Figure 9.

Figure 9 demonstrates that Head 8 provides the most concentrated attention, characterized by sharper and more localized activation patterns. In comparison, Head 0 exhibits a broadly distributed focus, capturing general contextual information across the dataset. A similar diffuse distribution is observed in Heads 3 and 5, indicating that the model does not strongly emphasize critical temporal or spatial features, but instead allocates attention uniformly. By contrast, CNN-LSTM and bidirectional LSTM architectures are able to capture localized spatial features, such as variations in acceleration and velocity, which are essential for effective collision detection. Additionally, the Transformer model suffers from limited feature diversity. With only 11 input features, the use of a 128-dimensional embedding space may be excessive, potentially resulting in underfitting or over-smoothing.

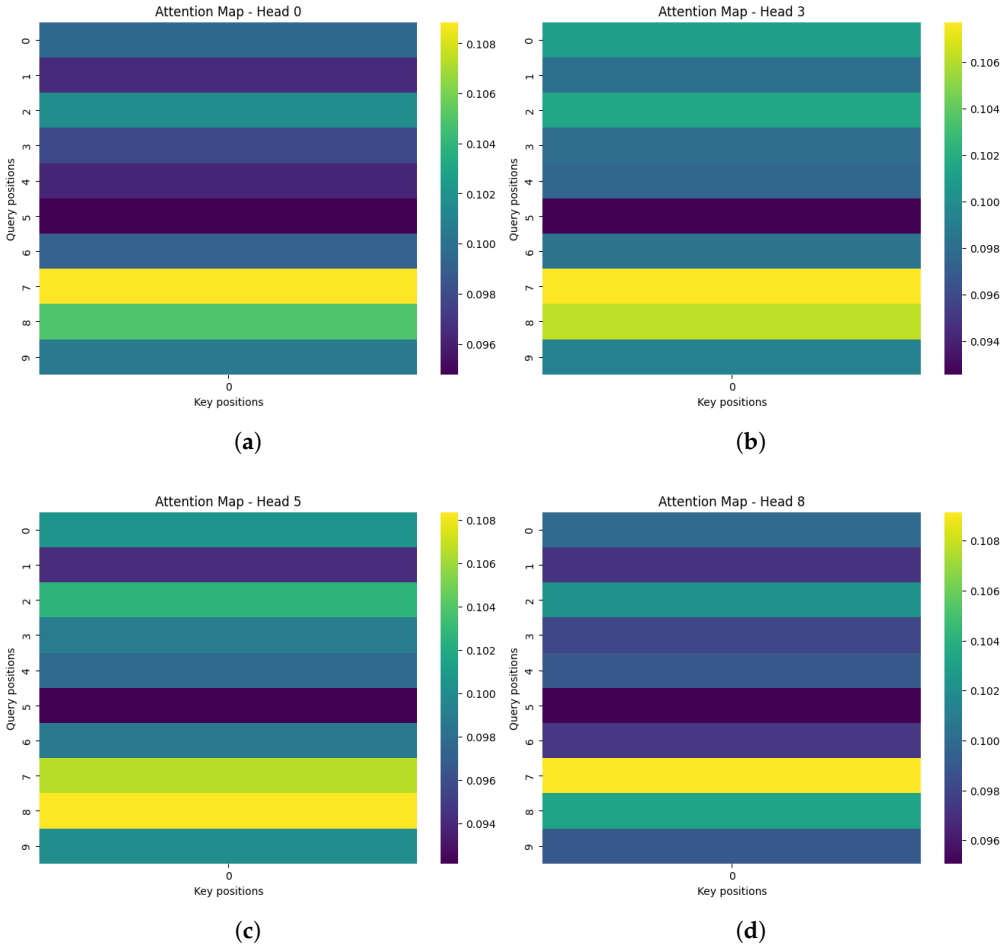


Figure 9. The attention layer produced feature maps corresponding to (**a**) the first head, (**b**) the fourth head, (**c**) the sixth head, and (**d**) the ninth head.

An evaluation of all models indicates that, while their reported accuracies appear high, the corresponding precision and recall values are considerably lower. This inflated accuracy stems from the dominance of normal instances within the validation and test sets. In certain scenarios, the proportion of positive cases is extremely limited, with ratios of 0.0057 in Scenario A and 0.0010 in Scenarios B1 and B2. Consequently, even a model that classifies all samples as normal achieves an accuracy exceeding 99%. To address this imbalance, our study emphasizes the importance of true TP and TN, aiming not only to improve overall accuracy but also to reduce FP while enhancing TP and TN performance.

Choosing the best performance among the proposed models is based on consistent performance in all metrics. However, the high FP rates lead to constant wrong alarms and reduce the eligibility of the proposed framework for autonomous driving systems. The results based on the confusion matrix are shown in Tables 6 and 7.

Table 6. The results of the confusion matrix for collision classification on scenario B1 for vehicle-to-cyclist collision using CNN-LSTM.

	Predicted +	**Predicted —**
Actual +	99.58%	2.71%
Actual —	0.42%	97.29%

Table 7. The results of the confusion matrix for collision classification on scenario B2 for vehicle-to-cyclist collision using bidirectional LSTM.

	Predicted +	**Predicted −**
Actual +	99.73%	3.57%
Actual −	0.27%	96.43%

Based on what is demonstrated in the aforementioned confusion matrix table, the main focus of the proposed model is to increase the TP while decreasing the FP simultaneously. The model performance is based on the threshold on the hyperparameter tuning; we focused on increasing the TP and decreasing the FN.

The proposed model for the second scenario focuses on forecasting the collision in two time steps ahead of the current time. This helps us to work on collision avoidance as well. By increasing the number of time steps for predicting the future, the performance of the model decreases, and the FP rate increases.

5. Discussion

Autonomous driving is part of a new area of automation. The process of a vehicle autonomous driving system requires providing safety for all objects on the roads. In this research, we have focused on identifying scenarios with collision possibilities, allowing them to be avoided sooner. The scenario involves checking for possible collisions between pedestrians and vehicles, as well as between vehicles and motorcyclists. The moving cars are considered to check the collision probability for both of them. The collisions between vehicles and motorcyclists are considered in two different paths. This study examined various collision scenarios involving vulnerable road users to enable the provision of effective early warnings.

The number of time steps for the future prediction is two. DL models, namely the Transformer model, the bidirectional LSTM, and CNN LSTM, are used for future predictions.

5.1. Comparison

This section presents a performance analysis of related studies and compares their results with the findings of the proposed approach. A comprehensive comparison between the proposed methods and existing research is presented in Table 8.

The proposed model comprises 34,945 trainable parameters and is structured with one convolutional layer, three LSTM layers, and a single dense layer. In comparison with related studies, it remains relatively lightweight and does not require extensive parameterization for training and evaluation. The studies used for comparison similarly relied on datasets organized according to the $<Sample, Features>$ format for model assessment. The main article for comparison is the work by Ribeiro et al. [30]. Ribeiro et al. [30] used LSTM for collision detection, and they reported 95% and 96% accuracies for B1 and B2 scenarios, respectively. The proposed methods in this research have outperformed them using CNN-LSTM and Bi-directional LSTM for scenarios B1 and B2, respectively. Another improvement over prior research lies in the reduction in false positives (FPs). The proposed method outperforms earlier approaches by lowering the number of FPs to 7 (from 39) in Scenario B1 and to 6 (from 33) in Scenario B2.

Compared to similar research, DL models based on time series architectures, namely LSTM and CNN LSTM, have shown promising results. The proposed model in the jaywalking scenarios has outperformed others with similar works, such as Parada et al. [11]. Parada et al. [11] worked on collision detection between vehicles and VRUs. Compared to [11], the proposed methods improved the accuracy by 3.73% (from 96 to 99.73). The

improvement was achieved using the CNN bidirectional LSTM compared to the LSTM. The proposed model outperformed similar projects using both TSMOTE and the combination of CNN-bidirectional LSTM. Increasing the number of collision samples using TSMOTE helped the model to reduce the FPs and increase the TPs. Changing the model architectures to a combination of CNN with bidirectional LSTM improved the ability of long-term feature extraction as well.

Table 8. Comparison of the proposed method with related studies based on key performance metrics, including dataset type.

Authors	Data	Dataset Description	Model	Accuracy (%)	Precision (%)	Recall (%)
Parada et al. [11]	V2X	Traffic dataset	LSTM	96	-	-
Zhang et al. [17]	SUMO	Simulated data	RF	80	-	-
Sharma et al. [19]	6G-V2X	Traffic dataset	LR	97	-	-
Oliveira et al. [12]	V2X	Traffic dataset	TCN-Attn	89	-	-
Prathiba et al. [22]	V2X (5G)	Traffic dataset	RL	92.5	-	-
Fu et al. [24]	NGSIM	Trajectory dataset	CNN-LSTM	96.7	-	-
Ribeiro et al. [30]	Scenario B1	Traffic dataset	LSTM	96	-	-
Ribeiro et al. [30]	Scenario B2	Traffic dataset	LSTM	95	-	-
Zeng et al. [31]	NGSIM	Trajectory dataset	Graph Attention	80	87	-
Proposed	Scenario A	Traffic dataset	CNN-LSTM	99.76	97.29	97.29
Proposed	Scenario B1	Traffic dataset	CNN-LSTM	99.58	99.77	99.76
Proposed	Scenario B2	Traffic dataset	Bi-LSTM	99.73	97.15	97.15

While the proposed model employs T-SMOTE as the primary sampling method, additional imbalance handling strategies such as focal loss and weighted sampling were also considered [49]. Focal loss was configured with gamma = 2 and alpha = 0.0057 for the jaywalking scenario, and alpha = 0.0010 for scenarios B1 and B2, corresponding to the respective sampling ratios and reflecting the substantial imbalance between collision and normal instances. The CNN-LSTM model was selected as the benchmark architecture for evaluating these strategies. The resulting performance metrics are presented in Table 9.

Table 9. Performance comparison of the CNN-LSTM model using focal loss and weighted sampling across different scenarios.

Scenario	Model	Accuracy (%)	Precision (%)	Recall (%)	Imbalance Handling
Jaywalking	CNN-LSTM	98.21	14.25	13.29	Focal Loss
Baseline Scenario B1	CNN-LSTM	98.26	23.48	10.69	Focal Loss
Baseline Scenario B2	CNN-LSTM	98.89	22.49	9.92	Focal Loss
Jaywalking	CNN-LSTM	97.78	27.46	26.45	Weighted Sampling
Baseline Scenario B1	CNN-LSTM	97.74	27.32	27.32	Weighted Sampling
Baseline Scenario B2	CNN-LSTM	97.48	25.40	25.40	Weighted Sampling

As shown in Table 9, the severity of the imbalance limited the effectiveness of both focal loss and weighted sampling strategies. Although the models achieved high accuracy—primarily due to the dominance of normal samples—their performance in terms of recall and precision remained weak, indicating poor sensitivity to collision events.

5.2. Ablation Study

Up to this point, our discussion has focused on simulation results. In this section, we turn to the real-world NGSIM dataset, where we introduce realistic conditions such as added noise, communication latency, and occlusion, and then re-evaluate the model under these scenarios. NGSIM was collected between 2005 and 2006 across locations in Los Angeles, Emeryville, and Atlanta. It provides detailed vehicle trajectory information, including records of cars, motorcycles, and other vulnerable road users. Using this real-world dataset, we evaluated our proposed model, and the corresponding results are presented in Table 10.

Table 10. Evaluating the proposed model on the new NGSIM datasets.

Scenario	Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
NGSIM	CNN-LSTM optimized by Grid search	99.39	98.57	98.57	99.19
NGSIM	Bidirectional LSTM optimized by Grid search	98.44	76.05	69.82	72.81
NGSIM	Transformer optimized by Grid search	98.53	98.35	98.35	98.35

As shown, the CNN–LSTM model optimized with grid search performed better than comparable studies. The proposed model was evaluated by introducing Gaussian noise, shifting frames, and applying an occlusion length of 10, with a maximum frame shift of two frames at a time. Noise was added to the velocity, acceleration, longitude, and latitude features. To assess performance in a real-world scenario, we further evaluated the model on a new dataset, and the results are presented in Table 11.

Table 11. Evaluating the proposed model on the new NGSIM datasets with added noise and conditions close to real-time scenarios.

Scenario	Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
NGSIM + Noise	CNN-LSTM optimized by Grid search	97.39	89.10	82.56	86.25
NGSIM + Noise	Bidirectional LSTM optimized by Grid search	96.65	75.49	63.80	69.15
NGSIM + Noise	Transformer optimized by Grid search	94.23	71.23	58.56	65.89

As presented in Table 11, introducing noise and modifying the frame index in the dataset leads to a decline in model performance, particularly in terms of recall, while also increasing the false-positive rate. Among the tested approaches, the CNN–LSTM architecture demonstrates the strongest resilience to these perturbations and frame shifts. Consequently, the CNN–LSTM optimized through grid search emerges as the most effective model for real-time collision detection, even under sensor noise conditions.

To further examine the effectiveness of the proposed model, we extended the evaluation to a real-time dataset. For this purpose, the NGSIM dataset was employed. The proposed approach was benchmarked against more recently introduced models, including the Graph Neural Network (GNN) [50] and the Temporal Fusion Transformer (TFT) [51]. We implemented a Graph Neural Network (GNN) comprising four graph convolutional layers, with a hidden dimension of 64 and a final output dimension of 2 to distinguish between normal and collision detection. To mitigate overfitting, we incorporated dropout layers and employed ReLU as the activation function following each graph convolutional layer.

This study employed a GNN consisting of four graph convolutional layers, configured with a hidden dimension of 64 and a final output dimension of 2 to differentiate between normal and collision detection. To reduce the risk of overfitting, dropout layers were incorporated, and the ReLU activation function was applied following each graph convolutional

layer. In addition, a TFT was implemented with a hidden size of 64 and four attention heads. The architecture comprised four Gated Residual Networks and two LSTM layers within the TFT, supplemented by two gated residual layers and a gated normalization layer. The total number of trainable parameters for the TFT was 207,000. The performance of both models, trained on Scenario B1, Scenario B2, jaywalking, and NGSIM datasets, is presented in Table 12. Table 12 indicates that the GNN outperforms the TFT. The relatively weaker performance of the TFT can be attributed to challenges similar to those faced by multi-head attention and gated recurrent networks, namely their limited ability to capture the diversity among input features. In contrast, the GNN generates embedding vectors from the input features and leverages vehicle IDs as edges to construct the network, enabling it to more effectively represent variations in acceleration and velocity compared to the TFT. Nevertheless, both evaluated models performed less favorably than the CNN LSTM optimized via random search, which achieved superior results for collision detection across all scenarios.

Table 12. Performance comparison of GNN and TFT across multiple scenarios.

Scenario	Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Jaywalking	Graph Neural Networks	99.41	96.83	96.41	96.12
Baseline Scenario B1	Graph Neural Networks	90.74	82.34	90.74	86.33
Baseline Scenario B2	Graph Neural Networks	90.71	82.14	90.71	85.14
NGSIM	Graph Neural Networks	97.78	95.61	97.78	96.68
Jaywalking	Temporal Fusion Transformers	33.45	33.46	33.45	33.45
Baseline Scenario B1	Temporal Fusion Transformers	49.20	47.10	49.20	47.66
Baseline Scenario B2	Temporal Fusion Transformers	49.87	48.35	49.87	47.52
NGSIM	Temporal Fusion Transformers	50.09	50.04	50.09	49.72

5.3. Early Warning System

The proposed model has demonstrated strong performance in real-time collision detection. However, developing an effective early warning system requires predicting collisions in advance. To evaluate the models' capability for future collision prediction, the next five and ten time steps were used as training targets. The corresponding results are presented in Table 13.

Table 13 shows that as the prediction horizon increases, the accuracy of the forecasts gradually declines. Among the evaluated models, CNN-LSTM and the Transformer achieve the best performance for future prediction. In contrast, the bidirectional LSTM performs the worst, as its memory units are less effective for long-term forecasting compared to CNN LSTM and the Transformer.

Table 13. Performance comparison of models for jaywalking detection and baseline scenarios at time steps 5 and 10.

Time Step	Scenario	Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
5	jaywalking	CNN + LSTM	91.93	91.90	91.90	91.90
5	jaywalking	Bidirectional LSTM	91.50	91.52	91.50	91.50
5	jaywalking	Transformer	85.32	94.88	72.65	82.39
5	Baseline Scenario B1	CNN + LSTM	90.50	90.52	90.50	90.50
5	Baseline Scenario B1	Bidirectional LSTM	90.10	90.12	90.10	90.10
5	Baseline Scenario B1	Transformer	83.02	92.88	70.65	80.27
5	Baseline Scenario B2	CNN + LSTM	89.50	89.52	89.50	89.50
5	Baseline Scenario B2	Bidirectional LSTM	89.10	89.12	89.10	89.10
5	Baseline Scenario B2	Transformer	82.02	91.88	69.65	79.27

Table 13. *Cont.*

Time Step	Scenario	Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
10	jaywalking	CNN + LSTM	87.02	87.06	87.02	87.02
10	jaywalking	Bidirectional LSTM	86.50	86.52	86.50	86.50
10	jaywalking	Transformer	83.02	92.88	70.65	80.27
10	Baseline Scenario B1	CNN + LSTM	85.02	85.06	85.02	85.02
10	Baseline Scenario B1	Bidirectional LSTM	84.50	84.52	84.50	84.50
10	Baseline Scenario B1	Transformer	82.02	91.88	69.65	79.27
10	Baseline Scenario B2	CNN + LSTM	83.02	83.06	83.02	83.02
10	Baseline Scenario B2	Bidirectional LSTM	82.50	82.52	82.50	82.50
10	Baseline Scenario B2	Transformer	81.02	90.88	68.65	78.27

5.4. Real-World Application

The proposed model in this research is part of the collision detection and avoidance system. The model utilizes the features of geo locations, acceleration, and velocity. The number of parameters for the CNN-Bidirectional LSTM is 236,945. The total number of parameters for the bidirectional LSTM model is 34,945. The response time for the CNN-LSTM in 1000 instances is 2 ms and 1 ms for the LSTM. Thus, the proposed methods can be mounted on the local device to receive the features and predict the collisions. All the instances between the vehicles and VRUs are recorded and sent to the cloud server for updating the models. The proposed system is shown in Figure 10. The proposed system helps the model in both collecting the proper dataset and avoiding possible collisions. Also, the proposed model can be further improved using the proposed model. The implementation of the proposed model in real-time scenarios is a gradual process that requires establishing infrastructure across edge, fog, and cloud computing environments. To ensure responsiveness and robustness, the model must be continuously trained with updated datasets. Financial constraints represent an additional challenge that must be acknowledged in the development process. To mitigate costs, the model should initially be deployed within edge and fog systems, with cloud integration as needed. Upon demonstrating success in predicting collisions under diverse scenarios, the approach can be scaled for deployment across all vehicles in the traffic network.

Beyond the technical evaluation, the system has clear pathways to real-world application. In autonomous driving, collision avoidance is a critical safety requirement. Integrating the proposed model into advanced driver assistance systems can enhance responsiveness in urban environments where VRUs such as pedestrians and cyclists are at high risk [52]. Similarly, in fleet management, logistics companies can deploy the model across connected vehicles to reduce accident rates, lower insurance costs, and improve operational safety. In addition, within the context of smart cities, the proposed system can be embedded into intelligent transportation infrastructures that combine edge, fog, and cloud computing. By leveraging real-time data from traffic sensors, connected vehicles, and pedestrian monitoring systems, the model can support proactive traffic management strategies such as dynamic rerouting, adaptive traffic signals, and early collision warnings. This integration aligns with broader smart city initiatives aimed at reducing accidents, improving mobility efficiency, and enhancing public safety [53].

5.5. Limitations and Future Work

One limitation of the proposed model is that it currently focuses only on detecting collisions between vehicles, motorcyclists, and pedestrians, excluding other types of VRUs, such as motorcyclists and scooters. Expanding the model to cover all VRUs would make it more comprehensive and enhance safety. Another limitation is response time, which could be improved by making the model smaller and more lightweight without compromising

accuracy. For future work, the model could be evaluated on new datasets to ensure robust performance under varied conditions and validated in real-world scenarios. The platform can be deployed on a Raspberry Pi 5 (16 GB), enabling the trained model to communicate with other devices via BSM messages and detect collisions in real time, demonstrating its potential for practical on-road applications. The proposed model in this study was developed using simulated data. Incorporating datasets that closely resemble real-world samples can further enhance the model's capability to detect collisions under actual conditions. In particular, collision detection can be evaluated across both light and heavy traffic scenarios by varying traffic light durations.

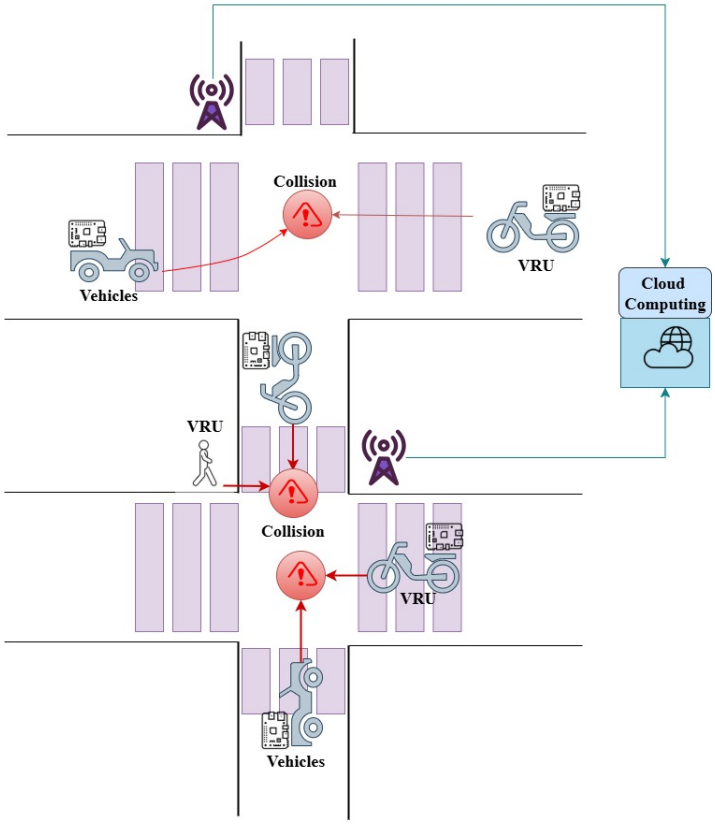


Figure 10. The architecture of the proposed avoidance system.

6. Conclusions

Road safety and traffic efficiency can be substantially enhanced through the application of deep learning (DL) techniques. In this study, a comprehensive review of prior research facilitated the identification of key findings and existing gaps. The evaluated models include Bidirectional Long Short-Term Memory (Bi-LSTM), CNN-LSTM, and Transformer architectures. Three scenarios representing potential collision situations involving pedestrians, vehicles, and motorcyclists were examined. Random search and grid search methods were applied to determine the optimal hyperparameters for all models. Experimental results show that, for pedestrian–vehicle collisions, Scenario A using CNN-LSTM achieved 99.76% accuracy, 99.77% precision, and 99.76% recall. For vehicle–motorcyclist collisions, Scenario B1 using Bi-LSTM attained 99.73% accuracy, 99.15% precision, and 99.15% recall, while Scenario B2 achieved 99.73% accuracy, 97.15% precision, and 97.15% recall using Bi-LSTM. The principal contributions of this research can be summarized as follows:

- **Framework Development:** We proposed a framework that integrates sampling techniques with deep learning model tuning to identify optimal architectures for collision avoidance.
- **Automatic Model Development:** The model architecture is selected automatically, emphasizing a lightweight design with rapid response time to reduce both false-positive and false-negative outcomes in collision detection.
- **Performance Improvement:** The proposed framework outperformed comparable deep learning models, enhancing accuracy as well as true-positive and true-negative rate predictions for collision avoidance involving vehicles, jaywalking pedestrians, and motorcyclists.
- **Practical Application:** Delivered a lightweight model suitable for real-time collision avoidance systems, achieving over 99% accuracy, along with a reliable model for early warning applications.

In conclusion, the comparative analysis of the three models indicates that CNN-LSTM achieved superior performance in Scenario A (jaywalking) and Scenario B1, while Bidirectional LSTM yielded better results in Scenario B2. Future research will aim to enhance the proposed models by exploring moderate architectural designs and employing a genetic algorithm optimizer, with performance compared against random and grid search hyperparameter tuning strategies.

Author Contributions: Conceptualization, M.H. and L.A.; Methodology, M.H.; Software, M.H.; Validation, M.H.; Formal Analysis, M.H.; Investigation, M.H.; Resources, M.H.; Data Curation, M.H.; Writing—original draft preparation, L.A.; Writing—review and editing, M.H.; Visualization, M.H.; Supervision, L.A.; Project Administration, L.A.; Funding Acquisition, M.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The dataset is available at https://zenodo.org/record/7376770, accessed on 31 October 2025.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
V2X	Vehicle-to-Everything
VRUs	Vulnerable Road Users
BSM	Basic Safety Message
LSTM	Long Short-Term Memory
DL	Deep Learning
GRU	Gated Recurrent Unit
MEC	Mobile Edge Computing
APT	Average Prediction Time
CDP	Correct Decision Percentage
SUMO	Simulation of Urban Mobility
VEINS	Vehicles in Network Simulation
NGSIM	Next-Generation Simulation
TCN-Attn	Temporal Convolutional Network Attention
IOT	Internet of Things
RF	Random Forest
RL	Reinforcement Learning
TP	True Positive

TN	True Negative
FP	False Positive
FN	False Negative

References

1. Shaheen, M.Y. Applications of Artificial Intelligence (AI) in healthcare: A review. *ScienceOpen Prepr.* **2021**. [CrossRef]
2. Liu, S.; Gao, C.; Chen, Y.; Peng, X.; Kong, X.; Wang, K.; Xu, R.; Jiang, W.; Xiang, H.; Ma, J.; et al. Towards vehicle-to-everything autonomous driving: A survey on collaborative perception. *arXiv* **2023**, arXiv:2308.16714.
3. Ahmed, S.K.; Mohammed, M.G.; Abdulqadir, S.O.; El-Kader, R.G.A.; El-Shall, N.A.; Chandran, D.; Rehman, M.E.U.; Dhama, K. Road traffic accidental injuries and deaths: A neglected global health issue. *Health Sci. Rep.* **2023**, *6*, e1240. [CrossRef]
4. Mohammadnazar, A.; Arvin, R.; Khattak, A.J. Classifying travelers' driving style using basic safety messages generated by connected vehicles: Application of unsupervised machine learning. *Transp. Res. Part C Emerg. Technol.* **2021**, *122*, 102917. [CrossRef]
5. Kalyani, Y.; Collier, R. A systematic survey on the role of cloud, fog, and edge computing combination in smart agriculture. *Sensors* **2021**, *21*, 5922. [CrossRef]
6. Zha, W.; Liu, Y.; Wan, Y.; Luo, R.; Li, D.; Yang, S.; Xu, Y. Forecasting monthly gas field production based on the CNN-LSTM model. *Energy* **2022**, *260*, 124889. [CrossRef]
7. Shiri, F.M.; Perumal, T.; Mustapha, N.; Mohamed, R. A comprehensive overview and comparative analysis on deep learning models: CNN, RNN, LSTM, GRU. *arXiv* **2023**, arXiv:2305.17473. [CrossRef]
8. Fu, H.; Guo, T.; Bai, Y.; Mei, S. What can a single attention layer learn? A study through the random features lens. *Adv. Neural Inf. Process. Syst.* **2023**, *36*, 11912–11951.
9. Belhadi, A.; Djenouri, Y.; Belbachir, A.N.; Michalak, T.; Srivastava, G. Knowledge Guided Visual Transformers for Intelligent Transportation Systems. *IEEE Trans. Intell. Transp. Syst.* **2025**, *26*, 3341–3349. [CrossRef]
10. Li, J.; Rombaut, E.; Vanhaverbeke, L. A systematic review of agent-based models for autonomous vehicles in urban mobility and logistics: Possibilities for integrated simulation models. *Comput. Environ. Urban Syst.* **2021**, *89*, 101686. [CrossRef]
11. Parada, R.; Aguilar, A.; Alonso-Zarate, J.; Vázquez-Gallego, F. Machine learning-based trajectory prediction for vru collision avoidance in v2x environments. In *Proceedings of the 2021 IEEE Global Communications Conference (GLOBECOM), Madrid, Spain, 7–11 December 2021*; IEEE: Piscataway, NJ, USA, 2021; pp. 1–6.
12. Miao, L.; Chen, S.F.; Hsu, Y.L.; Hua, K.L. How does C-V2X help autonomous driving to avoid accidents? *Sensors* **2022**, *22*, 686. [CrossRef]
13. Ahmed, A.; Farhan, M.; Eesaar, H.; Chong, K.T.; Tayara, H. From detection to action: A multimodal AI framework for traffic incident response. *Drones* **2024**, *8*, 741. [CrossRef]
14. Makris, S.; Aivaliotis, P. AI-based vision system for collision detection in HRC applications. *Procedia CIRP* **2022**, *106*, 156–161. [CrossRef]
15. Tavakolian, A.; Hajati, F.; Rezaee, A.; Fasakhodi, A.O.; Uddin, S. Fast COVID-19 versus H1N1 screening using optimized parallel inception. *Expert Syst. Appl.* **2022**, *204*, 117551. [CrossRef] [PubMed]
16. Bi, J.; Zhang, X.; Yuan, H.; Zhang, J.; Zhou, M. A hybrid prediction method for realistic network traffic with temporal convolutional network and LSTM. *IEEE Trans. Autom. Sci. Eng.* **2021**, *19*, 1869–1879. [CrossRef]
17. Nair, A.; Tanwar, S. AI-based accident severity detection scheme for V2X communication beyond 5G networks. In *Proceedings of the 2023 IEEE International Conference on Communications Workshops (ICC Workshops), Rome, Italy, 28 May–1 June 2023*; IEEE: Piscataway, NJ, USA, 2023; pp. 1002–1007.
18. Zagajewski, B.; Kluczek, M.; Raczko, E.; Njegovec, A.; Dabija, A.; Kycko, M. Comparison of random forest, support vector machines, and neural networks for post-disaster forest species mapping of the krkonoše/karkonosze transboundary biosphere reserve. *Remote Sens.* **2021**, *13*, 2581. [CrossRef]
19. Raja, G.; Begum, M.; Gurumoorthy, S.; Rajendran, D.S.; Srividya, P.; Dev, K.; Qureshi, N.M.F. AI-empowered trajectory anomaly detection and classification in 6G-V2X. *IEEE Trans. Intell. Transp. Syst.* **2022**, *24*, 4599–4607. [CrossRef]
20. Park, M.; Lee, S.Y.; Hong, J.S.; Kwon, N.K. Deep deterministic policy gradient-based autonomous driving for mobile robots in sparse reward environments. *Sensors* **2022**, *22*, 9574. [CrossRef] [PubMed]
21. Zhang, D.; Yang, J.; Li, F.; Han, S.; Qin, L.; Li, Q. Landslide risk prediction model using an attention-based temporal convolutional network connected to a recurrent neural network. *IEEE Access* **2022**, *10*, 37635–37645. [CrossRef]
22. Prathiba, S.B.; Raja, G.; Kumar, N. Intelligent cooperative collision avoidance at overtaking and lane changing maneuver in 6G-V2X communications. *IEEE Trans. Veh. Technol.* **2021**, *71*, 112–122.
23. Xue, W.; Kolaric, P.; Fan, J.; Lian, B.; Chai, T.; Lewis, F.L. Inverse reinforcement learning in tracking control based on inverse optimal control. *IEEE Trans. Cybern.* **2021**, *52*, 10570–10581. [CrossRef]

24. Fu, Y.; Li, C.; Yu, F.R.; Luan, T.H.; Zhang, Y. A survey of driving safety with sensing, vehicular communications, and artificial intelligence-based collision avoidance. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 6142–6163. [CrossRef]
25. Raj, N.; Dutta, K.K.; Kamath, A. Lane Prediction by Autonomous Vehicle in Highway Traffic using Artificial Neural Networks. In *Proceedings of the 2021 Fourth International Conference on Electrical, Computer and Communication Technologies (ICECCT), Erode, India, 15–17 September 2021*; IEEE: Piscataway, NJ, USA, 2021; pp. 1–6.
26. Moers, T.; Vater, L.; Krajewski, R.; Bock, J.; Zlocki, A.; Eckstein, L. The exid dataset: A real-world trajectory dataset of highly interactive highway scenarios in germany. In *Proceedings of the 2022 IEEE Intelligent Vehicles Symposium (IV), Aachen, Germany, 4–9 June 2022*; IEEE: Piscataway, NJ, USA, 2022; pp. 958–964.
27. Mienye, I.D.; Swart, T.G.; Obaido, G. Recurrent neural networks: A comprehensive review of architectures, variants, and applications. *Information* **2024**, *15*, 517. [CrossRef]
28. Kandhro, I.A.; Ali, F.; Panhwar, A.O.; Larik, R.S.A.; Fatima, K. Artificial intelligence (AI) empowered anomaly detection for autonomous vehicles in 6G-V2X. *Mehran Univ. Res. J. Eng. Technol.* **2023**, *42*, 79–88. [CrossRef]
29. Canese, L.; Cardarilli, G.C.; Di Nunzio, L.; Fazzolari, R.; Giardino, D.; Re, M.; Spanò, S. Multi-agent reinforcement learning: A review of challenges and applications. *Appl. Sci.* **2021**, *11*, 4948. [CrossRef]
30. Ribeiro, B.; Nicolau, M.J.; Santos, A. Using machine learning on v2x communications data for vru collision prediction. *Sensors* **2023**, *23*, 1260. [CrossRef]
31. Zeng, M.; Hashim, M.S.M.; Ayob, M.N.; Ismail, A.H.; Zang, Q. Intersection collision prediction and prevention based on vehicle-to-vehicle (V2V) and cloud computing communication. *PeerJ Comput. Sci.* **2025**, *11*, e2846. [CrossRef]
32. Vrahatis, A.G.; Lazaros, K.; Kotsiantis, S. Graph attention networks: A comprehensive review of methods and applications. *Future Internet* **2024**, *16*, 318. [CrossRef]
33. Varga, A. A practical introduction to the OMNeT++ simulation framework. In *Recent Advances in Network Simulation: The OMNeT++ Environment and Its Ecosystem*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 3–51.
34. Cunha, B.; Brito, C.; Araújo, G.; Sousa, R.; Soares, A.; Silva, F.A. Smart traffic control in vehicle ad-hoc networks: A systematic literature review. *Int. J. Wirel. Inf. Netw.* **2021**, *28*, 362–384. [CrossRef]
35. *IEEE 802.11p*; IEEE Standard for Information Technology—Local and Metropolitan Area Networks—Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments. IEEE: Piscataway, NJ, USA, 2010.
36. Jia, B.B.; Zhang, M.L. Multi-dimensional classification via sparse label encoding. In *Proceedings of the International Conference on Machine Learning*; PMLR: New York, NY, USA, 2021; pp. 4917–4926.
37. Tavakolian, A.; Rezaee, A.; Hajati, F.; Uddin, S. Hospital readmission and length-of-stay prediction using an optimized hybrid deep model. *Future Internet* **2023**, *15*, 304. [CrossRef]
38. Zhao, P.; Luo, C.; Qiao, B.; Wang, L.; Rajmohan, S.; Lin, Q.; Zhang, D. T-SMOTE: Temporal-oriented Synthetic Minority Oversampling Technique for Imbalanced Time Series Classification. In Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22), Vienna, Austria, 23–29 July 2022; pp. 2406–2412.
39. Jagtap, A.D.; Karniadakis, G.E. How important are activation functions in regression and classification? A survey, performance comparison, and future directions. *J. Mach. Learn. Model. Comput.* **2023**, *4*, 21–75. [CrossRef]
40. Rabinowicz, A.; Rosset, S. Tree-based models for correlated data. *J. Mach. Learn. Res.* **2022**, *23*, 1–31.
41. Han, K.; Xiao, A.; Wu, E.; Guo, J.; Xu, C.; Wang, Y. Transformer in transformer. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 15908–15919.
42. Rimal, Y.; Sharma, N.; Alsadoon, A. The accuracy of machine learning models relies on hyperparameter tuning: Student result classification using random forest, randomized search, grid search, bayesian, genetic, and optuna algorithms. *Multimed. Tools Appl.* **2024**, *83*, 74349–74364.
43. Imrana, Y.; Xiang, Y.; Ali, L.; Abdul-Rauf, Z. A bidirectional LSTM deep learning approach for intrusion detection. *Expert Syst. Appl.* **2021**, *185*, 115524. [CrossRef]
44. Qazi, E.U.H.; Almorjan, A.; Zia, T. A one-dimensional convolutional neural network (1D-CNN) based deep learning system for network intrusion detection. *Appl. Sci.* **2022**, *12*, 7986.
45. Amatriain, X.; Sankar, A.; Bing, J.; Bodigutla, P.K.; Hazen, T.J.; Kazi, M. Transformer models: An introduction and catalog. *arXiv* **2023**, arXiv:2302.07730.
46. Jiang, H.; Zhao, B.; Hu, C.; Chen, H.; Zhang, X. Multi-Modal Vehicle Motion Prediction Based on Motion-Query Social Transformer Network for Internet of Vehicles. *IEEE Internet Things J.* **2025**, *12*, 28864–28875.
47. Zhuang, Z.; Liu, M.; Cutkosky, A.; Orabona, F. Understanding adamw through proximal methods and scale-freeness. *arXiv* **2022**, arXiv:2202.00089. [CrossRef]
48. Pal, S.; Roy, A.; Palaiahnakote, S.; Pal, U. Adapting a swin transformer for license plate number and text detection in drone images. In *Proceedings of the Artificial Intelligence and Applications*; Springer: Berlin/Heidelberg, Germany, 2023; Volume 1, pp. 129–138.
49. Li, X.; Lv, C.; Wang, W.; Li, G.; Yang, L.; Yang, J. Generalized focal loss: Towards efficient representation learning for dense object detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 3139–3153.

50. Pagnier, L.; Chertkov, M. Physics-informed graphical neural network for parameter & state estimations in power systems. *arXiv* **2021**, arXiv:2102.06349. [CrossRef]
51. Lim, B.; Arık, S.Ö.; Loeff, N.; Pfister, T. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *Int. J. Forecast.* **2021**, *37*, 1748–1764. [CrossRef]
52. Abbasi, A.; Sobral, J.L.; Rodrigues, R. AI-Driven Virtual Power Plant Scheduling: CUDA-Accelerated Parallel Simulated Annealing Approach. *Smart Cities* **2025**, *8*, 192. [CrossRef]
53. Prieto, J.; Siano, P.; Vergura, S.; Witlox, F. Smart Cities—Announcing the Updated Scope and Sections. *Smart Cities* **2025**, *8*, 195. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.